SQLskills Immersion Event

IEPTO2: Performance Tuning and Optimization

Module 2: Extended Events

Erin Stellato

Erin@SQLskills.com

Overview

- Extended Events core concepts and architecture
- Actions and predicates
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

What are Extended Events?

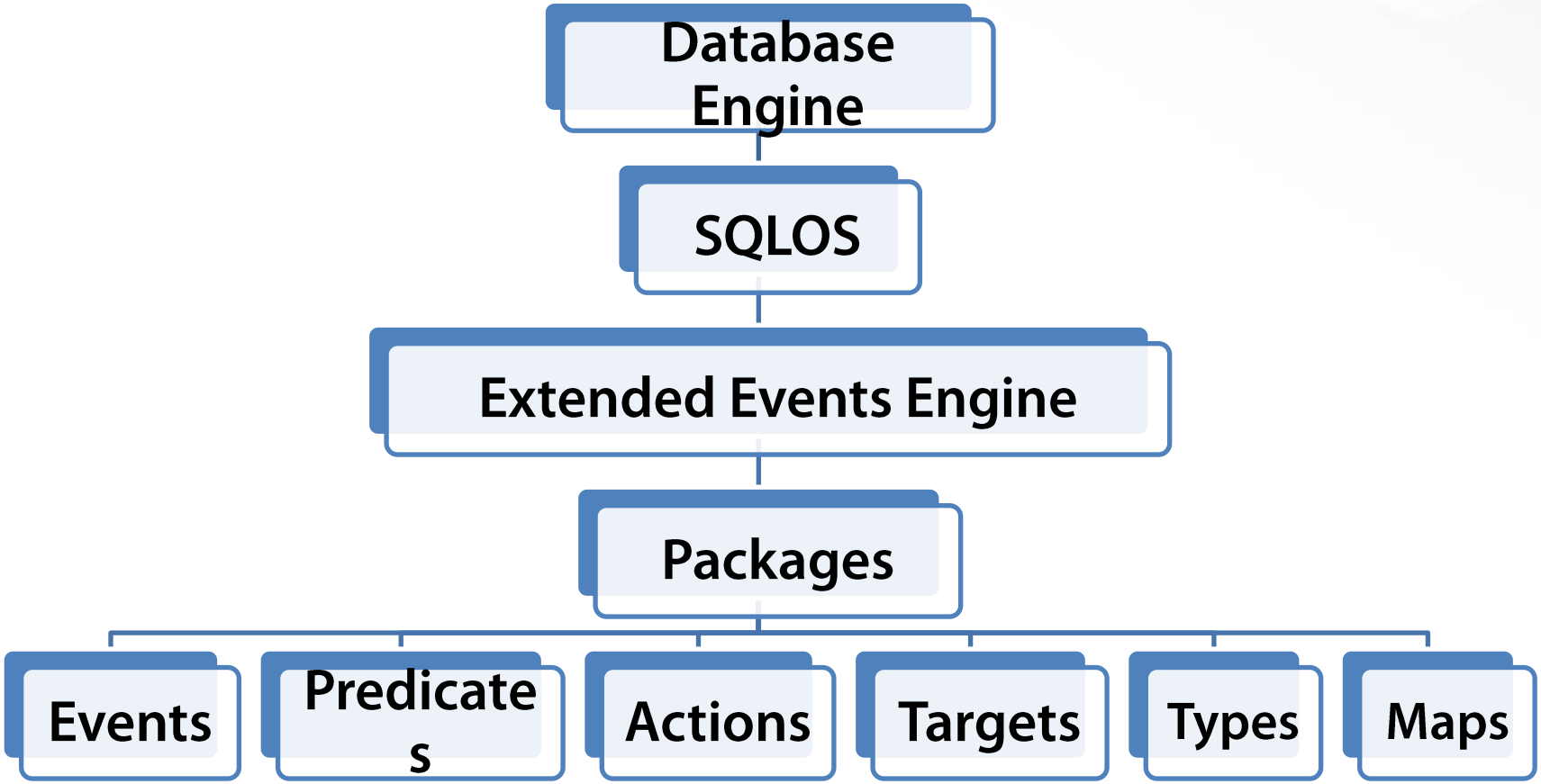
- **"…general event-handling system for server systems…"**
- **"The Extended Events infrastructure supports the correlation of data from SQL Server"**
- **This infrastructure provides the ability to create complex sessions to collect event information**
- **Introduced in SQL Server 2008, it is provided by SQLOS**
- **Provides the ability 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
  - Capture the first N executions of an event
  - Use the query_hash and query_plan_hash to capture execution plans and statement text

Overview

- Extended Events core concepts and architecture
- Predicates and actions
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

Extended Events Architecture

- The XE engine is a collection of the services and objects that it manages, and is an interaction point for other SQL Server processes



Core Concepts: Packages

- **Packages are loaded by individual modules at run-time**
  - Modules: sqlservr.exe, sqlos.dll, sqlmin.dll, sqllang.dll, sqldk.dll, hkengine.dll, hkruntime.dll, hkcompile.dll, qds.dll
- **Packages are containers that define the available Extended Events objects and their definitions**
- **Packages are not a functional boundary of usage**
  - They are boundaries of metadata
  - Objects from one package can be used with objects from another package

Core Concepts: Events

- **An event corresponds to a well-known point in the code, e.g. a T-SQL statement finished executing; a deadlock occurred**
  - SP: StmtCompleted
  - Deadlock graph
  - Data File Auto Grow
  - Sort Warning
- **Events deliver a basic payload of information**
  - Payload is defined by a versioned schema
  - Certain data elements are always returned by an event; these are non-customizable
  - Events may contain optional (customizable) data elements that are only collected when specified
- **Events are defined using the Event Tracing for Windows (ETW) model to allow integration with ETW**
- **An event can be changed or no longer used if the module code changes**

Event Comparison

Trace	Extended Events
SP: StmtCompleted	sp_statement_completed
SQL: StmtRecompile	sql_statement_recompile
Data File Auto Grow	database_file_size_changed*
Deadlock graph	xml_deadlock_report
Audit: Login	~~login~~

- Event names between Trace and Extended Events are similar, but not always exactly the same
- All events from Trace have a comparable event in Extended Events as of SQL Server 2012 (Audit events are an exception)
- Use the trace_xe_event_map DMV for mapping

Changes in XEvents by SQL Server Version

SQL Server Version	Number of Events	Notes
2008 SP3	253	
2008 R2 SP2	262	
2012	600+	Includes *all* events available in Trace
2014	900+	
2016	1300+	
2017	1500+	
2019	1800+	

- There are only 180 Trace events in each of the versions listed here

Core Concepts: Predicates

- **The predicate is a filter that defines whether or not the event will fire**
  - Implemented as Boolean expressions that can be grouped into logical blocks for complex analysis
- **Support short-circuit evaluation**
  - First false evaluation of a logical block of predicates prevents the event from firing
- **Predicates can operate on event payload data or global predicate source fields**
  - The global predicate source fields that can be used predicates are *not* the same as Actions, even though the same names *might* be used

Core Concepts: Actions

- **An action is an additional operation performed when the event fires**
  - *Collect* the database ID
  - *Collect* the session ID
  - *Create* a mini dump for the current thread
- **Actions *only* execute *after* predicate evaluation determines the event will fire**
- **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 and asynchronous targets exist**
- **Basic targets collect raw event data**
  - event_file
  - ring_buffer
- **Aggregate targets group data based on criteria**
  - histogram (event bucketizer)
  - event_counter
  - pair_matching (match events)
- **ETW target (etw_classic_sync_target) allows end-to-end tracing with Windows Kernel**

