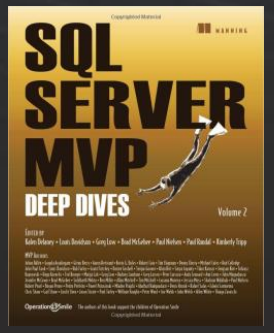
SPR201: Leveraging SQL Server 2012 Features and Tools

Kimberly L. Tripp & Jonathan Kehayias
SQLskills.com

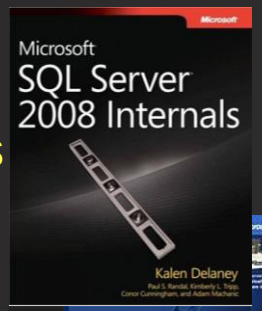
SQLskills.com

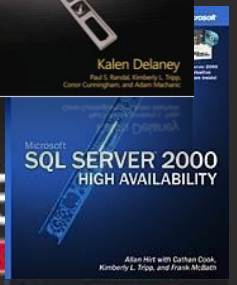
SQL Server CONNECTIONS

Author/Instructor: Kimberly L. Tripp



- Consultant/Trainer/Speaker/Writer
- President/Founder, SYSolutions, Inc.
  - e-mail: Kimberly@SQLskills.com
  - blog: http://www.SQLskills.com/blogs/Kimberly
  - Twitter: @KimberlyLTripp
- Writer/Editor for SQL Magazine www.sqlmag.com
- Author/Instructor for SQL Server Immersion Events
- Instructor for multiple rotations of both the SQL MCM and Sharepoint MCM
- Author/Manager of SQL Server 2005 Launch Content, co-author/Manager for SQL Server 2008 Jumpstart Content, Author/Speaker at TechEd, SQL Connections, SQLPASS, ITForum, Conference Co-chair for SQL Connections
- Author of several SQL Server Whitepapers on MSDN/TechNet: Partitioning, Snapshot Isolation, Manageability, SQLCLR for DBAs
- Author/Presenter for more than 25 online webcasts on MSDN and TechNet (two series and other individual webcasts)
- Co-author MSPress Title: SQL Server 2008 Internals, the SQL Server MVP Project (1 & 2), and SQL Server 2000 High Availability
- Presenter/Technical Manager for SQL Server HOL DVDs
- We love this stuff… feel free to ask questions!





Author/Instructor:
Jonathan Kehayias

- Consultant/Trainer/Author/Speaker
- Principal Consultant, SQLskills.com
  - Email: jonathan@sqlskills.com
  - Blog: http://sqlskills.com/blogs/jonathan
  - Twitter: @SQLPoolBoy

- Contributing Author: SQL Server 2008 Internals and Troubleshooting
- Author: Troubleshooting SQL Server: A Guide for the Accidental DBA
- Microsoft Certified Master: SQL Server 2008, SQL Server MVP
- Author of Using Extended Events whitepaper on MSDN
- Regular presenter at PASS Summit, SQL Saturdays, SQLBits and SQL Connections conferences
- Developer Extended Events Manager SSMS Addin for SQL Server 2008 open source project on Codeplex
- Course instructor for *Microsoft Certified Master – SharePoint* qualification

SQL Server CONNECTIONS

Overview

- Tour of the new Tools
- AlwaysOn feature overview
  - Availability Groups and Contained Databases
  - Failover Clustering changes
  - ONLINE index rebuilds with LOBs
- Columnstore indexes
- Partitioning
- Extended Events UI in Management Studio
- Distributed Replay

Conference Sessions

- **MSSQL05:** Team Database Development with SSDT, Visual Studio and Team Foundation 2012
- **MSSQL07:** Database Development with SSDT and Visual Studio 2012
- **SQL313:** Real-Life SQL Server 2012 Availability Group Lessons Learned

SQL Server CONNECTIONS

SQL Server Management Studio

- Shortcuts
  - Removed the SQL Server 2000 keyboard shortcuts (bummer!)
  - Defaults to new combinations of shortcuts; can set them to Visual Studio 2010 compatible
- Intellisense
  - Much better – do not need the start of the word for searching
  - ~~Allowed when SQLCMD mode set to the default~~
- Copy/paste cycling
- Multi-row editing (region editing)
- Increased view option (*great for presenting*)

Demo

A quick look at a couple of cool things in the new tools

SQL Server Management Studio
Debugging

- Support for SQL Server 2005 SP2+
- If connected to a non-local instance must allow remote debugging. For more information, see Configure the Transact-SQL Debugger
- Must run as a sysadmin fixed server role
- Debugging is not possible when the server is in single-user mode
- Improvements for breakpoints:
  - Breakpoint actions, conditions, filters, hit count

SQL Server Management Studio

Restore/Recovery

- Database Recovery Advisor
- Restore plans – including more complicated forks – are supported
- Point-in-time recovery using a visual timeline
- Page restore UI

Demo

Advanced Restore Scenarios

Overview

- Tour of the new Tools
- **AlwaysOn feature overview**
  - **Availability Groups and Contained Databases**
  - **Failover Clustering changes**
  - **ONLINE index rebuilds with LOBs**
- Columnstore indexes
- Partitioning
- Extended Events UI in Management Studio
- Distributed Replay

Availability Group Overview

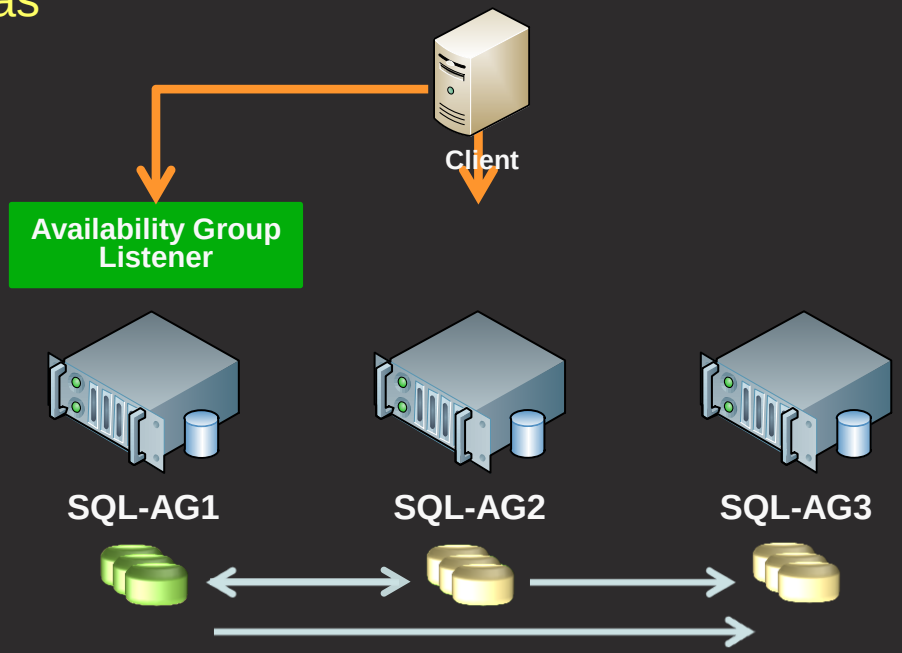
- Multiple database coordinated failover for applications that require multiple databases on a single instance (e.g. SharePoint)
- New Availability Group Listener provides automated connection redirection for failovers and read only routed connections
- Readable secondary replicas allow offloading readable workloads to additional servers in the environment
- Simplified configuration wizards, PowerShell integration and Availability Group Dashboard for monitoring

Availability Group Architecture

- Leverages WSFC for health detection, failover coordination, and to provide the Availability Group Listener functionality for connectivity across any node
- Up to 5 nodes per Availability Group, 1 Primary and 4 Secondary, allowing two Secondary nodes to be synchronous with automated failover in the environment
- Asynchronous replicas can exist in remote data centers for disaster recovery without requiring secondary technologies like Log Shipping

Application Connection Failover

- Clients connect using the Availability Group Listener Virtual Name
- During a failover the Listener is taken offline and moved to a new node in the Availability Group
- WSFC tells the AG Resource DLL to bring the AG online on a new node
- Clients reconnect as soon as the Listener is back online
- WSFC provides notification of the new primary server to all secondary replicas

