

Module 12

Collecting Performance Data



Overview

- The Default Trace
- Reading trace data using T-SQL
- ReadTrace
- SQLDiag
- SQL Nexus
- Deadlock analysis
- SQL Server 2012 Extended Events UI
- Extended Events advanced troubleshooting

The Default Trace

- Lightweight trace defined by Microsoft that collects events of interest in troubleshooting common problems with SQL Server
- The Books Online description:
“Default trace provides troubleshooting assistance to database administrators by ensuring that they have the log data necessary to diagnose problems the first time they occur.”
- If this was 100% true it would be fantastic, but a number of things can affect the usefulness of the Default Trace

Is the Default Trace Enabled?

- sys.configurations

```
-- Find out if Default Trace is Enabled
SELECT *
FROM sys.configurations
WHERE name = 'default trace enabled'
```

- sp_configure

```
-- Find out if Default Trace is Enabled
EXEC sys.sp_configure 'show advanced options', 1
RECONFIGURE WITH OVERRIDE
GO
EXEC sys.sp_configure 'default trace enabled'
GO
EXEC sys.sp_configure 'show advanced options', 0
RECONFIGURE WITH OVERRIDE
```

What Events are collected?

- sys.fn_trace_geteventinfo and sys.trace_events

```
-- Get the list of Events being collected
SELECT DISTINCT e.name
FROM sys.fn_trace_geteventinfo(1) t
JOIN sys.trace_events e ON t.eventID = e.trace_event_id
```

Default Trace Events

- Database
 - Data file auto grow
 - Data file auto shrink
 - Database mirroring status change
 - Log file auto grow
 - Log file auto shrink
- Errors and Warnings
 - Errorlog
 - Hash warning
 - Missing Column Statistics
 - Missing Join Predicate
 - Sort Warning
- Full-Text
 - FT Crawl Aborted
 - FT Crawl Started
 - FT Crawl Stopped
- Objects
 - Object Altered
 - Object Created
 - Object Deleted
- Security Audit
 - Audit Add DB user event
 - Audit Add login to server role event
 - Audit Add Member to DB role event
 - Audit Add Role event
 - Audit Add login event
 - Audit Backup/Restore event
 - Audit Change Database owner
 - Audit DBCC event
 - Audit Database Scope GDR event (Grant, Deny, Revoke)
 - Audit Login Change Property event
 - Audit Login Failed
 - Audit Login GDR event
 - Audit Schema Object GDR event
 - Audit Schema Object Take Ownership
 - Audit Server Starts and Stops
- Server
 - Server Memory Change

Common Questions

- Can I change the default trace?
 - No.. The default trace is not customizable in any way
- Can/Should I disable the default trace?
 - It depends...
 - If you think its impacting performance you are probably wrong and should not disable it
 - If it is just overlap of your own custom trace, then perhaps yes
- How far back in time does the default trace hold data?
 - Again It depends...
 - Every SQL Server is different and activity really impacts retention of the default trace data
 - Could be a few days to as long as a few months back

Disabling the Default Trace

- sp_configure

```
-- How to disabled the Default Trace
EXEC sys.sp_configure 'show advanced options', 1
RECONFIGURE WITH OVERRIDE
GO
EXEC sys.sp_configure 'default trace enabled', 0
RECONFIGURE WITH OVERRIDE
GO
EXEC sys.sp_configure 'show advanced options', 0
RECONFIGURE WITH OVERRIDE
GO
```

Getting information – What's there?

- Any DBCC event (SHRINKDATABASE, SHRINKFILE, UPDATEUSAGE, CHECKDB, etc)
- Who deployed an updated stored procedure?
- Auto-growth of log and data files
- Auto-shrink of log and data files
- Errors and exceptions raised
- Failed logins
- Service stops and starts

Getting Information – Continued

```
-- Get Current Filename
DECLARE @filename VARCHAR(MAX)
SELECT @filename = CAST(value AS VARCHAR(MAX))
FROM fn_trace_getinfo(DEFAULT)
WHERE property = 2
      AND value IS NOT NULL

SELECT gt.EventClass,
       te.Name AS EventName,
       gt.TEXTData,
       gt.NTUserName,
       gt.NTDomainName,
       gt.HostName,
       gt.ApplicationName,
       gt.LoginName,
       gt.SPID,
       gt.StartTime,
       gt.EndTime,
       gt.ObjectName,
       gt.DatabaseName,
       gt.FileName
FROM [fn_trace_gettable] (@fileName, DEFAULT) gt
JOIN sys.trace_events te ON gt.EventClass = te.trace_event_id
ORDER BY StartTime;
```

