

SQLskills Immersion Event IE2: Performance Tuning

Module 6: Extended Events

Paul S. Randal
Paul@SQLskills.com



Overview

- Extended Events core concepts
- Execution lifecycle and dispatching
- DDL and event session basics
- Event targets
- Usage scenario demos
- Appendix

What are Extended Events?

- Advanced event-collection infrastructure introduced in SQL Server 2008 and provided by SQLOS
- Highly-flexible implementation allows complex configurations for event collection that simplify problem identification
- Able to perform performance troubleshooting that's not possible any other way:
 - Identify stored procedures that exceed previous max duration, CPU, or I/O values
 - Identify statement timeouts/attention events
 - Capturing the first N executions of an event
 - Using the plan_handle and tsql_stack to capture execution plans and statement text
 - Examine the details of proportional fill
 - Watch page splits occurring

Demo

Intro demo: wait statistics

Core Concepts: Packages

- Packages are loaded by individual modules at run-time
- Packages are containers that define the available Extended Events objects and their definitions
- Packages are not a functional boundary of usage
 - Objects from one package can be used with objects from another package
- Example modules: sqlservr.exe, sqllos.dll

Core Concepts: Events

- Events correspond to well-known points in the code
 - E.g. a T-SQL statement finished executing; a deadlock occurred
- Events are stored as XML, as this is extensible
- Events deliver a basic payload of information
 - Payload is defined by a schema of information
 - Events may contain optional (customizable) data elements that are only collected when specified
 - Events will always return all non-customizable data elements
- Events are defined using the Event Tracing for Windows (ETW) model to allow integration with ETW

Core Concepts: Predicates

- **Predicates are Boolean expressions that define the conditions required for an event to actually fire**
- **Predicates support short-circuit evaluation**
 - First false evaluation prevents the event from firing
- **Predicates can use basic arithmetic operators, or textual comparators for more complex expressions**
 - E.g. Bitmask checking, greater than last max value, less than last min value, and divides evenly by int64 can perform event sampling not possible with basic Boolean expressions
- **Predicates can operate on global or event payload data**
- **Predicates can store state (this is pretty cool)**
 - E.g. count the number of times an event fires and only publish it every N times

Core Concepts: Actions

- **Actions only execute after predicate evaluation determines the event will fire**
- **Actions perform additional operations when an event fires**
 - E.g. collecting additional state data, or performing a memory dump
- **Actions execute synchronously on thread that fired the event.**
- **Any action may be used with any event, but state data may not be available at the point in code where an event fires**

Core Concepts: Targets

- **Event consumers**
 - Process single events or a buffer full of events
- **Both synchronous or asynchronous targets exist**
- **Basic targets collect raw event data**
 - Event file
 - Ring buffer
- **Aggregating targets aggregate data based on criteria**
 - Event bucketizer (histogram)
 - Event counter
 - Event pairing (matching events)
- **ETW target allows end-to-end tracing with Windows Kernel**

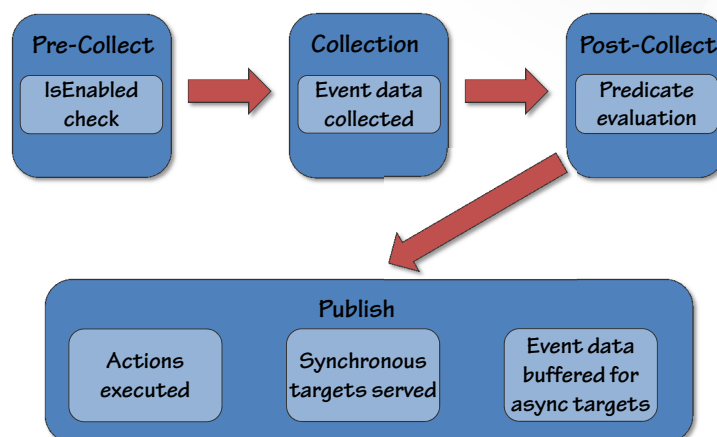
Core Concepts: Types and Maps

- **Types define the type of data for an event column, action, or global predicate source**
- **Types are contained within packages**
- **Maps are used as types in the Extended Events Engine**
 - Defining a predicate on a map based event column requires using the map_key value from the sys.dm_xe_map_values DMV
- **Maps provide a lookup to convert a value from one format to another**
 - E.g. a numeric lock mode to a string describing the lock mode – it's easier to understand mode = "X" than mode = 5