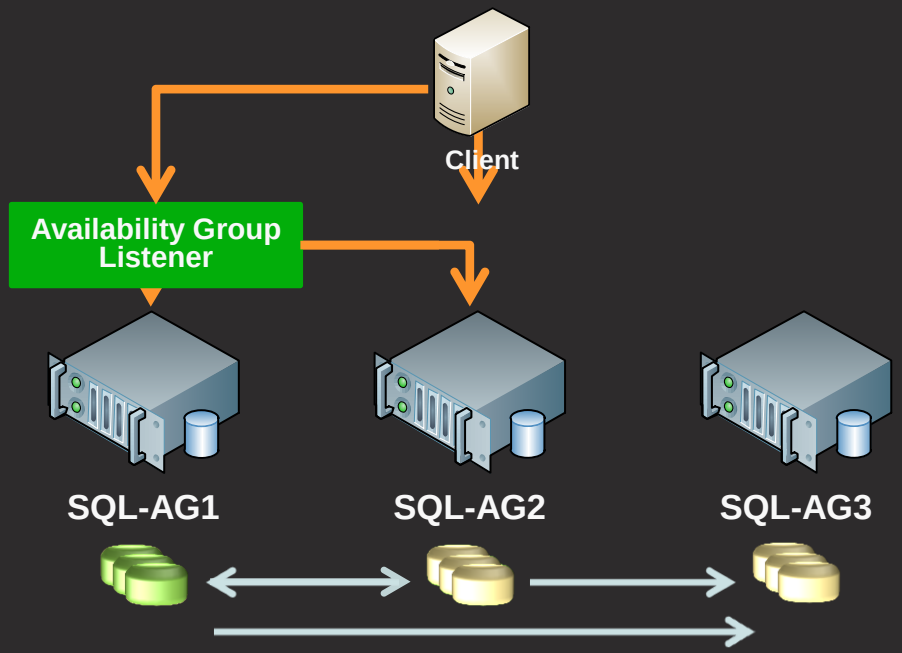


Readable Secondaries

- Allow offloading of readonly workloads to secondary replicas in the environment
  - Uses snapshot isolation to avoid blocking replay of transactions on the secondary
  - If additional statistics are required, they are created as temporary objects in tempdb
  - If additional indexes are required, the must be created in the primary database
- Allows offloading of transaction log backups to secondary replicas
  - All log backups form a single chain regardless of what server the backup occurred from

Application Connection Failover

- Clients connect using the Availability Group Listener Virtual Name and specify the Application Intent=ReadOnly parameter
- During a failover the ReadOnly connections are automatically rerouted by the Listener to a new node if the current node becomes the Primary



Demo

Readable Secondaries

Partial Database Containment

- Contained Database Authentication stores credentials inside the database removing the need to manually manage logins across environments or Availability Group nodes
  - Requires enabling the 'contained database authentication' sp_configure option for the instance
  - Individual databases can be converted to a contained database using ALTER DATABASE <database_name> SET CONTAINMENT = PARTIAL
  - Existing SQL Server Logins can be converted to a contained database user with the sp_migrate_user_to_contained system stored procedure
  - New contained database users can be created using the new DDL feature CREATE USER <username> WITH PASSWORD = ‘password’

Partial Database Containment

- Defines an application boundary for development that reduces the instance level dependencies as a part of the development considerations
  - Temporary Tables
    - Use the collation of the contained database instead of the tempdb collation
    - Cannot contain named constraints, or refer to user-defined types, XML schema collections, or user-defined functions
  - Instance level features like replication, change data capture, or change tracking can not be used
- Feature usage that crosses the application boundary can be tracked with sys.dm_db_uncontained_entities or the database_uncontained usage Extended Event
  - Any Dynamic SQL used will be flagged as uncontained usage since its actual containment cannot be determined until run time

Demo

Configuring Partial Containment

SQL Server CONNECTIONS

Failover Clustering Enhancements

- Multi-site clustering across subnets
  - IP Address now a AND/OR dependency
  - DNS HostRecordTTL settings considerations
- Flexible failover policy
  - Provides predictable failover configuration to reduce false positive IsAlive/LooksAlive checks for unhealthy instances
- Support for network attached storage (NAS) using SMB
  - Failover Clustering no longer requires a SAN due to rewrites of the SMB stack that provide as much as 97% of DAS perfomance
- Support for tempdb on a local drive

Flexible Failover

- Based on the output of sp_server_diagnostics

Level	Failover Type	Description	Example
0	No automatic failover or restart	Never Fail, Log Only	
1	Failover or restart on SQL Server down	Server Down	Instance Not Starting
2	Failover or restart on SQL Server unresponsive	Server Hang	17884 Deadlock
3	Failover or restart on critical SQL Server errors	System Unhealthy	Stack Dumps Occurring
4	Failover or restart on moderate SQL Server errors	Resource Unhealthy	Low on Memory
5	Failover or restart on any qualified failure	QP Unhealthy	Queries Stalled

Configuring Failover Clustering on SMB Storage

- Skip Cluster Validation Disk checks
- Use Node and File Share quorum configuration if an even number of nodes are participating in the cluster
  - Requires that the Virtual Computer Object (VCO) for the cluster have Full Control over the share
- During SQL Server cluster installation specify UNC paths for the file location configuration
  - Requires that the service account for the Database Engine have Full Control over the share locations

- Blog Post: http://tinyurl.com/2012FConSMB

Demo

Failover Clustering Enhancements

ONLINE Index Operations

- SQL Server 2005 Enterprise Edition:
  - Online index operations for complete index-level operations as long as the index does not include a LOB column
    - If the table has a LOB column defined then the clustered index cannot be built or rebuilt using online operations
    - If the nonclustered index INCLUDEs a LOB column then only OFFLINE builds/rebuilds are possible
  - Offline index operations for partition-level operations

ONLINE Index Operations

- SQL Server 2012 Enterprise Edition:
  - Online index operations for complete index-level operations even if a LOB column exists
    - Trick: Can move LOB data if you change the table's definition from non-partitioned to partitioned or from one partition scheme to another
      - See SQLServerPro Q&A article: **What about moving LOB data?** (Mar 2012)
  - **NOT CHANGED:** Offline index operations for partition-level operations (bummer)

Demo

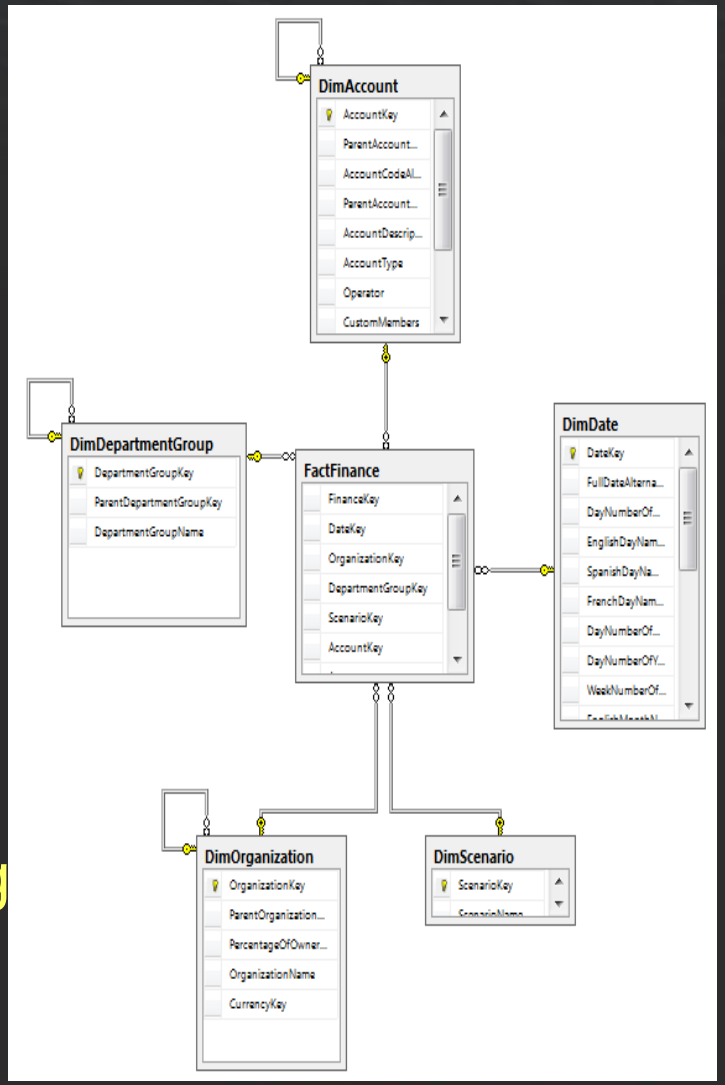
ONLINE Index Operations

Overview

- Tour of the new Tools
- AlwaysOn feature overview
  - Availability Groups and Contained Databases
  - Failover Clustering changes
  - ONLINE index rebuilds with LOBs
- **Columnstore indexes**
- Partitioning
- Extended Events UI in Management Studio
- Distributed Replay

