STONERIDGE SOFTWARE HEALTH CHECK REPORT

Prepared by: Internal Systems

Email: solutions@stoneridgesoftware.com

[enable your business]

Table of Contents

INTRODUCTION

The Health Check for Microsoft Dynamics® AX is a proactive service designed to deliver proactive problem identification and suggested resolution of issues for selected components of your Microsoft Dynamics AX deployment. It is designed to uncover potential performance and configuration risks and provide recommendations for any identified issues.

The Health Check offers formal recommendations in a scorecard deliverable that covers each reviewed component of the Microsoft Dynamics AX implementation.

Remediation can be added to the Health Check. This provides time for a Technical and or functional consultant to work with you on how to implement the recommendations in your environment.

Scorecard Reporting Overview

The findings are summarized with a risk assessment category denoted by 1 of the 3 icons detailed below:

Icon	Description	Meaning
	CRITCAL	This icon means the item requires ***immediate attention***. Failure to take action could result in reduced performance and data loss. Remediation should take highest priority.
	Concern	This icon means the item may require attention or a recommended suggestion has been provided. The item is operating below a critical status but above an ok status. The item, if not remediated may result in reduced performance or escalation to a critical status later on.
	Satisfactory	This item does not require attention and is operating normally.

Remediation and Follow-up

It is essential to follow up on issues that are discovered after the engagement, and then address them to ensure that the Microsoft Dynamics AX infrastructure functions correctly and optimally. It is incumbent upon the customer to remediate the issues identified unless an additional engagement or understanding is in place.

Executive Summary

Dynamics AX

The configuration of Dynamics AX has a few areas in which some slight adjustments could help improve the performance and stability of your system. The alerts batch job consistently runs for long periods with tables that are in your 'top 10 tables by activity' with alerts set up on them. Adjustments in the area of what alerts are setup and/or the addition of processing power for batch jobs could assist in the performance of not only batch jobs, but also the Dynamics AX system. The AOS Servers appear to be performing well with the memory utilization for the AOS service to be acceptable for the number of users. Based on the windows event logs the AOS service is healthy without crashes but there are a few event log errors that should be addressed.

The largest concern is the lack of change management around development and process around code deployment. There is currently no way to track what customizations have been requested, no central location for functional design documents (FDD), no development practices around documenting code or writing up a technical design document for a customization/FDD nor TFS with work items to link code to a functional design/customization. This not only makes it difficult to design testing plans or communicate to end users what needs to be tested but if an issue is found it is challenging to fix or remove from a build. There is also no consistency in how code is deployed to another environment – some deployments have been via xpo and others via a modelstore. This could very likely lead you to a state of model id conflicts and possibly data corruption. These are all items that if addressed could significantly affect your release schedule and stability.

SQL Server

For SQL there are more optimal configuration settings that could be implemented to optimize performance. The recommendation would be to gradually enact the changes in a few specific areas right away. The performance counters and numerous errors in the Windows Event logs indicate critical disk issues that should be addressed immediately. The existing drives are not at the recommended allocation unit sizes for SQL so as you address the disk performance this should be considered as part of that resolution plan. As you address the disk performance issue, keep in mind that the tempdb would benefit from having more than one data file. With the number of processors you have it would be recommended to have eight tempdb data files all sized the same. The other configuration change that should be made is to set your max degree of parallelism to 1 as this is a best practice for Dynamics AX. Holistically on the performance side, there are some areas for improvement through performance tuning. This tuning can be in the form of recommended indexes, dropping duplicate indexes, and reviewing the top worst performing queries.

For high availability (HA) and disaster recovery (DR) planning for the Dynamics AX database, there is not a specific SQL Server HA/DR features used. The recommendation is to consider leveraging SQL Server specific features such as database mirroring, log shipping, failover clustering or AlwaysOn Availability groups to provide a robust HA/DR plan. Also, be sure to review your RPO and RTO objectives to verify that your existing solution(s) can meet your needs.

Based on the health assessment of your systems, the following items are the Top 5 recommendations we suggest to implement. These recommendations are based on the potential overall impact to the system. Please be aware that this list is not all inclusive of the recommendations provided in this health check.

Top 5 Recommendations to Implement

Recommendation	Reason
Change SQL Server **Max Degree of Parallelism** to 1	This is a Microsoft product specific best practice
Implement change control and code deployment process.	Currently there is no method for tracking customizations in the system, which not only affects your ability to test changes but research and resolve issues found.
Clean up existing alerts and develop process for adding alerts to the system. Also, address the batch configuration for alerts.	Currently the batch job for alerts runs at times for hours which impacts the start time of other batch jobs in the system.
Address disk performance issues at the SQL Server and include then the increased number of TempDB data files from 1 to 8.	The read and write performance on the SQL Server are above the recommended values and some are in the range that demand immediate attention. The file number recommendation is a performance optimization that will lead to optimal TempDB configuration.
Consider archiving or purging data from the following tables: • **BATCHISTORY** • **AIFDOCUMENTLOG**	This will provide the benefit of reclaiming disk space from potentially unnecessary data and increase query performance against these tables.

Scorecard Summary

The following is a summary review of the components and areas that were assessed. Additional details on each of these items are outlined in further sections.

Dynamics AX

Component	Risk Assessment
System Configuration	
Application Object Server	
Change management and code deployment	

SQL Server

Component	Risk Assessment
Error Logs	
Windows Event Logs	
Backup Strategy	
Index and Statistics Maintenance Plan	
Settings Configuration	
TempDB Configuration	
Drives Configuration	
Virtual Log File Count	

SQL Server Reporting Services

Component	Risk Assessment
Error Logs	
Windows Event Logs	

Top Issues and Recommendations

Microsoft Dynamics AX

Application Object Server

Issue	Recommendation	Severity
Various Event Log Errors and Warnings	Review details under the event log section for each issue.	
Large Tables set to cache table	Change table cache options for tables too large to fit in cache	
Table Cache Candidates	Review the tables that could fit in cache	
Batch Server Configuration	Make the recommended changes for optimal performance	

Dynamics AX Configuration

Issue	Recommendation	Severity
Continuous number sequences	Review and determine if these should be set as continuous or consult Stoneridge Software Functional Tech to assist.	
Alerts	Remove alerts setup on high transaction volume tables and address batch job configuration to assist with performance of alerts batch job.	

Change management and code deployment

Issue	Recommendation	Severity
No change management process	Implement TFS	
Code deployment methodology not established	Configuration development environment so each developer has an all in one machine and implement a build server. Deploy modelstore exports to other environments as method of deploying code.	

Microsoft SQL Server

Database Services

Issue	Recommendation	Severity
SQL Server version is out of date	Patch SQL Server to more recent version	
No current SQL HA/DR feature	Consider implementing a SQL specific HA/DR feature. Or review current HA/DR plans to ensure RPO/RTO objectives are met	
Server configuration changes	Review the recommendations for more optimal configuration	
Excessive log growths	Pre-size the data and log files to prevent excessive growths and address VLF count issue	
TempDB file number may be incorrect	Review Tempdb configuration recommendation	
Change max degree of parallelism	Change max degree of parallelism to 1 for optimal AX performance	
Enable optimize for ad hoc workloads	Enable optimize for ad hoc workloads for overall performance	

Non-Optimal Drive Allocation Format	Consider reformatting drives with different allocation unit size for optimal performance	
Trace Flags	Consider adding recommended trace flags	
Missing Indexes	Review missing indexes and performance test adding	
Indexes Missing From AOT	Review missing indexes in AOT	
Duplicate Indexes	Consider removing duplicate indexes	
Unused Indexes	Consider removing unused indexes (review recommendation and notes)	
Disk performance counters outside normal range	Address disk performance issues with SAN vendor	
Database Maintenance Plans	Database and log backups are occurring and indexes and statistics are up to date	
Errors and Warnings in Event log and error log	Review Error log and event log recommendations	

Reporting Services

Issue	Recommendation	Severity
Errors and Warnings in Event log and error log	Review Error log and event log recommendations	
Excessive log growths	Pre-size the data and log files to prevent excessive growths and address VLF count issue	

Analysis Services

Issue	Recommendation	Severity
No logging and model refreshes are occurring		

Dynamics AX

Overview

Description	Value
Kernel Version	6.3.1000.1310
Application Version	6.3.1000.309
Continuous number sequences	36 of 94 number sequences are continuous
Database logging	Not configured
Alerts	121 Alerts
Virtual Company	None
Concurrency mode	Optimistic Per Table
Table caching	3 tables are too large for cache 88 tables could be cached
Users with Sys Admin Permission	4 of 47 Total Users

Continuous Number Sequence

Sequence Code	Sequence Code	Sequence Code	Sequence Code
Acco_18	Acco_60	Acco_66	Sale_7
Acco_19	APL	Purc_7	Sale_10
Acco_23	Inve_21	Bank_1	Sale_17
Acco_3	Inve_29	Bank_13	Sale_18
Acco_36	Inve_33	Basi_4	Sale_22
Acco_42	Prod_10	Basi_12	Sale_23
Acco_5	Prod_12	Basi_13	Sale_28
Acco_53	Prod_16	Gene_1	Sale_33
Acco_56	Prod_18	Gene_15	Sale_41

Recommendation
Review and determine if these should be set as continuous or consult Stoneridge Software Functional Tech to assist. Number sequences can be continuous or non-continuous. A continuous number sequence does not skip any numbers, but numbers may not be used sequentially. Numbers from a non-continuous number sequence are used sequentially, but the number sequence may skip numbers. For example, if a user cancels a transaction, a number is generated, but not used. In a continuous number sequence, that number is recycled later. In a non-continuous number sequence, the number is not used. Continuous number sequences are typically required for external documents, such as purchase orders, sales orders, and invoices. However, **continuous number sequences can adversely affect system response times because the system must request a number from the database every time that a new document or record is created**. If you use a non-continuous number sequence, you can enable Preallocation on the Performance FastTab of the Number sequences form. When you specify a quantity of numbers to preallocate, the system selects those numbers and stores them in memory. New numbers are requested from the database only after the preallocated quantity has been used. ***Unless there is a regulatory requirement that you use continuous number sequences, we recommend that you use non-continuous number sequences for better performance***.

Database Logging Configuration

Table Name	Field Name	Change Type
Batch job		Update
Batch Transactions		Update
Sales orders		Update
Customers		Update

Recommendation
Remove batch options. Additionally, review sales orders and customers to determine if specific fields can be added for logging.

Alerts by Table

Table Name	User	Rule
TSTIMESHEETLINE	John Doe	Field Approval status in table Timesheet line is set to: Returned
CUSTTABLE	John Doe	A record is inserted
ProjTransBudget	John Doe	Field Amount in table ProjeTransBudge has changed

Recommendation
None, currently only 1 alert is set.

Cached Tables Too Large To Fit In Cache

Table Name
LOGISTICSADDRESSCOUNTRYREGIONTRANSLATION
AIFACTION
SRSANALYSISENUMS

Recommendation
Change **CacheLookup** from **EntireTable** to **None** for the tables listed above.

Entire Table Cache Candidates

Table Name
XREFDIALOGUPDATE
WORKFLOWNOTIFICATIONSTAGING

Recommendation
Review these tables and consider setting the **CacheLookup** from None to **EntireTable**. This can help reduce roundtrips between the AOS and SQL increasing performance. These tables should be considered only if they are updated infrequently.

Batch Server Configuration

Batch Group Name	AOS Server Name
	01@PROD
DataUpdate	01@PROD
Subledger	01@PROD
WF	01@PROD
test	01@PROD

Recommendation

Batch Server	Batch Schedule	Threads	Comments
01@PROD	12:00 AM – 12:00 PM	8	Reduce thread count from 8 to 6 to match CPU core count.

Users with Sys Admin Permission

User Names
AXadmin
AXAOS
User1
User2

Recommendation
The AOS Service Account (AXAOS) is setup as a user with Sys Admin permissions. This is generally not a best practice unless there are specific reasons for doing so. Review users and verify that **System Administrator** permissions are required. Remove System administrator permissions from the accounts that do not require this complete access permission.

Code Deployment process

The environments that you are using to get customizations developed and to production are development, test, UAT, and production. Your process to move code is to develop the customization in the development environment. The change is then moved to the test environment via an xpo. Once the testing is completed, you move the customizations that passed testing to UAT via an xpo. In the UAT environment, you are then running a full x++ and CIL compile. Code is moved from UAT to production to utilizing a modelstore export. There have also been instances where modelstores are moved from development to test to uat and finally to production.

It is great that you utilize modelstore imports for moving code to production. The change that I would suggest is that you consider implementing change management and then use the model store concept to move code between all environments. The benefit of this is that you reduce the risk significantly of introducing id conflicts. You also ensure that when testing you are testing the full solution that will end up in production and reduce the risk of interdependencies causing invalid testing. The other thing that happens is you do the code builds in the development environment and eliminate the need for the builds in the uat environment.