Demo

Default Trace



Analyzing Trace Data with TSQL

- Methodology
 - Load trace data into a table with `fn_trace_gettable()` for analysis with TSQL
- Problems
 - Adhoc workloads have identical statements with different inline values
 - Calling `sp_get_query_template` to normalize statements can be expensive

Using fn_trace_gettable()

```

SELECT
  CAST(TextData AS NVARCHAR(MAX)) AS TextData,
  SUM(Duration) AS Duration,
  SUM(READS) AS READS,
  SUM(Writes) AS Writes,
  SUM(CPU) AS CPU
FROM sys.fn_trace_gettable('C:\SQLdiagTuning\output\w2k3-SQL2K8_SQLDIAG__sp_trace.trc', 1)
WHERE EventClass IN (41, 45) --SQL:stmtcompleted, SP:stmtcompleted
GROUP BY CAST(TextData AS NVARCHAR(MAX));

```

	Duration	Reads	Writes	CPU
	0	0	0	0
l.SalesOrderID = soh.SalesOrderID WHERE CarrierTrackingNumber = N'4911-403C-28'	204100	3444	0	187
l.SalesOrderID = soh.SalesOrderID WHERE CarrierTrackingNumber = N'4911-403C-84'	208983	3444	0	203
l.SalesOrderID = soh.SalesOrderID WHERE CarrierTrackingNumber = N'4911-403C-98'	199217	3606	0	172
l.SalesOrderID = soh.SalesOrderID WHERE CarrierTrackingNumber = N'4911-403C-99'	1092772	3444	0	173
tureWorks.Sales.SalesOrderDetail sod ON sod.SalesOrderID = soh.SalesOrderID ...	6505858	3849	0	298

Demo

Using fn_trace_gettable()

Analyzing Trace Data with ReadTrace from RML Utilities

- Methodology
 - Uses ReadTrace from RML Utilities to parameterize adhoc workloads for analysis
 - Trace data is loaded into a SQL database and pre-aggregated for consumption by RML Utilities Reporter application
- Problems
 - Requires filtering of trace data to exclude replication operations
 - MARS support is limited at best

ReadTrace Common Options

- -I – (capital ‘i’) file name of the starting input file.
- -i – indicates the TRC files are inside a CAB/ZIP file
- -S – SQL Server to connect to
- -d – database to load the data into
- -E – use Integrated Security to connect
- -U – SQL login to connect
- -P – password for SQL Login specified with -U