Core Concepts: Types and Maps

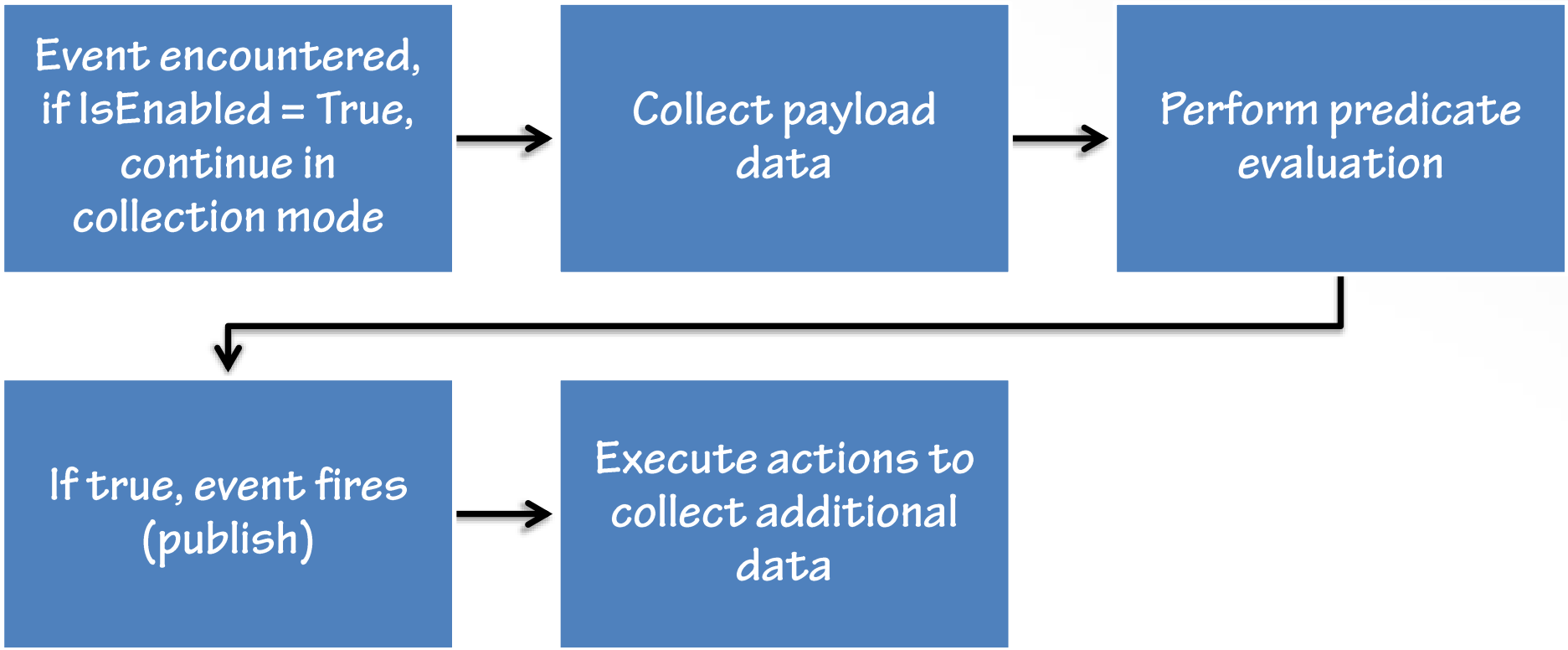
- **Types define the data type for an event column, action, or global predicate source (e.g. integer, string)**
- **Types are contained within packages**

- **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
- **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
- **Be aware that map values can change from one version of SQL Server to the next (e.g. wait_types)**

Core Concepts: Event Sessions

- **An event session is a collection of events, their configured actions and predicates, and targets to store 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* the events are dispatched to asynchronous targets**
- **Event sessions can be configured to:**
  - Track causality of an event firing (i.e. linking events together)
  - Start automatically at server startup

The Big Picture: Extended Events Life Cycle Part 1



XEvent Profiler

- **Available in SSMS starting in version 17.3**
- **Designed to help users quickly start an event session**
- **Provides two options for event sessions**
  - Standard
  - TSQL
- **Event session is created and started immediately, with data displayed**
  - Data is not written to any target
- **Event session definition persists after stopping, until an instance restart**

SQL Server Profiler

- **Available in Azure Data Studio as an Extension**
- **This is actually using Extended Events; this is not Profiler**
- **Provides three options for event session:**
  - Standard_OnPrem
  - Standard_Azure
  - TSQL_OnPrem
- **Event session is created and started immediately, with data displayed**
  - Data is written to the ring_buffer target
- **Event session definition persists after stopping, until an instance restart**
- **The event sessions for SSMS and ADS are named differently, so theoretically you could run the same session twice**

Demo

Exploring XE core concepts through the UI

Overview

- Extended Events core concepts and architecture
- Predicates and actions
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

The Power of the Predicate

- **Predicates can use basic arithmetic operators, or textual comparators for more complex expressions**
  - e.g. greater than last max value, less than last min value, and divides evenly by int64 can perform event sampling which is not possible with basic Boolean expressions
- **Predicates can store state (this is pretty cool)**
  - e.g. count the number of times an event fires and only publish the event every N times
- **Inverse predicates can be defined using NOT, !=, or <>**
- **If you filter on an element that is not part of the default payload, the engine has to first collect that information *before* it can perform predicate evaluation**

Writing Predicates

- **Predicates are broken up into logical blocks, which are denoted with parenthesis ( )**
- **The first false evaluation of a logical block prevents further evaluation – *write your predicates carefully***
  - When using (AND), the first logical block that is false prevents further evaluation
  - When using (OR), if there are multiple predicates in a single block, if one is false, evaluation continues until a true evaluation or the end of the block is reached

Writing Predicates: Consider Order

```
WHERE (
   database_id = 5
   AND duration > 1000000
   AND logical_reads > 500000
   )
```

```
WHERE (
   logical_reads > 500000
   AND duration > 1000000
   AND database_id = 5
   )
```

SQLskills
immerse yourself in sql server

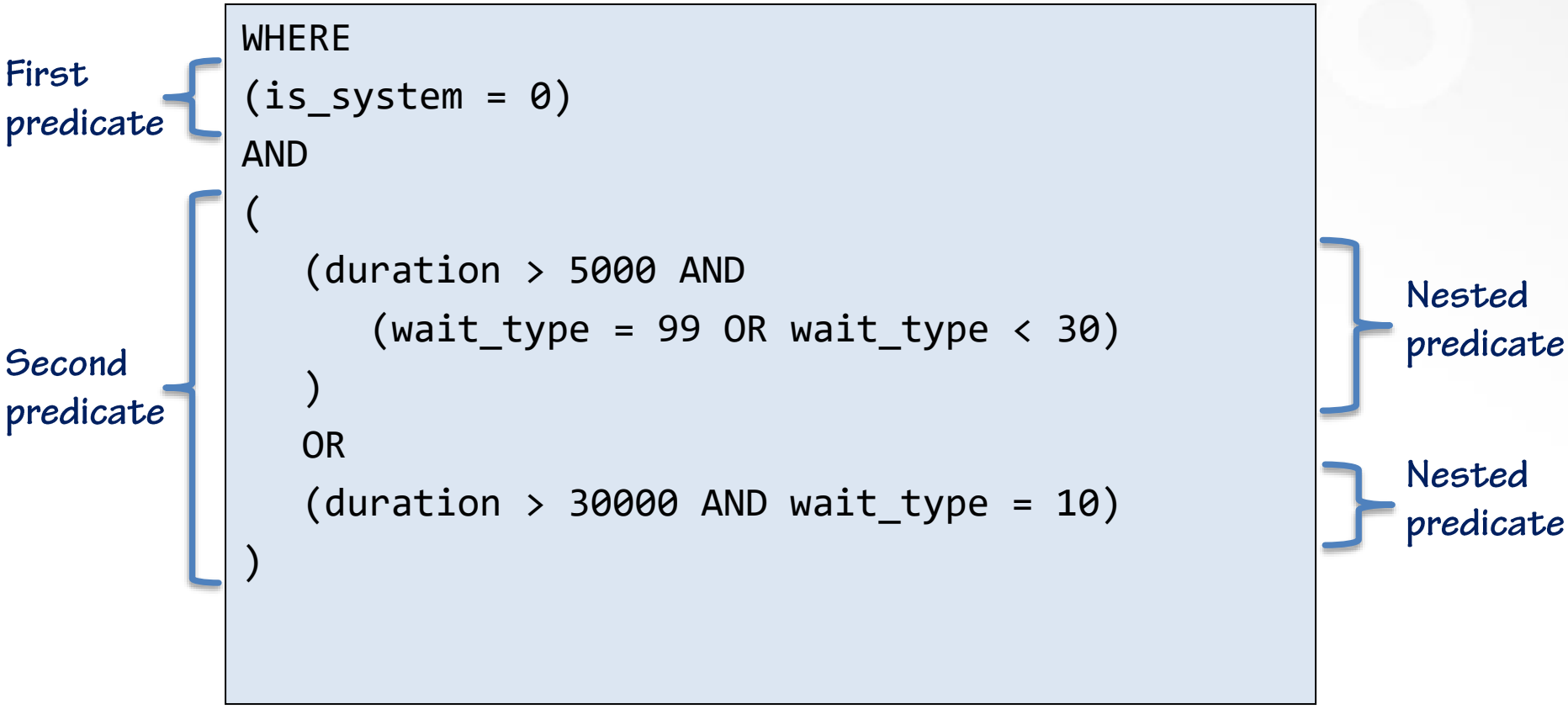
Writing Predicates: Short-Circuiting

```
WHERE (
   logical_reads > 500000
   AND duration > 1000000
   AND database_id = 5
   )
```

```
WHERE (
   logical_reads > 500000
   OR duration > 1000000
   OR database_id = 5
   )
```