Improved Data Warehouse Query performance

- Columnstore indexes provide an easy way to significantly improve data warehouse and decision support query performance against very large data sets
- Performance improvements for “typical” data warehouse queries from 10x to 100x
- Ideal candidates include queries against star schemas that use filtering, aggregations and grouping against very large fact tables



What Happens When…

- You need to execute high performance DW queries against very large data sets?
  - In SQL Server 2008 and SQL Server 2008 R2
    - OLAP (SSAS) MDX solution
    - ROLAP and T-SQL + intermediate summary tables, indexed views and aggregate tables
      - Inherently inflexible
      - Requires more specific/focused tuning – often requiring changes over time

What Happens When…

- You need to execute high performance DW queries against very large data sets?
  - In SQL Server 2012
    - You can create a columnstore index on a very large fact table referencing all columns with supporting data types
      - Utilizing T-SQL and core Database Engine functionality
      - Minimal query refactoring or intervention
      - Easy, a single index creates all of the required “column stores”
    - Upon creating the columnstore index, your table becomes “read only” – but you can still use partitioning to switch in and out data OR drop/rebuild indexes periodically

How Are These Performance Gains Achieved?

- Two complimentary technologies:
  - **Storage**
    - Data is stored in a compressed columnar data format (stored by column) instead of row store format (stored by row).
      - Columnar storage allows for less data to be accessed when only a sub-set of columns are referenced
      - Data density/selectivity determines how compression friendly a column is – example “State” / “City” / “Gender”
      - Translates to improved buffer pool memory usage
  - **New “batch mode” execution**
    - Data can then be processed in batches (1,000 row blocks) versus row-by-row
    - Depending on filtering and other factors, a query may also benefit by “segment elimination” - bypassing million row chunks (segments) of data, reducing I/O

Column vs. Row Store

Row Store (Heap / B-Tree)

- **Leaf-level pages consist of multiple “rows” as defined by the table or index**
- **Support compression but might not be as compressible**

data page 1000

Product ID	OrderDate	Cost
310	20010701	2171.29
311	20010701	1912.15
312	20010702	2171.29
313	20010702	413.14

data page 1001

Product ID	OrderDate	Cost
314	20010701	333.42
315	20010701	1295.00
316	20010702	4233.14
317	20010702	641.22

Column vs. Row Store

Column Store

- **Only that column is stored on the same page – potentially HIGHLY compressible!**
- **Data is segmented into groups of 1000 rows for better batch processing**
- **SQL Server can do "segment elimination" (just like partition elimination) and further reduce the batch(es) processed!**

data page 2000

Product ID
310
311
312
313
314
315
316
317
318
319
320
321

data page 2001

OrderDate
20010701
…
20010702
…
…
20010703
…
…
…
…
20010704
…

data page 2002

Cost
2171.29
1912.15
2171.29
413.14
333.42
1295.00
4233.14
641.22
24.95
64.32
1111.25

Batch Mode

- Allows processing of 1,000 row blocks as an alternative to single row-by-row operations
  - Enables additional algorithms that can reduce CPU overhead significantly
  - Batch mode “segment” is a partition broken into million row chunks with associated statistics used for Storage Engine filtering
- Batch mode can work to *further* improve query performance of a columnstore index, but this mode isn’t always chosen:
  - Some operations aren’t enabled for batch mode:
    - E.g. outer joins to columnstore index table / joining strings / NOT IN / IN / EXISTS / scalar aggregates
  - Row mode might be used if there is SQL Server memory pressure or parallelism is unavailable
  - Confirm batch vs. row mode by looking at the graphical execution plan

Columnstore format + batch mode Variations

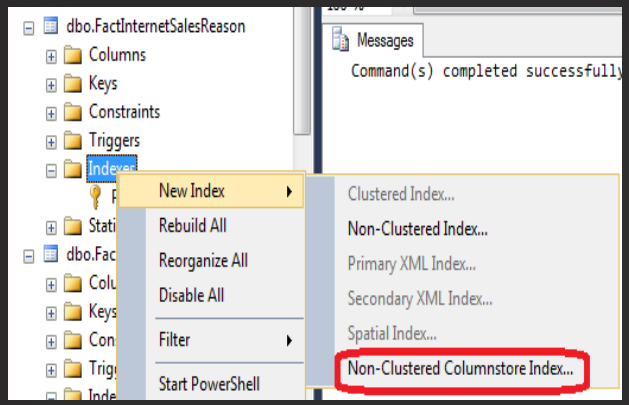
- Performance gains can come from a combination of:
  - Columnstore indexing alone + traditional row mode in QP
  - Columnstore indexing + batch mode in QP
  - Columnstore indexing + hybrid of batch and traditional row mode in QP

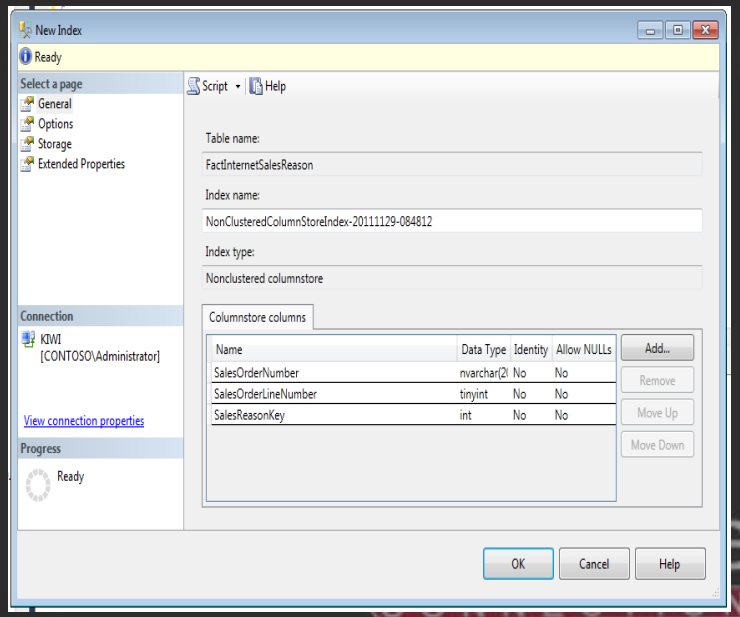
Creating a Columnstore index

T-SQL

```
CREATE NONCLUSTERED COLUMNSTORE INDEX [IX_CS_FactProductInventory]
ON dbo.FactProductInventory
(
        ProductKey,
        DateKey,
        UnitCost,
        UnitsIn,
        UnitsOut,
        UnitsBalance
);
```

SSMS





Good Candidates for Columnstore Indexing

- Table candidates:
  - Very large fact tables (for example – billions of rows)
  - Larger dimension tables (millions of rows) with compression friendly column data
  - If unsure, it is easy to create a columnstore index and test the impact on your query workload
- Query candidates (against table with a columnstore index):
  - Scan versus seek (columnstore indexes don't support seek operations)
  - Aggregated results far smaller than table size
  - Joins to smaller dimension tables
  - Filtering on fact / dimension tables – star schema pattern
  - Sub-set of columns (being selective in columns versus returning ALL columns)
  - Single-column joins between columnstore indexed table and other tables

    (NOTE: joins to smaller dimensions NOT large tables)

Defining the Columnstore Index

- Index type
  - Columnstore indexes are always non-clustered and non-unique
  - They cannot be created on views, indexed views, sparse columns
  - They cannot act as primary or foreign key constraints
- Column selection
  - Unlike other index types, there are no “key columns”
    - Instead you choose the columns that you anticipate will be used in your queries
    - Up to 1,024 columns – and the ordering in your CREATE INDEX doesn’t matter
    - No concept of “INCLUDE”
    - No 900 byte index key size limit
- Column ordering
  - Use of ASC or DESC sorting not allowed – as ordering is defined via columnstore compression algorithms