Using ReadTrace

```

C:\Program Files\Microsoft Corporation\RMLUtils>readtrace -IC:\SQLDiagTuning\Output\M2K3-SQL2K8_SQLDIAG__sp_trace.trc

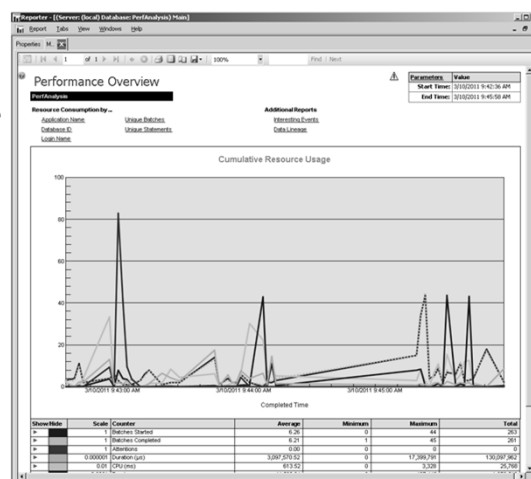
03/10/11 10:40:49.445 [0X00000FA0] Parse errors: 0
03/10/11 10:40:49.555 [0X00000FA0] Table ReadTrace.tblUniqueBatches: loaded ~87 rows
03/10/11 10:40:49.586 [0X00000FA0] Table ReadTrace.tblUniqueStatements: loaded ~634 rows
03/10/11 10:40:49.602 [0X00000FA0] Table ReadTrace.tblUniquePlans: loaded ~253 rows
03/10/11 10:40:49.617 [0X00000FA0] Table ReadTrace.tblUniquePlanRows: loaded ~2769 rows
03/10/11 10:40:49.633 [0X00000FA0] Table ReadTrace.tblBatches: loaded ~263 rows
03/10/11 10:40:49.680 [0X00000FA0] Table ReadTrace.tblStatements: loaded ~7494 rows
03/10/11 10:40:49.695 [0X00000FA0] Table ReadTrace.tblPlans: loaded ~2089 rows
03/10/11 10:40:49.727 [0X00000FA0] Table ReadTrace.tblPlanRows: loaded ~14998 rows
03/10/11 10:40:49.742 [0X00000FA0] Table ReadTrace.tblInterestingEvents: loaded ~134 rows
03/10/11 10:40:49.758 [0X00000FA0] Table ReadTrace.tblConnections: loaded ~156 rows
03/10/11 10:40:49.758 [0X00000FA0] WARNING: One or more warning conditions exist that may affect the quality of the analysis data. See PerfAnalysis.ReadTrace.tblWarnings table and the ReadTrace log for complete details.
03/10/11 10:40:49.774 [0X00000FA0] Indexing tables...
03/10/11 10:40:51.117 [0X00000FA0] Doing post-load data cleanup...
03/10/11 10:40:51.649 [0X00000FA0] Computing partial aggregates...
03/10/11 10:40:52.570 [0X00000FA0] Data load completed.
03/10/11 10:40:52.570 [0X00000FA0] Using execution path: C:\Program Files\Microsoft Corporation\RMLUtils
03/10/11 10:40:52.570 [0X00000FA0] Launching [C:\Program Files\Microsoft Corporation\RMLUtils\Reporter.exe]
03/10/11 10:40:52.820 [0X00000FA0]
*****
* ReadTrace encountered one or more WARNINGS. A warning condition typically *
* continues processing with reduced functionality, but the ReadTrace output *
* may be adversely affected. Review the log file for details. *
*****
03/10/11 10:40:52.867 [0X00000FA0]
C:\Program Files\Microsoft Corporation\RMLUtils>
  
```

©SQLskills.com
SQLskills Immersion Event

17

RML Utilities Reporter

- Shows aggregate query resource usage
- Click through reports allow drill down to specific statements
- Opens automatically after ReadTrace loads data



©SQLskills.com
SQLskills Immersion Event

18

Demo

Using ReadTrace



Using SQLDiag to Collect Data

- SQLDiag is a data capture utility for collecting diagnostic data from SQL Server
 - PerfMon logs
 - SQL Trace files
 - Windows Event Logs
 - SQL Server Error Logs
 - Msinfo32 information
- Installed by default in SQL Server 2005/2008
 - C:\Program Files\Microsoft SQL Server\[90|100]\Tools\Binn\sqldiag.exe
- This is what we use to drive our remote health checks

Running SQLDiag

```

C:\Program Files\Microsoft SQL Server\100\Tools\Binn>sqldiag
2011/03/07 20:26:54.62 SQLDIAG Collector version
2011/03/07 20:26:54.62 SQLDIAG

IMPORTANT: Please wait until you see "Collection started" before attempting to
reproduce your issue

2011/03/07 20:26:54.62 SQLDIAG Output path: C:\Program Files\Microsoft SQL Serve
r\100\Tools\Binn\SQLDIAG\
2011/03/07 20:26:54.75 SQLDIAG Collecting from 1 logical machine(s)
2011/03/07 20:26:54.76 W2K3-SQL2K8\* SQL Server version: 10
2011/03/07 20:26:54.76 W2K3-SQL2K8\* Machine name: W2K3-SQL2K8 (this machine)
2011/03/07 20:26:54.76 W2K3-SQL2K8\* Target machine is not a cluster
2011/03/07 20:26:54.76 W2K3-SQL2K8\* Instance: (Default) (32-bit)
2011/03/07 20:26:56.75 SQLDIAG Initialization starting...
2011/03/07 20:27:01.65 W2K3-SQL2K8\* MsInfo: Get MSINF032
2011/03/07 20:27:01.72 W2K3-SQL2K8\* MsInfo: Get default traces
2011/03/07 20:27:01.81 W2K3-SQL2K8\* MsInfo: Get SQLDumper log
2011/03/07 20:27:02.79 W2K3-SQL2K8\* Collecting diagnostic data
2011/03/07 20:27:02.79 SQLDIAG Initialization complete

2011/03/07 20:27:03.79 SQLDIAG Collection started. Press Ctrl+C to stop.

2011/03/07 20:27:09.22 SQLDIAG Ctrl+C pressed. Shutting down the collector
2011/03/07 20:27:12.79 W2K3-SQL2K8\* Shutting down the collector
2011/03/07 20:27:12.79 W2K3-SQL2K8\* Getting SQLConfig report(s)

2011/03/07 20:27:50.81 SQLDIAG Ending data collection. Please wait while the pr
ocess shuts down and files are compressed (this may take several minutes)

2011/03/07 20:27:50.84 SQLDIAG Collection complete. Collector exiting

C:\Program Files\Microsoft SQL Server\100\Tools\Binn>

