SQL - Blocking Process

These scripts are using to help how to determine what is blocking process are occurring.

```
SELECT text,*
  FROM sys.dm_exec_requests
  CROSS APPLY sys.dm_exec_sql_text(sql_handle)
  WHERE DB_NAME(database_id) = '[dBName]'
    AND blocking_session_id <> 0
```

```
SELECT  L.request_session_id AS SPID,
        DB_NAME(L.resource_database_id) AS DatabaseName,
        O.Name AS LockedObjectName,
        P.object_id AS LockedObjectId,
        L.resource_type AS LockedResource,
        L.request_mode AS LockType,
        ST.text AS SqlStatementText,
        ES.login_name AS LoginName,
        ES.host_name AS HostName,
        TST.is_user_transaction as IsUserTransaction,
        AT.name as TransactionName,
        CN.auth_scheme as AuthenticationMethod
FROM    sys.dm_tran_locks L
        JOIN sys.partitions P ON P.hobt_id = L.resource_associated_entity_id
        JOIN sys.objects O ON O.object_id = P.object_id
        JOIN sys.dm_exec_sessions ES ON ES.session_id = L.request_session_id
        JOIN sys.dm_tran_session_transactions TST ON ES.session_id = TST.session_id
        JOIN sys.dm_tran_active_transactions AT ON TST.transaction_id = AT.transaction_id
        JOIN sys.dm_exec_connections CN ON CN.session_id = ES.session_id
        CROSS APPLY sys.dm_exec_sql_text(CN.most_recent_sql_handle) AS ST
WHERE   resource_database_id = db_id()
ORDER BY L.request_session_id;
GO
```

```
SELECT * FROM sys.dm_exec_input_buffer (66,0);
GO
```

```
WITH cteBL (session_id, blocking_these) AS
(SELECT s.session_id, blocking_these = x.blocking_these FROM sys.dm_exec_sessions s
CROSS APPLY    (SELECT isnull(convert(varchar(6), er.session_id),'') + ', '
                FROM sys.dm_exec_requests as er
                WHERE er.blocking_session_id = isnull(s.session_id ,0)
                AND er.blocking_session_id <> 0
                FOR XML PATH('') ) AS x (blocking_these)
)
SELECT s.session_id, blocked_by = r.blocking_session_id, bl.blocking_these
, batch_text = t.text, input_buffer = ib.event_info, *
FROM sys.dm_exec_sessions s
LEFT OUTER JOIN sys.dm_exec_requests r on r.session_id = s.session_id
INNER JOIN cteBL as bl on s.session_id = bl.session_id
OUTER APPLY sys.dm_exec_sql_text (r.sql_handle) t
OUTER APPLY sys.dm_exec_input_buffer(s.session_id, NULL) AS ib
WHERE blocking_these is not null or r.blocking_session_id > 0
ORDER BY len(bl.blocking_these) desc, r.blocking_session_id desc, r.session_id;
GO
```

```
WITH cteHead ( session_id,request_id,wait_type,wait_resource,last_wait_type,is_user_process,request_cpu_time
,request_logical_reads,request_reads,request_writes,wait_time,blocking_session_id,memory_usage
,session_cpu_time,session_reads,session_writes,session_logical_reads
,percent_complete,est_completion_time,request_start_time,request_status,command
,plan_handle,sql_handle,statement_start_offset,statement_end_offset,most_recent_sql_handle
,session_status,group_id,query_hash,query_plan_hash)
AS ( SELECT sess.session_id, req.request_id, LEFT (ISNULL (req.wait_type, ''), 50) AS 'wait_type'
    , LEFT (ISNULL (req.wait_resource, ''), 40) AS 'wait_resource', LEFT (req.last_wait_type, 50) AS 'last_wait_type'
    , sess.is_user_process, req.cpu_time AS 'request_cpu_time', req.logical_reads AS 'request_logical_reads'
    , req.reads AS 'request_reads', req.writes AS 'request_writes', req.wait_time,
req.blocking_session_id,sess.memory_usage
    , sess.cpu_time AS 'session_cpu_time', sess.reads AS 'session_reads', sess.writes AS 'session_writes',
sess.logical_reads AS 'session_logical_reads'
    , CONVERT (decimal(5,2), req.percent_complete) AS 'percent_complete', req.estimated_completion_time AS
'est_completion_time'
    , req.start_time AS 'request_start_time', LEFT (req.status, 15) AS 'request_status', req.command
    , req.plan_handle, req.[sql_handle], req.statement_start_offset, req.statement_end_offset,
conn.most_recent_sql_handle
    , LEFT (sess.status, 15) AS 'session_status', sess.group_id, req.query_hash, req.query_plan_hash
    FROM sys.dm_exec_sessions AS sess
    LEFT OUTER JOIN sys.dm_exec_requests AS req ON sess.session_id = req.session_id
```

```
    LEFT OUTER JOIN sys.dm_exec_connections AS conn on conn.session_id = sess.session_id
    )
, cteBlockingHierarchy (head_blocker_session_id, session_id, blocking_session_id, wait_type, wait_duration_ms,
wait_resource, statement_start_offset, statement_end_offset, plan_handle, sql_handle, most_recent_sql_handle,
[Level])
AS ( SELECT head.session_id AS head_blocker_session_id, head.session_id AS session_id,
head.blocking_session_id
    , head.wait_type, head.wait_time, head.wait_resource, head.statement_start_offset,
head.statement_end_offset
    , head.plan_handle, head.sql_handle, head.most_recent_sql_handle, 0 AS [Level]
    FROM cteHead AS head
    WHERE (head.blocking_session_id IS NULL OR head.blocking_session_id = 0)
    AND head.session_id IN (SELECT DISTINCT blocking_session_id FROM cteHead WHERE blocking_session_id !=
0)
    UNION ALL
    SELECT h.head_blocker_session_id, blocked.session_id, blocked.blocking_session_id, blocked.wait_type,
    blocked.wait_time, blocked.wait_resource, h.statement_start_offset, h.statement_end_offset,
    h.plan_handle, h.sql_handle, h.most_recent_sql_handle, [Level] + 1
    FROM cteHead AS blocked
    INNER JOIN cteBlockingHierarchy AS h ON h.session_id = blocked.blocking_session_id and
h.session_id!=blocked.session_id --avoid infinite recursion for latch type of blocking
    WHERE h.wait_type COLLATE Latin1_General_BIN NOT IN ('EXCHANGE', 'CXPACKET') or h.wait_type is null
    )
SELECT bh.*, txt.text AS blocker_query_or_most_recent_query
FROM cteBlockingHierarchy AS bh
OUTER APPLY sys.dm_exec_sql_text (ISNULL ([sql_handle], most_recent_sql_handle)) AS txt;
GO
```

```
SELECT [s_tst].[session_id],
[database_name] = DB_NAME (s_tdt.database_id),
[s_tdt].[database_transaction_begin_time],
[sql_text] = [s_est].[text]
FROM sys.dm_tran_database_transactions [s_tdt]
INNER JOIN sys.dm_tran_session_transactions [s_tst] ON [s_tst].[transaction_id] = [s_tdt].[transaction_id]
INNER JOIN sys.dm_exec_connections [s_ec] ON [s_ec].[session_id] = [s_tst].[session_id]
CROSS APPLY sys.dm_exec_sql_text ([s_ec].[most_recent_sql_handle]) AS [s_est];
GO
```

Links

Microsoft

Revision #0
Created 14 October 2025 18:54:29 by Admin
Updated 14 October 2025 18:59:26 by Admin