Microsoft

SQL Server 2014 Mission Critical Performance

(Level 300 Deck)

Microsoft

SQL Server 2014 Mission Critical Performance

(Level 300 Deck)

Microsoft

SQL Server 2014 Mission Critical Performance

(Level 300 Deck)

SQL Server 2014 Mission Critical Performance

Level 300

Microsoft

SQL Server 2014 and the data platform

Mission-critical performance

Faster insights from any data

Platform for hybrid cloud

Mission-critical performance

In-memory built-in

Average 10x faster for new and existing SQL Server apps

Secure and scalable

Most secure with enterprise scale using Windows Server

High availability

The 9's you need, with AlwaysOn

Mission-critical support

Live support designed for mission-critical solutions

In-memory built-in

Built-in

On average 10x faster, without need to rewrite entire app

Uses full Microsoft SQL Server capabilities

Flexible

Selects only highly-utilized tables to be in-memory

Optimizes in-memory to fit existing hardware

Spans all workloads

In-memory performance across OLTP, data warehouses, and BI

All in a single SKU

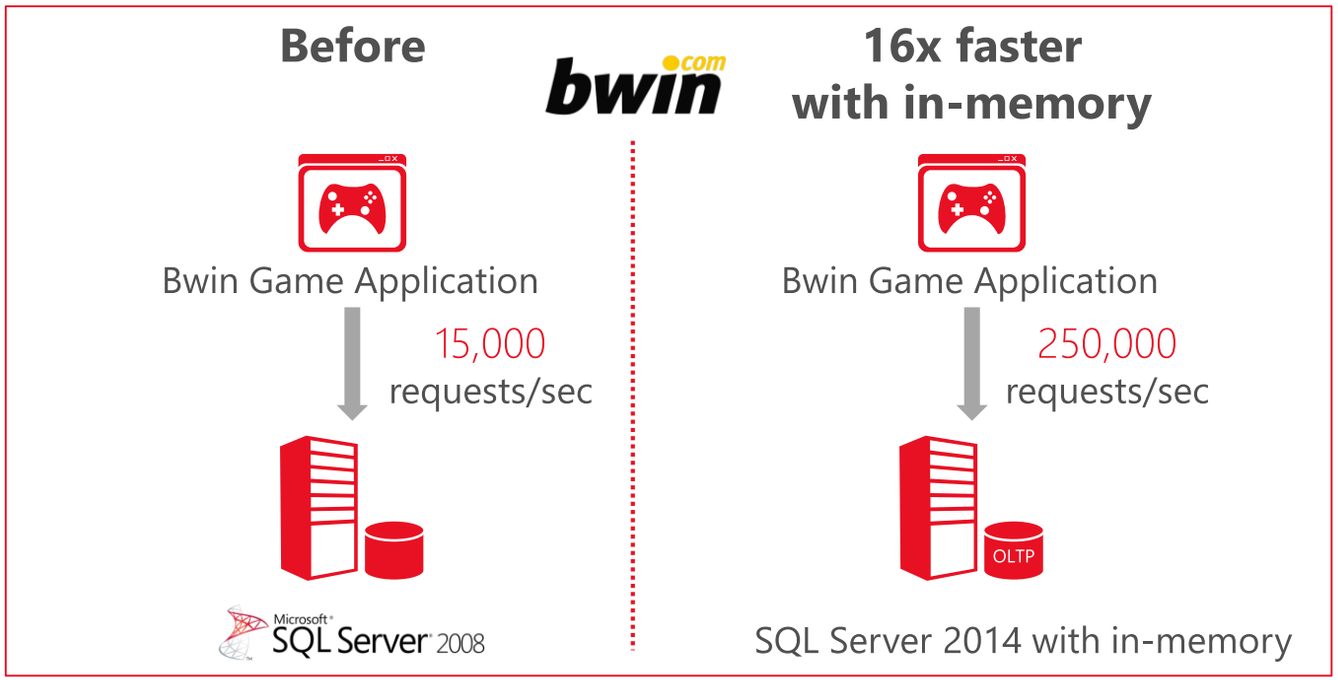
Key Features

New In-Memory OLTP

Enhanced In-Memory Columnstore for DW

In-Memory BI with PowerPivot

Buffer Pool Extension to SSDs and Enhanced Query Processing



Our Strategy for In-Memory Computing

- Deliver end-to-end solution, with frictionless scale and breadth of delivery model
- Built into your existing platform, and seamlessly spans memory-only and hybrid environments
- Business Acceleration via a high velocity virtuous cycle from raw data to insights to business transformation

Microsoft in-memory technologies

OLTP

- In-Memory OLTP

Microsoft SQL Server 2014
Community Technology Preview 2 (CTP2)

DW & Analytics

- In-Memory Analytics in PowerPivot for Excel
- In-Memory Analytics in SQL Server Analysis Services
- In-Memory DW, Fast Track Data Warehouse, and Parallel Data Warehouse

Microsoft SQL Server 2014
Community Technology Preview 2 (CTP2)

Microsoft SQL Server 2012

Office Excel 2013

Streaming Data

- StreamInsight

Microsoft SQL Server 2014
Community Technology Preview 2 (CTP2)