Supported Data Types

- Char / nchar / varchar / nvarchar
  - (max) types, legacy LOB types and FILESTREAM are not supported
- Decimal/numeric
  - Precision greater than 18 digits NOT supported
- Tinyint, smallint, int, bigint
- Float/real
- Bit
- Money, smallmoney
- Date and time data types
  - Datetimeoffset with scale > 2 NOT supported

Maintaining Data in a Columnstore Index

- Once built, the table becomes “read-only” and INSERT/UPDATE/DELETE/MERGE is no longer allowed
- ALTER INDEX REBUILD / REORGANIZE not allowed
- Other options *are* still supported:
  - Partition switches (IN and OUT)
  - Drop columnstore index / make modifications / add columnstore index
  - UNION ALL (but be sure to validate performance)

Limitations

- Columnstore indexes cannot be used in conjunction with
  - Change Data Capture and Change Tracking
  - Filestream columns (supported columns from same table are supported)
  - Page, row and vardecimal storage compression
  - Replication
  - Sparse columns
- You can prevent a query from using the columnstore index using the IGNORE_NONCLUSTERED_COLUMNSTORE_INDEX query hint

Demo

Columnstore Indexes

SQL Server CONNECTIONS

Columnstore Summary

- SQL Server 2012 offers significantly faster query performance for data warehouse and decision support scenarios
  - 10x to 100x performance improvement depending on the schema and query
    - I/O reduction and memory savings through columnstore compressed storage
    - CPU reduction with batch versus row processing, further I/O reduction if segmentation elimination occurs
  - Easy to deploy and requires less management than some legacy ROLAP or OLAP methods
    - No need to create intermediate tables, aggregates, pre-processing and cubes
  - Interoperability with partitioning

Overview

- Tour of the new Tools
- AlwaysOn feature overview
  - Availability Groups and Contained Databases
  - Failover Clustering changes
  - ONLINE index rebuilds with LOBs
- Columnstore indexes
- **Partitioning**
- Extended Events UI in Management Studio
- Distributed Replay

Improvements for Table Partitioning

- SQL Server 2012 RTM supports up to 15,000 partitions
  - No need for a service pack to gain functionality
- Partition statistics are created using a row sub-set sampling when an index is rebuilt or created - versus scanning all rows to create the statistics
- Additional partition management wizard options can assist with executing or scripting out common partition operations
- Partitioning can be used in conjunction with tables that have a columnstore index in order to switch in and out data

What Happens When…

- You need to partition data by day for 3 years of data or more? Or you need to partition data by hour for a year's worth of data?
  - In SQL Server 2008 & SQL Server 2008 R2
    - Limited to 1,000
      - Unless you installed 2008 SP2 or 2008 R2 SP1 – which allowed for 15,000 partitions when enabled via sp_db_increased_partitions
        - » This prevented moving from 2008 SP2 to 2008 R2 RTM
        - » Also prevented moving SQL Server 2008 SP2 database with 15,000 partitions enabled to SQL Server 2008 R2
        - » Created other restrictions for Log Shipping, Database Mirroring, Replication, SSMS manageability
  - In SQL Server 2012
    - 15,000 partitions are supported in RTM

15,000 Partitions

- You now have the option – as appropriate
  - Flexibility to partition based on common data warehousing increments (hours / days / months) without hitting the limit
    - This doesn't remove the need for an archiving strategy or mindful planning
  - You have native support for log shipping, availability groups, database mirroring, replication and SSMS management
- Exceptions:
  - > 1000 partitions for x86 is permitted but not supported
  - > 1000 partitions for non-aligned indexes is permitted but not supported
  - For both exceptions – the risk is in degraded performance and insufficient memory

NOTE: Statistics are still table-level not partition-level

What Happens When…

- Your partitioned index is rebuilt or created:
  - In SQL Server 2008 and SQL Server 2008 R2
    - All table rows are scanned in order to create the statistics histogram
  - In SQL Server 2012
    - A default sampling algorithm is used instead
      - May or may not have an impact on performance
      - You can still choose to scan all rows by using CREATE STATISTICS or UPDATE STATISTICS with FULLSCAN

2008 ->

Results | Messages

	Name	Updated	Rows	Rows Sampled	Steps	Density
1	ClusteredIndex-20111201-133710	Dec 1 2011 1:46PM	483184	483184	199	0.003460169

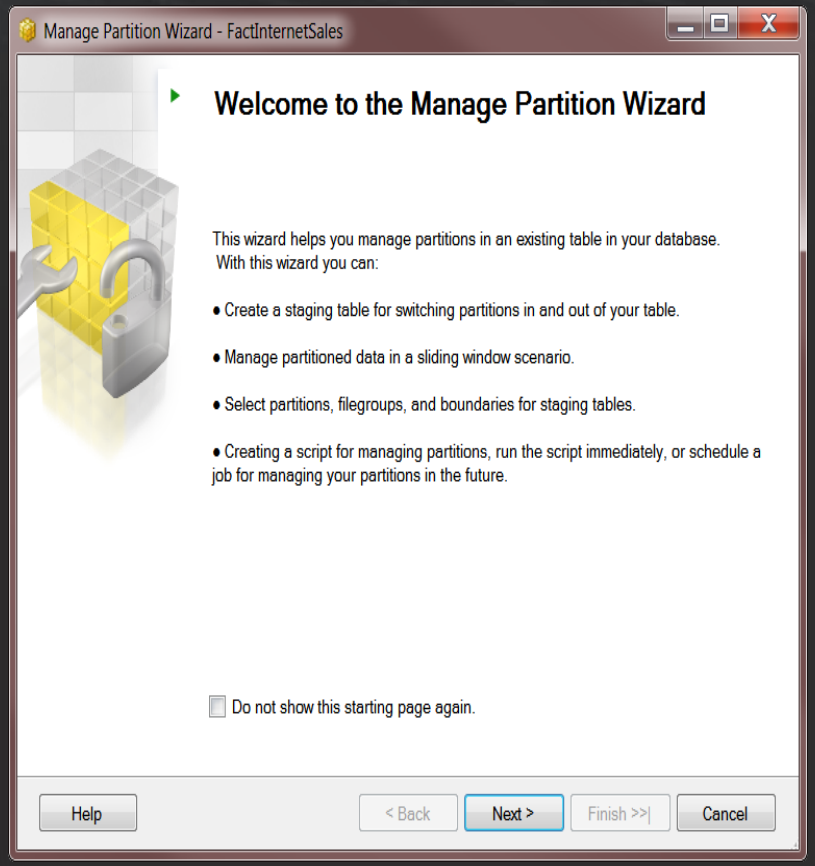
2012 ->

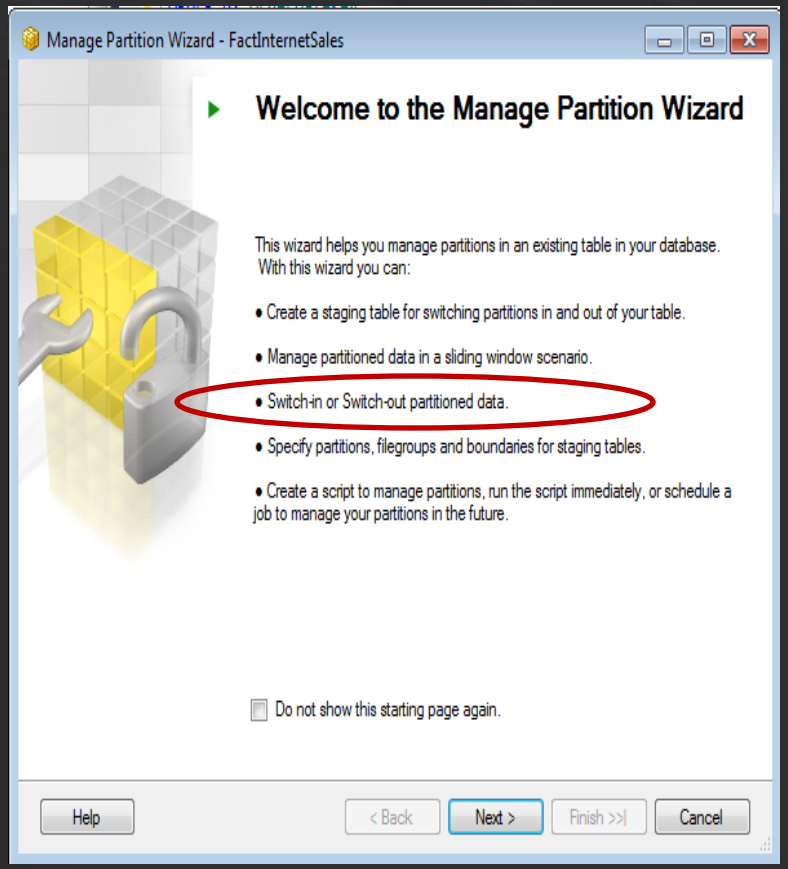
Results | Messages

	Name	Updated	Rows	Rows Sampled	Steps	Density
1	ClusteredIndex-20111201-133710	Dec 1 2011 1:44PM	483184	60143	200	0.01499414