```

©SQLskills.com
SQLskills Immersion Event

21

SQL

Demo

Using SQLDiag

SQL skills

SQLDiag Configuration

- Utilizes XML configuration file
 - Default file created on first execution
- Can be edited using any text edit (e.g. Notepad) or the Business Intelligence Development Studio (BIDS) environment from Visual Studio
 - Editing with BIDS simplifies editing by making subsections collapsible minimizing the viewable XML
- Contains machine and instance level collectors
- Customizations must be saved as a new file name and utilized with the /I command line switch
 - The default SQLDiag.xml file is overwritten at SQLDiag startup

Machine Collectors

- Collect information from Windows Server
- EventLogCollector
 - Collects Windows Event Logs for analysis
- PerfmonCollector
 - Collects Perfmon counters for analysis

Machine Collectors:

EventLogCollector

- Disabled by default
- Collects Windows Event Logs
 - **startup** sets the collection to run when SQLDiag starts
 - **shutdown** sets the collection to run when SQLDiag shuts down
- Individual logs can be enabled or disabled as needed

```
<EventlogCollector enabled="false" startup="false" shutdown="true" >
  <Eventlogs>
    <EventlogType name="Application" enabled="true"/>
    <EventlogType name="Security" enabled="true"/>
    <EventlogType name="System" enabled="true"/>
  </Eventlogs>
</EventlogCollector>
```

Machine Collectors:

PerfmonCollector

- Disabled by default
- Collects performance counter data
 - **pollinginterval** sets the frequency of collection in seconds
 - **maxfilesize** sets the maximum file size in megabytes(MB)
- Individual objects and counters can be enabled or disabled

```
<PerfmonCollector enabled="false" pollinginterval="5" maxfilesize="256">
  <PerfmonCounters>
    <PerfmonObject name="\MSFTESQL:Service" enabled="true">
      <PerfmonCounter name="\*" enabled="true" />
      <PerfmonCounter name="\Heartbeats" enabled="true" />
      <PerfmonCounter name="\Total wordbreakers" enabled="true" />
      <PerfmonCounter name="\Total stemmers" enabled="true" />
    </PerfmonObject>
  </PerfmonCounters>
</PerfmonCollector>
```

Instance Collectors

- Collect information from SQL Server Instances
- Default collects information for all SQL Server Instances installed on a server
 - Can be targeted to a specific instance or multiple instances by modifying the XML configuration
- SQLDiagCollector
- BlockingCollector
- ProfilerCollector
- CustomDiagnostics

Instance Collectors:

SQLDiagCollector

- Enabled by default
- Collects DMV and system information using `sp_sqldiag[ver]`
 - **startup** sets the collection to run when SQLDiag starts
 - **shutdown** sets the collection to run when SQLDiag shuts down
- Output
 - `<ServerName>__sp_sqldiag_Shutdown.OUT`

```
<Collectors>
  <SqlDiagCollector enabled="true" startup="false" shutdown="true" />
```

Instance Collectors:

BlockingCollector

- Disabled by default
- Collects additional blocking information using the blocked process report SQL Trace Event
 - **pollinginterval** sets the blocked process threshold sp_configure option in seconds
 - **maxfilesize** sets the maximum trace file size in megabytes(MB)
 - **filecount** sets the maximum number of files to collect
- Output:
 - <ServerName>_SQLDIAG__sp_trace_blk.trc

```
<BlockingCollector enabled="false" pollinginterval="5" maxfilesize="350" filecount="1"/>
```

Instance Collectors:

