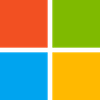
**Lesson Learned #19: How to obtain the deadlocks of your Azure SQL Database or Managed Instance?**

[Jose_Manuel_Jurado's avatar](https://techcommunity.microsoft.com/users/jose_manuel_jurado/247500)

[Jose\_Manuel\_Jurado](https://techcommunity.microsoft.com/users/jose_manuel_jurado/247500)

Microsoft

Mar 14, 2019

**First published on MSDN on Jan 21, 2017, Updated Dec 2022**

In Azure SQL Database we have already running an Extended Event that captures the deadlocks without any additionally action

for the customer side. In this post we are going to learn how to obtain the deadlocks, how to obtain the deadlock graphic and how to reproduce a deadlock for testing.

**Working with high volume of deadlocks**

Assuming your system experiences a high volume of deadlocks using the above technique might cause an issue.

When querying the system function it fetch all the XEL files at once

This process uses the memory clerk "XE Engine" - MEMORYCLERK\_SQLQUERYPLAN and it might impact your master database availability and eventually lead to availability issues with your managed instance / SQL Database.

Once fetching the data is completed, we need to go and parse the XML information for each deadlock event.

which is also an intensive resource consuming process.

Having that in mind we strongly suggest customers which have more than just a few deadlock events to avoid using this method

Instead, we suggest customers create XE session to track deadlock information on their database.

When setting customer XE session you may choose destination as Ring Buffer or blob storage to host the captured events.

**Validating how many deadlock your database experience**

To get the current number of deadlock events happening on your database please run the following query.

Depending on the number you make, take a decision about the next steps.

SELECT

#OfDeadlockEvents = format(sum(event\_count),'#,#')

FROM

sys.event\_log

WHERE

event\_type = 'deadlock'

**How to obtain the deadlocks**

You need to run the query against master database the following TSQL (no results will be shown if ran against user database )

WITH CTE AS (

SELECT

CAST(event\_data AS XML) AS [target\_data\_XML]

FROM

sys.fn\_xe\_telemetry\_blob\_target\_read\_file('dl', null, null, null)

)

SELECT

target\_data\_XML.value('(/event/@timestamp)[1]', 'DateTime2') AS Timestamp,

target\_data\_XML.query('/event/data[@name=''xml\_report'']/value/deadlock') AS deadlock\_xml,

target\_data\_XML.query('/event/data[@name=''database\_name'']/value').value('(/value)[1]', 'nvarchar(100)') AS db\_name

FROM CTE

**How to obtain the deadlock graph**

* + Copy the deadlock\_xml column results from the previous query and load into a text file. If more than one row is returned, you will want to do each row result separate.
  + Save the file as a '.xdl' extension, (e.g. deadlock.xdl) which can be viewed in tools such as SQL Server Management Studio as a deadlock report/graphic.

**How to reproduce a deadlock for testing**

-- ==================================

-- Create the tables.

-- ==================================

CREATE TABLE [dbo].[Person](

[Id] [int] IDENTITY(1,1) NOT NULL,

[FirstName] [varchar](20) NOT NULL,

[BusinessEntityID] [int] NOT NULL,

CONSTRAINT [PK\_Person] PRIMARY KEY CLUSTERED

([Id] ASC))

CREATE TABLE [dbo].[PersonPhone](

[PhoneNumber] [varchar](50) NOT NULL,

[BusinessEntityID] [int] NOT NULL,

CONSTRAINT [PK\_PersonPhone] PRIMARY KEY CLUSTERED

([PhoneNumber] ASC))

-- ==================================

-- Add some rows

-- ==================================

insert into [PersonPhone] values ( '999-555-1212',1)

insert into [Person] values( 'Chris',1)

-- ==================================

-- Open the SQL SERVER Management Studio and open two queries.

-- ==================================

-- ====================================

-- Execute this query using a query window for Deadlock Thread 1

-- ====================================

BEGIN TRANSACTION

UPDATE [PersonPhone] SET PhoneNumber = '999-555-1212' WHERE [BusinessEntityID] = 1

WAITFOR DELAY '00:00:50'

UPDATE [Person] SET [FirstName] = 'Chris' WHERE [BusinessEntityID] = 1

ROLLBACK TRANSACTION

-- ====================================

-- Very quickly execute this query using a query window for Deadlock Thread 2

-- ====================================

BEGIN TRANSACTION

UPDATE [Person] SET [FirstName] = 'Chris' WHERE [BusinessEntityID] = 1

UPDATE [PersonPhone] SET PhoneNumber = '999-555-1212' WHERE [BusinessEntityID] = 1

WAITFOR DELAY '00:00:10'

ROLLBACK TRANSACTION

**Or you can use the following Powershell script to generate more deadlocks**

[https://github.com/yochananrachamim/AzureSQL/edit/master/GenerateDummyDeadlocks/DeadlockWithPowerShellV2.ps1](https://raw.githubusercontent.com/yochananrachamim/AzureSQL/master/GenerateDummyDeadlocks/DeadlockWithPowerShell.ps1)