
Rule WLM651: Lock Contention was high for the indicated structure

Finding: The lock contention for the indicated structure was higher than guidance provided by IBM for normal lock contention.

Impact: This finding can have a LOW IMPACT, MEDIUM IMPACT, or HIGH IMPACT on the signalling performance of the sysplex. The level of impact depends on the amount of lock contention.

Logic flow: This a basic finding. There are no predecessor rules.

Discussion: *Locking* is the mechanism used to reserve all or part of a database so that other programs will not be able to update the data until you have finished processing the data. By locking the data, users can be sure that the information they are processing is current. Without locking, users might lose updates or access invalid or incomplete data. Locking is necessary, of course, only if one or more of the users of the data will be performing updates. If no updating of the data is performed, locking is unnecessary; the data may be concurrently accessed by any number of user without worry that the data is incomplete or invalid.

Lock contention occurs when one user wishes to access data and some other user has placed a lock on the data. The user wishing to access the data usually is suspended until the data is available (that is, until the lock is released). Techniques such as separating data, choosing locking parameters, and monitoring for contention can be used to provide a balance between concurrency of access, isolation and integrity of data, and efficient use of system resources. Lock contention is analyzed by CPEXpert in Rule WLM651.

SMF Type 74 (Subtype 4 - Coupling Facility Activity) records contain information describing the requests for data, the number of requests that were delayed because of lock contention, and the number of requests that encountered false lock contention. CPEXpert analyzes this information to determine whether an excessive percentage of requests encountered lock contention.

CPEXpert divides R744SSCN (the number of times any request encountered lock contention) by R744STRC (the total number of lock-related requests), to yield the percent of requests that experienced lock contention. CPEXpert compares this percentage with the **LOCKCONT** guidance variable in USOURCE(WLMGUIDE).

CPEXpert produces Rule WLM651 when the percent of lock contention exceeds the value specified by the LOCKCONT variable.

The default value for the LOCKCONT variable is 2%, indicating that CPEXpert should produce Rule WLM651 when more than 2% of the requests were delayed because of lock contention. IBM documents suggest that most CICS/DBCTL workloads should have less than 1% lock contention, and most IMS/DB2 workloads should have less than 2% lock contention.

The following example illustrates the output from Rule WLM651:

| | | | |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------|-----------------|--------------|
| RULE WLM651: LOCK CONTENTION WAS HIGH | | | |
| DB2DBP2_LOCK1: The lock contention for this structure was higher than normal. High lock contention can result in an increase in central processor utilization and a reduction in throughput. If this finding continues to occur, you should review the alternatives listed in the WLM Component User Manual. If you are unable to take action, you should consider increasing the LOCKCONT guidance variable, located in USOURCE(WLMGUIDE). The LOCKCONT variable currently is 2%. | | | |
| | TOTAL LOCK | REQUESTS WITH | PERCENT LOCK |
| MEASUREMENT INTERVAL | REQUESTS | LOCK CONTENTION | CONTENTION |
| 12:45-13:00,02OCT1996 | 17,696 | 910 | 5 |
| 16:45-17:00,02OCT1996 | 12,320 | 757 | 6 |
| 17:15-17:30,02OCT1996 | 3,741 | 371 | 10 |

Suggestion: A frequent cause of high lock contention is batch jobs running against shared databases. If this is the cause, you may be able to reschedule the batch jobs to resolve the lock contention problem.

Additionally, CPEXpert suggests that you consider the following alternatives, depending on the type of lock structure experiencing the contention:

- If the structure involved is CICS/DBCTL, you should refer to the CICS-IMS DBCTL Guide for a discussion about lock contention and suggestions on how to prevent the lock contention. IBM provides the following recommendations on ways to reduce lock contention:
 - All BMPs and applications should issue frequent checkpoints to avoid locking out other resource users.
 - All BMPs and applications must be restartable from last checkpoint. This is because records in the same database may have since been updated, and these updates would be lost if the database were restored from a previous backup.

-
- BMPs and applications should not hold on to locks for long periods without issuing checkpoints or syncpoints (either explicitly or implicitly).
 - Review the use of control records; that is, records that are accessed by most applications. If they have to be updated, it is important to remember that the CI or physical block is locked from other subsystems until the updates are committed.
 - If the structure involved is DB2, you should refer to the following sections of the indicated documents (shown in the References section of this rule description) for the appropriate version of DB2 running on your system:
 - "Tuning your use of Locks" section and the "Improving Concurrency" section in the Data Sharing: Planning and Administration document.
 - "Improving Concurrency" section in the DB2 Administration Guide
 - "Archive Log Data Set Parameters" section in the DB2 Installation Guide

If you decide that the DB2 application design is causing lock contention, you should refer to the DB2 Application Programming and SQL Guide for detailed suggestions about how to avoid or minimize lock contention.

- If the structure involved is JES2 checkpoint, the structure will be a *serialized list structure* with locking controlled by JES2 in a multi-access spool (MAS) environment.

Each JES2 member of the MAS will acquire and hold the checkpoint for the duration specified in the HOLD parameter on the MASDEF initialization statement. Upon releasing the lock on the checkpoint, each JES2 member of the MAS will wait for the interval specified in the DORMANCY parameter on the MASDEF initialization statement before attempting to again acquire the checkpoint.

Please refer to the "Accessing the CKPTn Data Set in a MAS" section of the JES2 Initialization and Tuning Guide for IBM's suggestions on setting the HOLD and DORMANCY parameters in a MAS environment in which the CKPTn data sets reside in structures on a coupling facility.

-
- Reference:** OS/390: DB2 for MVS/ESA Version 4
Data Sharing: Planning and Administration (SC26-3269)
Administration Guide (SC26-3265)
Installation Guide (SC26-3456)
Application Programming and SQL Guide (SC26-3266)
- OS/390: DB2 for MVS/ESA Version 5
Data Sharing: Planning and Administration (SC26-8961)
Administration Guide (SC26-8957)
Installation Guide (SC26-8970)
Application Programming and SQL Guide (SC26-8958)
- OS/390: DB2 for MVS/ESA Version 6
Data Sharing: Planning and Administration (SC26-9007)
Administration Guide (SC26-9003)
Installation Guide (SC26-9008)
Application Programming and SQL Guide (SC26-9004)
- OS/390: JES2 Initialization and Tuning Guide (SC28-1791)
- OS/390: RMF Performance Management Guide (SC28-1951)
- z/OS: JES2 Initialization and Tuning Guide (SA28-7532)
- z/OS: RMF Performance Management Guide (SC33-7992)
- Washington System Center Flash 9609 ("CF Reporting Enhancements to RMF 5.1")
- "Parallel Sysplex Performance: tuning tips and techniques,"
Kelley, Joan (IBM, Poughkeepsie, NY), SHARE 86, February 1996.