ProfilerCollector

- Disabled by default
- Collects SQL Trace information
 - **template** sets the trace template to use.
 - **pollinginterval** sets the blocked process threshold sp_configure option in seconds
 - **maxfilesize** sets the maximum trace file size in megabytes(MB)
 - **filecount** sets the maximum number of files to collect

```
<ProfilerCollector enabled="false" template="_GeneralPerformance90.xml" pollinginterval="5" maxfilesize="350" filecount="1">
  <Events>
    <EventType name="Broker">
      <Event id="124" name="Broker:Conversation" enabled="false" description="Reports the progress of a Service Broker"
      <Event id="136" name="Broker:Conversation Group" enabled="false" description="Occurs when Service Broker creates"
      <Event id="138" name="Broker:Connection" enabled="false" description="Reports the status of a transport connecti
```

Instance Collectors:

CustomDiagnostics

- Enabled by default
- Collects MSINFO32 output
 - <ServerName>_MSINFO32.TXT
- Copies the following to the output path
 - Default Trace files
 - Any SQL Dump files in the Log path

```
<CustomDiagnostics>
  <CustomGroup name="msinfo" enabled="true" />
  <CustomTask enabled="true" groupname="MsInfo" taskname="Get MSINFO32" type="Utility"
  <CustomTask enabled="true" groupname="MsInfo" taskname="Get default traces" type="Copy"
  <CustomTask enabled="true" groupname="MsInfo" taskname="Get SQLDumper log" type="Copy"
</CustomDiagnostics>
```

Creating a Custom Collector

- Custom collectors can be created and added to SQLDiag by adding to the CustomDiagnostics section of the XML configuration
- Custom collector types:
 - TSQL Command
 - TSQL Script
 - Utility (.cmd files or command line strings)
 - VB Script
 - Copy File
 - Registry Query
- Custom collectors can be grouped using a CustomGroup specification

Creating a Custom Collector (2)

- Create an Event Session at SQLDiag startup
- Collect the data from the Event Session at SQLDiag shutdown

```
<CustomGroup name="SQLskills Tasks" enabled="true" />
<CustomTask enabled="true"
  groupname="SQLskills Tasks"
  taskname="SQLskills Create Event Session"
  type="TSQL_Script"
  point="Startup"
  wait="No"
  cmd="SQLskills_Create_Event_Session.sql"/>
<CustomTask enabled="true"
  groupname="SQLskills Tasks"
  taskname="SQLskills Collect Event Session Data"
  type="TSQL_Script"
  point="Shutdown"
  wait="OnlyOnShutdown"
  cmd="SQLskills_Collect_Event_Session_Data.sql"/>
```

©SQLskills.com
SQLskills Immersion Event

33

SQL

Command Line Options

- /I cfgfile – sets the configuration file to use
- /O outputpath – sets the output location
- /N # - folder management
 - 1=overwrite, 2=rename
- /X – snapshot mode (collect diagnostics then exit immediately)
- /C # – sets file compression type:
 - 0=none, 1=NTFS, 2=CAB
- /B YYYYMMDD_HH:MM:SS – sets a start time
- /E YYYYMMDD_HH:MM:SS – sets an end time

©SQLskills.com
SQLskills Immersion Event

34

SQL

Installing as a Service

- /R – registers SQLDiag as a service
- /A – sets an application name
- /U – removes specified SQLDiag service
- All options specified when the service registers are maintained when the service starts
 - E.g. (sqldiag /R /A SQLDiagTuning /I C:\SQLDiagTuning\SQLDiagTuning.XML /O C:\SQLDiagTuning\Output /N 2 /C 2)
- To control service:
 - sqldiag START /A SQLDiagTuning
 - sqldiag STOP /A SQLDiagTuning

Demo

Customizing SQLDiag configuration

SQLDiag: Perfstats Script

- Part of the SQLNexus project
- Provides multiple SQLDiag configurations for extended collection.
- Adds collectors for DMV data
- Captures additional blocking information

SQL Nexus

- Analysis tool originally developed by Ken Henderson for use by Product Support Services to simplify analysis of the information collected by PSSDiag.
- Released to the community as a open source project on Codeplex
- Offline analysis of data previously collected with SQLDiag and Perfstats script only (NOT a real-time monitoring tool)

