

SQLskills Immersion Event IE2: Performance Tuning

Module 13: Performance Issue Patterns

Joe Sack
Joe@SQLskills.com



Overview

- There also several patterns and scenarios which repeat themselves with only minor variations
- These patterns are rarely identical, but can be recognized based on common attributes
- Following a methodology as we've been discussing this week will help you avoid missing upstream issues
- This module will provide an interactive investigation into various performance issue patterns



2

© SQLskills, All rights reserved.
<http://www.SQLskills.com>



Finding the Root Cause

- Live analysis or post-mortem?
- Plenty of tools to choose from
- I'll scope them for the purposes of this session
- The challenge – identify the right tools for root cause analysis, keeping in mind economy of effort
- Common issues, variations, multi-jointed problems

Scenario "A" Symptoms

- Production system experiencing random drops in throughput during business hours – from the regular 200+ batch requests/sec to less than 80 batch requests/sec
- Large team of developers with unrestricted access and continual development phases and code "refinement"
- In the past, drops in throughput are not in conjunction with code changes
- They have not witnessed waiting tasks that have a persistent theme

Scenario "B" Symptoms

- Hybrid application workload - around 50% OLTP and 50% complex reporting
- Smaller transaction throughput seems fine, but some of the reporting workloads seem to have trouble executing concurrently
- Reporting queries seem to run "fine" when tested in isolation and aren't getting blocked according to block detection alerts

Scenario "C" Symptoms

- A recent third-party code release to the Credit database has resulted in unexpected tempdb usage, discovered via out-of-space errors
- Average duration for statements has increased for some workloads, but not others
- Customer wants to know – "what changed" and "why"?

Scenario "D" Symptoms

- Frequent deadlock 1205 error messages
- The issue happened recently as additional application load was increased
- Need help resolving the workload involved

Scenario "E" Symptoms

- "Reduced throughput" for key workloads
- PAGELATCH_X high wait stats in sys.dm_os_wait_stats and ongoing in sys.dm_os_waiting_tasks

Scenario "F" Symptoms

- Customer has had reported inconsistent workload performance (reported as "random" performance issues)
- Restarting SQL Server usually helps for a bit, and then random issues later
- They had a consultant come in to "fix" the various stored procedures to perform better
- Now the performance is more consistent, but the CPU is pegged during peak application workload periods

Scenario "G" Symptoms

- CPU usage has been slowly climbing over 80% persistently
- Increase I/O usage as well
- DBA on staff has been troubleshooting the TOP X queries, adding indexes as needed, but what they are tuning is NOT helping
- What workloads should they be troubleshooting?

Scenario "H" Symptoms

- Customer has developed a reporting query
- It used to run in parallel and now it does not and runs more slowly than before
- They tried tuning it to go parallel – but with no success

Scenario "I" Symptoms

- Database development team with a long history of turnover, handoffs and varying coding philosophies
- Read-centric activity reported as sufficient, but write-centric activity has had increasing latency over time
- Customer has provided an example INSERT + sub-SELECT that does not perform as quickly as it used to (and they want you to optimize)

Scenario "J" Symptoms

- Slow performance at peak business periods or during maintenance periods
- High CPU (>95% across all 4 schedulers)
- Quickly increasing percentage of PAGEIOLATCH_XX waits

Scenario "K" Symptoms

- A specific stored procedure is associated with excessive I/O
- The cardinality estimates are always "wrong" – but which table they are incorrect for seems to change periodically
- CPU usage is fine – but excessive I/O is the main overall bottleneck
- Can you help get the cardinality estimates to be correct AND solve the excessive I/O issue?

Review

- There also several patterns and scenarios which repeat themselves with only minor variations
- These patterns are rarely identical, but can be recognized based on common attributes
- Following a methodology will help you avoid missing upstream issues
- This module provided an interactive investigation into various performance issue patterns

Questions?

Bonus Scenario 0

- A developer is executing a “simple” query against a view
- The performance is not what the developer expects
- The developer would like you to look at the execution plan and see if there is anything that can be done to improve the performance

Bonus Scenario 1

- DBA is seeing performance issues for a set of queries that follow a similar pattern
- The query pattern includes predicates that reference two or more columns from the same table
- For larger reference tables, the DBA sometimes sees Sort and Hash Join spills to disk
- The DBA wants to know why this is happening and how it can be fixed

Bonus Scenario 2

- A developer is seeing performance issues for a set of queries that follow a similar pattern
- The query pattern includes predicates that compare two columns from the same table
- The specific query is “simple” so the developer would like to understand why this is happening and how it can be fixed

Bonus Scenario 3

- A developer is using a new function introduced in SQL Server 2012
- The query pattern allows for a more flexible way to enforce predicates
- The query isn't performing well and the developer needs your help

Bonus Scenario 4

- A developer is seeing poor performance related to a workload and has looked at the query execution plan
- There are multiple statements, and the developer is having trouble figuring out how to improve the performance
- The developer needs your help to improve the performance of the overall workload

Bonus Scenario 5

- A developer is seeing poor performance related to a query that searches for people based on first and last name
- The developer found a cardinality estimate skew, but doesn't know how to fix it

Bonus Scenario 6

- A DBA created a filtered index on a table in order to benefit the performance of a specific mission-critical stored procedure
- The stored procedure doesn't seem to be performing any better than it was before, and the DBA wants to understand why and also have you find a solution

Bonus Scenario 7

- A report developer is producing an aggregated charge amount report that seems to be driving a high amount of I/O according to the DBA
- The report developer wants to understand why the report is driving high I/O – and also wants you to provide a solution if possible