SQLskills
immerse yourself in sql server

Writing Predicates: Logical Blocks



Actions that Add Data

- **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**
- **Actions cause 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!)

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 Server 2012+ instead**
- **Large actions like sql_text and sql_context may require additional consideration for event sizing during session configuration to prevent event loss**

Demo

Predicates and actions

Overview

- Extended Events core concepts and architecture
- Predicates and actions
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

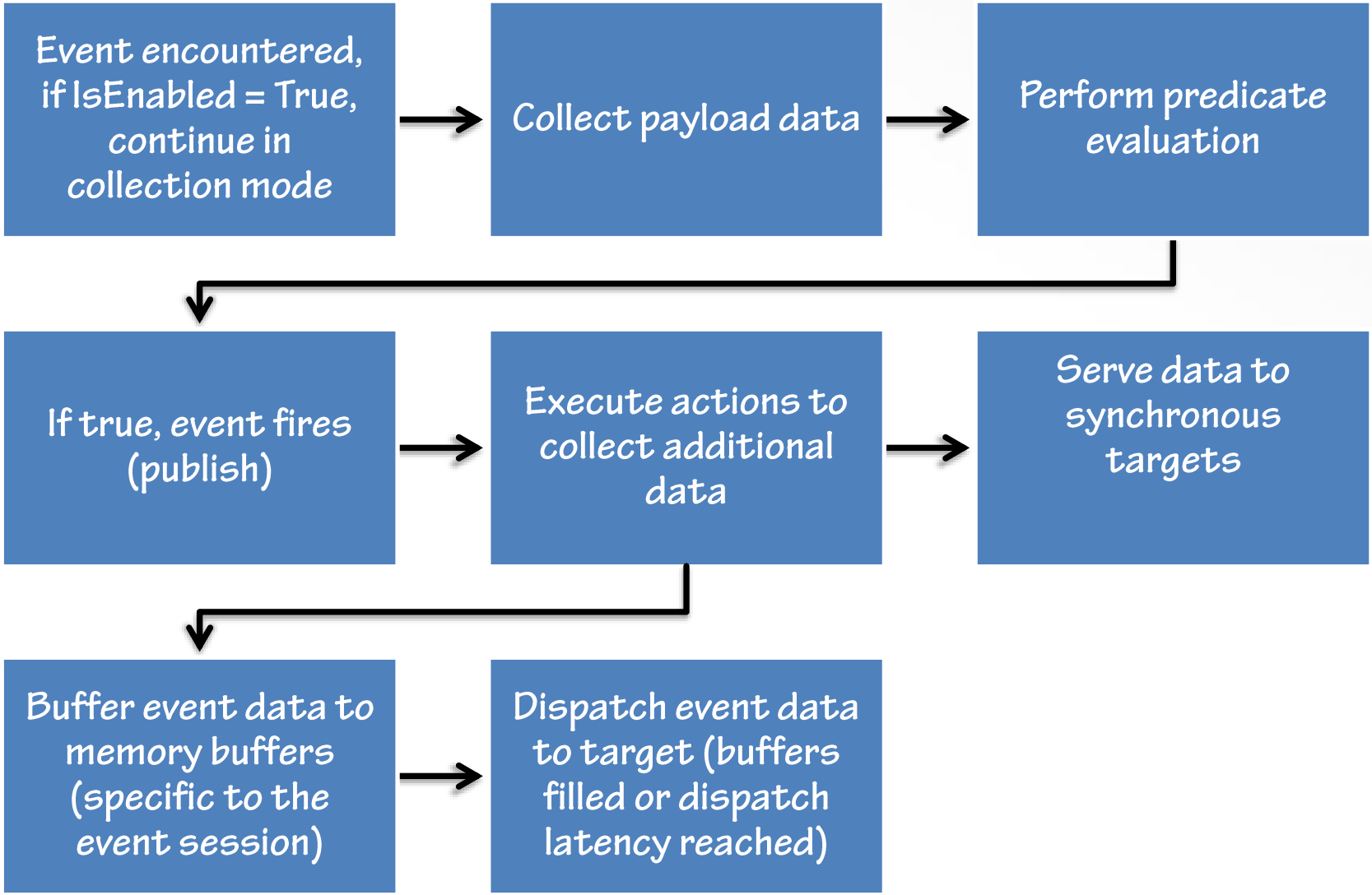
Targets

- Targets consume the events, and can store the data (temporarily or permanently) in either raw or aggregate format
- Synchronous targets are served immediately by the thread that fires the event
- Asynchronous targets are served after a delay – either when the intermediate buffers are filled, or when the dispatch latency is reached

Dispatching

- **The dispatching of 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

The Big Picture: Extended Events Life Cycle Part 2



Ring Buffer Target

- **General purpose, in-memory target that does FIFO event collection**
- **Ideal target for small data sets where event loss is acceptable**
- **Data is in binary format, but materialized as XML when the target is queried via the sys.dm_xe_session_targets DMV**
  - If viewed using the UI, the XML is shown, not the processed events
- **Options**
  - MAX_MEMORY: the maximum amount of memory in KB to be used, so older events are dropped when this limit is reached
  - MAX_EVENT_LIMIT: the maximum number of events to keep, existing events are dropped when this limit is reached
  - OCCURRENCE_NUMBER: sets preferred number of each event type to keep
- **Data is lost when the event session is stopped**
- **DMV limitations may result in unreadable XML from sys.dm_xe_session_targets**

Event File Target

- **Similar to the trace file in SQL Trace, this target collects event data in a file on disk**
  - Data persists after an event session is stopped
  - Files can be shared
  - Useful for long-term and/or detailed analysis
- **Event data has the same XML schema as the ring_buffer target**
- **Can be read using the UI in SQL Server 2012 and higher**
- **In SQL Server 2008 and 2008 R2:**
  - Target data must be queried through the SQL Server database engine, there is no external API available for reading the file
  - File target used two files, log and metadata, and both must exist to read the data
  - The metadata file simply describes the structure of the log file, not the events themselves

Event Counter Target

- Counts how many events occurred for the event session
- Use when defining an event session for an unknown workload to determine how many events you can expect based on your event session definition