SQL Nexus Features

- Simplified data loading
 - SQL Trace files, T-SQL script outputs, and Performance Monitor logs
- Simplified reporting
 - Includes five SSRS reports for analyzing data
- Aggregates trace data
 - Uses ReadTrace to aggregate data to find the top most expensive queries
- Analyzes wait stats
 - Provides visual representation of resource contention
- Extensible
 - Custom reports and importers can be built and added to the application

©SQLskills.com
SQLskills Immersion Event

39

SQL

SQL Nexus Requirements

- Requires SQL Server 2005 or higher database to import data into
- Requires RML Utilities (ReadTrace)
- Microsoft Report Viewer Redistributable 2008 SP1

©SQLskills.com
SQLskills Immersion Event

40

SQL

Using SQL Nexus

- Connect to a non-production SQL Server when the application loads
- A sqlnexus database will be created when the application first connects
- Click the Import link in the Data pane to load the data into the database

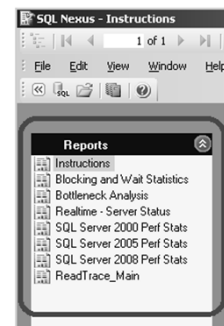


©SQLskills.com
SQLskills Immersion Event

41

Built in Reports

- Blocking and Wait Statistics
 - Provides details about the top wait types and blocking chains
- Bottleneck Analysis
 - Provides details about the top resource bottlenecks during the collection period
- SQL 2000/2005/2008 Perf Stats
 - Version specific reports of the data collected by the PerfStats script
- ReadTrace Report
 - Aggregate trace information for TOP Unique Batches and resource usage by Query Template



©SQLskills.com
SQLskills Immersion Event

42

Demo

Using SQL Nexus



Overview

- The Default Trace
- Reading trace data using T-SQL
- ReadTrace
- SQLDiag
- SQL Nexus
- Deadlock analysis
- SQL Server 2012 Extended Events UI
- Extended Events advanced troubleshooting

Everything you need to know to prevent Deadlocks

**Use WITH(NOLOCK)
for all SELECT statements**

Everything you DON'T need to know to prevent Deadlocks



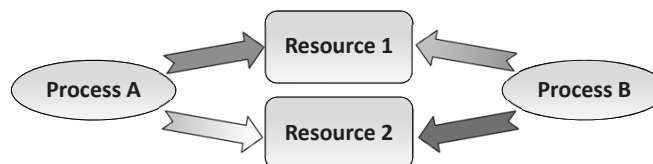
This is not the Answer!

Deadlock Analysis

- What is a deadlock (reminder)
- Collecting deadlock graphs
- Anatomy of a deadlock
- Reading deadlock graphs
- Resolving deadlocks

What is a Deadlock

A condition when two or more processes are holding locks on resources and are each waiting for the other to release it's locks to progress, or where two or more processes are waiting for locks on resources in a circular chain



SQL Server and Deadlocks

- Lock Monitor thread
- Error 1205
- Victim selection
 - Deadlock priority
 - Cost of rollback

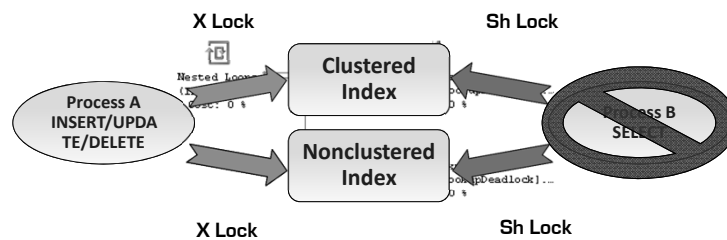
Collecting Deadlock Graphs

- Trace Flags (1222, 1205, 1204)
- SQL Trace / Profiler
- Event Notifications / WMI
- Extended Events

Diagram illustrating a query plan with nested loops and index seek operations:

- Nested Loops (Inner Join)**
Cost: 0 %
 - Index Seek**
[tempdb].[dbo].[KeyLookupDeadlock]...
Cost: 50 %
 - Key Lookup**
[tempdb].[dbo].[KeyLookupDeadlock]...
Cost: 50 %