Microsoft SQL Server 2012

Microsoft

In-Memory OLTP

Microsoft

In-Memory OLTP

In-Memory OLTP

Benefits

Low latency

Up to 30 times the improvement in performance

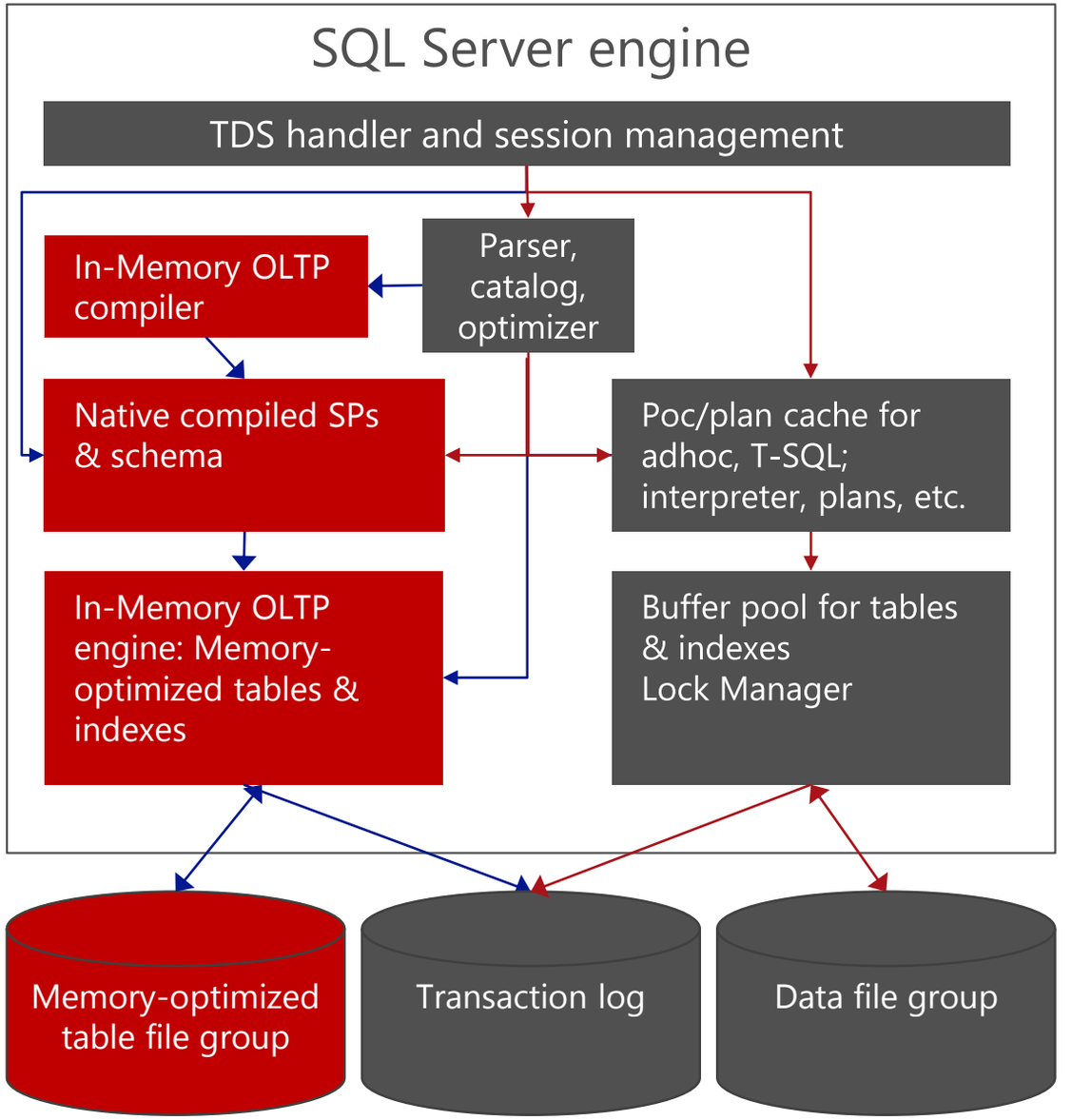
2 to 5 times the improvement in scalability

Takes advantage of investments in Microsoft SQL Server

How it works

New high-performance, memory-optimized online transaction processing (OLTP) engine integrated into SQL Server and architected for modern hardware trends

- Integrated into SQL Server relational database
- Full ACID support
- Memory-optimized
- Non blocking multi-version optimistic concurrency control (no locks or latches)
- T-SQL compiled to native code



Suitable Application Characteristics

- Application is suited for in-memory processing
  - All performance critical data already fits in memory
  - Transaction locking or physical latching causing stalls and blocking
- Application is “OLTP-Like”
  - Relatively short-lived transactions
  - High degree of concurrent transactions from many connections
  - Examples: Stock trading, travel reservations, order processing
- Application porting simplified if
  - Stored procedures used
  - Performance problems isolated to subset of tables and SPs