Histogram Target

- Specialized in-memory target that collects events into defined "buckets" to simplify analysis of event frequency
  - e.g. database_id, object_id
- Does not store actual event data, only the count of occurrences

Event Pairing 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 is critical, otherwise 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 event_file target**
- **Same output XML as the ring_buffer for event data**

Target Summary

Target	Information Captured	Synch vs. Asynch	Data Location
Ring Buffer	Raw data	Asynchronous	Memory
Event File	Raw data	Asynchronous	File
Event Counter	Aggregate	Synchronous	Memory
Histogram	Aggregate	Asynchronous	Memory
Event Pairing	Raw data	Asynchronous	Memory

Demo

Working with targets

Overview

- Extended Events core concepts and architecture
- Predicates and actions
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

Event Session Options (1)

- **STARTUP_STATE**
  - Start the event session automatically with SQL Server
- **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

Event Session Options (2)

- **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
- **MAX_DISPATCH_LATENCY**
  - Dispatch events to targets faster
  - May impact the dispatcher pool if multiple event sessions exist

Event Session Options (3)

- **MAX_MEMORY**
  - MAX_MEMORY is not the actual max memory for the event session since buffers align on 64KB boundaries
- **MAX_EVENT_SIZE**
  - Determine what the maximum event size may be based on the event definitions
- **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

Creating an Event Session

```
CREATE EVENT SESSION [Track_Queries]
   ON SERVER
   ADD EVENT sqlserver.sp_statement_completed (
     SET collect_statement=(1)
   ACTION (
     sqlserver.database_id,
     sqlserver.session_id)
   WHERE (
     [sqlserver].[is_system]=(0)
     AND logical_reads > 25000)
     )
   ADD TARGET package0.ring_buffer
     (SET max_memory=(2048))
   WITH (MAX_MEMORY=4096 KB,
     MAX_DISPATCH_LATENCY=30 SECONDS);
```

SQLskills
immerse yourself in sql server

Working with Event Sessions (1)

```
ALTER EVENT SESSION [Track_Queries]
   ON SERVER
   STATE=START;
GO

ALTER EVENT SESSION [Track_Queries]
   ON SERVER
   ADD EVENT sqlserver.sql_statement_starting (
     SET collect_statement=(1)
   ACTION (
     sqlserver.database_id,
     sqlserver.session_id)
   );
GO
```

Working with Event Sessions (2)

```
ALTER EVENT SESSION [Track_Queries]
   ON SERVER
   DROP EVENT sqlserver.sp_statement_completed;
GO

ALTER EVENT SESSION [Track_Queries]
   ON SERVER
   DROP TARGET package0.ring_buffer;
GO

ALTER EVENT SESSION [Track_Queries]
   ON SERVER
   STATE=STOP;
GO
```

SQLskills
immerse yourself in sql server

Management DLL (1)

- **All DDL requires:**
  - ALTER ANY EVENT SESSION in SQL Server 2012+
  - CONTROL SERVER in SQL Server 2008/2008R2
- **CREATE EVENT SESSION**
  - Creates a new event session based on the events, actions, predicates, targets, and session options provided
  - Requires at least one event
  - All event sessions are created in a stopped state

Management DDL (2)

- **ALTER EVENT SESSION**
  - Alter the state of an event session to start or stop
  - Add or remove events and targets from an event session
  - Change session configuration options for a stopped event session
- **DROP EVENT SESSION**
  - Removes an event session from the system entirely
  - Memory resident targets are not available after an event session is dropped

Metadata Views (1)

- **sys.dm_xe_packages**
  - Contains an entry for each of the packages registered in the engine
  - Each package will have a unique guid, 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 engine
  - Objects have the package_guid of the package that loaded them
  - The object_type column determines the type of object

Metadata Views (2)

- **sys.dm_xe_object_columns**
  - Contains information about the columns, or data elements, that exist for a specific object
    - readonly - system metadata about an event
    - data - the data elements that are returned 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
    - Linked by object_package_guid to the package that created them

The system_health Session (1)

- **The system_health session is an Extended Events session that's always running**
  - Good starting point to just look at XE data
  - Target data to ring_buffer and event_file in SQL Server 2012
    - Only writes to the ring_buffer in 2008/2008R2
  - It is not recommended to alter the system_health session configuration
    - But…the script to create it is in the default installation folder:
    - C:\Program Files\Microsoft SQL Server\MSSQL14.SQL2017\MSSQL\Install\u_tables.sql
- **Historical information about what's happened in SQL Server**
  - Information is different than what's in the ERRORLOG, default trace
  - Because the data is persisted to disk, it is often the only way to view information from select DMVs after a restart

The system_health Session (2)

- **Includes:**
  - Information for any sessions that encounter an error with severity >=20
  - Information for any sessions that encounter a memory type of error
  - Deadlock information
  - Information for any sessions that have waited on latches (>15s) or locks (>30s)
  - Information for any sessions that have waited for “external” waits or “pre-emptive waits”

Demo

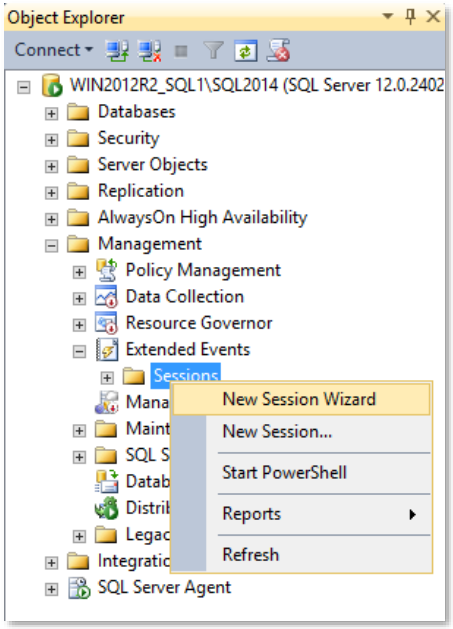
Modifying event sessions and using the DMVs

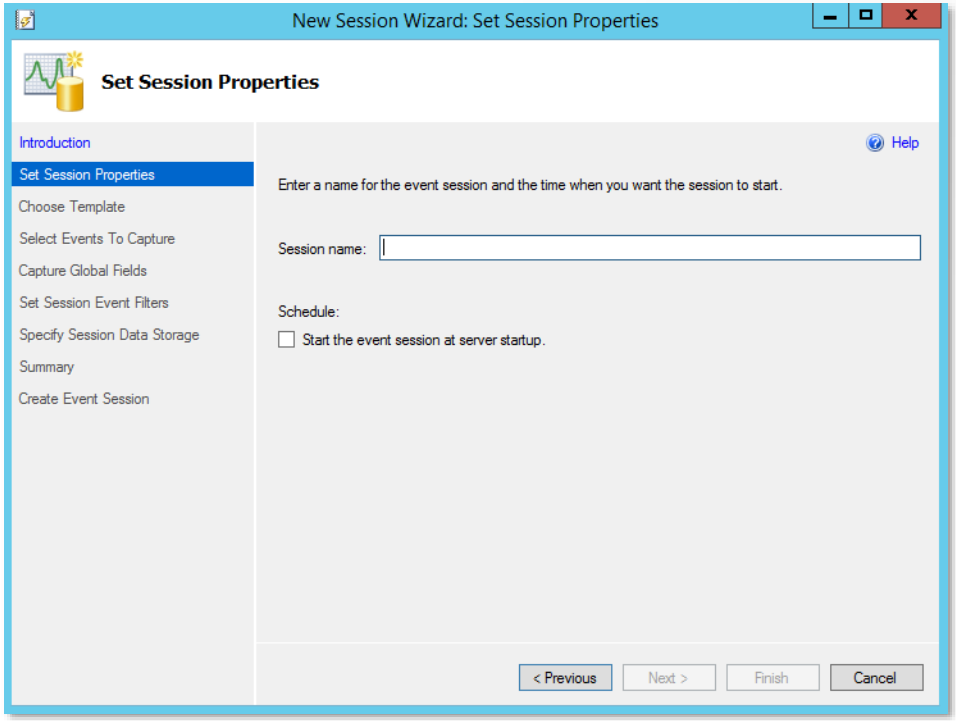
Overview

- Extended Events core concepts and architecture
- Predicates and actions
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

