SQL Server: Introduction to Extended Events

Module 3: Extended Events Objects

Jonathan M. Kehayias
Jonathan@SQLskills.com

pluralsight
hardcore developer training

Introduction

- **Extended Events introduce a new complexity for management in the number of objects exposed as a part of the implementation**
- **Understanding the objects and how they are properly used is critical for understanding how to leverage Extended Events in SQL Server**

- **In this module we'll cover:**
  - Metadata views
  - Packages
  - Events
  - Predicates
  - Actions
  - Targets
  - Types and Maps

Packages

- **Packages are loaded by individual modules at runtime**
  - Default package0 package is loaded by the Extended Events engine and contains generic objects that are not specific to any single module
    - E.g.: all targets, generic types, predicate comparators, and some actions
- **Packages are containers that define the available objects and their definitions**
- **Packages are not a functional boundary of usage**
  - Objects from one package can be used with objects from another package
- **Examples of packages: sqlservr.exe, sqlos.dll**

Events

- **Events correspond to well-known points in the code**
  - E.g. a Transact-SQL statement finished executing; a deadlock occurred
- **Events deliver a basic payload of information**
  - The payload is defined by a (versioned) schema of information immediately available to the event
  - 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 (channel, keyword) to allow integration with ETW**

Predicates

- **Predicates are Boolean expressions that define the conditions required for an event to actually fire**
- **Predicates support short-circuit evaluation**
  - The first false evaluation prevents event from firing
- **Predicates can use basic arithmetic operators, or textual comparators for more complex expressions**
- **Predicates can operate on event-column data or global fields**
  - Global fields require synchronous collection of the additional data first
- **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
  - 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

Actions

- **Actions only execute after predicate evaluation determines the event will fire**
- **Actions execute synchronously on the thread that fired the event**
- **Actions collect additional state data to add to the event data**
- **Some actions have side effects like performing a memory dump**
- **Any action may be used with any event, but state data may not be available at the point in code where an event fires**
  - Parallel subtasks may not have the same context as the parent task

Targets

- **Targets are event consumers**
  - Process single events or a buffer full of events
- **Synchronous and asynchronous targets exist**
- **Basic targets collect raw event data**
  - Event File
  - Ring Buffer
- **Aggregating targets aggregate data based on criteria**
  - Event Bucketizer (providing a histogram)
  - Event Counter
  - Event Pairing (which matches events)
- **ETW target allows end-to-end tracing from the application with Windows kernel events**

Types and Maps

- **Types define the type of data for an event column, action, or global predicate source**
- **Types are contained within each individual package and may repeat across 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

Metadata Views

- **sys.dm_xe_packages**
  - Contains an entry for each of the packages registered in the Extended Events engine
  - Each package has 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

Metadata Views (2)

- **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 ETW
    - 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

Summary

- **The package metadata that has been loaded into the Extended Events engine can be queried using the metadata catalog views**
- **Each SQL Server module loads a package that contains the events, predicates, actions, types, and maps associated with the module**
- **Targets and general actions and types are loaded into the engine by the Package0 package by default**
- **Events only provide state information for the point in execution that the event was fired, additional information can be gathered through the use of actions**

- **The next module will look at:**
  - Event session management