Change Management process

At this time, there is no formal change management system in place. This is a risk in being able to track down what code is in production, providing documentation on what is released into production, and having users validating and signing off on testing of features and bugs. Without a change management, system it can be hard to manage reported issues and follow them through the life cycle of being reported to being resolved to moving into production. It would be our recommendation that you install Team Foundation Server and start using this to log bugs and feature requests. The developer(s) would then be assigned bugs and/or feature requests. After they are finished coding for a feature they would check in their code on that work item. This would also allow users to go in and pass or fail the work item for a bug or feature so that it is promoted to production or returned for a developer to review.

Microsoft SQL Server

Version Information

Edition	Enterprise
Version	11.0.5343
Level	SP2
Platform	X64
Clustered	No
AlwaysOn	No
Lock Pages in Memory	Yes
Instant File Initialization	Yes

Recommendations

Version
11.0.5343 was released on July 14, 2015. The latest release was on September 21, 2015.
Recommendation
Upgrade to the latest CU8 - http://support.microsoft.com/kb/3082561/en-us

Clustered/AlwaysOn
A high availability and potential disaster recovery solution that allows for failover to a secondary server in the event of a failure on the primary.
Recommendation
Currently there is not a high availability mechanism in place. In the event of a disaster, you would need to stand up a new environment and restore the database and transaction logs. Implement a high availability strategy.

Lock Pages in Memory
A Windows policy that determines which accounts can use a process to keep data in physical memory, preventing the system from paging the data to virtual memory on disk
Recommendation
None, currently set.

Instant File Initialization
Data and log files are initialized to overwrite any existing data left on the disk from previously deleted files. Data and log files are first initialized by filling the files with zeros when you perform one of the following operations: – Create a database. – Add files, log or data, to an existing database. – Increase the size of an existing file (including autogrow operations). – Restore a database or filegroup. SQL Server, data files can be initialized instantaneously. This allows for fast execution of the previously mentioned file operations. Instant file initialization reclaims used disk space without filling that space with zeros. Instead, disk content is overwritten as new data is written to the files. Log files cannot be initialized instantaneously.
Recommendation
None, currently set.

Database Configuration

Database Name	AXPROD
Data File	D:\DataFiles\AXProd.mdf
Data File Size (MB)	4,027 MB

Data File Free space	1,765 MB
Data Auto growth	1,024 MB
Log File	E:\LogFiles\AXPROD_log.ldf
Log File Size (MB)	5,348 MB
Log File Free Space	5,274 MB
Log Auto growth	1,024 MB
VLF Count	52

Recommendations

Data/Log File Free space
The amount of free space within the data file. Free space allows the file to grow without triggering an auto growth event.
Recommendation
Currently the data file appears to be sufficiently sized to allow future growth. The transaction log has ample space to accommodate log growth.

Data Auto growth/Log Auto growth
The size in MB or percentage that the data or log file should grow by in the event of an auto growth event.
Recommendation
File size growths are adequately sized should a file growth occur.

VLF Count
The number of virtual log files within the log file. Large VLF counts (5,000+) can lead to performance issues with backup, restore and recovery.
Recommendation
None, there is no issue with the VLF count.

Microsoft SQL Server Configuration Settings

Name	Configured	Current	Recommended
max degree of parallelism	4	4	No
max server memory (MB)	0	0	No
min server memory (MB)	0	5000	No
optimize for ad hoc workloads	0	0	No

Recommendations

max degree of parallelism
Microsoft Dynamics AX OLTP workloads generally perform better when intra-query parallelism is disabled. However, the upgrade process benefits from parallelism.
Recommendation
Use the following settings to configure the value for a normal operation or an upgrade: 1. During normal operation of the Microsoft Dynamics AX database, **set the max degree of parallelism to 1**. 2. During an upgrade to a new release of Microsoft Dynamics AX, set the max degree of parallelism to the least of: • 8 • The number of physical processor cores • The number of physical processor cores per non-uniform memory access (NUMA) node

max server memory (MB)
This value determines the maximum amount of memory that the SQLServer.exe will be able to use. This setting varies based on the version of SQL Server. SQL Server 2012 and above this setting limits the total memory SQL can consume with some exceptions. SQL Server 2008 R2 and before this setting only limits the buffer pool size.
Recommendation
None, current value is appropriate.

min server memory (MB)
This value determines the minimum amount of memory that SQL will not release back to the OS even if not needed.
Recommendation
Recommended default value of 0 (default)

optimize for ad hoc workloads
This will generate a plan stub from execution plans which helps to reduce plan cache size for ad-hoc queries.
Recommendation
Recommended value of 1 sp_configure 'optimize for ad hoc workloads',1; RECONFIGURE; GO

Drive Configuration

Label	Drive	Size (GB)	Free Space (GB)	% Free	% Used	Cluster Size
OS	C:	1526	54	40.0	60.0	4k
Data	D:	150	121	80.0	20.0	4k
Logs	E:	100	50	50.0	50.0	4k
TempDB	F:	100	85	90.0	10.0	4k

Recommendations

Cluster Size
SQL Server IO Best Practices recommends setting the file allocation unit size to 64k. This aligns with SQL Server page extent structures which are 64 KB in size.
Recommendation
Reformat the drives excluding the C: drive to a 64k cluster size for optimal IO throughput.

TempDB Configuration

FileID	Name	Path	Size (MB)	Growth (MB)
1	tempdev	T:\TempdbFiles\tempdb.mdf	1024	1024
2	templog	T:\TempdbFiles\templog.ldf	1024	1024

Recommendations

Size/Growth/Count
It is recommended to have .25 to 1 data files (per filegroup) for each CPU (excluding hyper-threading). Too many tempdb data files can cause performance problems. If you have a workload that uses query plan operators that require lots of memory (e.g. sorts), the odds are that there won't be enough memory on the server to accommodate the operation and it will spill out to tempdb. If there are too many tempdb data files,

the writing of the temporarily spilled data can be slowed while the allocation system does round-robin allocation. Guidelines: • Greater than 8 cores = # of processors in the system (excluding hyper-threads) • Less than or equal to 8 cores = 8 files • If you are seeing in-memory contention, add 4 files at a time
Recommendation
Create 7 additional tempdb files.

Trace Flags

Existing Trace Flag	Description
None	

Recommendations

Trace Flag	Description
1118	Changes the default mixed page allocations in TempDB to full extent allocations.
4199	Enable all optimizer changes implemented in SP's, CU's and Hotfixes.
1224	Disables lock escalation, lock escalation only occurs under extreme memory pressure.
2371	Changes default auto-update statistics thresholds from default values to sliding row size
1117	Evenly grow all database files

Indexes and Statistics

Index and Statistics > 5 days old

Database	Rows	Mods	% Mod	Table	Type	Index	ID	Last Update

Recommendations

Description
When indexes and statistics are out of date, this can lead to suboptimal query plans.
Recommendation
None, indexes and statistics are up to date.

Missing Indexes

Table	Create Statement
GENERALJOURNALACCOUNTENTRY	CREATE INDEX [missing_index_57270_57269_GENERALJOURNALACCOUNTENTRY] ON [SSIDAX].[dbo].[GENERALJOURNALACCOUNTENTRY] ([GENERALJOURNALENTRY]) INCLUDE ([RECID])
COSTCONTROLTRANSCOMMITTEDCOST	CREATE INDEX [missing_index_60754_60753_COSTCONTROLTRANSCOMMITTEDCOST] ON [SSIDAX].[dbo].[COSTCONTROLTRANSCOMMITTEDCOST] ([OPEN_], [SOURCEDOCUMENTLINE], [PARTITION])

Recommendations

Description
When the query optimizer generates a query plan, it analyzes what are the best indexes for a particular filter condition. If the best indexes do not exist, the query optimizer generates a suboptimal query plan, but still stores information about these indexes.
Recommendation

Do not implement these indexes without testing. The following indexes may provide additional performance. It is HIGHLY recommended to use SQL Server Database Tuning Advisor to fine tune the index recommendations based on work loads

Indexes Missing From AOT

Table Name	Index Name	Description

Recommendations

Description
INDEXES should ALWAYS be defined in the AOT as synchronization will remove them. This also violates Dynamics AX Best Practices to not have an index defined in the AOT. It's ok to add an index for testing purposes on the SQL side as long as you add it to the AOT once you know you are going to keep the index
Recommendation
None. There are no missing indexes from the AOT.

Indexes with RecVersion column

Table Name	Index Name	Index Keys

Recommendations

Description
INDEXES that have **RECVERSION** in the key or included list. **RECVERSION** should NOT be included in Dynamics AX indexes due to the frequency of updates
Recommendation
None, there are no indexes with RECVERSION in the key list.

Duplicate Indexes

Table Name	Index Name	Index Keys

Recommendations

Description
Tables that have 2 or more indexes with the exact same key. The indexes could be duplicated on the keys but have unique set of included columns. In that case they should be combined into a singular index for performance reasons.
Recommendation
None, there are currently no duplicate indexes.

Unused Indexes

Size (MB)	Days	Table Name	Index Name

Recommendations

Description
Unused Indexes incur a performance write penalty. If these indexes are not needed, it would be prudent to drop these indexes. If an index enforces a uniqueness or constraint, it cannot be dropped. **DO NOT DELETE THESE INDEXES UNLESS YOU ARE SURE YOU HAVE RUN EVERY PROCESS IN YOUR DYNAMICS DATABASE INCLUDING YEAR END.**
Recommendation
None, there are currently no unused index.

Database File Settings

Name	Snapshot Isolation	RCSI	Recovery Mode	Page Verify	Auto Close	Auto Shrink	Auto Create Stats	Auto Update Stats	Auto Update Async
master	ON	OFF	SIMPLE	CHECKSUM	OFF	OFF	ON	ON	OFF
tempdb	OFF	OFF	SIMPLE	CHECKSUM	OFF	OFF	ON	ON	OFF
model	OFF	OFF	SIMPLE	CHECKSUM	OFF	OFF	ON	ON	OFF
msdb	ON	OFF	SIMPLE	CHECKSUM	OFF	OFF	ON	ON	OFF
AXProd	OFF	ON	FULL	Page_Verify	OFF	OFF	ON	ON	OFF
AXProd_model	OFF	ON	SIMPLE	CHECKSUM	OFF	OFF	ON	ON	OFF

Recommendations

Page Verify
Checksum creates a checksum value using the content of the entire page, and saves that value in the header. When a page is read from disk, a checksum is created again and compared to the saved checksum. Torn page detection is a 2 bit marker on the database page. It is inaccurate as it can miss corruption and page changes outside of the bit markers.
Recommendation
Change Page verify on AXProd_model to CHECKSUM **ALTER DATABASE AX SET PAGE_VERIFY CHECKSUM** **GO** Re-index the database to have checksums calculated on all pages. The checksum is only calculated on write, so enabling this option only affects newly written pages. Performing an index rebuild will resolve this.

Database Maintenance Plans

Job Name	Schedule	Frequency
Cleanup Tasks.Delete Expired Backups	12 AM	Daily
Daily Database Backups.Backup Databases	11 PM	Daily
Daily System Database Backups.Backup System Databases	10 PM	Daily
Daily Transaction Log Backups.Daily Transaction Log Backups	6 PM	Daily
Index and Stats Rebuild.Rebuild Indexes and Update Statistics	12 AM	Weekly - Sunday
SSIDAX Hourly TLog Backups.Backup Transaction Log	12 – 12 AM	Hourly

Recommendations

Job Definitions

Recommendations
None.

Blocking Occurrences > 5 Seconds

Start Date	End Date	Count
No blocking Detected		

Top 5 Tables by Wait Times

No blocking Detected

Top 10 Waited on Resources

Total Wait Time (ms)	Resource	Blocks	Table	Index
No blocking Detected				

Top 5 Blocking Statements by Wait Time

Wait Time (s)	Resource	Blocked Statement	Blocking Statement

No blocking Detected

Recommendations

Blocking
Blocking is a state when a transaction requires a lock on a resource on which an incompatible lock is already placed by another transaction. Blocking can decrease transactional throughput and increase wait times. There is a heavy amount of prolonged blocking taking place. Blocking is a natural process in SQL Server, however, there appears to be extended wait periods of blocking with heavy contention on certain resources (see the top 5 resources above). Review the statements and attempt to optimize them.
Recommendations
None, no blocking detected.

Deadlocking Occurrences

Start Date	End Date	Count
None, no deadlocks detected.		

Query Deadlocks from SystemHealth

Query
SELECT xed.value('@timestamp', 'datetime') as [Deadlock Time], xed.query('.') AS [Deadlock Report] FROM (SELECT CAST([target_data] AS XML) AS Target_Data FROM sys.dm_xe_session_targets AS xt INNER JOIN sys.dm_xe_sessions AS xs ON

```
                xs.address = xt.event_session_address
        WHERE
                xs.name = N'system_health' AND xt.target_name = N'ring_buffer') AS XML_Data
CROSS APPLY
        Target_Data.nodes('RingBufferTarget/event[@name="xml_deadlock_report"]') AS XEventData(xed)
ORDER BY
        [Deadlock Time] DESC
```

Recommendations

Deadlocking
There is frequent deadlocking occurring in the system which impacts transactional performance. Deadlock are a blocking situation where two transactions are holding locks and waiting to acquire lock on the resource held by the other transaction. Deadlocks are detected and handled automatically by the SQL Server Database Engine. Once detected, the Database Engine chooses one of the transactions as the deadlock victim and lets the other transaction complete its process. The victim process is rolled back and the application must resubmit the transaction again or abort.
Recommendations
None, no deadlocks detected.

