­­



**Quality of Service – Scoring Details**

August 7, 2015 - Version 1.5

**Contents**

[1 Purpose 4](#_Toc422833746)

[2 Overview 5](#_Toc422833747)

[3 The Sample 6](#_Toc422833748)

[4 User Experience 7](#_Toc422833749)

[4.1 Measurements 7](#_Toc422833750)

[4.1.1 Search Analysis 7](#_Toc422833751)

[4.1.2 User Experience Sample 7](#_Toc422833752)

[4.2 Scoring 7](#_Toc422833753)

[5 Infrastructure Performance 8](#_Toc422833754)

[5.1 Measurements 8](#_Toc422833755)

[5.1.1 CPU Utilization (Web + SQL Servers) 8](#_Toc422833756)

[5.1.2 Available Memory (Web + SQL Servers) 8](#_Toc422833757)

[5.1.3 File Level Latency (SQL Only) 8](#_Toc422833758)

[5.1.4 SQL Server Memory Signal State & Paging (SQL Only) 9](#_Toc422833759)

[5.1.5 Virtual Log Files (SQL Only) 9](#_Toc422833760)

[5.1.6 Waits Score (SQL Only) 9](#_Toc422833761)

[5.2 Scoring 10](#_Toc422833762)

[5.2.1 Scoring the Web Servers 10](#_Toc422833763)

[5.2.2 Scoring the SQL Servers 10](#_Toc422833764)

[5.2.3 Overall Score 10](#_Toc422833765)

[6 Recoverability and Integrity 11](#_Toc422833766)

[6.1 Measurements 11](#_Toc422833767)

[6.2 Scoring 11](#_Toc422833768)

[6.2.1 Database Backups and Database Consistency Checks (DBCCs) 11](#_Toc422833769)

[6.2.2 Maximum Data Loss 11](#_Toc422833770)

[6.2.3 Time to Recover 12](#_Toc422833771)

[6.3 Overall Scoring 12](#_Toc422833772)

[7 Uptime 13](#_Toc422833773)

[7.1 Measurements 13](#_Toc422833774)

[7.2 Scoring 13](#_Toc422833775)

[8 Performance Dashboard – Component Details 14](#_Toc422833776)

[8.1 Agents: 14](#_Toc422833777)

[8.1.1 QoS Manager 14](#_Toc422833778)

[8.1.2 QoS Worker 14](#_Toc422833779)

[8.1.3 WMI Worker 14](#_Toc422833780)

[8.1.4 Trust Worker 14](#_Toc422833781)

[8.2 SQL Stored Procedures: 15](#_Toc422833782)

[8.2.1 QoS\_LookingGlass 15](#_Toc422833783)

[8.2.2 QoS\_WorkspaceAnalysis: 15](#_Toc422833784)

[9 Scoring Data Protection 16](#_Toc422833785)

[10 Best in Service and the Trust site 17](#_Toc422833786)

[10.1 The Trust Site 17](#_Toc422833787)

[10.2 Entry Point to Best in Service 17](#_Toc422833788)

[Appendix A: Search Complexity 18](#_Toc422833789)

[Appendix B– SQL Wait Types 19](#_Toc422833790)

[Proprietary Rights 22](#_Toc422833791)

1. Purpose

Performance Dashboard (PDB) is an environment monitoring and performance assessment tool that provides Relativity infrastructure administrators with feedback and alerts on the performance of their Relativity environments. This document describes the information and algorithms PDB uses to collect, analyze, and score environments.

1. Overview

The Quality of Service scoring system is what PDB uses to calculate an overall weekly score. The Quality of Service scoring system measures the following four key aspects of an environment:

* User experience
* Infrastructure performance
* Recoverability and integrity
* Uptime

PDB examines each category individually on a weekly basis and determines an overall weekly score by averaging the score of all four concepts together. All four categories are equally weighted when calculating the average.

The overall quarterly score is an average of the previous twelve overall weekly scores.