➤ Bookmark lookup deadlock



Demo

Bookmark Lookup Deadlock

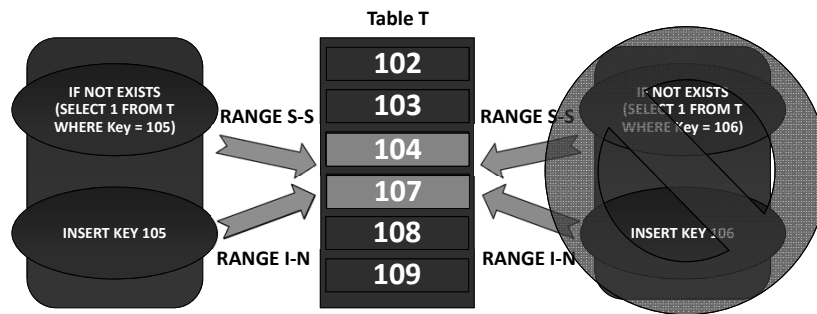


Reading Deadlock Graphs

- deadlock victim
- process-list
 - process id, waitresource, spid, priority, isolationlevel, inputbuf
- resource-list
 - owner-list, waiter-list, objectname, indexname

Anatomy of a Deadlock

➤ Serializable lock conversion deadlock



©SQLskills.com
SQLskills Immersion Event

55

SQL

Demo

Serializable Lock Conversion Deadlock

SQL
skills

Resolving Deadlocks



This is not the Answer!

Resolving Deadlocks

- Change indexing to cover queries
- Enable Read Committed Snapshot Isolation
 - Writers won't block readers
 - Reads won't block writers
- Change isolation Level
- Use locking hints to force specific lock types to prevent lock conversion
- Change application code to handle 1205 errors, log them and then resubmit the transaction to SQL Server again

Overview

- The Default Trace
- Reading trace data using T-SQL
- ReadTrace
- SQLDiag
- SQL Nexus
- Deadlock analysis
- SQL Server 2012 Extended Events UI
- Extended Events advanced troubleshooting

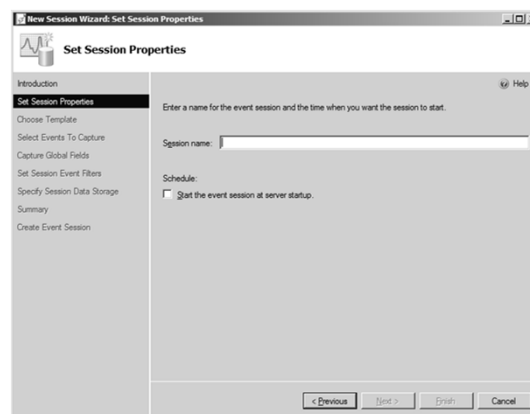
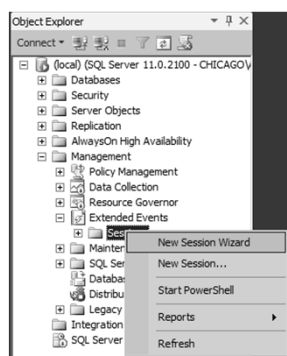
©SQLskills.com
SQLskills Immersion Event

59

SQL

Extended Events UI

- New Session Wizard
 - Create new sessions rapidly from templates or by using a minimal set of configurable options



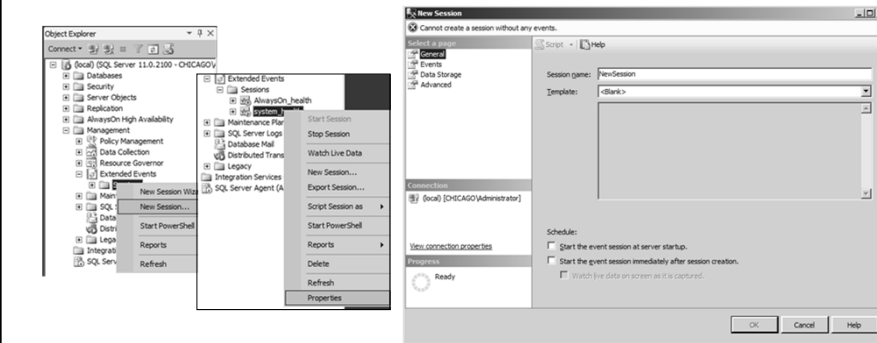
©SQLskills.com
SQLskills Immersion Event

60

SQL

Extended Events UI

- New Session Dialog
 - Used for advanced session creation using all available options and targets
 - Used for modifying existing event sessions



