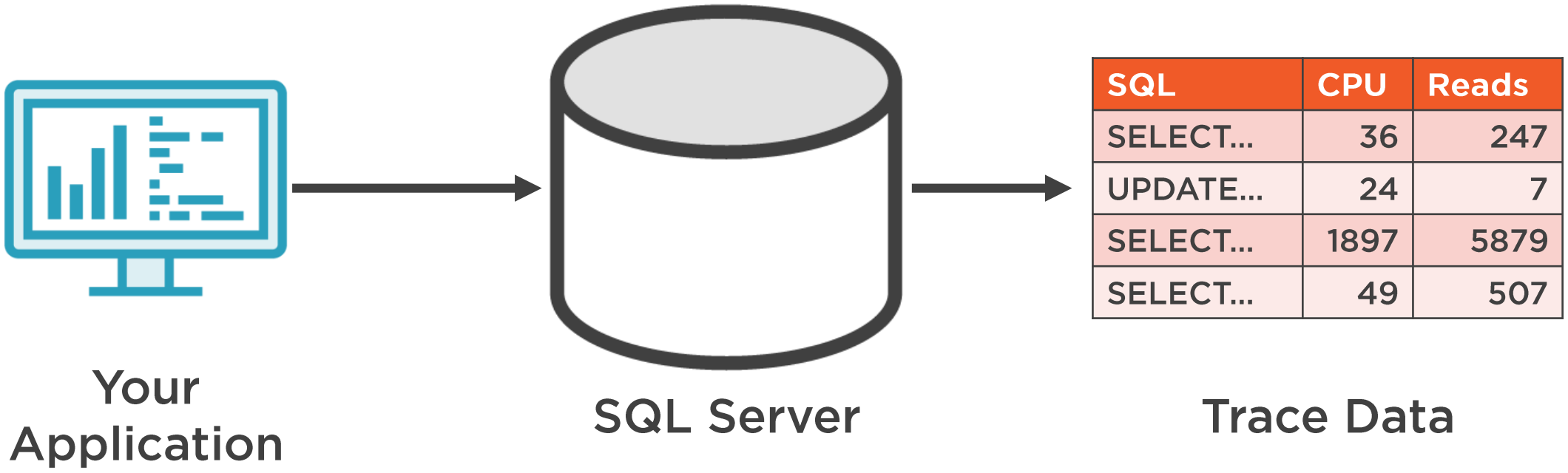
Capturing What Your Application is Doing Inside SQL Server

David Berry
@davidcberry13 buildingbettersoftware.blogspot.com

What is a SQL Trace?



SQL	CPU	Reads
SELECT...	36	247
UPDATE...	24	7
SELECT...	1897	5879
SELECT...	49	507

Understand Program Execution

- What statements are being run?
- In what order?

Troubleshoot Performance Issues

- Stats recorded for each statement

Where Tracing Can Be Used

DEV

Understand how an application works

TEST

Log detailed results from a load test

PROD

Track statements that exceed thresholds

Tracing Tools for SQL Server

SQL Server 2008R2 and prior	SQL Server 2012 and later, SQL Azure
SQL Profiler	SQL Server Extended Events

Required Permissions

Tracing requires high level permissions in SQL Server

Production tracing will likely require a DBA

Understand the capability so you can ask for it

Module Outline

Example traces using SQL Profiler and SQL Server Extended Events

Setting up a trace

Logging events

Filtering trace data

Capturing data for later review

Analyzing trace data

SQL Profiler

Use for SQL Server 2008R2 or earlier

Deprecated in SQL Server 2012, 2014 but still available

Not available on SQL Server 2016, SQL Azure

Setting the Application Name in a Connection String

```
<add name="MyConnectionString"
    connectionString="Server=localhost\sqlexpress;
        Database=<<database name>>;
        User Id=<<username>>;Password=<<password>>;
        Application Name=<<friendly application name>>;"/>
```

SQL Server Extended Events

Use for SQL Server 2012 and later, SQL Azure

Uses Microsofts Event Tracing for Windows (ETW) framework

Access to a richer set of events, more filtering options

Permissions for Extended Events

SQL Server 2008R2	CONTROL SERVER permission required
SQL Server 2012+	ALTER ANY EVENT SESSION permission VIEW SERVER STATE permission needed to use SSMS GUI

Setup Differences for SQL Azure

Most of the setup is the same between SQL Azure and on-premises SQL Server

Differences will be covered in a dedicated clip at module end

Server Side Tracing

More efficient than interactive tracing

Use SQL Profiler to define the trace

Generate SQL file of commands to run trace

Basic Query for Server Side Trace Information

```
-- Get info on all server side traces
SELECT * FROM sys.fn_trace_getinfo(0);
```

Better Query for Server Side Trace Information

```
SELECT
    traceid As TraceId,
    max(case when property = 2 then value end) AS TraceFile,
    max(case when property = 3 then value end) AS MaxSize,
    max(case when property = 4 then value end) AS StopTime,
    max(case when property = 5 then
        case value
            when 1 then 'Running'
            when 0 then 'Stopped'
        end
    end) AS CurrentStatus
FROM sys.fn_trace_getinfo(0)
GROUP BY traceid;
```

Controlling the Trace

```
-- Stop the Trace
exec sp_trace_setstatus @TraceID, 0

-- (Re)start the trace
exec sp_trace_setstatus @TraceID, 1

-- Remove the trace definition from SQL Server
exec sp_trace_setstatus @TraceID, 2
```

Trace Capture Analysis

- Capture all the SQL from a process
- Determine highest average CPU, duration
- Determine which statements had the most *total* CPU usage and time to execute

SQL Azure and Extended Events

Setup Azure storage to write extended events files to

Module Summary

SQL Server tracing allows us to see what is happening inside the database

Capture exact sequence of statements being executed

Capture performance of each statement

Capture how many times a statement has executed

Tooling Support

SQL Profiler	Use for SQL Server 2008R2 and earlier
SQL Server Extended Events	Use for SQL Server 2012 and later, SQL Azure