1. The Sample

PDB takes a weekly sample that represents the busiest periods of time (hours) for each workspace across the entire Relativity environment and uses that data to compute the user experience and infrastructure performance scores. This focuses the scoring for these categories on times when an environment is most likely to compromise infrastructure performance and user experience.

PDB determines the busiest hours per workspace by user arrival rate. The user arrival rate is the average arrival of queries per second (total requests within an hour/3,600 seconds).

PDB bases a second sample extraction from the larger weekly sample based on concurrency (total execution time within an hour (seconds) / 3600 seconds). This concurrency sample is used in the calculation of the user experience score.

The scoring process ignores hours when there are no users – hence no activity – in a workspace. This eliminates maintenance windows from the infrastructure performance scoring process and allows off-hours reviewers and project managers to run operations that otherwise affect the experience of other users in the system.

1. User Experience

The execution time of simple document query audits is used to determine the user experience score. This is because it reflects a user’s experience running searches and navigating views within in Relativity.

* 1. Measurements
     1. Search Analysis

PDB analyzes user searches and assigns a complexity rating based on the criteria within the search (nesting, use of Is Like, etc.) Searches fall into the following two categories that carry different scoring weights:

* + - 1. Simple Query
      2. Complex query

If a search’s execution time is greater than 2 seconds, it is classified as a Long Running Query (LRQ).

* + 1. User Experience Sample

The user experience category is an average of the arrival rate sample and the concurrency sample. PDB averages the scores generated for each of these samples to form the overall weekly score for the user experience category.

* 1. Scoring

Each user receives a score based on the percentage of simple document searches that are not long-running. Long-running searches between 2-5 seconds count against the score as though they were run twice, and searches exceeding 5 seconds count as though they were run three times. The server’s score for the hour is an average of the user scores. User experience is the average of the following two component scores:

* **Arrival rate sample hours** (top 20% by arrival rate) - the average hourly score in the arrival rate sample hours forms one component of the user experience score. The top 33 hours by arrival rate in the last week are used for the sample.
* **Concurrency sample hours** - the average hourly score in the concurrency sample hours forms the second component of the user experience score. The top 33 hours by execution time in the last week that are also in the arrival rate sample are used for the concurrency sample.

1. Infrastructure Performance

Infrastructure performance scoring reflects the underlying infrastructure’s ability to handle the volume of activity. This category is an analysis of the performance of the web and SQL servers.

* 1. Measurements

PDB analyses the following items to determine an infrastructure performance score:

* + 1. CPU Utilization (Web + SQL Servers)

This metric analyses the CPU utilization on the SQL Servers and the Web Servers. PDB collects this information every five minutes and averages it every hour.

* + - 1. Scoring Details:
* If CPU utilization is less than or equal to 60 percent, the score is 100.
* If CPU utilization greater than 85 percent, the score is zero.
* If CPU utilization is between 60 and 85 percent, a linear scale determines the score.
  + 1. Available Memory (Web + SQL Servers)

This metric analyses the available memory on the SQL servers and the web servers. PDB collects this information every five minutes and averages it every hour.

* + - 1. Scoring Details:
* Web Servers
  + If the available memory is greater than or equal to 1GB, the score is 100.
  + If the available memory is less than 1GB, a logarithmic scale determines the score.
* SQL Servers
  + If the available memory is greater than or equal to 4GB, the score is 100.
  + If the available memory is less than 4GB, a logarithmic scale determines the score.  
    1. File Level Latency (SQL Only)

PDB checks the read and write latency on database each hour, using SQL Server’s virtual file stats function (sys.dm\_io\_virtual\_file\_stats). If a SQL Server has a high amount of RAM, latency on reads becomes less of an issue, hence the file level latency is only scored when PAGEIOLATCH waits are the highest wait type.

* + - 1. Scoring Details:

File latencies are checked against the following thresholds:

* Data file reads (100ms)
* Data file writes (30ms)
* Log file writes (10ms)

If file latency exceeds the threshold and PAGEIOLATCH waits are the highest wait type, the score is zero; otherwise, the score is 100.