©SQLskills.com
SQLskills Immersion Event

61

Live Data Viewer

- Reads a live stream of event buffers from an Extended Events session on a server
 - Provides a "Profiler like" view of the data being generated
- Will disconnect if it can't keep up with event generation to prevent impact to the instance
 - Boosting MAX_MEMORY option for the instance and/or changing memory partitioning may prevent disconnects from occurring

Displaying 150 Events

name	timestamp
sql_batch_completed	2012-02-27 16:30:44.4455716
error_reported	2012-02-27 16:30:44.4836575
error_reported	2012-02-27 16:30:44.4836575
error_reported	2012-02-27 16:30:44.4836575
sql_batch_completed	2012-02-27 16:30:44.4846341
sql_batch_completed	2012-02-27 16:30:44.4846341
sql_statement_completed	2012-02-27 16:30:44.4855403
sql_statement_completed	2012-02-27 16:30:44.4855403
sql_batch_completed	2012-02-27 16:30:44.4855403
sql_statement_completed	2012-02-27 16:30:44.4963528
sql_statement_completed	2012-02-27 16:30:44.4963528
sql_batch_completed	2012-02-27 16:30:44.4963528

Event: error_reported (2012-02-27 16:30:30.7026587)

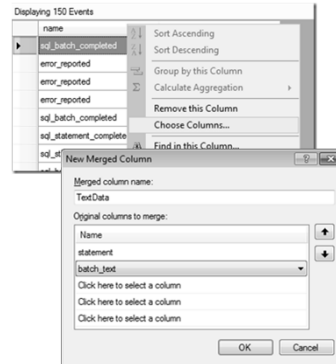
Field	Value
attach_activity_id	3384E93C-AA4D-436A-AE96-A8168767811
attach_activity_id	1
attach_activity_id	29680246-C486-4168-8006-15707AC4248
attach_activity_id	0
server	SERVER
client_app_name	Microsoft SQL Server Management Studio - Query
database_id	10
destination	USER
error_number	5701

©SQLskills.com
SQLskills Immersion Event

62

Data Viewer Customization

- Default view is limited but columns can be added using the column chooser
 - New Merged columns can be created to track similar information provided by different events in a single column in the grid view
- Display settings can be saved for reuse with later Event Sessions or event_file processing



Grouping and Aggregating Data

- Simplified complex data analysis of disconnected data within the data viewer for Extended Events
 - Group by any available columns in the current view
 - Aggregate one or many columns based on the current grouping settings to find trends of interest
- Groups can be expanded for detail level viewing



name	timestamp	cpu_time	duration	logical_reads
query_hash: 2019622035648856120 (24)		SUM: 180000	SUM: 142563	SUM: 2494
query_hash: 11466592891013851518 (24)		SUM: 48000	SUM: 138661	SUM: 2478
query_hash: 9126238797566703259 (19)		SUM: 983000	SUM: 937490	SUM: 23698
sql_statement_completed	2012-03-19 12:25:47.5438887	47000	42968	1246
sql_statement_completed	2012-03-19 12:25:58.7541428	47000	40988	1246
sql_statement_completed	2012-03-19 12:25:58.7541428	62000	52734	1246
sql_statement_completed	2012-03-19 12:25:58.8631273	62000	52734	1246

Demo

Extended Events UI



Extended Events Advanced Troubleshooting

- Tracking bad page splits
- Tracking tempdb allocation bitmap contention
- Tracking error handling in TSQL code
- Walking the tsql_stack to discover what caused a procedure to execute
- Tracking auto parameterized statements
- Debugging lock escalation issues
- Tracking database and object usage

Review

- The Default Trace
- Reading trace data using T-SQL
- ReadTrace
- SQLDiag
- SQL Nexus
- Deadlock analysis
- SQL Server 2012 Extended Events UI
- Extended Events advanced troubleshooting

Resources

- Bart Duncan's Blog
<http://blogs.msdn.com/b/bartd/archive/tags/sql+deadlocks/>
- Deadlock posts
http://sqlblog.com/blogs/jonathan_kehayias/archive/tags/Deadlock/default.aspx
- Jon's posts on SQL Server Central
http://www.sqlservercentral.com/Authors/Articles/Jonathan_Kehayias/244648/
- Jon's videos on SQL Share
<http://www.sqlshare.com/Search.aspx?terms=deadlock>