SQL Server CONNECTIONS

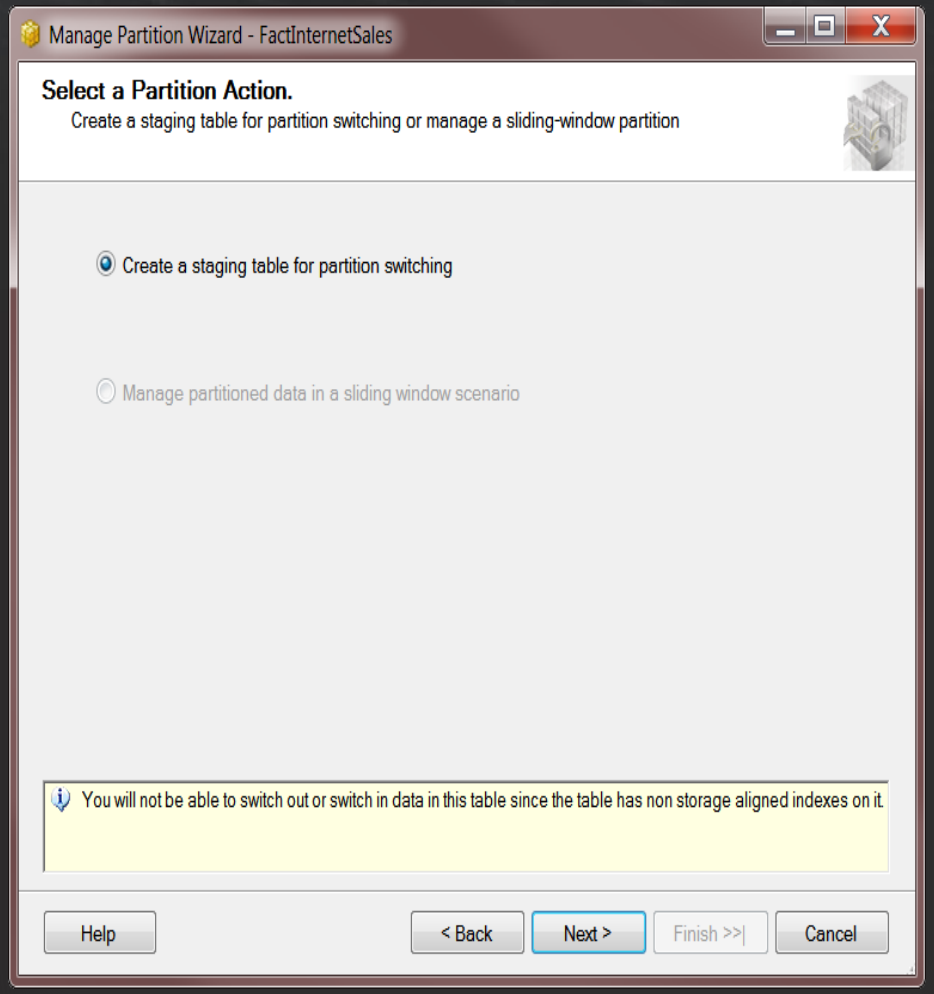
Enhanced Manage Partition Wizard

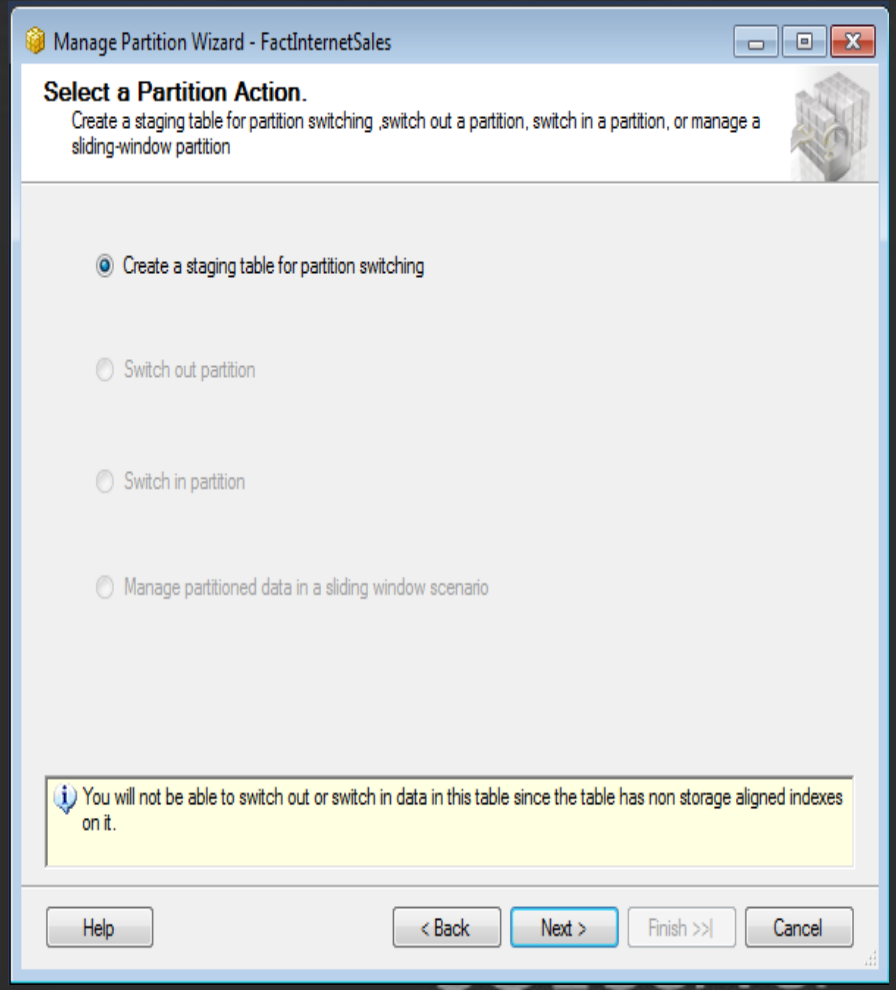




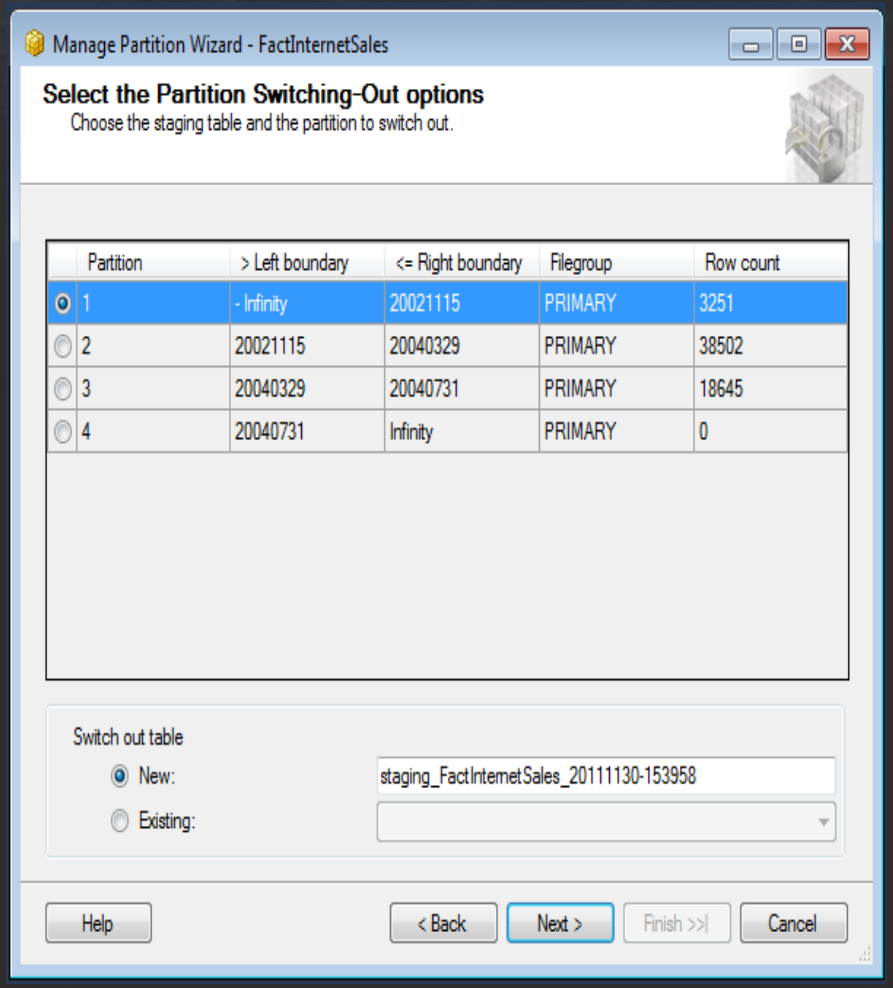
SQL Server CONNECTIONS

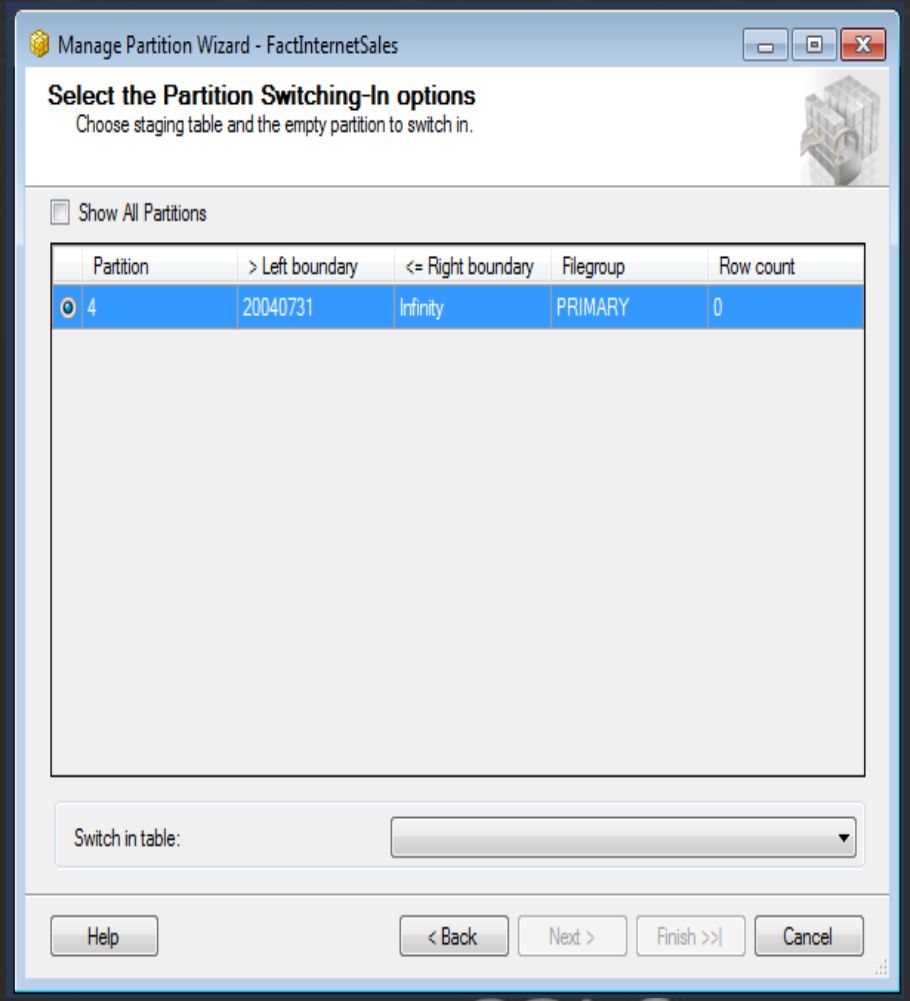
Manage Partition Wizard





Switch Out / In Partition





SQL Server CONNECTIONS

Partitioning Summary

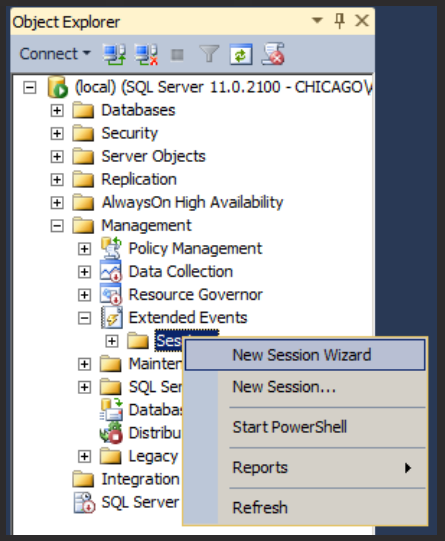
- An increased number of partitions, helping address common data warehouse requirements
- Prevention of lock starvation during SWITCH operations
