Troubleshoot Database Concurrency in SQL Server with sp_locks

BY YANIV ETROGI

Usage

–Return only the second result set

EXEC sp_Locks @Mode = 1, @Wait_Duration_ms = 1000;

– Return the first and second result sets

EXEC sp_Locks @Mode = 2, @Wait_Duration_ms = 1000;

– Return all three result sets

EXEC sp_Locks @Mode = 3, @Wait_Duration_ms = 1000;

Capture blocking information to table

To save blocking information to table use the provided script *Capture_blocking_info.sql* which uses an INSERT…EXEC statement with sp_locks in @Mode = 2.

Inside the locks

To get an insight view into a process that is blocked or blocking and returned by sp_locks you can use the bellow script that details the reason for why that process is blocked or blocking.

I use it when I troubleshoot and reproduce a blocking scenario while filtering on the relevant spid so I can see only the data I am interested in and then get a picture of the granted resources and their lock mode.

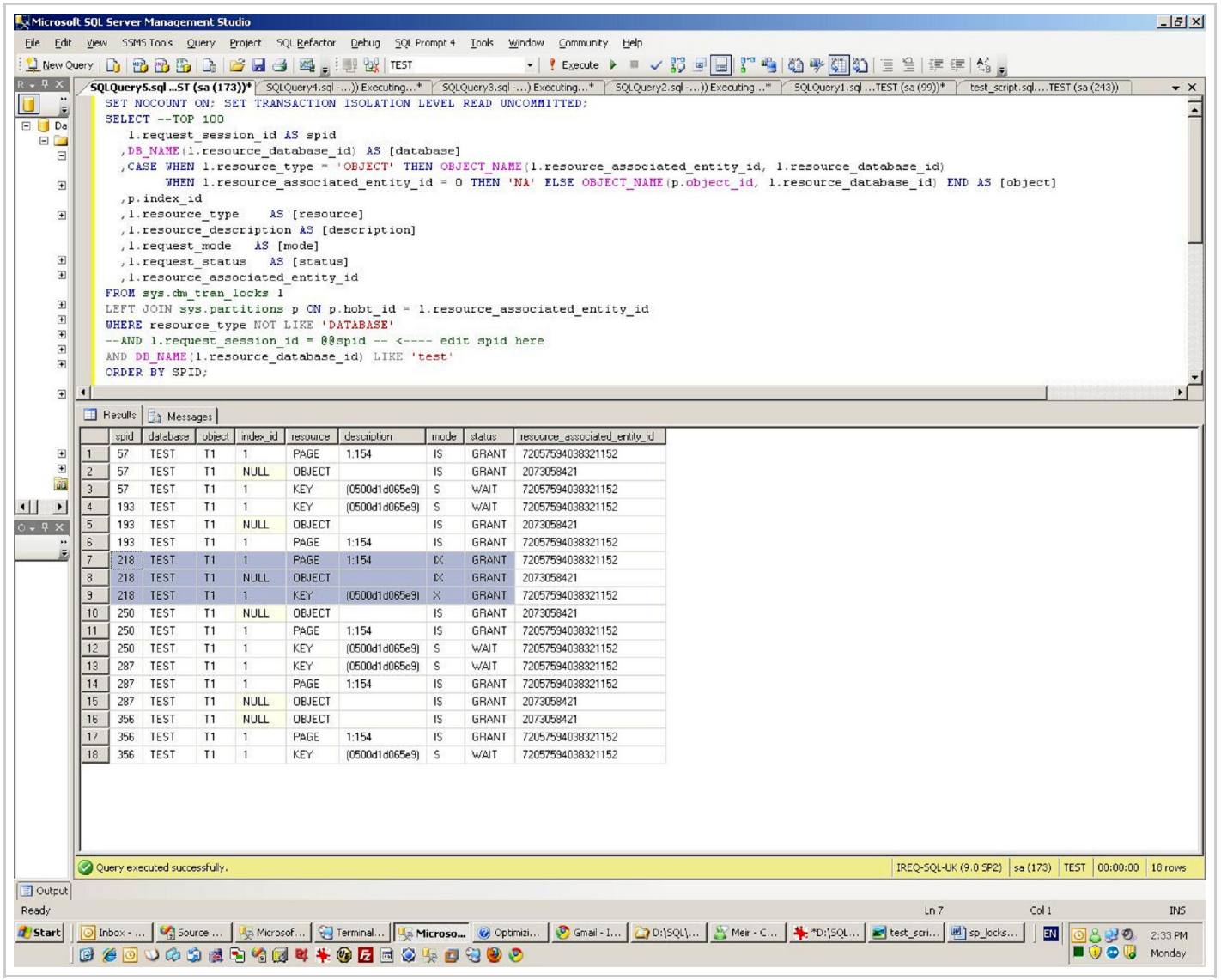
SET NOCOUNT ON;

```
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

SELECT –TOP 100

    l.request_session_id AS spid

    ,DB_NAME(l.resource_database_id) AS [database]

    ,CASE WHEN l.resource_type = 'OBJECT' THEN OBJECT_NAME(l.resource_associated_entity_id, l.resource_database_id)

            WHEN l.resource_associated_entity_id = 0 THEN 'NA' ELSE OBJECT_NAME(p.object_id, l.resource_database_id) END AS [object]

    ,p.index_id

    ,l.resource_type        AS [resource]

    ,l.resource_description AS [description]

    ,l.request_mode         AS [mode]

    ,l.request_status       AS [status]

    ,l.resource_associated_entity_id

FROM sys.dm_tran_locks l

LEFT JOIN sys.partitions p ON p.hobt_id = l.resource_associated_entity_id

WHERE resource_type NOT LIKE 'DATABASE'

–AND l.request_session_id = @@SPID – <—- edit spid here

–AND DB_NAME(l.resource_database_id) NOT LIKE 'distribution'
```

When running the script on the test scenario while filtering on the database name so I get to see all processes involved it clearly shows that the lead blocker process, spid 218 holds an exclusive (X) lock on an index key (clustered index in this case) and this is the actual lock on the row being updated. The exclusive row lock has also to be marked at the page level and this is done with an Intense Exclusive (IX) lock and you can see that this is page

number 154 in data file number 1. The Intense Exclusive (IX) page lock has also to be marked at the table level and that is done with Intense Exclusive (IX) lock on the object resource.

All the other processes trying to read the data are in a WAIT status on the KEY resource held by the UPDATE statement.



Permissions

The DMVs (Dynamic Management Views) and DMF (Dynamic Management Function) used by the stored procedure requires the caller to have the VIEW SERVER STATE permissions and of course the permissions to EXECUTE the procedure in the master databases.

```
/*

USE [master];

GRANT VIEW SERVER STATE TO Paul;

GRANT EXECUTE ON sp_locks TO Paul;

*/
```