Database Error Logs

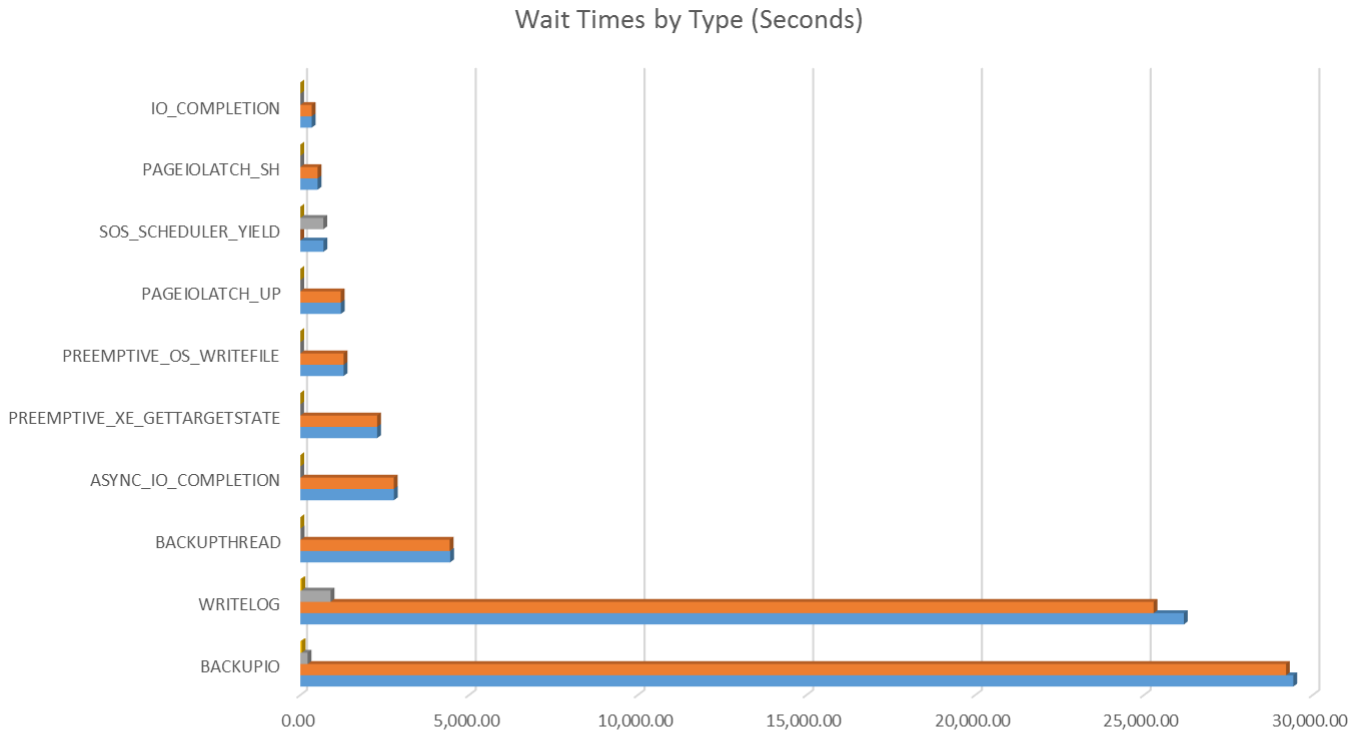
Event Date
No errors reported
Event Description
Notes
Recommendation

Reporting Services Error Logs

Event Date
No errors reported
Event Description
Notes
Recommendation

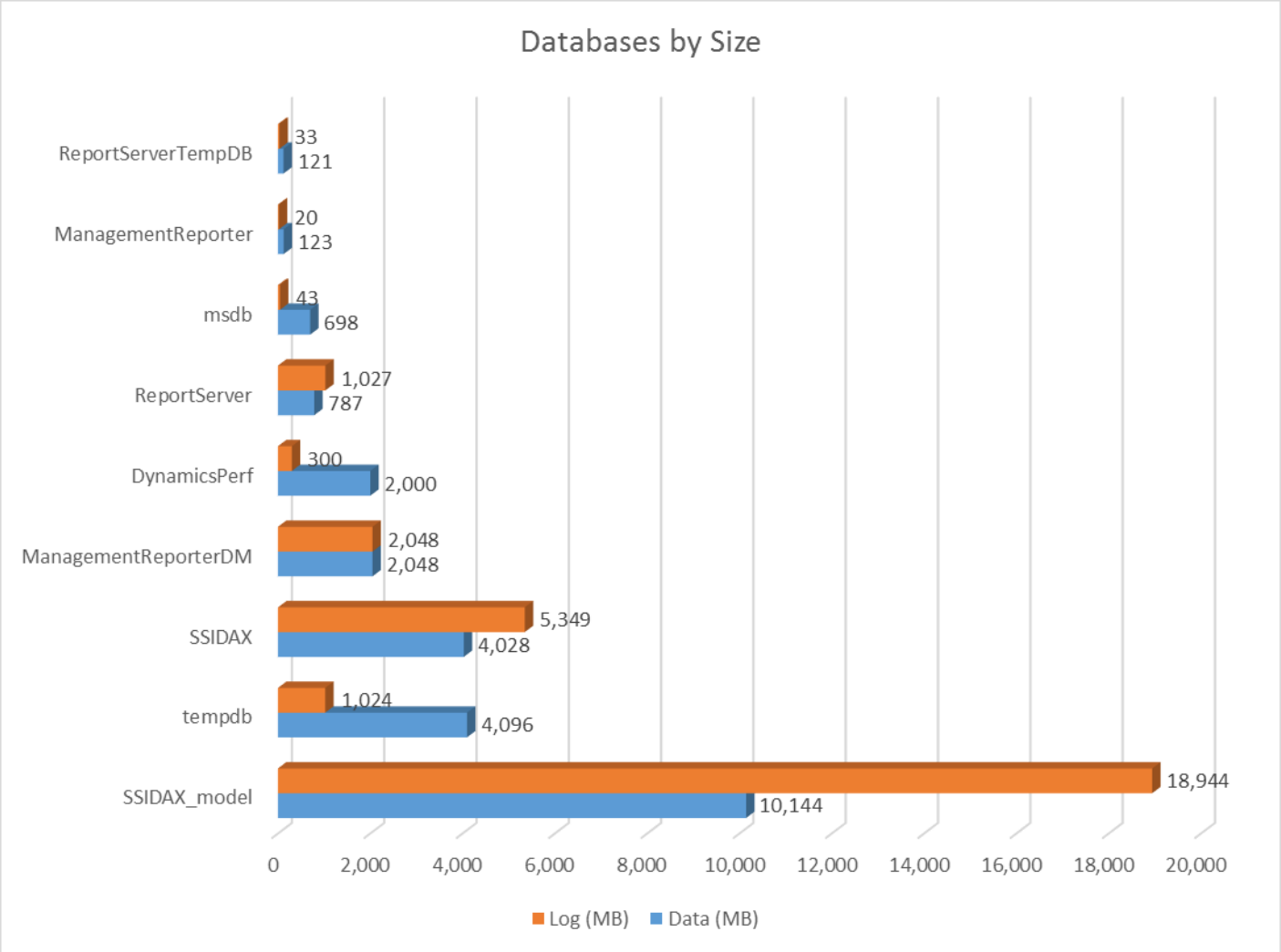
Performance Charts

Wait Statistics (Since Last Start)

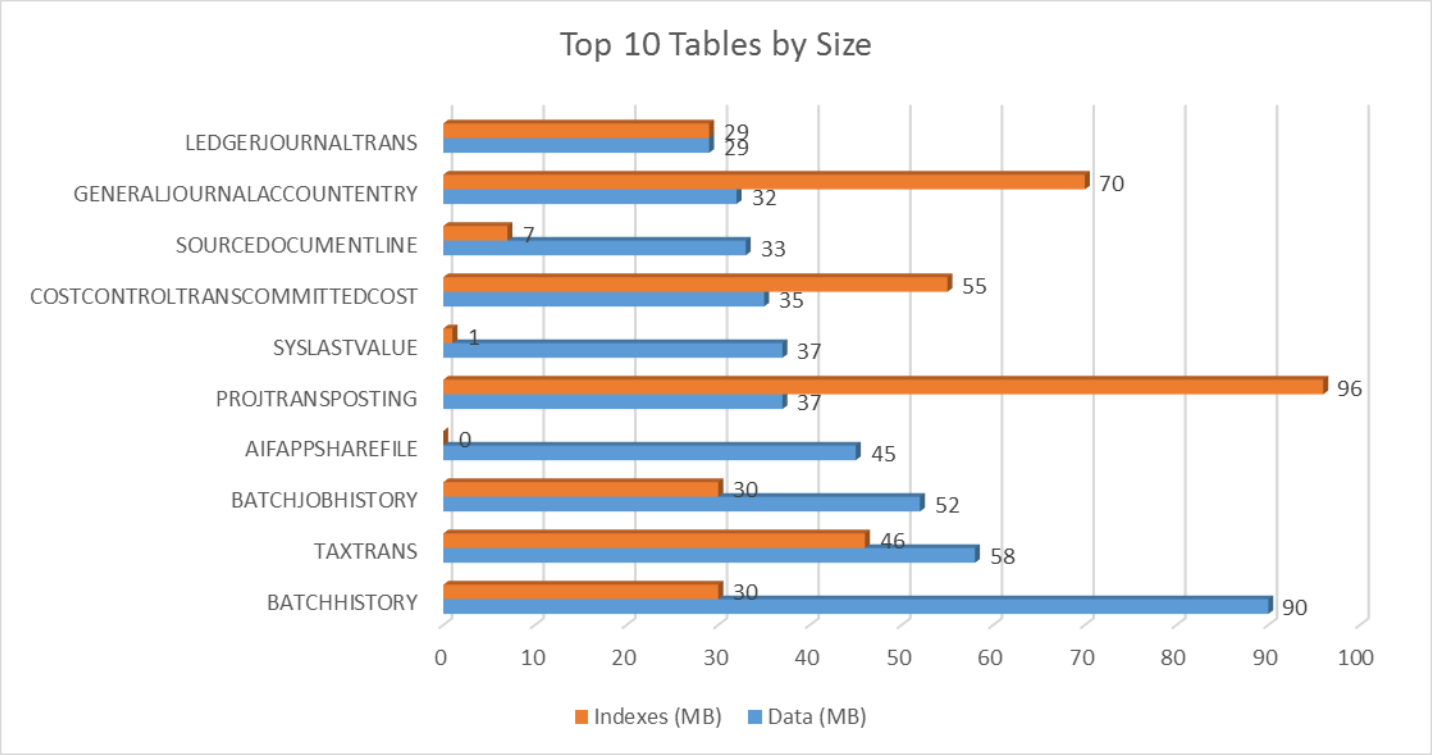


	BACKUPIO	WRITELOG	BACKUPTHREAD	ASYNC_IO_COMPLETION	PREEMPTIVE_XE_GETTARGETSTATE	PREEMPTIVE_OS_WRITEFILE	PAGEIOLATCH_UP	SOS_SCHEDULER_YIELD	PAGEIOLATCH_SH	IO_COMPLETION
Percentage	41.21	36.67	6.22	3.88	3.19	1.80	1.68	0.96	0.72	0.48
Signal_S	217.66	897.62	14.51	2.28	0.00	0.00	1.34	684.45	0.81	0.13
Resource_S	29,210.22	25,286.53	4,428.00	2,771.63	2,277.75	1,283.12	1,200.43	2.65	512.58	339.18
Wait_S	29,427.88	26,184.15	4,442.50	2,773.91	2,277.75	1,283.12	1,201.77	687.09	513.39	339.30