- Reduced statistics generation footprint (not scanning ALL rows by default)
- An enhanced manageability experience, enabling wizard-based SWITCH IN and SWITCH OUT assistance

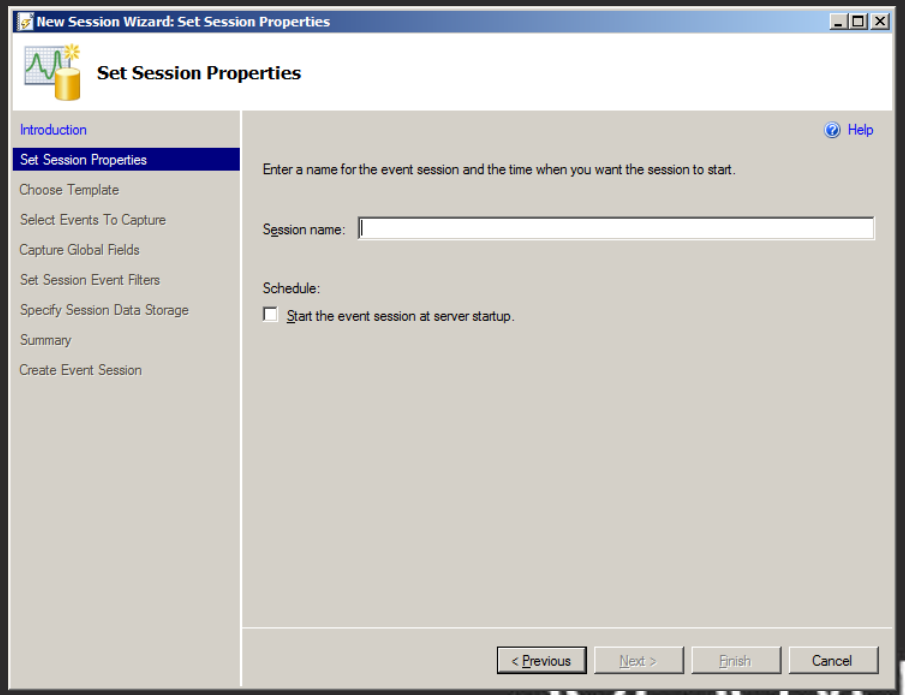
Overview

- Tour of the new Tools
- AlwaysOn feature overview
  - Availability Groups and Contained Databases
  - Failover Clustering changes
  - ONLINE index rebuilds with LOBs
- Columnstore indexes
- Partitioning
- **Extended Events UI in Management Studio**
- Distributed Replay