Extended Events UI

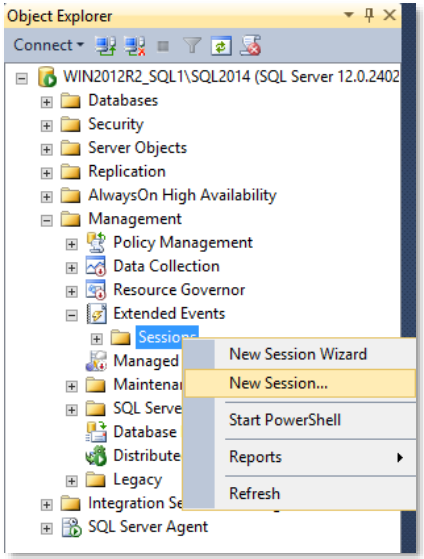
- **New Session Wizard**
  - Create new sessions rapidly from templates or by using a minimal set of configurable options

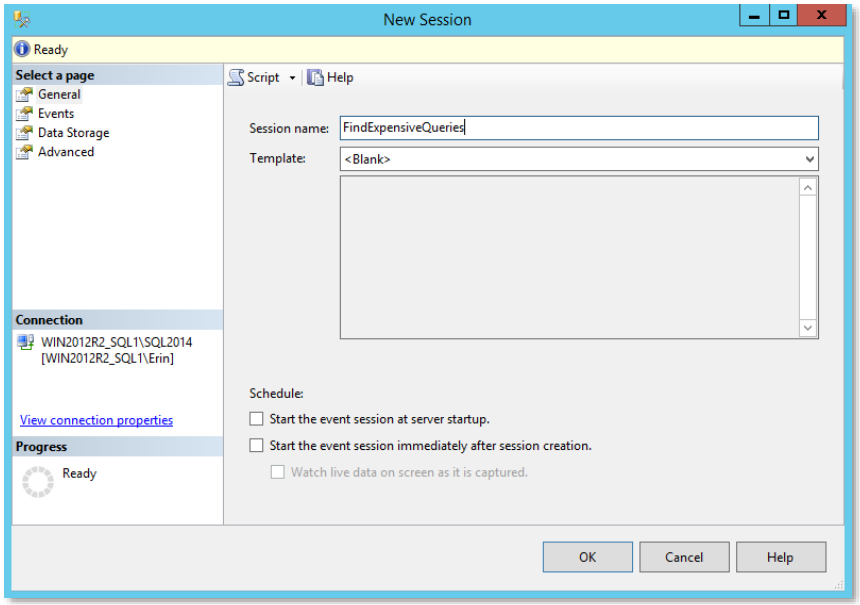


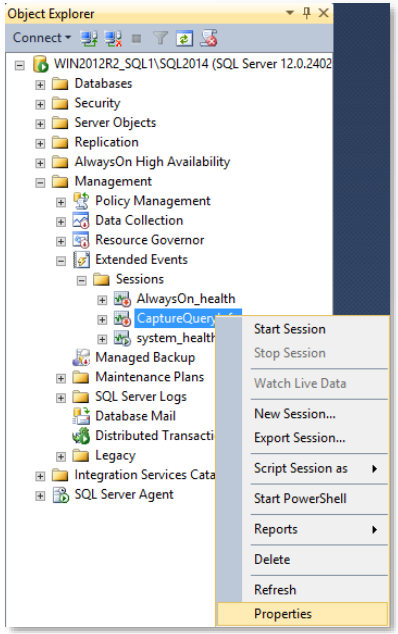


Extended Events UI

- **New Session Dialog**
  - Used for advanced session creation using all available options and targets
  - Via Properties option, use to modify existing event sessions





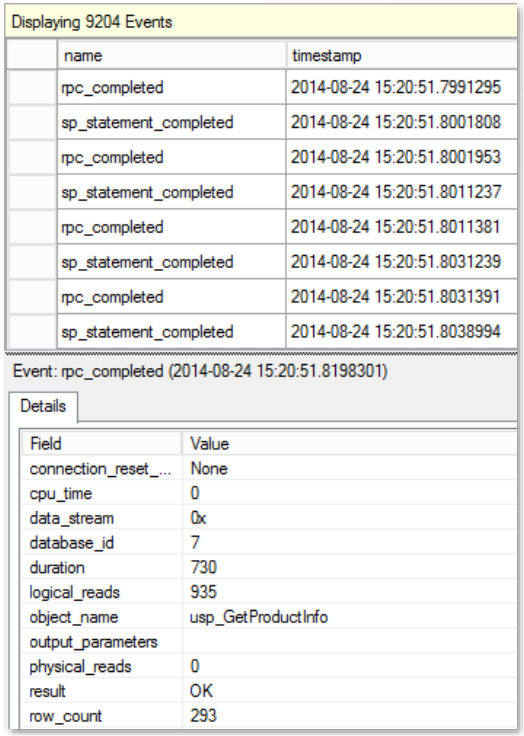


Templates

- **Templates are an easy way to automate set up of an Extended Events session**
- **Export an existing session to create a template**
- **User templates default to:**
  - %USERPROFILE%\Documents\SQL Server Management Studio\Templates\XEventTemplates
- **Default templates exist in:**
  - C:\Program Files (x86)\Microsoft SQL Server\110\Tools\Templates\sql\xevent
- **Edit the .xml file to add a custom category, the template name, and the description**

Live Data Viewer

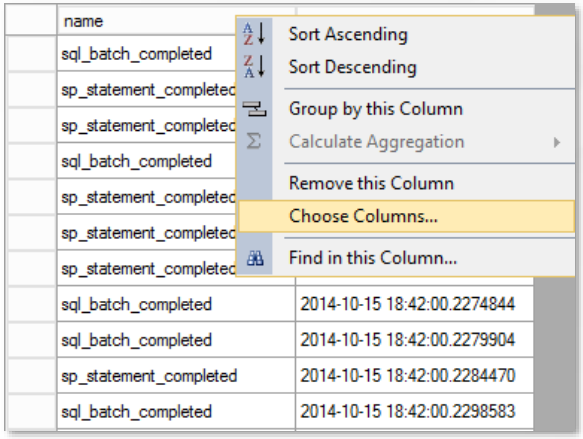
- **Reads a live stream of event buffers from an Extended Events session on a server**
  - Provides a "Profiler like" view of the data being generated
- **Will disconnect if it can't keep up with event generation to prevent impact to the instance**
  - Boosting MAX_MEMORY option for the instance and/or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