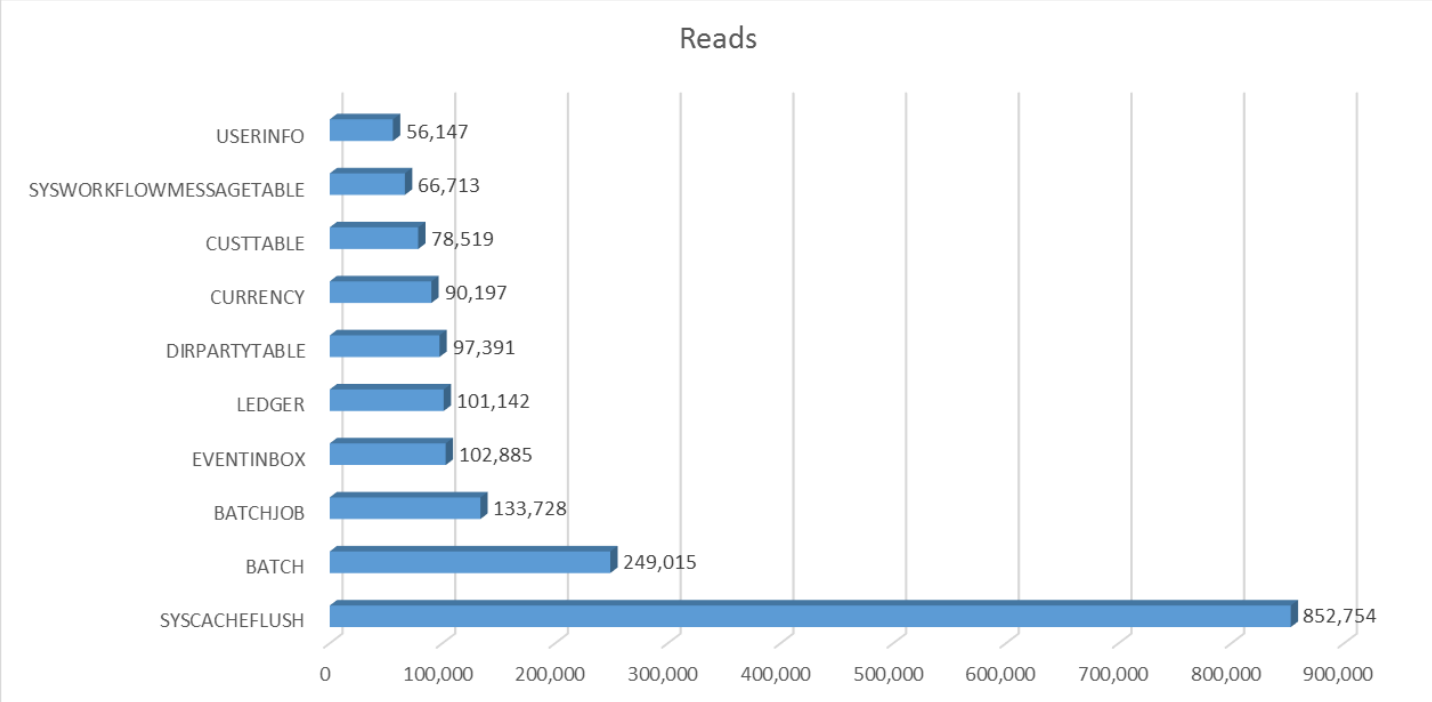
Databases by Size

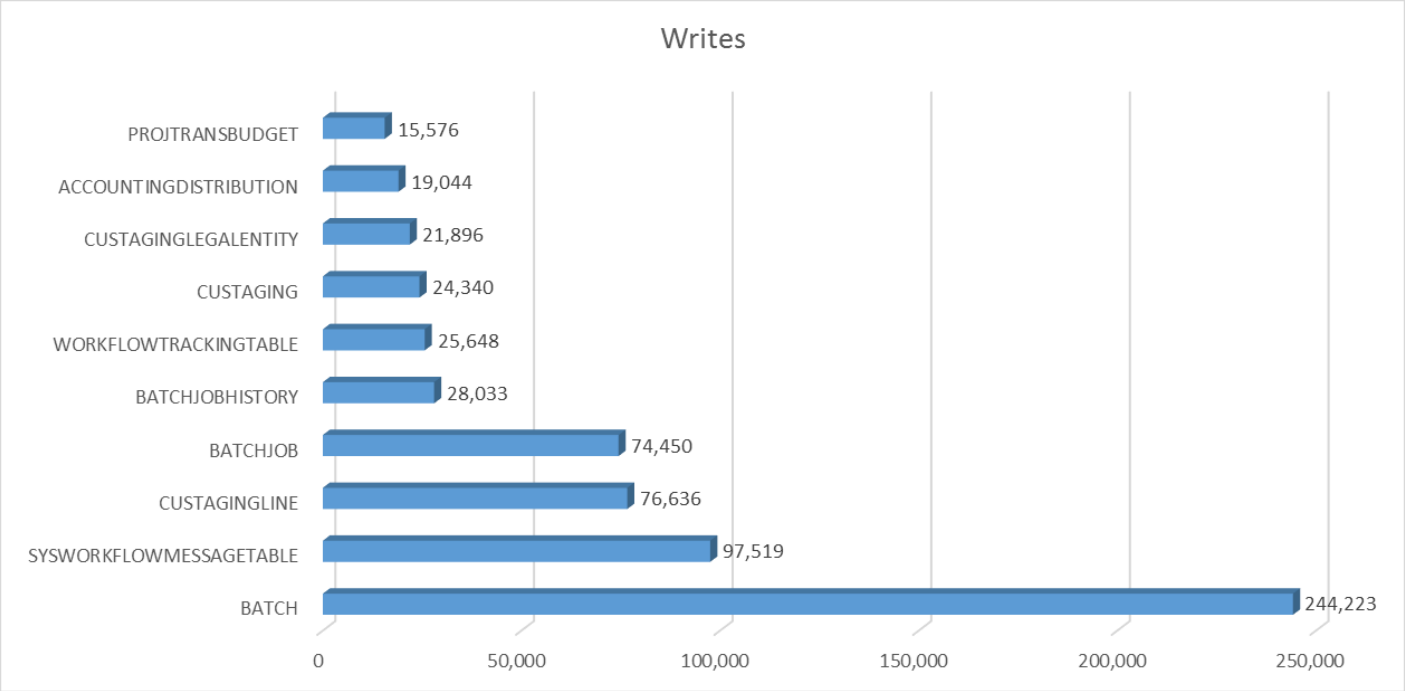


Top 10 Tables by Size

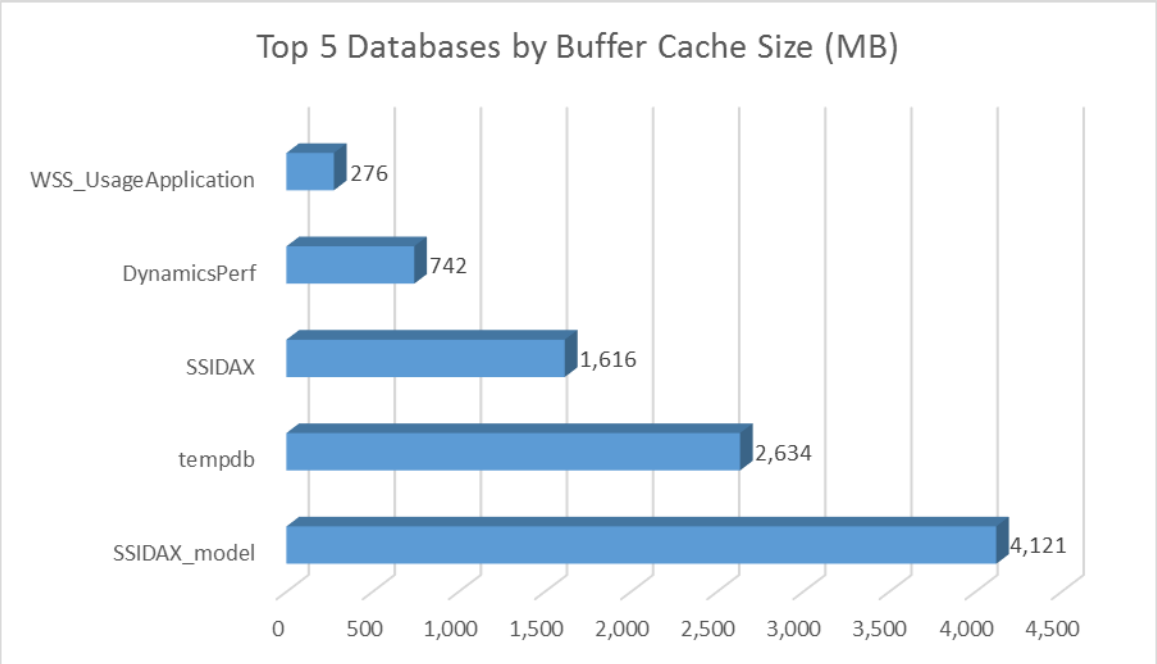


Top 10 Tables by Activity

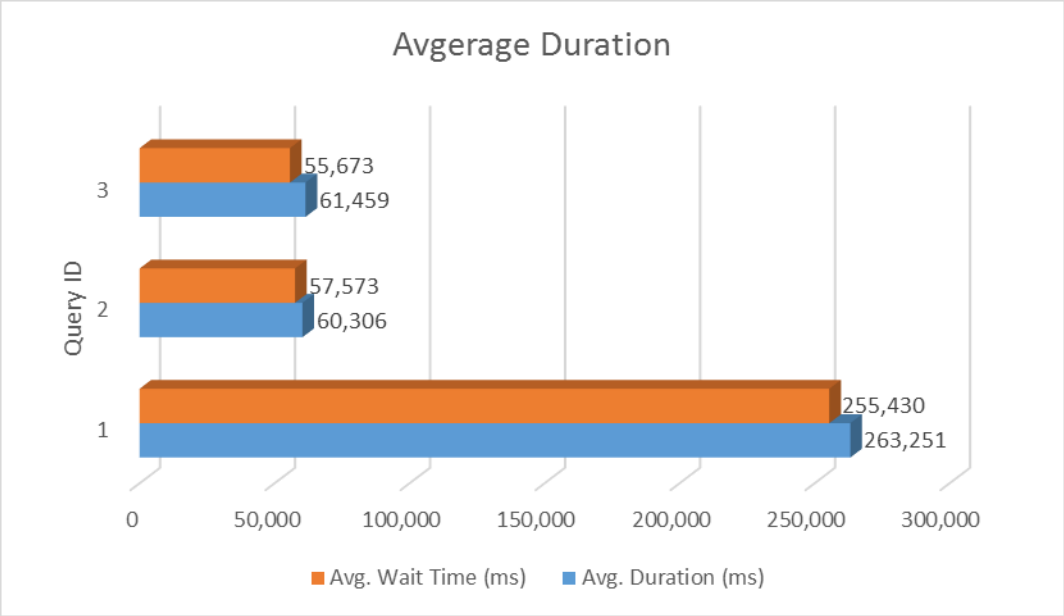




Top 5 Databases by Buffer Cache Size



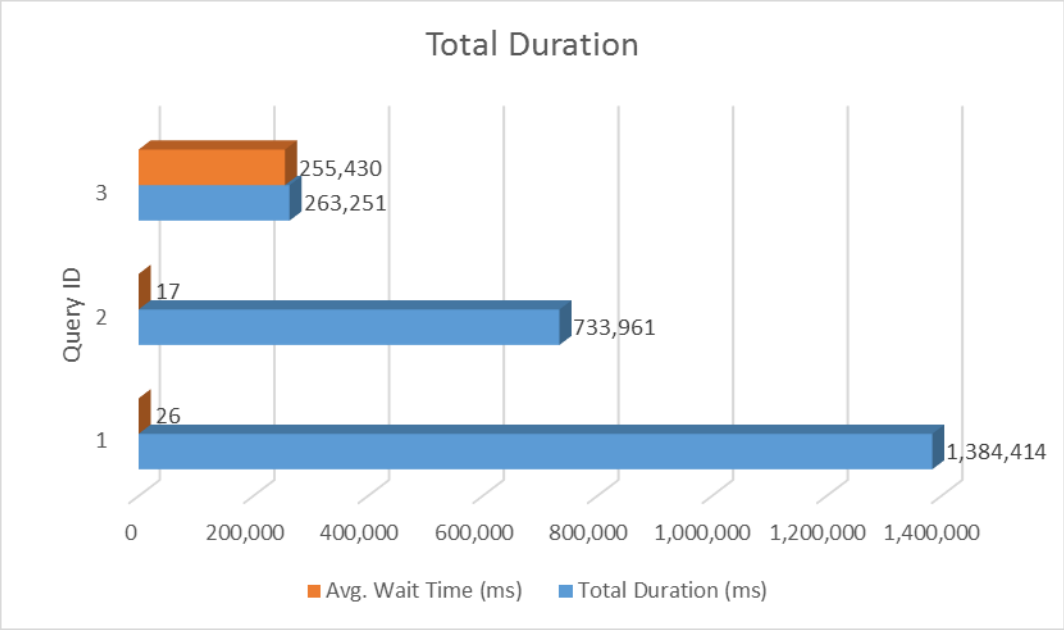
Top 3 Most Expense Queries by Avg. Duration