Core Concepts: Event Sessions

- An event sessions is a collection of events, their configured actions and predicates, and targets to consume the event data
- Event sessions are the functional boundary for configuration
 - Multiple sessions can have the same events with different predicates and targets
- Event sessions have buffers in memory assigned to them to store data generated by events before it's dispatched to the targets by the dispatchers
- Event sessions can be configured to track causality of an event firing (i.e. linking events together) and to start automatically at server startup

Event Execution Lifecycle



Dispatching

- Dispatching events to the asynchronous targets is handled by the dispatcher pool in the Extended Events Engine (in SQLOS)
- Dispatch occurs under two conditions
 - The memory buffer becomes full
 - The event data in the buffer exceeds the event session's MAX_DISPATCH_LATENCY configuration option

Management DDL

- **CREATE EVENT SESSION**
 - Creates a new event session based on the events, actions, predicates, targets, and session options provided
 - All event sessions are created in a stopped state
- **ALTER EVENT SESSION**
 - Add or remove events and targets from an event session
 - Change session configuration options for a stopped event session
 - Alter the state of an event session to start or stop
- **DROP EVENT SESSION**
 - Removes an event session from the system entirely
 - Memory resident targets are not available after a event session is dropped

Creating an Event Session

```
CREATE EVENT SESSION [long_duration_statements]
ON SERVER
ADD EVENT sqlserver.sql_statement_completed
( ACTION
  ( sqlserver.tsql_stack,
    sqlserver.sql_text )
WHERE
  ( sqlserver.is_system = 1
    AND duration > 5000000 ))
ADD TARGET package0.ring_buffer
  (SET MAX_MEMORY = 1024)
WITH (MAX_DISPATCH_LATENCY = 15 SECONDS)
```



15

© SQLskills. All rights reserved.
<http://www.SQLskills.com>

Working with Event Sessions

```
ALTER EVENT SESSION [long_duration_statements]
ON SERVER
STATE=START | STOP

ALTER EVENT SESSION [long_duration_statements]
ON SERVER
ADD EVENT sqlserver.sql_statement_starting
ADD TARGET package0.asynchronous_file_target
( SET filename =
  'C:\SQLskills\long_duration_statements.xel',
  metadatafile =
  'C:\SQLskills\long_duration_statements.mta')
```



16

© SQLskills. All rights reserved.
<http://www.SQLskills.com>

Data-Adding Actions on Events

- **Allow side effects to occur when an event fires**
 - Performing a memory dump of a single thread
 - Performing a memory dump of all threads
 - Inserting a debug break (this really does what it says it does!)
- **Large actions like `sql_text` and `sql_context` may require additional consideration for event sizing during session configuration to prevent event loss**
- **Actions execute synchronously on the firing thread and should be used only where they actually add benefit to the event data**
- **Don't add actions to do event correlation use causality tracking instead**

Event Session Options

- **TRACK_CAUSALITY**
 - Attaches a GUID and sequence number to events for correlation of what events led to other events
 - E.g. watching all the page splits that occur from a single insert/update
- **STARTUP_STATE**
 - Start the event session automatically with SQL Server
- **MAX_DISPATCH_LATENCY**
 - Dispatch events to targets faster
 - May impact the dispatcher pool if multiple event sessions exist
- **EVENT_RETENTION_MODE**
 - Determines whether single events, entire buffers, or no events can be lost by the event session
 - No event loss can significantly impact performance

asynchronous_file_target Target

- Similar to the trace file in SQL Trace, this target collects event data for long-term detailed analysis
- Event data has the same XML schema as the ring_buffer target
- Target data must be queried through the SQL Server database engine, there is no external API available for reading the file
- The metadata file simply describes the structure of the data file, not the events themselves
- Renamed to the 'event_file target' in SQL Server 2012+

ring_buffer Target

- General purpose in-memory target that performs FIFO collection of events
- DMV limitations in SQL Server 2008 may result in unreadable XML from sys.dm_xe_session_targets
- Ideal target for small data sets where event loss is acceptable
- Options
 - MAX_MEMORY: the maximum amount of memory in KB to be used, so older events are dropped when this limit is reached
 - OCCURRENCE: sets the preferred number of events of each type to keep