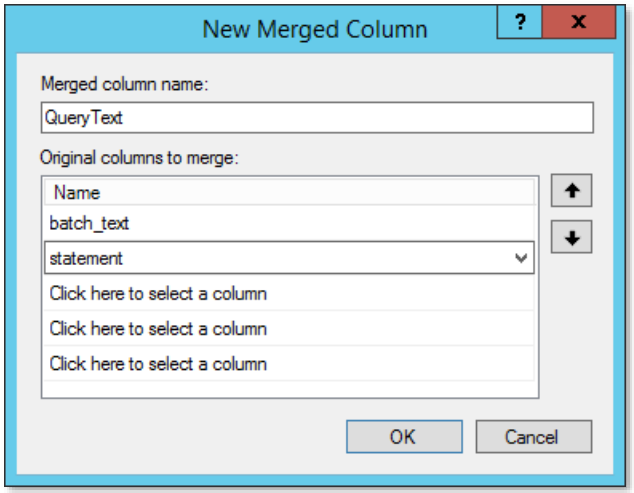


57

Data Viewer Customization

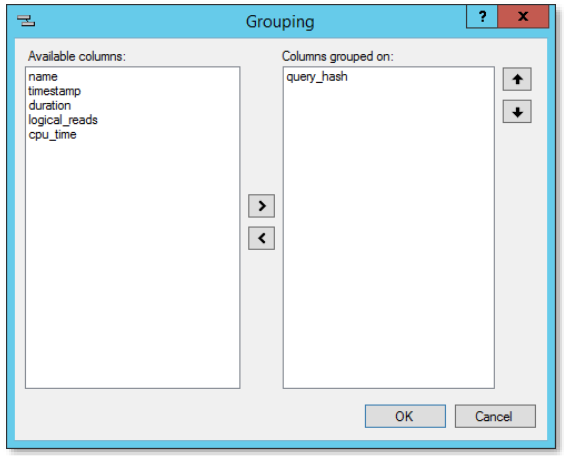
- **Default view is limited but columns can be added using the column chooser**
  - New *merged columns* can be created to track similar information provided by different events in a single column in the grid view
- **Display settings can be saved for reuse with later event sessions or event_file processing**

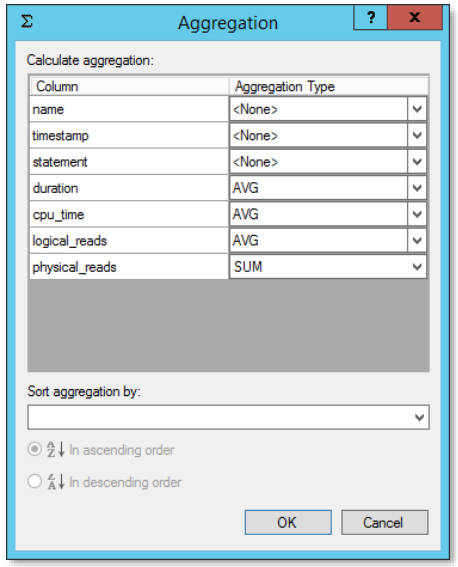




Grouping and Aggregating Data (1)

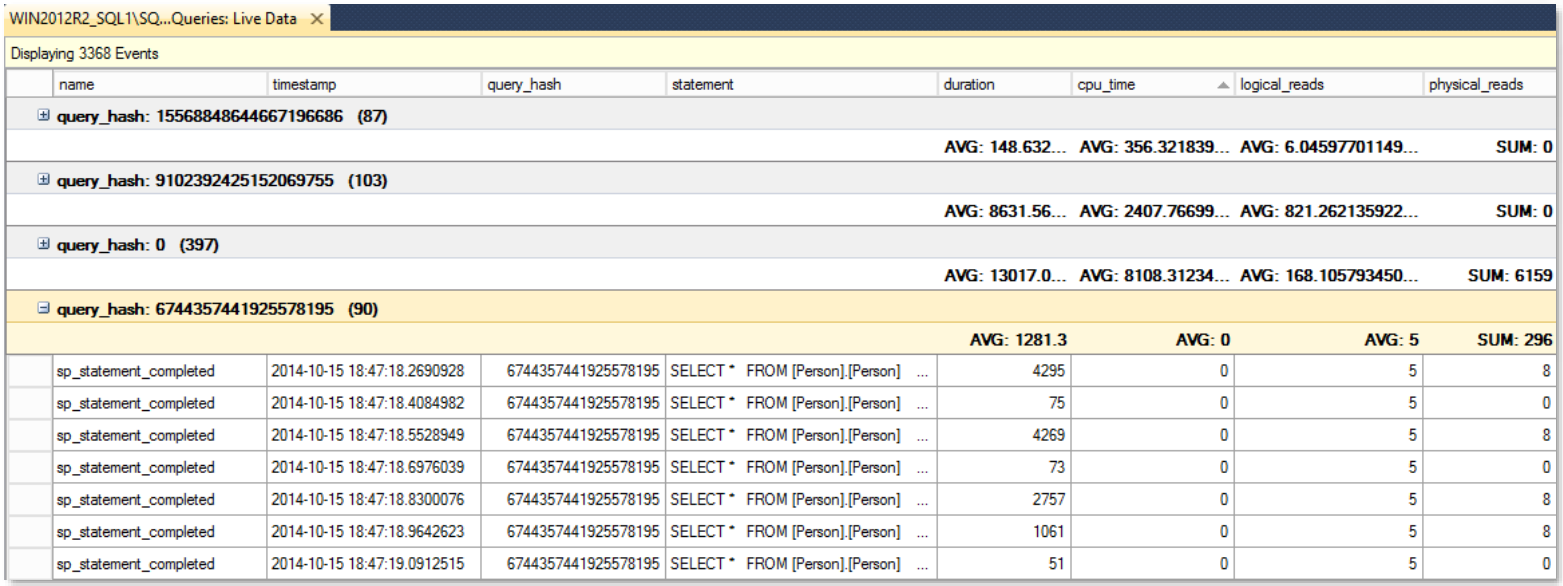
- **Simplify complex data analysis of disconnected data within the data viewer for Extended Events**
  - Group by any available columns in the current view
  - Aggregate one or many columns based on the current grouping settings to find trends of interest





Grouping and Aggregating Data (2)