ID	Query Text
1	DELETE FROM SYSTRACETABLESQL WHERE ((PARTITION=@P1) AND (DATAAREAID=@P2))
2	SELECT T1.RECID,T1.AOTNAME,T2.SECURITYROLE,T2.SECURITYSUBROLE,T2.RECID,T3.SECURITYROLE,T3.SECURITYTASK,T3.RECID,T4.SECURITYTASK,T4.SECURITYSUBTASK,T4.RECID,T5.SECURITYTASK,T5.ENTRYPOINT,T5.RECID,T6.NAME,T6.ENTRYPOINTTYPE,T6.USERLICTYPE,T6.RECID FROM [SSIDAX_model].[dbo].SECURITYROLE T1 CROSS JOIN [SSIDAX_model].[dbo].SECURITYROLEEXPLODEDGRAPH T2 CROSS JOIN [SSIDAX_model].[dbo].SECURITYROLETASKGRANT T3 CROSS JOIN [SSIDAX_model].[dbo].SECURITYTASKEXPLODEDGRAPH T4 CROSS JOIN [SSIDAX_model].[dbo].SECURITYTASKENTRYPOINT T5 CROSS JOIN tempdb."DBO".t101349_A1EFF88624C64E97A82D3C9A4D78E3BB T6 WHERE (T2.SECURITYROLE=T1.RECID) AND (T3.SECURITYROLE=T2.SECURITYSUBROLE) AND (T4.SECURITYTASK=T3.SECURITYTASK) AND (T4.SECURITYSUBTASK=T5.SECURITYTASK) AND (((T6.PARTITION=@P1) AND (T6.DATAAREAID=@P2)) AND (((T5.ENTRYPOINT=T6.SECURABLEOBJECT) AND (T6.USERLICTYPE<>@P3)) AND (((T6.ACCESSTYPE=@P4) AND (T5.PERMISSIONGROUP=@P5)) OR ((T6.ACCESSTYPE=@P6) AND (T5.PERMISSIONGROUP>@P7)))))
3	SELECT T1.UTILLEVEL,T1.RECORDTYPE,T1.PARENTID,T1.NAME,T1.BASEVERSION,T1.VERSION,T1.SAVECOUNT,T1.ID,T1.MODIFIEDDATETIME,T1.DEL_MODIFIEDTIME,T1.MODIFIEDBY,T1.CREATEDDATETIME,T1.DEL_CREATEDTIME,T1.CREATEDBY,T1.RECVERSION,T1.RECID,T1.CODE FROM [SSIDAX_model].[dbo].UTILIDELEMENTS T1 WHERE (((((RECORDTYPE=@P1) OR (RECORDTYPE=@P2)) OR (RECORDTYPE=@P3)) OR (RECORDTYPE=@P4)) OR (RECORDTYPE=@P5)) ORDER BY T1.RECORDTYPE DESC,T1.ID DESC,T1.NAME DESC,T1.UTILLEVEL DESC OPTION(FAST 20)

ID	Rows	Count	Total Dur. (ms)	Avg. Dur. (ms)	Total Logical Reads	Avg. Logical Reads	Total Physical Reads	Avg. Physical Reads	Total Writes	Avg. Logical Writes
1	1	263,251	263,251	1,356,311	1,356,311	62,089	62,089	152,050	152,050	1
2	1	60,306	60,306	530,699	530,699	10,282	10,282	2,902	2,902	1
3	1	61,459	61,459	3,253,188	3,253,188	50,181	50,181	5,661	5,661	1

Top 3 Most Expense Queries by Total Duration

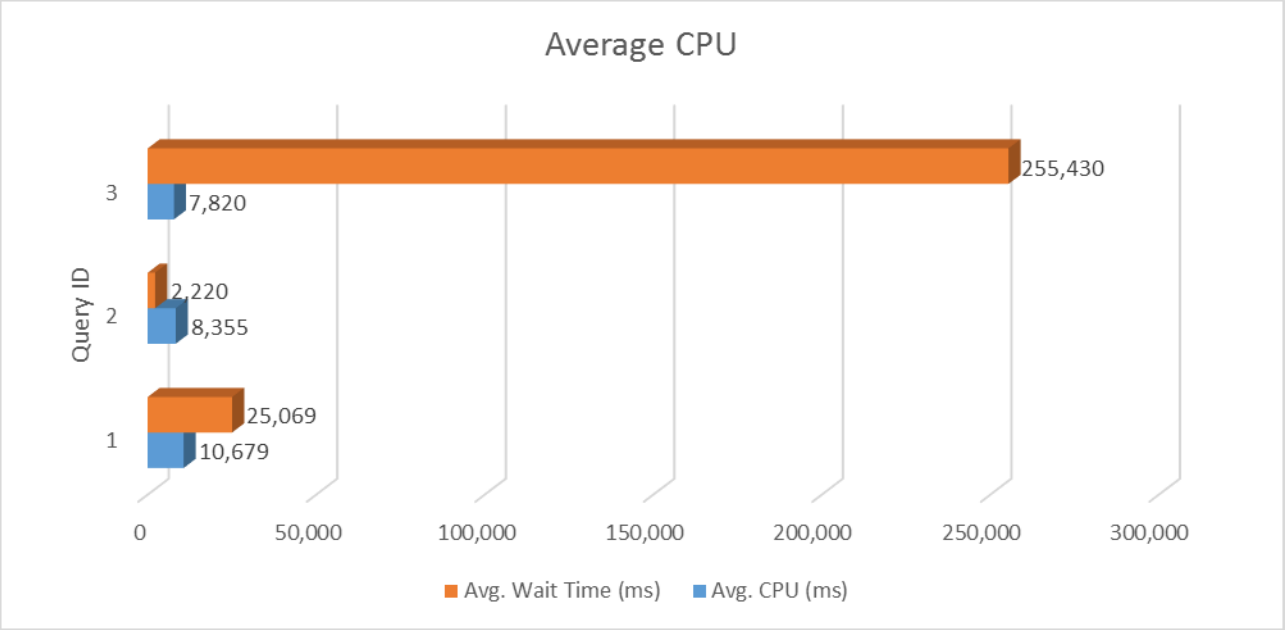


ID	Query Text
1	DELETE FROM SYSTRACETABLESQL WHERE ((PARTITION=@P1) AND (DATAAREAID=@P2))
2	SELECT T1.RECID,T1.AOTNAME,T2.SECURITYROLE,T2.SECURITYSUBROLE,T2.RECID,T3.SECURITYROLE,T3.SECURITYTASK,T3.RECID,T4.SECURITYTASK,T4.SECURITYSUBTASK,T4.RECID,T5.SECURITYTASK,T5.ENTRYPOINT,T5.RECID,T6.NAME,T6.ENTRYPOINTTYPE,T6.USERLICTYPE,T6.RECID FROM [SSIDAX_model].[dbo].SECURITYROLE T1 CROSS JOIN [SSIDAX_model].[dbo].SECURITYROLEEXPLODEDGRAPH T2 CROSS JOIN [SSIDAX_model].[dbo].SECURITYROLETASKGRANT T3 CROSS JOIN [SSIDAX_model].[dbo].SECURITYTASKEXPLODEDGRAPH T4 CROSS JOIN [SSIDAX_model].[dbo].SECURITYTASKENTRYPOINT T5 CROSS JOIN tempdb."DBO".t101349_A1EFF88624C64E97A82D3C9A4D78E3BB T6 WHERE (T2.SECURITYROLE=T1.RECID) AND (T3.SECURITYROLE=T2.SECURITYSUBROLE) AND (T4.SECURITYTASK=T3.SECURITYTASK) AND (T4.SECURITYSUBTASK=T5.SECURITYTASK) AND (((T6.PARTITION=@P1) AND (T6.DATAAREAID=@P2)) AND (((T5.ENTRYPOINT=T6.SECURABLEOBJECT) AND (T6.USERLICTYPE<>@P3)) AND (((T6.ACCESSTYPE=@P4) AND (T5.PERMISSIONGROUP=@P5)) OR ((T6.ACCESSTYPE=@P6) AND (T5.PERMISSIONGROUP>@P7)))))
3	SELECT T1.UTILLEVEL,T1.RECORDTYPE,T1.PARENTID,T1.NAME,T1.BASEVERSION,T1.VERSION,T1.SAVECOUNT,T1.ID,T1.MODIFIEDDATETIME,T1.DEL_MODIFIEDTIME,T1.MODIFIEDBY,T1.CREATEDDATETIME,T1.DEL_CREATEDTIME,T1.CREATEDBY,T1.RECVERSION,T1.RECID,T1.CODE FROM [SSIDAX_model].[dbo].UTILIDELEMENTS T1 WHERE (((((RECORDTYPE=@P1) OR

	(RECORDTYPE=@P2)) OR (RECORDTYPE=@P3)) OR (RECORDTYPE=@P4)) OR (RECORDTYPE=@P5)) ORDER BY T1.RECORDTYPE DESC,T1.ID DESC,T1.NAME DESC,T1.UTILLEVEL DESC OPTION(FAST 20)

ID	Rows	Count	Total Dur. (ms)	Avg. Dur. (ms)	Total Logical Reads	Avg. Logical Reads	Total Physical Reads	Avg. Physical Reads	Total Writes	Avg. Logical Writes
1	48,177,640	7,481	1,384,414	185	17,176,384	2,296	143	0	0	0
2	6,440	6,440	733,961	114	134,853,881	20,940	472	0	0	0
3	287,686	1	263,251	263,251	1,356,311	1,356,311	62,089	62,089	152,050	152,050

Top 3 Most Expense Queries by Avg. CPU

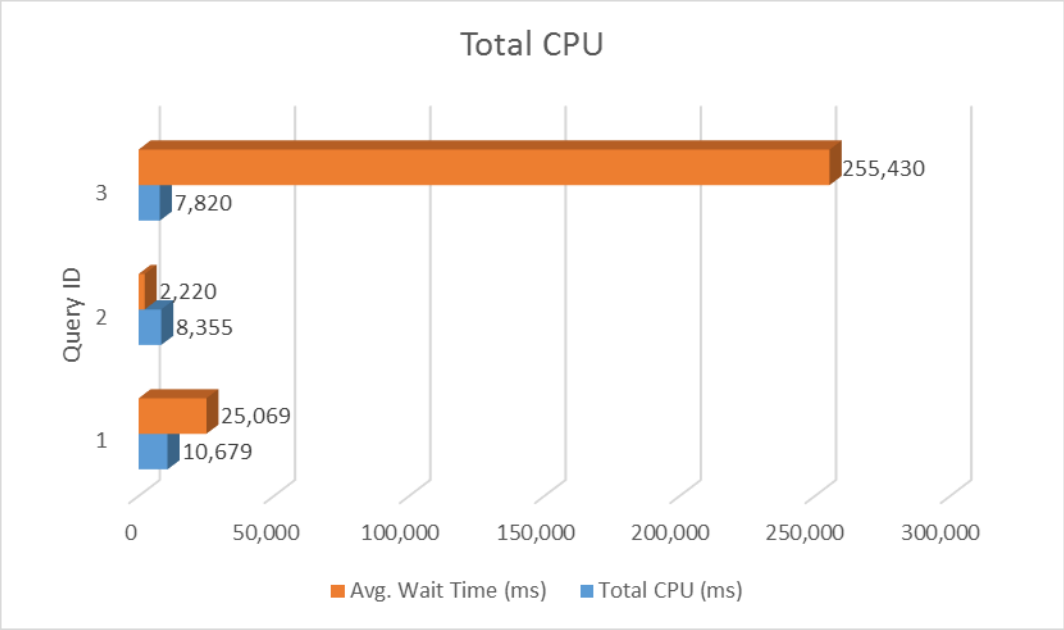


ID	Query Text
1	SELECT T1.RESOURCE_ AS f1,T1.RESOURCEDATAAREA AS f2,T1.RESOURCECALENDAR AS f3,T1.EMPLOYMENTSTART AS f4,T1.EMPLOYMENTEND AS f5,SUM(T2.HOURCAPACITY) AS f6,T2.WORKTIMECONTROL AS f7,T2.TRANSDATE AS f8,T2.TRANSDATE AS f9,T3.USERSESSION AS f10,1 AS RECVERSION,5637144576 AS PARTITION,IDENTITY(bigint,1,1) AS RECID INTO [##ax_tmp_dat15_71_102760] FROM PSARESOURCETABLE T1 CROSS JOIN WORKCALENDARDATELINECAPACITYVIEW T2 CROSS JOIN PSARESOURCESET T3 WHERE ((T1.PARTITION=@P1) AND ((((T1.USERSESSION=@P2) AND (T1.RESOURCESET=@P3)) AND (T1.RESOURCEDATAAREA=@P4)) AND (T1.RESOURCECALENDAR=@P5))) AND (((((T2.PARTITION=@P6) AND (T2.DATAAREAID=@P7)) AND ((T2.PARTITION#2=@P8) OR (T2.PARTITION#2 IS NULL))) AND ((T2.DATAAREAID#2=@P9) OR (T2.DATAAREAID#2 IS NULL))) AND ((((T2.CALENDARID=@P10) AND (T2.WORKTIMECONTROL=@P11)) AND (T2.TRANSDATE>=@P12)) AND (T2.TRANSDATE<=@P13))) AND ((T3.PARTITION=@P14) AND ((T3.USERSESSION=@P15) AND (T3.RESOURCESET=@P16))) AND EXISTS (SELECT 'x' FROM WORKCALENDARDATELINE T4 WHERE (((T4.PARTITION=@P17) AND (T4.DATAAREAID=@P18)) AND (((T4.DATAAREAID=@P19) AND (T4.CALENDARID=@P20)) AND (T4.TRANSDATE=T2.TRANSDATE))) AND NOT (EXISTS (SELECT 'x' FROM PSARESOURCECAPACITY T5 WHERE ((T5.PARTITION=@P21) AND ((((T5.USERSESSION=@P22) AND (T5.RESOURCE_=T1.RESOURCE_)) AND (T5.RESOURCEDATAAREA=T1.RESOURCEDATAAREA)) AND (T5.STARTDATE=T2.TRANSDATE)))))) GROUP BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DA

	TAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION ORDER BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION
2	SELECT T1.RESOURCE_ AS f1,T1.RESOURCEDATAAREA AS f2,T1.RESOURCECALENDAR AS f3,T1.EMPLOYMENTSTART AS f4,T1.EMPLOYMENTEND AS f5,SUM(T2.HOURCAPACITY) AS f6,T2.WORKTIMECONTROL AS f7,T2.TRANSDATE AS f8,T2.TRANSDATE AS f9,T3.USERSESSION AS f10,1 AS RECVERSION,5637144576 AS PARTITION,IDENTITY(bigint,1,1) AS RECID INTO [##ax_tmp_dat15_233_102760] FROM PSARESOURCETABLE T1 CROSS JOIN WORKCALENDARDATELINECAPACITYVIEW T2 CROSS JOIN PSARESOURCESET T3 WHERE ((T1.PARTITION=@P1) AND ((((T1.USERSESSION=@P2) AND (T1.RESOURCESET=@P3)) AND (T1.RESOURCEDATAAREA=@P4)) AND (T1.RESOURCECALENDAR=@P5))) AND (((((T2.PARTITION=@P6) AND (T2.DATAAREAID=@P7)) AND ((T2.PARTITION#2=@P8) OR (T2.PARTITION#2 IS NULL))) AND ((T2.DATAAREAID#2=@P9) OR (T2.DATAAREAID#2 IS NULL))) AND ((((T2.CALENDARID=@P10) AND (T2.WORKTIMECONTROL=@P11)) AND (T2.TRANSDATE>=@P12)) AND (T2.TRANSDATE<=@P13))) AND ((T3.PARTITION=@P14) AND ((T3.USERSESSION=@P15) AND (T3.RESOURCESET=@P16))) AND EXISTS (SELECT 'x' FROM WORKCALENDARDATELINE T4 WHERE (((T4.PARTITION=@P17) AND (T4.DATAAREAID=@P18)) AND (((T4.DATAAREAID=@P19) AND (T4.CALENDARID=@P20)) AND (T4.TRANSDATE=T2.TRANSDATE))) AND NOT (EXISTS (SELECT 'x' FROM PSARESOURCECAPACITY T5 WHERE ((T5.PARTITION=@P21) AND ((((T5.USERSESSION=@P22) AND (T5.RESOURCE_=T1.RESOURCE_)) AND (T5.RESOURCEDATAAREA=T1.RESOURCEDATAAREA)) AND (T5.STARTDATE=T2.TRANSDATE)))))) GROUP BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION ORDER BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION
3	DELETE FROM SYSTRACETABLESQL WHERE ((PARTITION=@P1) AND (DATAAREAID=@P2))

ID	Rows	Count	Total Dur. (ms)	Avg. Dur. (ms)	Total Logical Reads	Avg. Logical Reads	Total Physical Reads	Avg. Physical Reads	Total Writes	Avg. Logical Writes	Total CPU (ms)	Avg. CPU (ms)
1	8,987	1	35,748	35,748	1,161,193	1,161,193	642	642	150	150	10,679	10,679
2	1,045	1	10,575	10,575	771,180	771,180	32	32	9	9	8,355	8,355
3	287,686	1	263,251	263,251	1,356,311	1,356,311	62,089	62,089	152,050	152,050	7,820	7,820

Top 3 Most Expensive Queries by Total CPU



ID	Query Text
1	SELECT T1.RESOURCE_ AS f1,T1.RESOURCEDATAAREA AS f2,T1.RESOURCECALENDAR AS f3,T1.EMPLOYMENTSTART AS f4,T1.EMPLOYMENTEND AS f5,SUM(T2.HOURCAPACITY) AS f6,T2.WORKTIMECONTROL AS f7,T2.TRANSDATE AS f8,T2.TRANSDATE AS f9,T3.USERSESSION AS f10,1 AS RECVERSION,5637144576 AS PARTITION,IDENTITY(bigint,1,1) AS RECID INTO [##ax_tmp_dat15_71_102760] FROM PSARESOURCETABLE T1 CROSS JOIN WORKCALENDARDATELINECAPACITYVIEW T2 CROSS JOIN PSARESOURCESET T3 WHERE ((T1.PARTITION=@P1) AND ((((T1.USERSESSION=@P2) AND (T1.RESOURCESET=@P3)) AND (T1.RESOURCEDATAAREA=@P4)) AND (T1.RESOURCECALENDAR=@P5))) AND (((((T2.PARTITION=@P6) AND (T2.DATAAREAID=@P7)) AND ((T2.PARTITION#2=@P8) OR (T2.PARTITION#2 IS NULL))) AND ((T2.DATAAREAID#2=@P9) OR (T2.DATAAREAID#2 IS NULL))) AND ((((T2.CALENDARID=@P10) AND (T2.WORKTIMECONTROL=@P11)) AND (T2.TRANSDATE>=@P12)) AND (T2.TRANSDATE<=@P13))) AND ((T3.PARTITION=@P14) AND ((T3.USERSESSION=@P15) AND (T3.RESOURCESET=@P16))) AND EXISTS (SELECT 'x' FROM WORKCALENDARDATELINE T4 WHERE (((T4.PARTITION=@P17) AND (T4.DATAAREAID=@P18)) AND (((T4.DATAAREAID=@P19) AND (T4.CALENDARID=@P20)) AND (T4.TRANSDATE=T2.TRANSDATE))) AND NOT (EXISTS (SELECT 'x' FROM PSARESOURCECAPACITY T5 WHERE ((T5.PARTITION=@P21) AND ((((T5.USERSESSION=@P22) AND (T5.RESOURCE_=T1.RESOURCE_)) AND (T5.RESOURCEDATAAREA=T1.RESOURCEDATAAREA)) AND (T5.STARTDATE=T2.TRANSDATE)))))) GROUP BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION ORDER BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION
2	SELECT T1.RESOURCE_ AS f1,T1.RESOURCEDATAAREA AS f2,T1.RESOURCECALENDAR AS f3,T1.EMPLOYMENTSTART AS f4,T1.EMPLOYMENTEND AS f5,SUM(T2.HOURCAPACITY) AS f6,T2.WORKTIMECONTROL AS f7,T2.TRANSDATE AS f8,T2.TRANSDATE AS f9,T3.USERSESSION AS f10,1 AS RECVERSION,5637144576 AS PARTITION,IDENTITY(bigint,1,1) AS RECID INTO [##ax_tmp_dat15_233_102760] FROM PSARESOURCETABLE T1 CROSS JOIN WORKCALENDARDATELINECAPACITYVIEW T2 CROSS JOIN PSARESOURCESET T3 WHERE ((T1.PARTITION=@P1) AND ((((T1.USERSESSION=@P2) AND (T1.RESOURCESET=@P3)) AND (T1.RESOURCEDATAAREA=@P4)) AND (T1.RESOURCECALENDAR=@P5))) AND (((((T2.PARTITION=@P6) AND (T2.DATAAREAID=@P7)) AND ((T2.PARTITION#2=@P8) OR (T2.PARTITION#2 IS NULL))) AND ((T2.DATAAREAID#2=@P9) OR (T2.DATAAREAID#2 IS NULL)))

	AND ((((T2.CALENDARID=@P10) AND (T2.WORKTIMECONTROL=@P11)) AND (T2.TRANSDATE>=@P12)) AND (T2.TRANSDATE<=@P13))) AND ((T3.PARTITION=@P14) AND ((T3.USERSESSION=@P15) AND (T3.RESOURCESET=@P16))) AND EXISTS (SELECT 'x' FROM WORKCALENDARDATELINE T4 WHERE (((T4.PARTITION=@P17) AND (T4.DATAAREAID=@P18)) AND (((T4.DATAAREAID=@P19) AND (T4.CALENDARID=@P20)) AND (T4.TRANSDATE=T2.TRANSDATE))) AND NOT (EXISTS (SELECT 'x' FROM PSARESOURCECAPACITY T5 WHERE ((T5.PARTITION=@P21) AND ((((T5.USERSESSION=@P22) AND (T5.RESOURCE_=T1.RESOURCE_)) AND (T5.RESOURCEDATAAREA=T1.RESOURCEDATAAREA)) AND (T5.STARTDATE=T2.TRANSDATE)))))) GROUP BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION ORDER BY T1.RESOURCE_,T1.RESOURCEDATAAREA,T1.RESOURCECALENDAR,T1.EMPLOYMENTSTART,T1.EMPLOYMENTEND,T2.DATAAREAID,T2.CALENDARID,T2.TRANSDATE,T2.WORKTIMECONTROL,T3.USERSESSION
3	DELETE FROM SYSTRACETABLESQL WHERE ((PARTITION=@P1) AND (DATAAREAID=@P2))

ID	Rows	Count	Total Dur. (ms)	Avg. Dur. (ms)	Total Logical Reads	Avg. Logical Reads	Total Physical Reads	Avg. Physical Reads	Total Writes	Avg. Logical Writes	Total CPU (ms)	Avg. CPU (ms)
1	8,987	1	35,748	35,748	1,161,193	1,161,193	642	642	150	150	10,679	10,679
2	1,045	1	10,575	10,575	771,180	771,180	32	32	9	9	8,355	8,355
3	287,686	1	263,251	263,251	1,356,311	1,356,311	62,089	62,089	152,050	152,050	7,820	7,820

AXPROD - SQL

Server Overview

Server Information	Value
Virtual	Yes
OS Name	Microsoft Windows Server 2012 R2 Standard
Processor	Intel® Xeon® CPU E5620 @ 2.40 GHz 4 cores, 4 logical
BIOS Version/Date	7/2/2013
Installed Physical Memory	32 GB
Total Virtual Memory	36.7 GB
Page File Space	4.75 GB
Page File Location	C:\
Page File Setting	Automatically Managed
Power Settings	High Performance
Process Setting	Background

Recommendations

BIOS Version
The BIOS version is over 2 years out of date.
Recommendation
Check for more recent BIOS update and determine if it resolves any existing issues or contains performance enhancements.

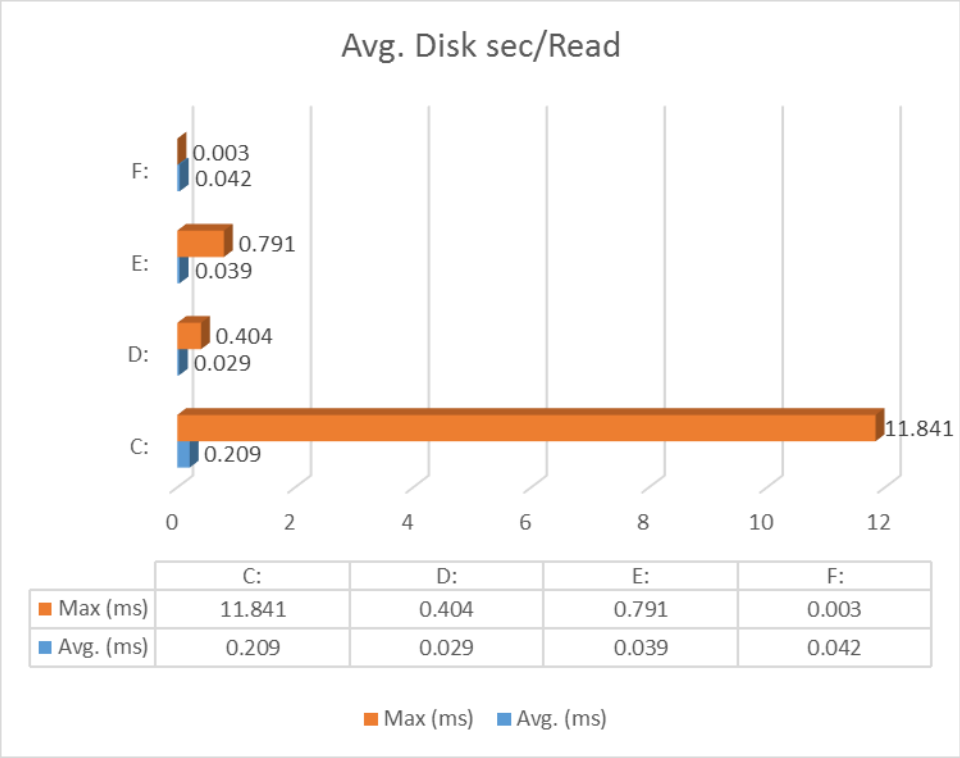
Page File Location/Setting
The page file is utilized by Windows to move pages of memory out of memory. These memory pages are stored in the page file.