a/synchronous_bucketizer Targets

- Specialized in-memory targets that collect events into buckets to simplify analysis of event frequency
- Does not store actual event data, only the count of occurrences
- Renamed to 'histogram' target in SQL Server 2012+

pair_matching Target

- Specialized target that matches events based on specified criteria and discards the matched pairs, leaving unmatched events only in the data
- Careful selection of the pairing criteria has to be made or invalid event pairings will occur resulting in incorrect analysis
 - E.g. lock_acquired and lock_released are not fired the same number of times when lock escalation occurs
- Troubleshooting incorrect matching should be done using TRACK_CAUSALITY and the ring_buffer or file target
- Same output XML as the ring_buffer for event data

synchronous_event_counter Target

- Counts how many events occurred for the event session
- Use this when defining an event session for an unknown workload to determine how many events you can expect based on your event sessions definition as a part of planning the appropriate targets to use for the session
- Renamed to 'event_counter' in SQL Server 2012+

Demo

Example usage scenarios

Review

- Extended Events core concepts
- Execution lifecycle and dispatching
- DDL and event session basics
- Event targets
- Usage scenario demos
- Appendix

Appendix: Metadata Views

- **sys.dm_xe_packages**
 - Contains a entry for each of the packages registered in the Extended Events Engine
 - Each package will have a unique guid, which is used to map its objects to it
- **sys.dm_xe_objects**
 - Contains information about the objects (events, actions, predicates, targets, types and maps) available in the packages registered in the Extended Events Engine
 - Objects have the package_guid of the package that loaded them.
 - The object_type column determines the type of object

Appendix: Metadata Views

- **sys.dm_xe_object_columns**
 - Contains information about the columns, or data elements, that exist for a specific object
 - Readonly: additional system metadata about an event, that allows the integration with Event Tracing for Windows
 - Data: the data elements that are returned by default when the event fires
 - Customizable: control collection of additional data elements in the event payload that have a higher cost to collect
- **sys.dm_xe_map_values**
 - Contains key/value pairs for each of the maps defined in the system
 - Maps are linked by their object_package_guid to the specific package that created them

Appendix: Session Definition Catalog Views

- **sys.server_event_sessions**
 - Contains the name and session level options for each event session that exist inside of the Extended Events Engine
- **sys.server_event_session_events**
 - Contains information about the events and predicates that are a part of an event session
- **sys.server_event_session_actions**
 - Contains one row for each action for each event in an event session

Appendix: Session Definition Catalog Views

- **sys.server_event_session_targets**
 - Contains one row for each of the configured targets that are defined for an event session
- **sys.server_event_session_fields**
 - Contains one row for each of the configured options for each target defined for an event session

Appendix: Active Session DMVs

- **sys.dm_xe_sessions**
 - Contains the event session name, memory buffer configuration, event loss information, and date/time the event session was started for each active event session (STATE=START) in the SQL Server Instance
- **sys.dm_xe_session_events**
 - Contains information about each of the events and the event predicate for events defined in an active event session
- **sys.dm_xe_session_event_actions**
 - Contains one row for each action that is defined on an event in an active event session

Appendix: Active Session DMVs

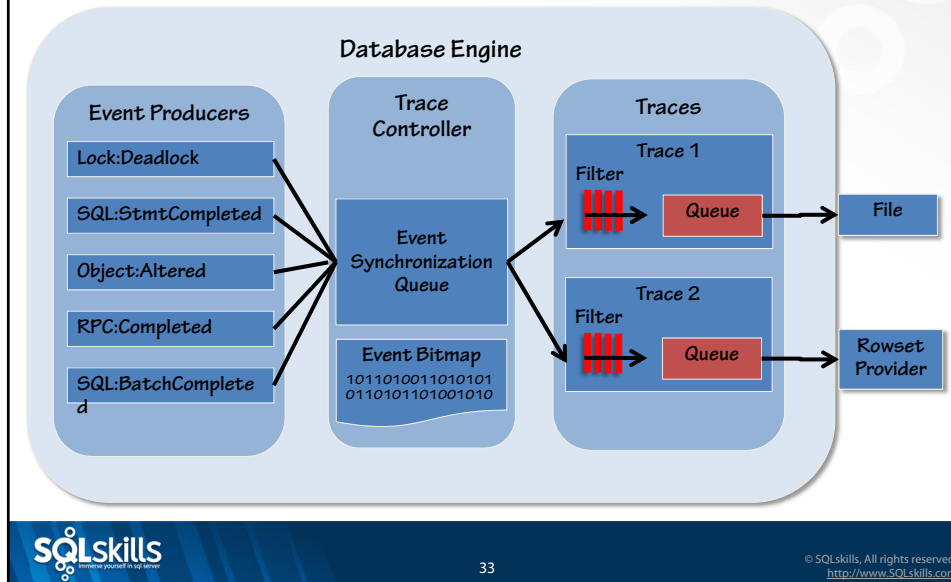
- **sys.dm_xe_session_targets**
 - Contains the name, type of target, execution statistics and an XML column for each target that exists for an active event session
 - The data for memory resident targets is in the xml column and for persisted targets this will contain execution statistics
- **sys.dm_xe_session_object_columns**
 - Contains information about the configuration options of each of the targets, as well as information about the customizable columns for events in the event session