- Groups can be expanded for detail level viewing



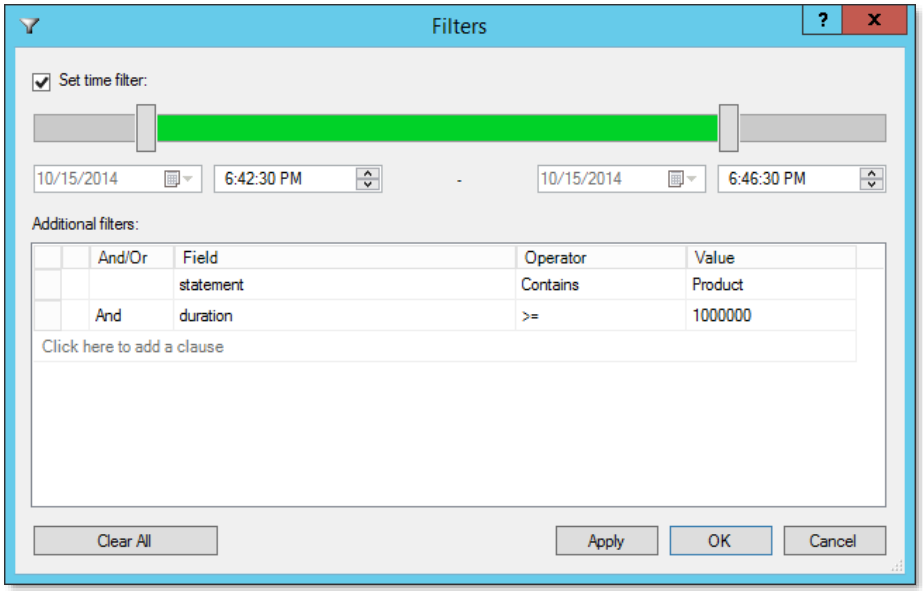
WIN2012R2_SQL1\SQ...Queries: Live Data

Displaying 3368 Events

	name	timestamp	query_hash	statement	duration	cpu_time	logical_reads	physical_reads
⊞	**query_hash: 15568848644667196686 (87)**							
					AVG: 148.632...	**AVG: 356.321839...**	**AVG: 6.04597701149...**	**SUM: 0**
⊞	**query_hash: 9102392425152069755 (103)**							
					AVG: 8631.56...	**AVG: 2407.76699...**	**AVG: 821.262135922...**	**SUM: 0**
⊞	**query_hash: 0 (397)**							
					AVG: 13017.0...	**AVG: 8108.31234...**	**AVG: 168.105793450...**	**SUM: 6159**
⊟	**query_hash: 6744357441925578195 (90)**							
					AVG: 1281.3	**AVG: 0**	**AVG: 5**	**SUM: 296**
	sp_statement_completed	2014-10-15 18:47:18.2690928	6744357441925578195	SELECT * FROM [Person].[Person] ...	4295	0	5	8
	sp_statement_completed	2014-10-15 18:47:18.4084982	6744357441925578195	SELECT * FROM [Person].[Person] ...	75	0	5	0
	sp_statement_completed	2014-10-15 18:47:18.5528949	6744357441925578195	SELECT * FROM [Person].[Person] ...	4269	0	5	8
	sp_statement_completed	2014-10-15 18:47:18.6976039	6744357441925578195	SELECT * FROM [Person].[Person] ...	73	0	5	0
	sp_statement_completed	2014-10-15 18:47:18.8300076	6744357441925578195	SELECT * FROM [Person].[Person] ...	2757	0	5	8
	sp_statement_completed	2014-10-15 18:47:18.9642623	6744357441925578195	SELECT * FROM [Person].[Person] ...	1061	0	5	8
	sp_statement_completed	2014-10-15 18:47:19.0912515	6744357441925578195	SELECT * FROM [Person].[Person] ...	51	0	5	0

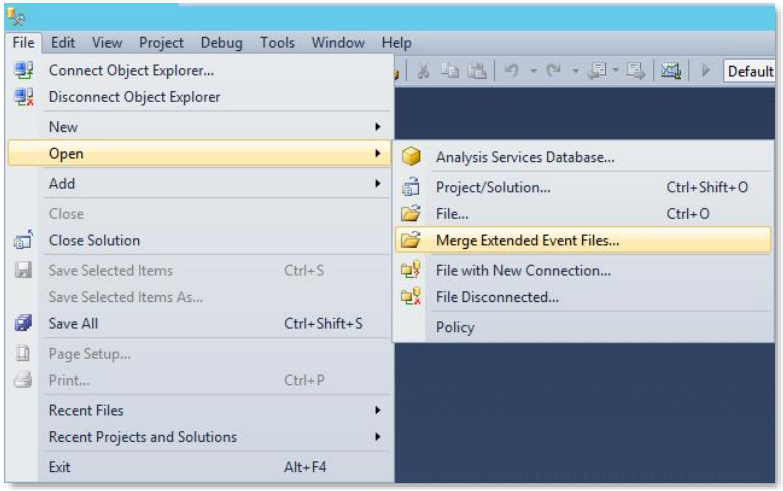
Data Viewer: Filtering

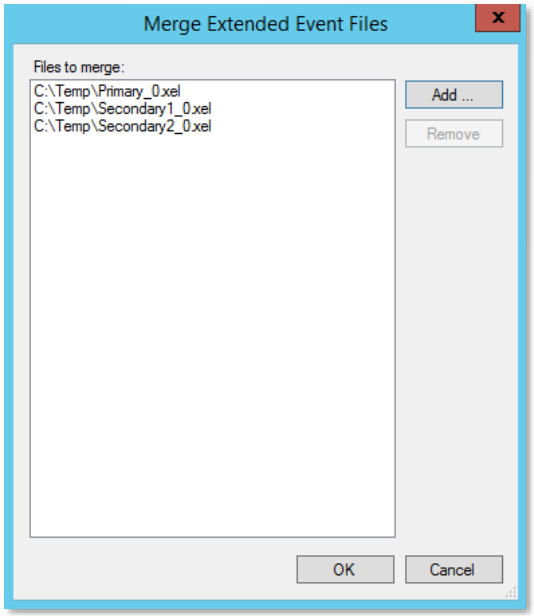
- **Data viewed in the UI can be filtered based on time and/or column values**
  - Use And/Or between columns to further reduce the result set



Data Viewer: Merging Files

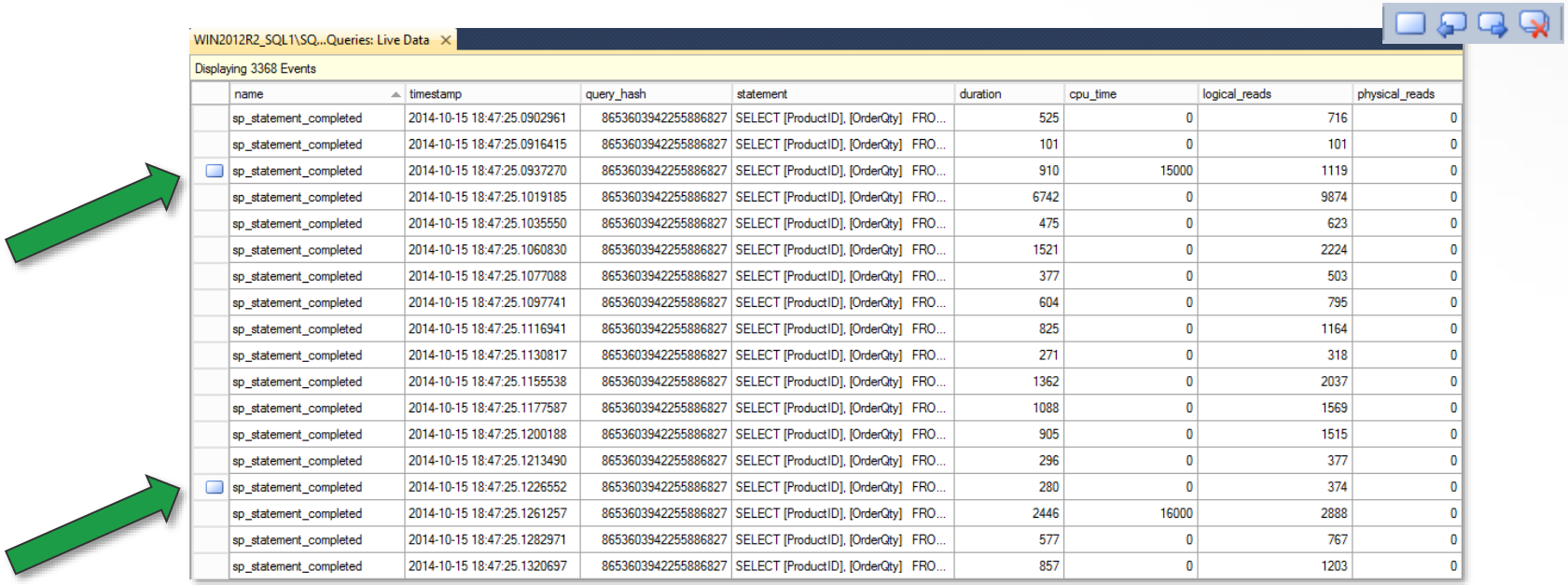
- Multiple files can be merged together in the UI to create one master file for analysis





Data Viewer: Bookmarking Events

- Within the UI, bookmark events to quickly find them again



WIN2012R2_SQL1\SQ...Queries: Live Data

Displaying 3368 Events

name	timestamp	query_hash	statement	duration	cpu_time	logical_reads	physical_reads
sp_statement_completed	2014-10-15 18:47:25.0902961	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	525	0	716	0
sp_statement_completed	2014-10-15 18:47:25.0916415	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	101	0	101	0
sp_statement_completed	2014-10-15 18:47:25.0937270	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	910	15000	1119	0
sp_statement_completed	2014-10-15 18:47:25.1019185	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	6742	0	9874	0
sp_statement_completed	2014-10-15 18:47:25.1035550	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	475	0	623	0
sp_statement_completed	2014-10-15 18:47:25.1060830	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	1521	0	2224	0
sp_statement_completed	2014-10-15 18:47:25.1077088	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	377	0	503	0
sp_statement_completed	2014-10-15 18:47:25.1097741	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	604	0	795	0
sp_statement_completed	2014-10-15 18:47:25.1116941	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	825	0	1164	0
sp_statement_completed	2014-10-15 18:47:25.1130817	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	271	0	318	0
sp_statement_completed	2014-10-15 18:47:25.1155538	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	1362	0	2037	0
sp_statement_completed	2014-10-15 18:47:25.1177587	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	1088	0	1569	0
sp_statement_completed	2014-10-15 18:47:25.1200188	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	905	0	1515	0
sp_statement_completed	2014-10-15 18:47:25.1213490	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	296	0	377	0
sp_statement_completed	2014-10-15 18:47:25.1226552	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	280	0	374	0
sp_statement_completed	2014-10-15 18:47:25.1261257	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	2446	16000	2888	0
sp_statement_completed	2014-10-15 18:47:25.1282971	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	577	0	767	0
sp_statement_completed	2014-10-15 18:47:25.1320697	8653603942255886827	SELECT [ProductID], [OrderQty] FRO...	857	0	1203	0