* + 1. SQL Server Memory Signal State & Paging (SQL Only)

SQL tracks the allocated memory state that indicates high or low memory. If the low memory state is flagged, SQL server attempts to trim in process memory. Performance suffers if large sections of in process memory are paged out.

* + - 1. Scoring Details
* If the SQL low memory state occurrence is greater than 80 percent, the score is zero.
* SQL low memory state occurrence is greater than 0 percent and page outs occurred, the score is zero.
* PDB determines the score with a linear scale from 0 to 80 percent low memory occurrences.
  + 1. Virtual Log Files (SQL Only)

Too many virtual log files (VLFs) can cause long startup and backup times. Depending on the database size, more than 1,000 VLFs indicate a problem. SQL Server issues a warning if there are more than 10,000 VLFs.

* + - 1. Scoring Details

If the number of VLFs is less than or equal to 10,000, the score is 100; otherwise the score is zero.

* + 1. Waits Score (SQL Only)

A high signal-to-resource wait ratio indicates CPU pressure. Threads spend an excessive amount of time on the runnable queue as opposed to actually running.

Poison waits are detrimental to the environment and should not appear during hours of user activity. If any poison wait type exceeds one second for the hour, the hourly system load score is zero.

* + - 1. Scoring Details
* If the signal waits ratio less than or equal to 10 percent and no there are no poison waits, the score is 100.
* If the Signal waits ratio is greater than or equal to 20%, the score is zero.
* If the Signal waits ratio is between 10 and 20 percent, a linear scale is used to calculate the score.
* If poison waits exceed one second for the hour, the score is zero.
  1. Scoring

PDB uses the following methods to determine a final infrastructure performance score.

* + 1. Scoring the Web Servers

Each web server’s score is the average of the scores of the following scores:

* CPU Utilization
* Available Memory

PDB aggregates all web server data (across sample hours) to generate one score.

* + 1. Scoring the SQL Servers

Each SQL server’s score uses the following weighted average of the scores:

* CPU Utilization – 22.5%
* Available Memory - 22.5%
* File Level Latency - 5%
* SQL Server Memory Signal State & Paging - 22.5%
* Virtual Log Files – 5%
* Waits Score – 22.5%

PDB scores each SQL server separately.

* + 1. Overall Score

The overall score for Infrastructure Performance is taken as the lowest score of all the SQL and web servers.

1. Recoverability and Integrity

Relativity workspace databases store all of the underlying metadata within Relativity. The databases include the entire work product and the reviewers coding decisions. Maintaining these critical databases is important to the integrity of the entire Relativity environment. This category analyses the frequency and coverage of database backups and integrity checks (DBCCs). PDB bases the analysis on the length of time it takes to recover a database from a backup and the maximum potential data loss.

* 1. Measurements

PDB uses the following four measurements to determine a score.

* Database backups (frequency and coverage)
* Database consistency checks (DBCCs) (frequency and coverage)
* Maximum data loss
* Time to recover
  1. Scoring
     1. Database Backups and Database Consistency Checks (DBCCs)

PDB calculates a database backup score according to frequency and coverage.

* + - 1. Frequency

Frequency represents the largest gap size that exceeds the allotted 10 day window.

PDB deducts five points per day until maximum gap size is reached (after 30 days).

* + - 1. Coverage

Coverage represents the percentage of databases in violation of the allotted 10 day window. PDB deducts points after 0.75% of databases exceed allotted 10 day window. If 10% or more of databases are in violation, the score is zero.

* + 1. Maximum Data Loss

This metric measures the maximum amount of data lost if corruption or accidental user deletion occurs. This is measured as the time between backups.

* Fifteen (15) minutes between backups is a score of 100.
* Four hours between backups is a score of 80.
* Greater than four hours, uses a linear scale down to 24 hours of lost time, where 24 hours between backups is a score of 0.
  + 1. Time to Recover

This metric measures the length of time it would take to restore the most recent full backup if corruption occurs within the database. It accounts for the most recent differential and any subsequent transaction log backups.

