

Negative Blocking Header

Last updated by | Holger Linke | Sep 26, 2022 at 3:42 AM PDT

Contents

- [Issue](#)
- [Investigation / Analysis](#)
- [Mitigation](#)
 - [Regarding session ID = -1](#)
 - [Regarding session ID = -2](#)
 - [Regarding session ID = -3](#)
 - [Regarding session ID = -4](#)
 - [Regarding session ID = -5](#)
- [Public Doc Reference](#)

Issue

SQL Server may report a blocking session ID as a negative integer value. The customer cannot find any user sessions related to those numbers and got concerned that their workloads are blocked by some platform task or by itself. They ask for mitigation and more details about these negative session IDs.

If the customer concern is specifically about a session ID = -5, see the related article [Blocking session id -5 \(spid -5\)](#).

Investigation / Analysis

SQL Server reports a blocking session ID as a negative integer value to indicate special conditions. The meaning of a negative blocking session ID depends on the actual ID and the context.

Here is the complete list of possible negative blocking session IDs:

Value	Description
-1	Orphaned lock, commonly a bug/defect in SQL Server where lock ownership has been incorrectly lost.
-2	Pending DTC transaction ☐. There are no enlisted sessions on the SQL Server but the transaction is still active. The client connection(s) associated with the DTC transaction have disconnected and the SQL Server DTC transaction object is waiting for the MSDTC Manager's state change notification. The client application needs to invoke commit or rollback on the transaction interface to complete the DTC transaction.
-3	Deferred recovery. The lock is actively held by a deferred transaction ☐.
-4	Latch transition occurring. Indicates a latch release is in progress.
-5	The session is waiting on an asynchronous operation to complete. Any task/session can release the latch. I/O latches are the most common occurrences.

Mitigation

Regarding session ID = -1

This might require an ICM if the issue occurs repeatedly or can be reproduced.

A restart of the SQL service through a reconfiguration/failover should clear the active locks and resolve the immediate issue. To initiate a failover, see the [Invoke-AzSqlDatabaseFailover](#) ☐ PowerShell cmdlet.

Regarding session ID = -2

It is unsure if the spid=-2 scenario applies to Azure SQL Database at all. Since MSDTC isn't available for PaaS applications in Azure, the ability to coordinate distributed transactions has been directly integrated into SQL Database and Managed Instance. Applications can connect to any database to launch distributed transactions, and one of the databases or servers will transparently coordinate the distributed transaction.

If you see this in Azure SQL Database, the cause is likely application-related. See [Distributed transactions across cloud databases](#) ☐ for more information about distributed transactions on PaaS.

A restart of the SQL service through a reconfiguration/failover might clear the active locks and resolve the immediate issue. To initiate a failover, see the [Invoke-AzSqlDatabaseFailover](#) ☐ PowerShell cmdlet.



Open an ICM if the failover does not help, or if the issue reoccurs again without a clear understanding of the reason.

Regarding session ID = -3

A deferred transaction is a transaction that remains uncommitted when the roll forward phase finishes but that cannot be rolled back because it encounters an error during rollback. It usually occurs because, while the

database was being rolled forward, an I/O error prevented reading a page that was required by the transaction. The redo operation reacquires all the necessary locks, and all data modified by the transaction remains locked until the transaction can roll back. Therefore the blocking by spid=-3 will persist until the deferred transaction can be cleared.

This might be caused either by a harmless, transient I/O error, or by a severe error at the file level involving database corruption. It might also be caused by a failed, partially-completed restore sequence.

The first step for [moving a transaction out of the DEFERRED state](#)  is to restart/failover the database. If the error was transient, there is a good chance that this resolves the issue. To initiate a failover, see the [Invoke-AzSqlDatabaseFailover](#)  PowerShell cmdlet.

Open an ICM if the failover does not help, or if the issue reoccurs again later without a clear understanding of the reason.


Regarding session ID = -4

This is very likely a short-lived, transient situation without any impact on its own. The customer very likely has general performance issues on their database and the investigation should start on the performance side, e.g. through ASC.

Regarding session ID = -5

See the "Mitigation" section in the related article [Blocking session id -5 \(spid -5\)](#) for more information.

Public Doc Reference

Review this article for more information about negative blocking sessions: [Negative Blocking Session Ids \(-5 = Latch ANY TASK RELEASOR\)](#) .

How good have you found this content?