Key Takeaways

- For SQL Server 2008R2 and below, Trace is typically still the first option for tracing query activity
- For SQL Server 2012 and higher, using Extended Events (instead of Profiler/Trace) is recommended
- Extended Events includes events for new features (e.g. Availability Groups , Columnstore, In-Memory OLTP) that Trace does not have
- Extended Events is flexible not just in its implementation, but also in the configuration of event sessions which allows for troubleshooting in a way not possible previously

Additional Resources

- **Pluralsight courses**
  - SQL Server: Replacing Profiler with Extended Events, Erin Stellato
    - http://bit.ly/2e5fByz
  - SQL Server: Introduction to Extended Events, Jonathan Kehayias
    - http://bit.ly/1uhYXv4
  - SQL Server: Advanced Extended Events, Jonathan Kehayias
    - http://bit.ly/1FhQVcK
- **Blog posts**
  - Stairway to SQL Server Extended Events, Erin Stellato
    - http://www.sqlservercentral.com/stairway/134867/
  - 31 Days of Extended Events, Jonathan Kehayias
    - https://www.sqlskills.com/blogs/jonathan/category/xevent-a-day-series/
  - Introducing the Extended Events User Interface
    - http://bit.ly/1c9qHN3

Additional Resources

- **TechNet**
  - Using the system_health Session
    - http://technet.microsoft.com/en-us/library/ff877955.aspx
  - Extended Events
    - http://technet.microsoft.com/en-us/library/bb630282.aspx
- **Utilities**
  - SQL Server 2008/2008R2 Extended Events Add-In for SSMS
    - http://extendedeventmanager.codeplex.com/
  - SQL Server 2012 Extended Events Add-in
    - https://www.sqlskills.com/free-tools/sql-server-2012-extended-events-add-in/

Review

- Extended Events core concepts and architecture
- Predicates and actions
- Target practice
- Event session basics and system health
- Extended Events UI
- Appendix

Questions?

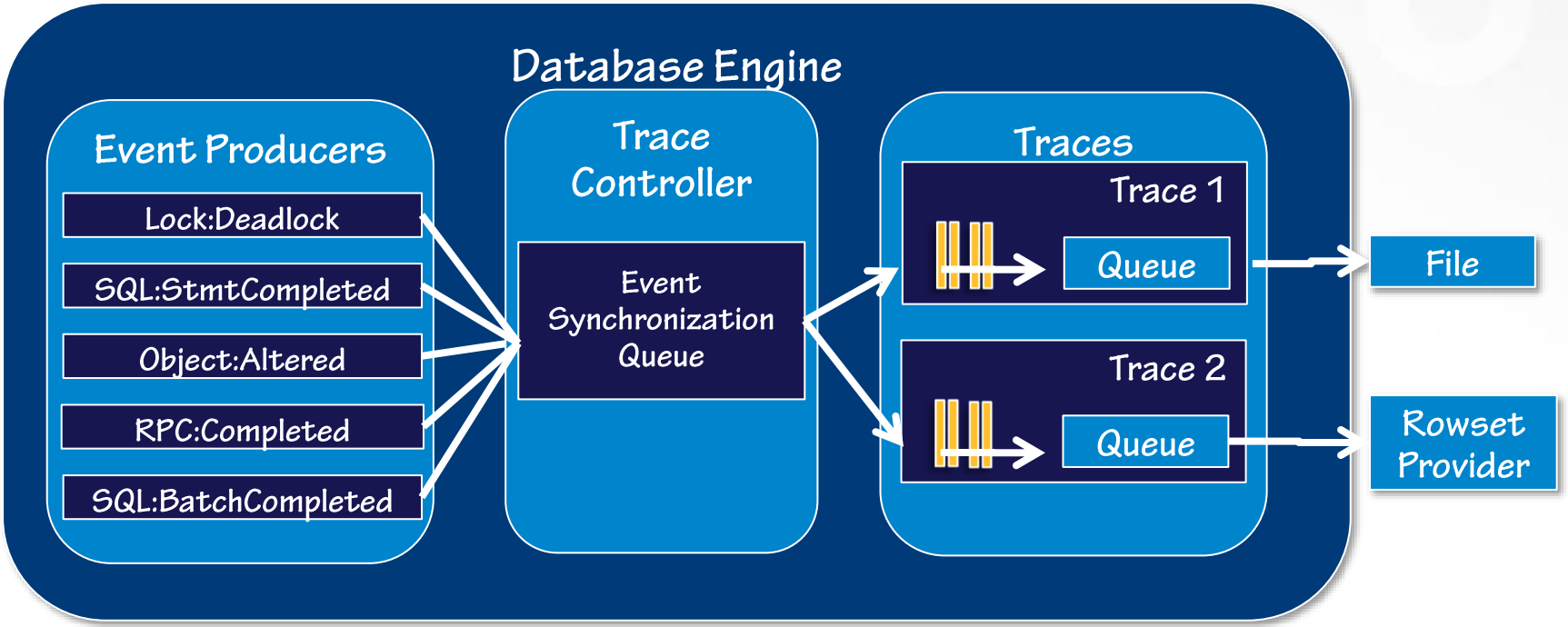
Appendix: A Brief History of Trace

- **SQL Trace was introduced in SQL Server 6.5 as a graphical tool, which was external to the SQL Server architecture**
  - Saved output to a trace or script
  - Very limited in what it could capture
- **SQL Profiler was introduced in SQL Server 7.0, and provided new capabilities for capturing and analyzing data**

Appendix: Basic Uses of SQL Trace

- **Provides real-time insight into SQL Server activity**
  - Start and stop can be automated
- **Allows DBAs to capture duration, frequency, and resource use of queries**
  - Filter on different columns
- **Can be used to capture a baseline, or replay a workload (one client)**
- **May be used to audit user activity**
- **Most frequently used to troubleshoot performance issues and errors**
- **Data can be analyzed using free, third-party tools (RML Utilities, Clear Trace, Qure)**

Appendix: Trace Architecture Review



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: 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

Appendix: Why Do You Need to Migrate to Extended Events?

- **Extended Events captures the same information as SQL Trace, *and more***
  - Trace is deprecated as of SQL Server 2012
- **Extended Events provides the same functionality as Trace:**
  - Capture information about what's going on inside SQL Server
  - Choose which fields to capture
  - Filter on different fields
  - Automate capture
- **Extended Events has a UI in SQL Server 2012+**
  - No need to use XQuery to view the data *(with some exceptions)*
- **Extended Events is more lightweight than Profiler and SQL Trace**
- **Extended Events is only method for tracing *new* SQL Server features**
- **Extended Events is the new way to look at diagnostic data collection**

Appendix: Session Definition Catalog Views

- **sys.server_event_sessions**
  - Contains the name and session level options for each event session that currently exists within the Extended Events Engine
- **sys.server_event_session_events**
  - Contains information about the events and predicates that are a part of an existing event session
- **sys.server_event_session_actions**
  - Contains one row for each action, for each event, in an existing 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 (1)

- **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 (2)

- **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