Recommendation
For optimal performance consider moving the page file from the C: (OS) drive to a different drive. Also pre-size the page file to prevent growing and contracting of the file by the OS. This will increase page file performance if utilized by the system.

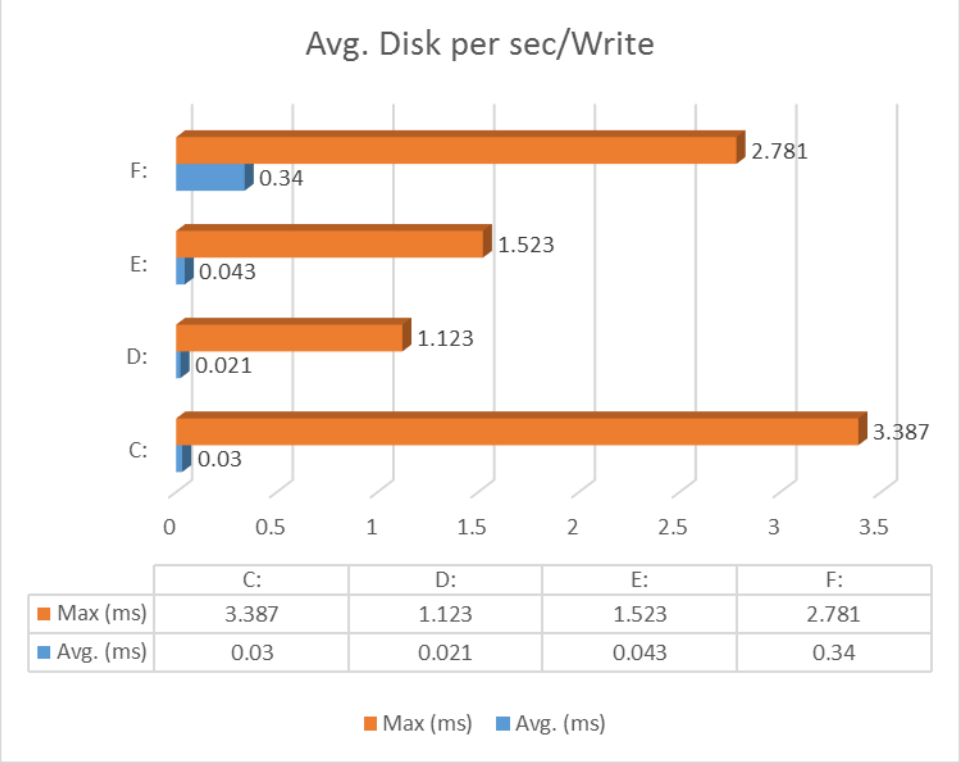
Power Settings
A power plan is a collection of hardware and system settings that manages how your computer uses power. This affects how your server performs as CPU parking and throttling can occur to reduce power consumption.
Recommendation
The **High Performance** power plan is recommended to ensure maximum server performance on the system and prevent CPU parking or throttling.

Performance Counters

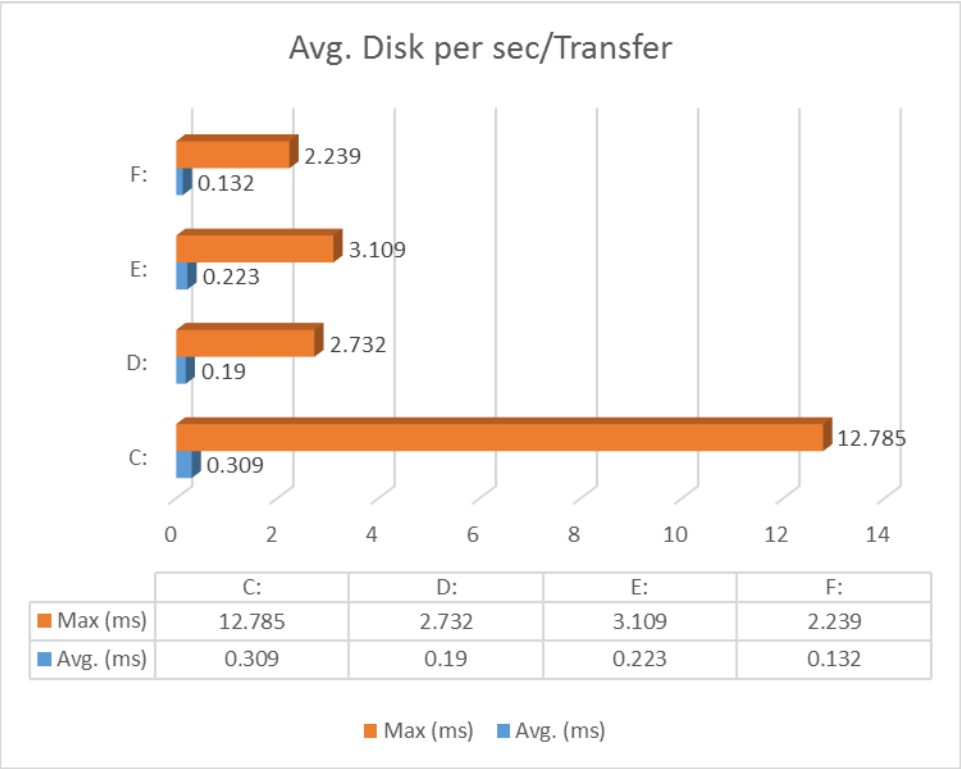
Counter
Avg. Disk sec/Read
Description
Avg. Disk sec/Read is the average time, in seconds, of a read of data to the disk. This analysis determines if any of the logical disks are responding slowly. The following thresholds are based on the access times of 5400 RPM disk drives. Hard drives that are faster than 5400 RPM such as 7200 RPM and solid state drives should have faster response times. Occasional spikes above 25 ms are normal. If the response times are less than 0.015 (15 milliseconds), then the disk subsystem is keeping up with demand. If the response times are greater than 0.025 (25 milliseconds), then the disk subsystem is likely overwhelmed.
Notes
There are performance issues with these disks.



Counter
LogicalDisk Avg. Disk sec/Write
Description
Avg. Disk sec/Write is the average time, in seconds, of a read of data to the disk. This analysis determines if any of the logical disks are responding slowly. The following thresholds are based on the access times of 5400 RPM disk drives. Hard drives that are faster than 5400 RPM such as 7200 RPM and solid state drives should have faster response times. Occasional spikes above 25 ms are normal. If the response times are less than 0.015 (15 milliseconds), then the disk subsystem is keeping up with demand. If the response times are greater than 0.025 (25 milliseconds), then the disk subsystem is likely overwhelmed
Notes
There are serious write performance issues that need to be addressed.

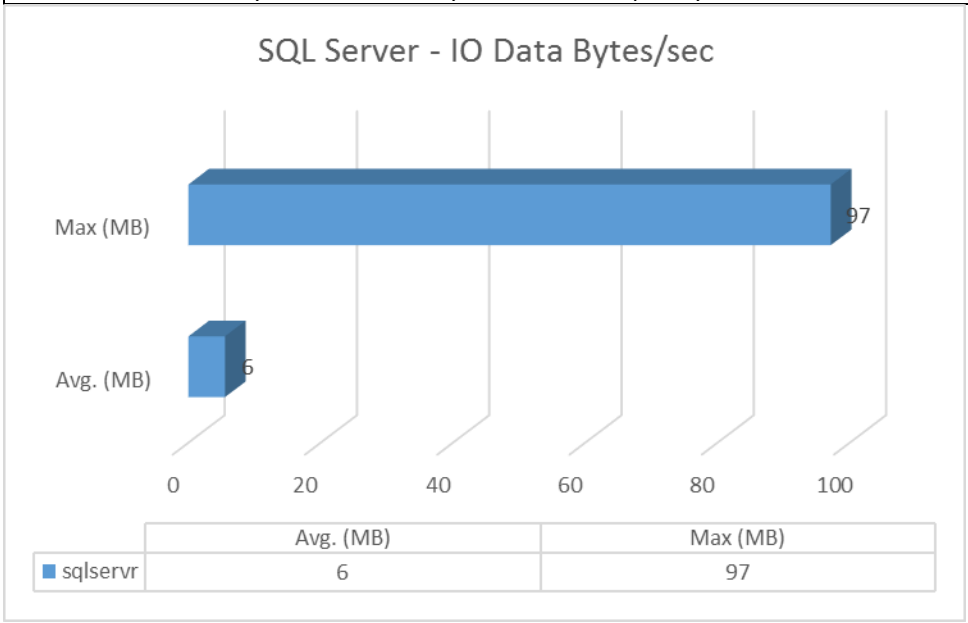


Counter
Avg. Disk sec/Transfer
Description
Avg. Disk sec/Transfer is the time, in seconds, of the average disk transfer.
Notes
There are serious IOP issues that need to be addressed.



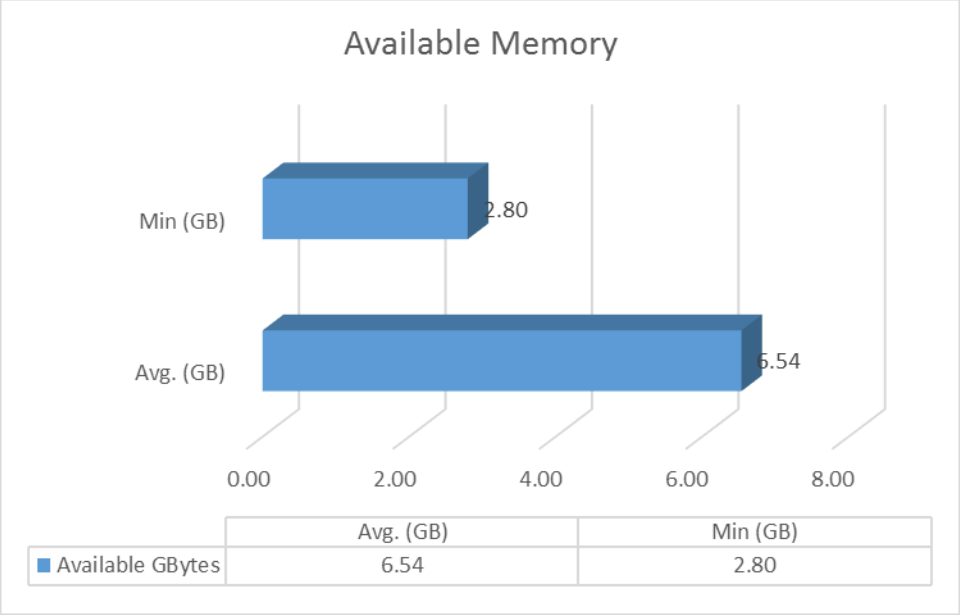
	C:	D:	E:	F:
Max (ms)	12.785	2.732	3.109	2.239
Avg. (ms)	0.309	0.19	0.223	0.132

Counter
IO Data Bytes/Sec
Description
The amount of data SQL Server generates in bytes per second.
Notes
There are periodic spikes of throughput, however the averages are well within normal ranges. The max range may however cause latency on the host subsystem if not adequately sized to handle workloads.