Extended Events UI

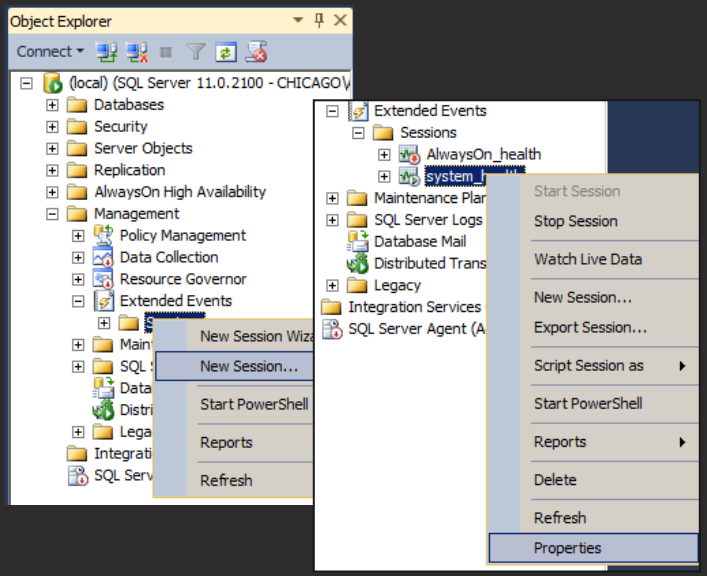
- New Session Wizard
  - Create new sessions rapidly from templates or by using a minimal set of configurable options

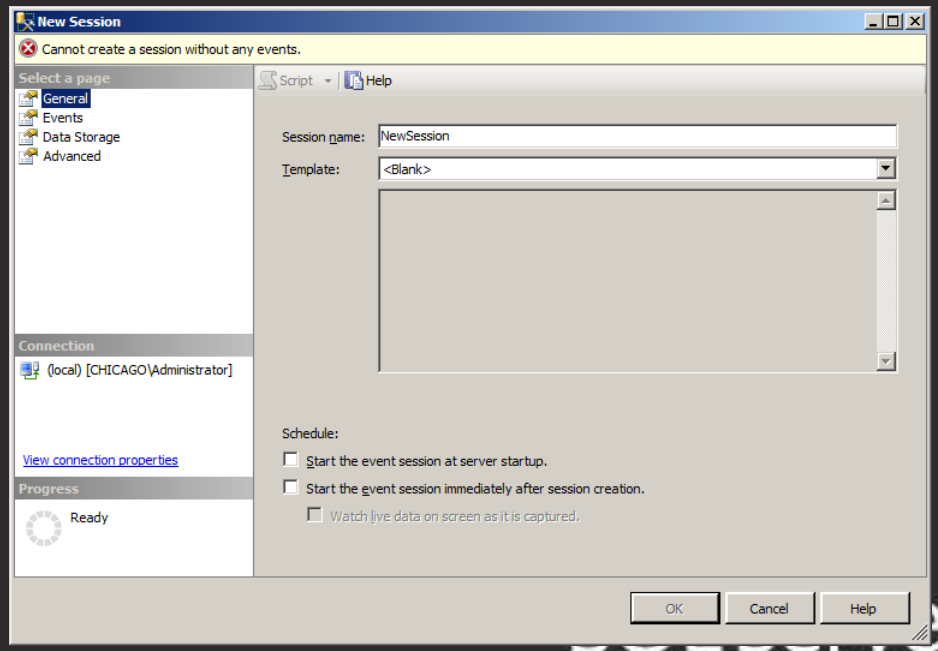




Extended Events UI

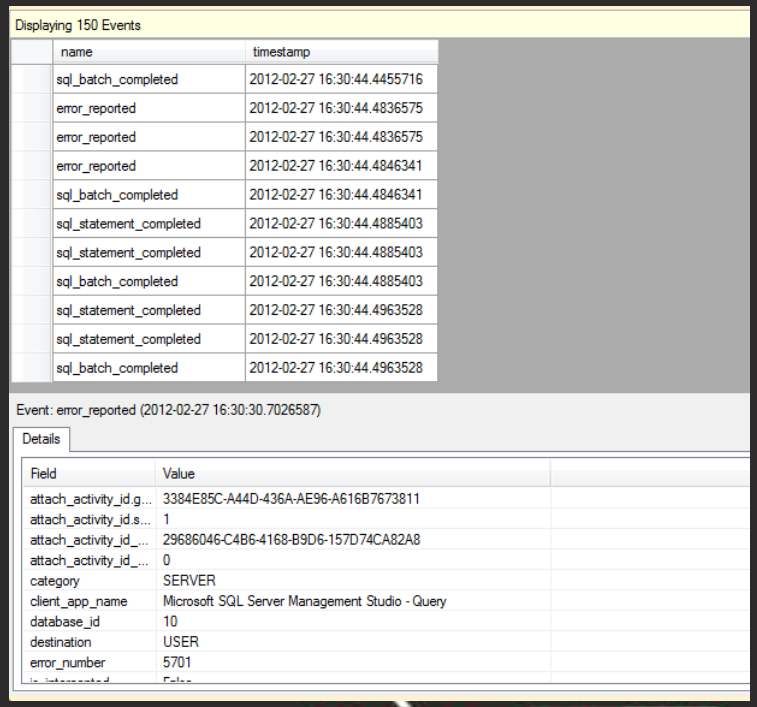
- New Session Dialog
  - Used for advanced session creation using all available options and targets
  - Used for modifying existing event sessions





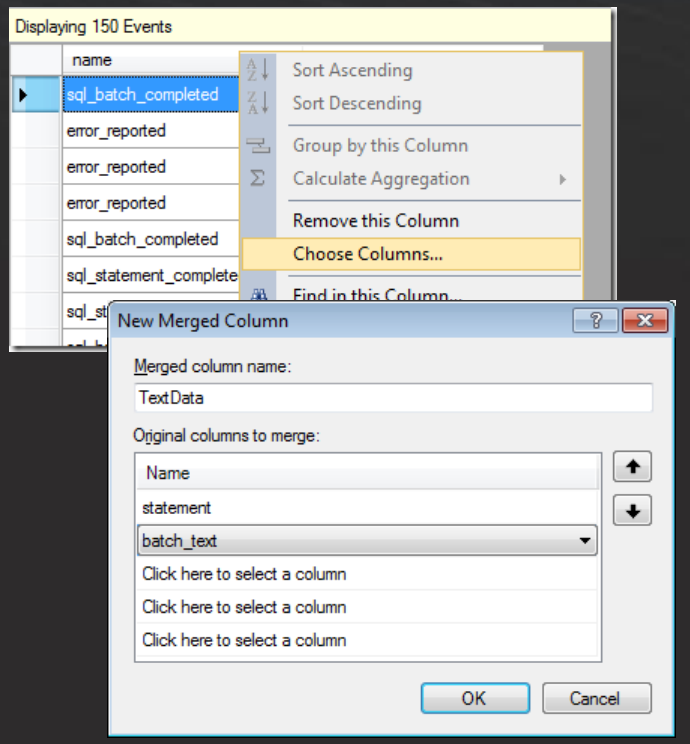
Live Data Viewer

- 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



Data Viewer Customization

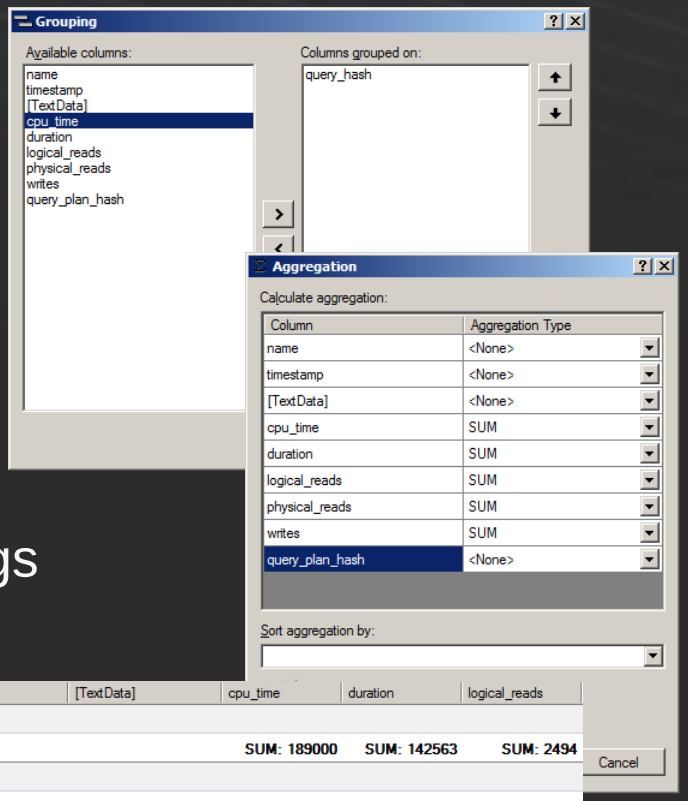
- Default view is limited but columns can be added using the column chooser
  - New Merged columns can be created to track similar information provided by different events in a single column in the grid view
- Display settings can be saved for reuse with later Event Sessions or event_file processing

