# Database-Level Performance Queries Part 2

Glenn Berry
Glenn@SQLskills.com
GlennAlanBerry

pluralsight
hardcore dev and IT training

# Database-Level Performance Queries

- **A group of queries to collect database-level performance metrics**
  - These must be run in the context of the database you are concerned with
  - These are database specific queries
- **Many SQL Server databases have performance issues**
  - These queries help you further focus your tuning efforts in the right area
- **My Pluralsight course *Scaling SQL Server 2012 – Part 1* covers best practice instance-level performance considerations**
  - http://bit.ly/1iL0NQR
- **Joe Sack's Pluralsight course *SQL Server: Common Performance Issue Patterns* is also a valuable resource**
  - http://bit.ly/1nTzupp

# Bad Nonclustered Indexes

- **This query helps you identify indexes with more writes than reads**
  - It uses the sys.dm_db_index_usage_stats DMV
    - MSDN link: http://bit.ly/1reC1Zh
  - It also uses the sys.indexes catalog view
    - MSDN link: http://bit.ly/1C22vY9
- **Be cautious about dropping indexes that show up in this query**
  - Don't drop Primary Key or Foreign Key indexes
  - Make sure the instance has been running long enough to see your complete workload and business cycle
  - Make sure you understand your workload before dropping indexes

# Missing Indexes for Current Database

- **This query returns information about candidate "missing indexes" for the current database**
  - sys.dm_db_missing_index_group_stats
    - MSDN link: http://bit.ly/1rU6o94
  - sys.dm_db_missing_index_groups
    - MSDN link: http://bit.ly/1hYbjTh
  - sys.dm_db_missing_index_details
    - MSDN link: http://bit.ly/1pU8yry
- **This query is both useful and potentially dangerous**
  - It can help find missing indexes with very high impacts
  - It encourages less experienced DBAs to over-index their databases
  - It sometimes recommends duplicate indexes
  - Never just blindly create every index that it recommends!

# Missing Index Warnings

- **This query uses the sys.dm_exec_cached_plans DMV cross-applied with the sys.dm_exec_query_plan DMF**
  - sys.dm_exec_cached_plans
    - MSDN link: http://bit.ly/1y5VsZq
  - sys.dm_exec_query_plan
    - MSDN link: http://bit.ly/1pEZzp3
- **This query finds cached query plans that have missing index warnings**
  - It can help you associate missing indexes with particular stored procedures
  - It will often find candidate indexes that don't show up in other queries
  - This query can take some time to complete on busy instances

# Buffer Usage

- **This query uses the sys.dm_os_buffer_descriptors DMV joined with sys.allocation_units and sys.partitions catalog views**
  - sys.dm_os_buffer_descriptors
    - MSDN link: http://bit.ly/1xV1hez
  - sys.allocation_units
    - MSDN link: http://bit.ly/1yNZhSp
  - sys.partitions
    - TechNet link: http://bit.ly/1xOjycd
- **This query shows which indexes are taking the most buffer space**
  - This helps you better understand your buffer pool contents
  - It also helps identify possible candidates for SQL Server data compression

# Table Sizes

- **This query uses the sys.partitions catalog view**
  - sys.partitions
    - TechNet link: http://bit.ly/1xOjycd
- **Tells you which tables have the most rows**
  - This helps you understand your overall workload
  - It also helps identify possible candidates for SQL Server data compression

# Table Properties

- **This query uses the sys.tables catalog view**
  - sys.tables
    - MSDN link: http://bit.ly/1zSFTo5
- **This gives you some useful information about your tables**
  - Is the table replicated?
  - Does the table have a replication filter?
  - Is the table tracked by change data capture (CDC)?
  - What is the lock escalation property?
  - Is it a memory optimized table (Hekaton)
  - If it is a Hekaton table, what type of durability is it using?

# Statistics Update

- **This query uses the sys.objects joined with the sys.indexes, sys.stats catalog views and sys.dm_db_partition_stats DMV**
  - sys.objects
    - MSDN link: http://bit.ly/11R07Uc
  - sys.indexes
    - MSDN link: http://bit.ly/1C22vY9
  - sys.stats
    - MSDN link: http://bit.ly/1HDAHtm
  - sys.dm_db_partition_stats
    - MSDN link: http://bit.ly/1tfiYik
- **Tells you when your indexed-based statistics were last updated**
  - This helps you evaluate the quality of your statistics

# Index Fragmentation

- **This query uses the sys.indexes catalog view joined with the sys.dm_db_index_physical_stats DMV**
  - sys.indexes
    - MSDN link: http://bit.ly/1C22vY9
  - sys.dm_db_index_physical_stats
    - MSDN link: http://bit.ly/1vGPssm
- **Returns fragmentation levels for indexes above a given size**
  - Helps you evaluate your index maintenance effectiveness
  - It is very common to over-maintain or under-maintain indexes

# Overall Index Usage: Reads

- **This query uses the sys.dm_db_index_usage_stats DMV joined with the sys.indexes catalog view**
  - sys.dm_db_index_usage_stats
    - MSDN link: http://bit.ly/1reC1Zh
  - sys.indexes
    - MSDN link: http://bit.ly/1C22vY9
- **Returns indexes that have the highest number of reads**
  - Shows which indexes are the most valuable for your workload
  - This also helps you understand your query workload
  - It can help identify possible data compression candidates

# Overall Index Usage: Writes

- **This query uses the sys.dm_db_index_usage_stats DMV joined with the sys.indexes catalog view**
  - sys.dm_db_index_usage_stats
    - MSDN link: http://bit.ly/1reC1Zh
  - sys.indexes
    - MSDN link: http://bit.ly/1C22vY9
- **Returns indexes that have the highest number of writes**
  - Shows which indexes are the most volatile for your workload
  - This also helps you understand your write workload
  - It can help identify possible indexes that are not as useful

# Recent Full Backups

- **This query uses the msdb.dbo.backupset system table**
  - msdb.dbo.backupset
    - TechNet link: http://bit.ly/1nWu3I7
- **Shows metrics about recent full database backups**
  - Shows whether native backup compression is being used
  - Shows the backup compression ratio
  - Shows the backup elapsed time and finish date
  - Help you track the growth of your database and your full backups

# Database Size History

- **This query uses the msdb.dbo.backupset system table**
  - msdb.dbo.backupset
    - TechNet link: http://bit.ly/1nWu3I7
- **Shows metrics about recent full database backups**
  - Shows whether native backup compression is being used
  - Shows the backup compression ratio
  - Help you track the historical growth of your database and your full backups, grouped by month

# Course Summary

- **These queries can detect most database-level performance issues**
  - They can also help you find instance-level settings that may be incorrect
- **They give you performance information about your stored procedures, indexes, and statistics**
- **They can help you find your database-level bottlenecks that warrant more detailed investigation and troubleshooting**
- **Make sure to also watch the other DMV courses for more queries**

- **Thanks for watching!**