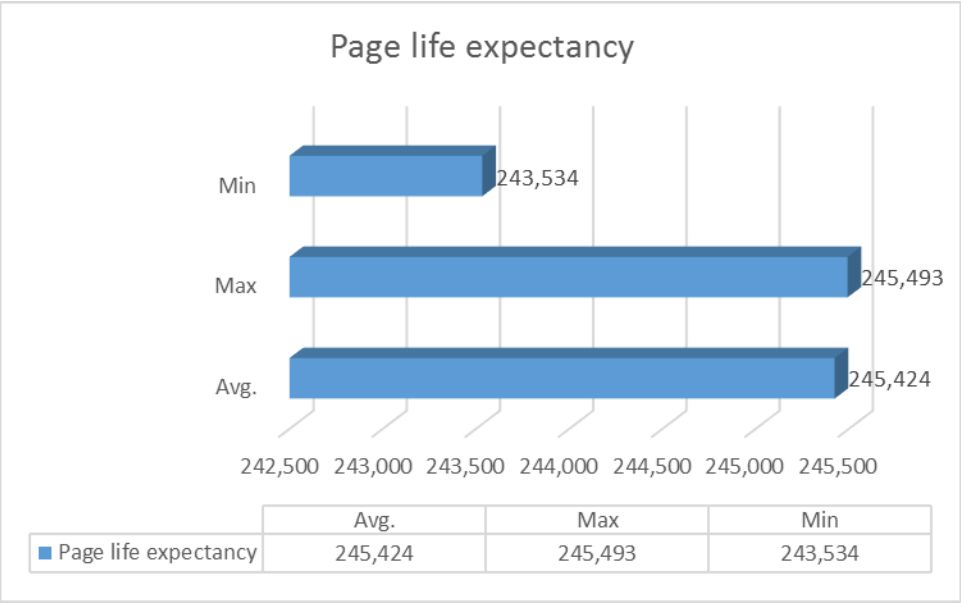


	Avg. (MB)	Max (MB)
sqlservr	6	97

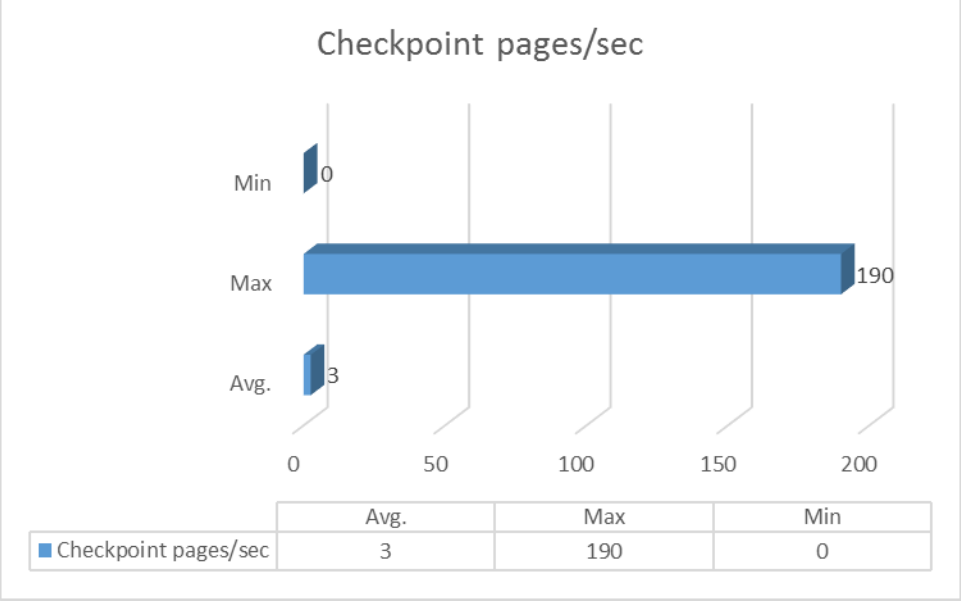
Counter
Memory Available MBytes
Description
Available MBytes is the amount of physical RAM, in megabytes, immediately available for allocation to a process or for system use. It is equal to the sum of memory assigned to the standby (cached), free and zero page lists. If this counter is low, then the computer is running low on physical memory (RAM).
Notes
Available memory did not drop below 300 MB, so there doesn't appear to be any issues.



Counter
Page life expectancy
Description
The number of seconds a page will stay in the buffer pool without references. In other words, if your pages stay in the buffer pool longer (memory) PLE will be higher. This leads to higher performance as every time a request comes in there will be a higher chance that it will the data required in cache versus disk (slow) to read the data into memory.
Notes
None, no issues present.

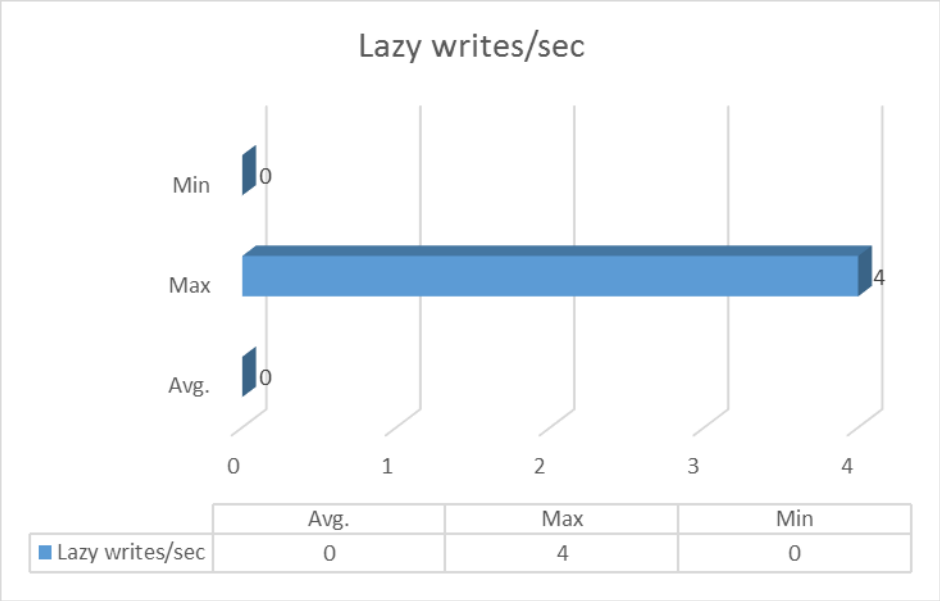


Counter
Checkpoint pages/sec
Description
The number of pages flushed to disk per second by a checkpoint or other operation that require all dirty pages to be flushed. If this counter is climbing, it might mean you are running into memory pressures that are causing dirty pages to be flushed to disk more frequently than normal. A baseline is required to accurately measure this counter.
Notes
None. Currently there is no issue with these values.

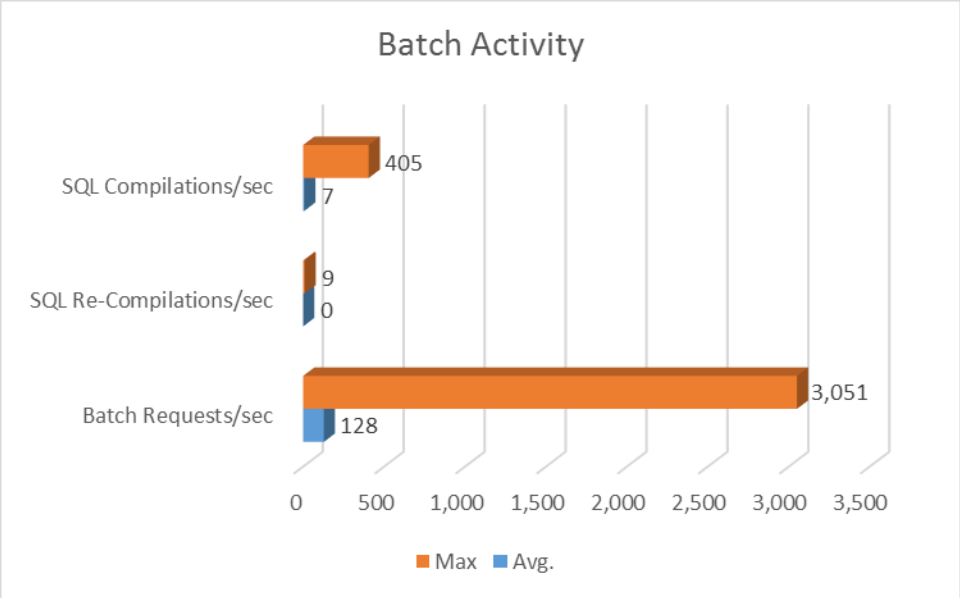


Counter
Lazy Writes/sec
Description

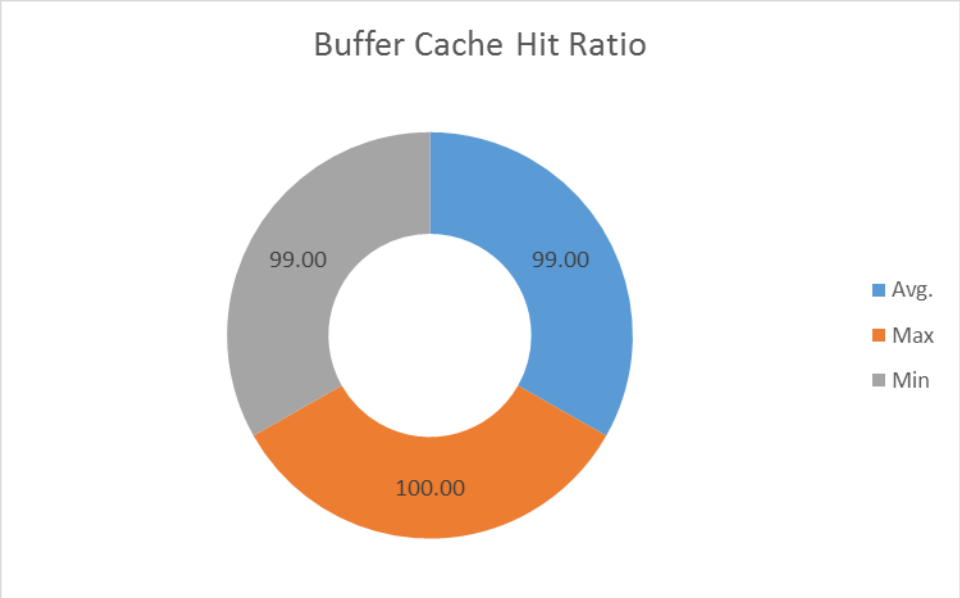
The lazy writer process periodically checks the available free space in the buffer cache between two checkpoints. If a dirty data page (a page read and/or modified) in the buffer hasn't been used for a while, the lazy writer flushes it to disk and then marks as free in the buffer cache. If SQL Server needs more memory and the buffer cache size is below the value set as the Maximum server memory parameter for the SQL Server instance, the lazy writer will take more memory. If SQL is under memory pressure, the lazy writer will be busy trying to free enough internal memory pages and will be flushing the pages extensively. The intensive lazy writer activity affects other resources by causing additional physical disk I/O activity and using more CPU resources
Notes
None, there is no issue with these values.



Counter
Batch Requests/sec SQL Compilations/sec SQL Re-Compilations/sec
Description
Batch Requests/Sec measures the number of batches SQL Server is receiving per second. The SQL Compilations/Sec measure the number of times SQL Server compiles an execution plan per second. Compiling an execution plan is a resource-intensive operation. The Re-compilations/Sec counter measures the number of time a re-compile event was triggered per second.
Notes
None. There is no issues with batch requests compared against compilations and recompilations.

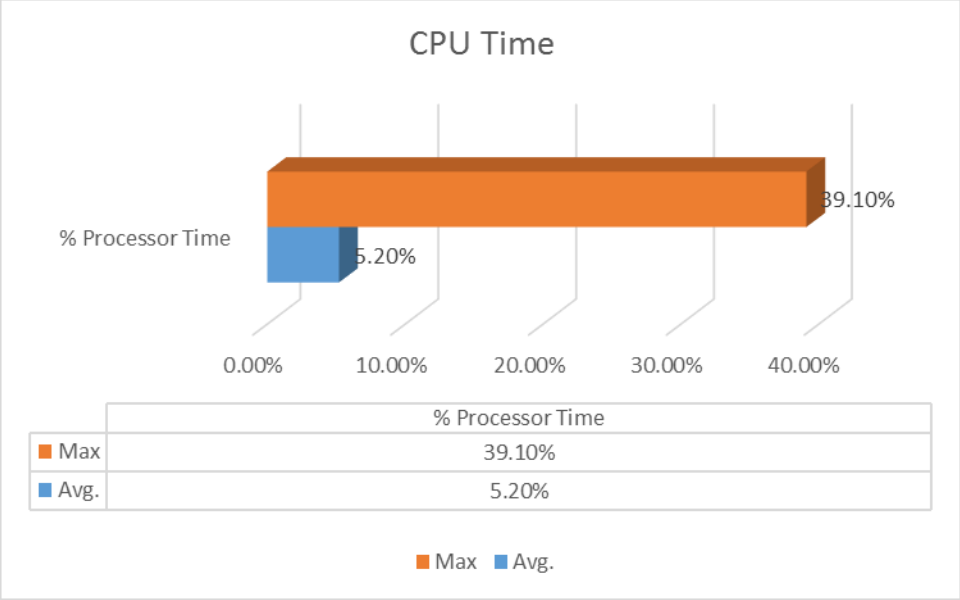


Counter
Buffer cache hit ratio
Description
The buffer cache hit ratio counter represents how often SQL Server is able to find data pages in its buffer cache when a query needs a data page. The higher this number the better, because it means SQL Server was able to get data for queries out of memory instead of reading from disk. You want this number to be as close to 100 as possible. Having this counter at 100 means that 100% of the time SQL Server has found the needed data pages in memory. A low buffer cache hit ratio could indicate a memory problem.
Notes
None, there are no issues.



Counter
Processor Time

Description
Percentage of time (cumulative) all cores spent in each thread compartment.
Notes
None, there is no issue with CPU consumption.



Windows Event Log Details

Application Event Log
Event Count
No Dynamics AX errors found
Event Description
Notes
Recommendation

System Event Log
Event Count
No Dynamics AX errors found
Event Description
Notes
Recommendation