* If recovery time is four hours, the score is 100.
* If recovery time is one day, the score is 80.
* Between 24 to 48 hours uses a linear scale to calculate the score, where 48 hours is a score of 0.
  1. Overall Scoring

PDB determines the overall recoverability and integrity score by averaging the scores of the above four categories.

1. Uptime

The uptime category tracks the user access readiness of the Relativity environment. It analyses at the availability of the EDDS SQL server, agent servers, and web servers.

* 1. Measurements

PDB uses the following measurements to determine a score.

1. Login audits in EDDS over the last 90 days (or earliest audit)
   1. If there are any logins in an hour, you’re “up” for the hour.
2. Web server uptime
   1. At least one web servers is up and accessible.
   2. Scoring

PDB uses the following methods to determine the final uptime score.

* If the uptime is 99.99 percent or greater, the score is 100. (~1 minute of downtime per week)
* If the uptime is 98 percent, the score is 80. (43 hours of downtime per quarter)
* If the uptime is 90 percent or less, the score is 0. (216 hours of downtime per quarter)

1. Performance Dashboard – Component Details
   1. Agents:

PDB relies on the following agents to gather the aggregate data needed to score your environment.

* + 1. QoS Manager

PDB requires only one QoS Manager agent in each environment.

The QoS Manager agent executes the QoS\_LookingGlass procedure in the background to coordinate the work performed by the QoS Worker agents. The QoS Manager also performs the score calculations and secures the data through Fraud Countermeasures (FCM).

* + 1. QoS Worker

PDB requires one or more QoS Worker agents in each environment. kCura recommends that you begin with four QoS Workers and add more if needed. Large environments with 500 databases or more benefit from having six to twelve QoS Workers.

The QoS Worker agents run in the background executing the QoS\_WorkspaceAnalysis procedure within each workspace that collects and analyzes search and audit data. The workers save all data to the EDDSPerformance database.

* + 1. WMI Worker

PDB requires one or more WMI Worker agents per environment. kCura recommends that you begin with two WMI Worker agents and add more if needed. Large environments with 20 or more Relativity servers benefit from having four or more WMI Worker agents.

The WMI Worker agents run in the background collecting WMI counters and SQL server statistics for scoring purposes. They save all collected data to the EDDSPerformance database.

* + 1. Trust Worker

PDB requires one Trust Worker agent per environment.

The Trust Worker agent is only required for Best in Service partners. This agent requires you to configure a Trust ID from kCura in the Performance Dashboard application. The Trust Worker agent sends Quality of Service scores to kCura for automated quarterly audits. If the client is participates in the Trust website, it publishes Quality of Service scores at <https://trust.kcura.com>.

* 1. SQL Stored Procedures:
     1. QoS\_LookingGlass

The QoS\_LookingGlass stored procedure is the parent module that processes workspace analysis output, calculates arrival rate and concurrency, constructs a sample set for scoring, and determines the final scores for all servers.

* + 1. QoS\_WorkspaceAnalysis:

The QoS\_WorkspaceAnalysis stored procedure gathers and analyses audit data across a Relativity environment to determine the user impact of long-running, simple document searches.

1. Scoring Data Protection

All scored data within PDB is protected from modification and forgery through private hashing and encryption algorithms. This protection system is known as Fraud Countermeasures (FCM).

1. Best in Service and the Trust site
   1. The Trust Site

Best in Service customers can choose to display weekly performance metrics in a simplified score summary on the Trust site. PDB uses the overall weekly score to generate the Trust score summary. PDB assigns the following values to represent an environment’s overall performance.



* + A score of 80 percent is
  + A score between 75 and 80 percent is
  + A score that is less than 75 percent is

The score is updated every Tuesday at midnight UTC time.

* 1. Entry Point to Best in Service

kCura evaluates applications to the Best in Service program with the overall quarterly score from PDB. You must have a score of 80 percent or higher to participate. All Best in Service partners must submit scores every week, regardless of Trust site participation.

