Blocked Sessions (Count) alert

The Blocked Sessions (Count) alert provides the number of sessions on the SQL Server instance blocked by other sessions holding requested locks. For this alert to display, enable monitoring of non-query activities with capturing of blocking events.

By default, blocked sessions wait for a lock wait forever, which is not optimum behavior. You can change this default behavior by making a connection, and then adding `SET LOCKTIMEOUT 60000` to the top of the first T-SQL batch.

Reduce the number of blocked sessions

If you experience more than one blocked session on an average day, your site may experience one of the following issues:

- T-SQL batches submitted containing a `BEGIN TRANSACTION` statement with no corresponding `COMMIT TRANSACTION` statement. You must correct the T-SQL.
- T-SQL batches submitted containing a `BEGIN TRANSACTION` statement but where the `COMMIT TRANSACTION` statement is in a following T-SQL batch that is only executed once the end-user confirms the transaction. You must correct the T-SQL.
- The site includes some long-running transactions processing at peak times.

When your transactions run longer than they should and clash, consider drastically reducing the time that a lock is held by:

- Doing as much work as possible (for example, SELECT statements that may be needed) before the transaction performs its first update, delete, or insert.
- Group all UPDATES, DELETES, and INSERTS as closely as possible together within a transaction with as few selects as possible separating them.
- Commit the transaction as soon as possible after the last DML statement.
- Once the transaction has begun do not have any stops for user input. Gather all user inputs before the transaction starts.
- Avoid the use of server side cursors during a transaction as they slow execution considerably.
- If a stored procedure and/or trigger are invoked inside a transaction minimize or eliminate the number of SQL re-compilations made to that object. See the SQL Re-compilations counter for ways to dramatically reduce recompiles.
- Increase the speed of transaction throughput such that it becomes less likely that one transaction waits for the preceding one by:
  - Adding more disks to your RAID solution.
  - Replacing your disks with faster disks.
  - Switching your RAID array from a RAID 5 to a RAID 10 solution as each write IO results in two writes using RAID 10 vs. four for RAID 5 (100% more efficient with writes).
  - So long as the RAID controller has some form of battery backup, then switch its cache mode from Write-through to Write-back as this increases the system's ability to handle write IOs by an order of magnitude.
  - Adding more cache memory to the RAID controller.
  - Adding more RAM to the server.
  - Adding another CPU to a SMP computer.
  - Upgrading the CPU, memory and motherboard with faster models.
  - Minimize the number of Context Switches by turning on Use NT Fibers in SQL Server.
  - Switch the Boost SQL Server Priority on.

Enable baseline thresholds

To enable alerting when this metric is outside its established baseline, click the **Baseline Thresholds Enabled (as a percentage of baseline)** check box in the Alert Configuration window.

Create an alert response bundle

Create an alert response bundle with the Blocked Sessions (Count) alert and related alerts. For additional information, see Create alert response bundles.