Appendix: XEvents vs. SQL Trace

SQL Trace

- **Events fire based on server wide bitmap**
- **All data buffered to trace controller and then sent to the traces**
 - Entire Events may be discarded by a trace at filtering
 - Excess columns will be discarded from the dataset if not part of the trace definition
- **Rowset provider and trace file collect all events and do not provide aggregation**

Appendix: SQL Trace Architecture

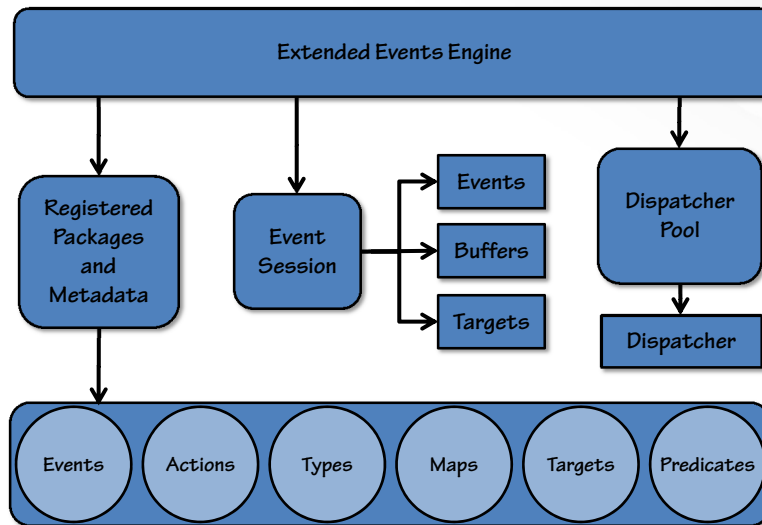


Appendix: XEvents vs. SQL Trace

Extended Events

- Events fire based on session level configuration
- Events only fire if predicate evaluation succeeds
 - Removing unnecessary processing
- Additional data only collected when event fires
- New targets provide specialized aggregation of data for complex analysis

Extended Events Architecture



Appendix: sqlserver.sql_text Action

- Provides the InputBuffer for the executing session and not the actual `sql_text`
 - InputBuffer masking of sensitive data using keyword match list for words like "password" blocks data collection
- Use `TRACK_CAUSALITY` and `sql_statement_starting/completed` event collect_statement customizable column in SQL 2012 instead
- Large actions like `sql_text` may require additional consideration for event sizing during session configuration to prevent event loss

Appendix: Event Session Options

- **MAX_MEMORY**
 - MAX_MEMORY is not the actual max memory for the event session since buffers align on 64K boundaries
- **MAX_EVENT_SIZE**
 - Determine what the maximum event size may be based on the event definitions.
 - Can not be smaller than a single memory buffer for the event session and the large buffers will be in addition to the regular buffers for the session.
- **MEMORY_PARTITION_MODE**
 - Determines the number of memory buffers created for the event session; none=3 buffers, per_node=3 buffers per NUMA node, per_cpu=2.5 buffers per scheduler

Appendix: etw_classic_sync_target Target

- Provides integration between SQL Server Eventing and Event Tracing for Windows
- Built on the classic provider from Windows Server 2000 and not the manifest provider from Windows Server 2008
- Creates an XE_DEFAULT_ETW_SESSION in windows, and only one exists for the entire server, making segregation of data from multiple instances impossible
- ETW session requires command line flush and close even if the Event Session in SQL is dropped

Questions?