Appendix A: Search Complexity

The QoS\_WorkspaceAnalysis procedure calculates a search complexity for each search analyzed. This is used in the scoring of the User Experience category.

The search complexity is calculated with the following method:

     (total # of words in search value fields)

  + (total # of characters in parsed search text)

  + (total # of characters in dtSearch text)

  + 1 point for full text searches

  + 1 point for DTSearches

  + 1 point for searches using "in" or "contains" operators

  + (# of search folders) \* (# of search operators excluding "in", "contains", "like")

  + (# of subsearches)

  + (# of "like" operators) \* 10

**PDB considers any search with a complexity score greater than nine points as complex.**

Appendix B– SQL Wait Types

PDB collects data on 36 different wait types, all of which are listed below.

\* Indicates “Poison” waits.

**LCK\_M\_SCH\_S** - occurs when a task is waiting to acquire a **Schema Share** lock.

**LCK\_M\_SCH\_M** - occurs when a task is waiting to acquire a **Schema Modify** lock.

**LCK\_M\_S** - occurs when a task is waiting to acquire a **Shared** lock.

**LCK\_M\_U** - occurs when a task is waiting to acquire an **Update** lock.

**LCK\_M\_X** - occurs when a task is waiting to acquire an **Exclusive** lock.

**LCK\_M\_IS** - occurs when a task is waiting to acquire an **Intent Shared (IS)** lock.

**LCK\_M\_IU** - occurs when a task is waiting to acquire an **Intent Update (IU)** lock.

**LCK\_M\_IX** - occurs when a task is waiting to acquire an **Intent Exclusive (IX)** lock.

**LCK\_M\_SIU** - occurs when a task is waiting to acquire a **Shared With Intent Update** lock.

**LCK\_M\_SIX** - occurs when a task is waiting to acquire a **Shared With Intent Exclusive** lock.

**LCK\_M\_UIX** - occurs when a task is waiting to acquire an **Update With Intent Exclusive** lock.

**LCK\_M\_BU** - occurs when a task is waiting to acquire a **Bulk Update (BU)** lock.

**LCK\_M\_RS\_S\*** - occurs when a task is waiting to acquire a **Shared** lock on the current key value and a **Shared Range** lock between the current and previous key.

**LCK\_M\_RS\_U\*** - occurs when a task is waiting to acquire an **Update** lock on the current key value and an **Update Range** lock between the current and previous key.

**LCK\_M\_RIn\_S\*** - occurs when a task is waiting to acquire a **Shared** lock on the current key value, and an **Insert Range** lock between the current and previous key.

**LCK\_M\_RIn\_U\*** – occurs when a task is waiting to acquire an **Update** lock on the current key value, and an **Insert Range** lock between the current and previous key.

**LCK\_M\_RIn\_X\*** - occurs when a task is waiting to acquire an **Exclusive** lock on the current key value, and an **Insert Range** lock between the current and previous key.

**LCK\_M\_RX\_S\*** - occurs when a task is waiting to acquire a **Shared** lock on the current key value and an **Exclusive Range** lock between the current and previous key.

**LCK\_M\_RX\_U\*** - occurs when a task is waiting to acquire an **Update** lock on the current key value and an **Exclusive Range** lock between the current and previous key.

**LCK\_M\_RX\_X\*** - occurs when a task is waiting to acquire an **Exclusive** lock on the current key value and an **Exclusive Range** lock between the current and previous key.

**PAGEIOLATCH\_KP** - occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in **Keep** mode. Long waits may indicate problems with the disk subsystem. Insufficient memory in SQL Server available to cache data, or a lack of indexes on the tables involved, or queries that aren’t sargable can also cause this.

**PAGEIOLATCH\_SH** - occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Shared mode. Long waits may indicate problems with the disk subsystem. Insufficient memory in SQL Server available to cache data, or a lack of indexes on the tables involved, or queries that aren’t sargable can also cause this.