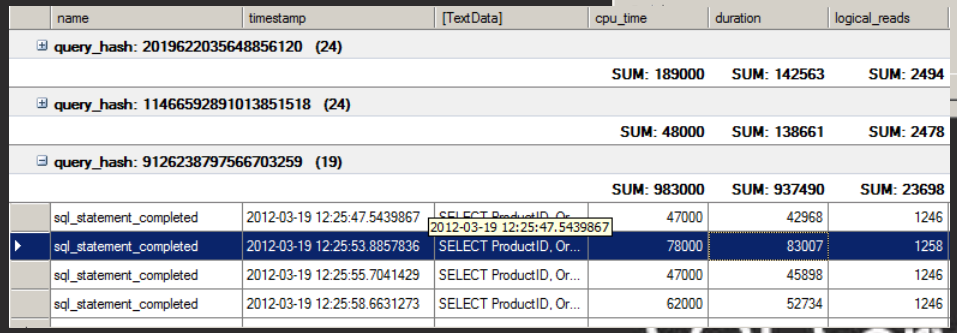


SQL Server CONNECTIONS

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





SQL Server CONNECTIONS

Demo

Extended Events UI

Overview

- Tour of the new Tools
- AlwaysOn feature overview
  - Availability Groups and Contained Databases
  - Failover Clustering changes
  - ONLINE index rebuilds with LOBs
- Columnstore indexes
- Partitioning
- Extended Events UI in Management Studio
- **Distributed Replay**

SQL Server CONNECTIONS

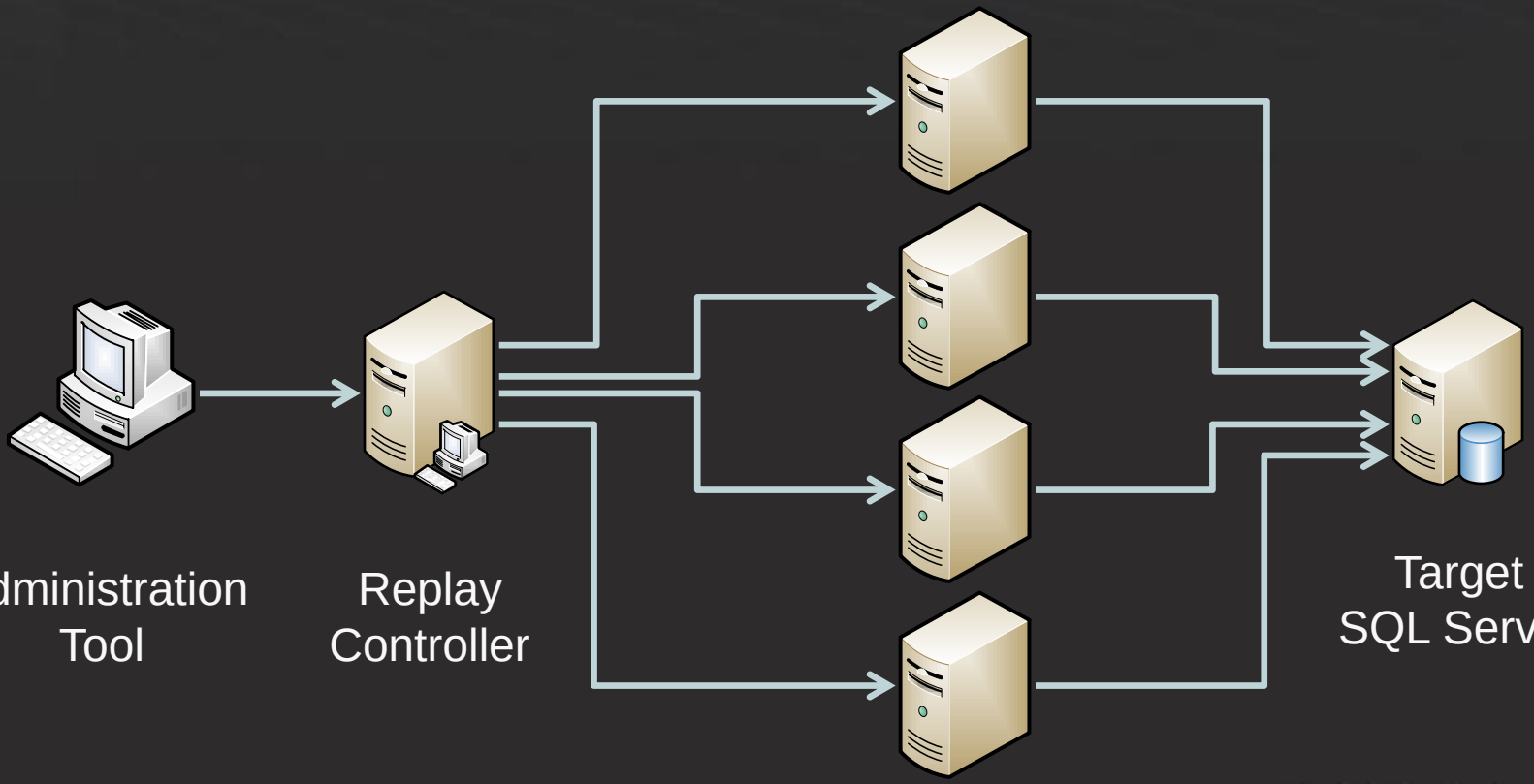
Distributed Replay

- New feature in SQL Server 2012 to help you assess the impact of hardware and operating system upgrades, as well as tuning and code changes within your SQL Server database
- More scalable than SQL Server Profiler or RML Utilities OSTRESS tool by allowing replay to be performed from multiple workstations to better simulate mission critical workloads
- Good for application compatibility, performance/stress, and scalability testing an environment or configuration

Components

- Distributed Replay administration tool
  - The console application used to communicate with the distributed replay controller to control the distributed replay
- Distributed Replay controller
  - A single machine in the environment that controls the actions of the distributed replay clients
- Distributed Replay clients:
  - One or more machines, up to 16, used to replay workloads against an instance of SQL Server
- Target server
  - The instance of SQL Server that the Distributed Replay clients use to replay the trace data against

Distributed Replay Components



Requirements

- The services must run under domain user accounts, local users are not supported
- Input trace data can be from SQL Server 2005+
- Target server must be SQL Server 2008+
- Only one controller can be used within a single Distributed Replay environment, and clients can only be connected to a single controller at a time

Replay Process

- Start controller service
- Preprocess trace file for replay using the DReplay.Exe command line executable
- Start client services and validate registration in the controller
- Configure Replay options in DReplay.Exe.Replay.Config file
- Perform replay using DReplay.Exe command line executable

Replay Options (DReplay.Exe.Replay.Config)

- Connect Time
  - Amount of time the trace should wait between the trace starting and the logon event starting
- Think Time
  - Amount of time between the completion of the previous event and the start of the next event
- MaxIdleTime
  - Amount of time the trace process can be idle before skipping to the next runnable command

Replay Options (DReplay.Exe.Replay.Config)

- HealthmonInterval
  - Tells the replay process how often to resolve deadlocks created by the replay process
- QueryTimeout
  - The number of seconds to wait for the database engine to return the first record before terminating a query

Demo

Distributed Replay

Review

- Tour of the new tools
- AlwaysOn feature overview
  - Availability Groups and Contained Databases
  - Failover Clustering changes
  - ONLINE index rebuilds with LOBs
- Columnstore indexes
- Partitioning
- Extended Events UI in Management Studio
- Distributed Replay

Thank you!

Please fill out a session evaluation form drop it off at the conference registration desk.

SPR201: Leveraging SQL Server 2012 Features and Tools

Questions?

SQLskills.com

SQL Server CONNECTIONS