**PAGEIOLATCH\_UP** - occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Update mode. Long waits may indicate problems with the disk subsystem. Insufficient memory in SQL Server available to cache data, or a lack of indexes on the tables involved, or queries that aren’t sargable can also cause this.

**PAGEIOLATCH\_EX** - occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Exclusive mode. Long waits may indicate problems with the disk subsystem. Insufficient memory in SQL Server available to cache data, or a lack of indexes on the tables involved, or queries that aren’t sargable can also cause this.

**PAGEIOLATCH\_DT** - occurs when a task is waiting on a latch for a buffer that is in an I/O request. The latch request is in Destroy mode. Long waits may indicate problems with the disk subsystem. Insufficient memory in SQL Server available to cache data, or a lack of indexes on the tables involved, or queries that aren’t sargable can also cause this.

**IO\_COMPLETION** - occurs while waiting for I/O operations to complete. This wait type generally represents non-data page I/Os. Data page I/O completion waits appear as PAGEIOLATCH\_\* waits.

**ASYNC\_IO\_COMPLETION** - occurs when a task is waiting for I/O operations to finish.

**ASYNC\_NETWORK\_IO** - occurs on network writes when the task is blocked behind the network. Verify that the client is processing data from the server. SQL Server built the query results, and it’s waiting for the application on the other end to consume the results faster. There’s nothing you can do to performance tune the SQL Server in this case. You must determine why the app can’t retrieve the data faster. It could be a slow network line between the app and the SQL Server (like a long distance wide area network), an underpowered client machine, or row-by-row processing on the application server.

**RESOURCE\_SEMAPHORE\*** - occurs when a query memory request can’t be granted immediately due to other concurrent queries. High waits and wait times may indicate excessive number of concurrent queries or excessive memory request amounts.

**THREADPOOL\*** - occurs when a task is waiting for a worker to run on. This can indicate that the maximum worker setting is too low or that batch executions are taking unusually long, thus reducing the number of workers available to satisfy other batches.

**SOS\_SCHEDULER\_YIELD** - occurs when a task voluntarily yields the scheduler for other tasks to execute. During this wait the task is waiting for its quantum to be renewed.

**BACKUPIO** - occurs when a backup task is waiting for data, or is waiting for a buffer in which to store data. This type is not typical, except when a task is waiting for a tape mount.

**WRITELOG** - occurs while waiting for a log flush to complete. Common operations that cause log flushes are checkpoints and transaction commits.

**CMEMTHREAD** - occurs when a task is waiting on a thread-safe memory object. The wait time might increase when there is contention caused by multiple tasks trying to allocate memory from the same memory object.

**CXPACKET** - occurs with parallel query plans when trying to synchronize the query processor exchange iterator. If waiting is excessive and cannot be reduced by tuning the query (such as adding indexes), consider adjusting the cost threshold for parallelism or lowering the degree of parallelism.

**RESOURCE\_SEMAPHORE\_QUERY\_COMPILE\*** - occurs when the number of concurrent query compilations reaches a throttling limit. High waits and wait times may indicate excessive compilations, recompiles, or un-cacheable plans.

Proprietary Rights

This documentation (“Documentation”) and the software to which it relates (“Software”) belongs to kCura Corporation and/or kCura’s third party software vendors. kCura grants written license agreements which contain restrictions. All parties accessing the Documentation or Software must: respect proprietary rights of kCura and third parties; comply with your organization’s license agreement, including but not limited to license restrictions on use, copying, modifications, reverse engineering, and derivative products; and refrain from any misuse or misappropriation of this Documentation or Software in whole or in part. The Software and Documentation is protected by the Copyright Act of 1976, as amended, and the Software code is protected by the Illinois Trade Secrets Act. Violations can involve substantial civil liabilities, exemplary damages, and criminal penalties, including fines and possible imprisonment.

**©2015 kCura LLC. All rights reserved. Relativity® and kCura® are registered trademarks of kCura Corporation.**