In-Memory OLTP Architecture

Benefits

- High-performance data operations
- Frictionless scale-up
- Efficient, business-logic processing
- Hybrid engine and integrated experience

In-Memory OLTP Tech Pillars

Main-memory optimized

- Optimized for in-memory data
- Indexes (hash and range) exist only in memory
- No buffer pool
- Stream-based storage for durability

High concurrency

- Multiversion optimistic concurrency control with full ACID support
- Core engine uses lock-free algorithms
- No lock manager, latches, or spinlocks

T-SQL compiled to machine code

- T-SQL compiled to machine code via C code generator and Visual C compiler
- Invoking a procedure is just a DLL entry-point
- Aggressive optimizations at compile time

SQL Server integration

- Same manageability, administration, and development experience
- Integrated queries and transactions
- Integrated HA and backup/restore

Drivers

Hardware trends

- Steadily declining memory price, NVRAM
- Many-core processors
- Stalling CPU clock rate

Business

- TCO

Design Considerations For Memory-optimized Tables

Benefits

High performance data operations

In-Memory OLTP Tech Pillars

Main-memory optimized

- Optimized for in-memory data
- Indexes (hash and ordered) exist only in memory
- No buffer pool
- Stream-based storage for durability

Drivers

Hardware trends

Steadily declining memory price, NVRAM

Table constructs

Fixed schema; no ALTER TABLE; must drop/recreate/reload
No LOB data types; row size limited to 8,060
No constraints support (primary key only)
No identity or calculated columns, or CLR

Data and table size considerations

Size of tables = (row size * # of rows)
Size of hash index = (bucket_count * 8 bytes)
Max size SCHEMA_AND_DATA = 512 GB

IO for durability

SCHEMA_ONLY vs. SCHEMA_AND_DATA
Memory-optimized filegroup
Data and delta files
Transaction log
Database recovery

Create Table DDL

```
CREATE TABLE [Customer](
    [CustomerID] INT NOT NULL
            PRIMARY KEY NONCLUSTERED HASH WITH (BUCKET_COUNT = 1000000),
    [Name] NVARCHAR(250) NOT NULL
        INDEX [IName] HASH WITH (BUCKET_COUNT = 1000000),
    [CustomerSince] DATETIME NULL
)
WITH (MEMORY_OPTIMIZED = ON, DURABILITY = SCHEMA_AND_DATA);
```

Hash index

Secondary indexes are specified inline

This table is memory optimized

This table is durable

Create Procedure DDL

```
CREATE PROCEDURE [dbo].[InsertOrder] @id INT, @date DATETIME
  WITH
    NATIVE_COMPILATION,
    SCHEMABINDING,
    EXECUTE AS OWNER
AS
BEGIN ATOMIC
  WITH
(TRANSACTION
    ISOLATION LEVEL = SNAPSHOT,
 LANGUAGE = 'us_english')

  -- insert T-SQL here
END
```

This proc is natively compiled

Native procs must be schema-bound

Execution context is required

Atomic blocks
- Create a transaction if there is none
- Otherwise, create a savepoint

Session settings are fixed at create time

High Concurrency Design Considerations

Benefits

Frictionless scale-up

In-Memory OLTP Tech Pillars

High concurrency

- Multiversion optimistic concurrency control with full ACID support
- Core engine uses lock-free algorithms
- No lock manager, latches, or spinlocks

Drivers

Hardware trends

Many-core processors

Impact of no locks or latches

Write-write conflict: design application for condition with try.catch
Applications dependent on locking; may not be a good candidate

Multiple versions of records

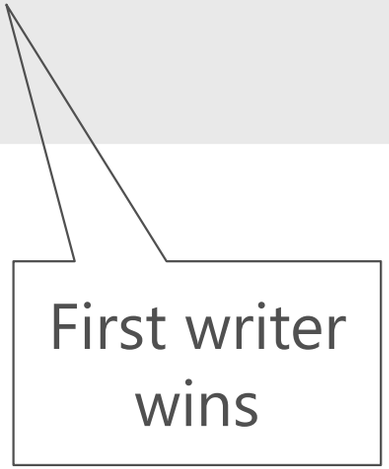
Increases the memory needed by memory-optimized tables
Garbage Collection used to reclaim old versions

Transaction scope

Support for Isolation Levels: Snapshot, Repeatable Read, Serializable
Commit time validation; again must retry logic to deal with failure

Example: Write Conflict

Time	Transaction T1 (SNAPSHOT)	Transaction T2 (SNAPSHOT)
1	BEGIN	
2		BEGIN
3		UPDATE t SET c1='bla' WHERE c2=123
4	UPDATE t SET c1='bla' WHERE c2=123 (write conflict)	



Guidelines for Usage

1. Declare isolation level – no locking hints
2. Use retry logic to handle conflicts and validation failures
3. Avoid using long-running transactions

Design considerations for native compiled stored procedures

Benefits

Efficient business-logic processing

In-Memory OLTP Tech Pillars

T-SQL compiled to machine code

- T-SQL compiled to machine code via C code generator and Visual C compiler
- Invoking a procedure is just a DLL entry-point
- Aggressive optimizations at compile-time

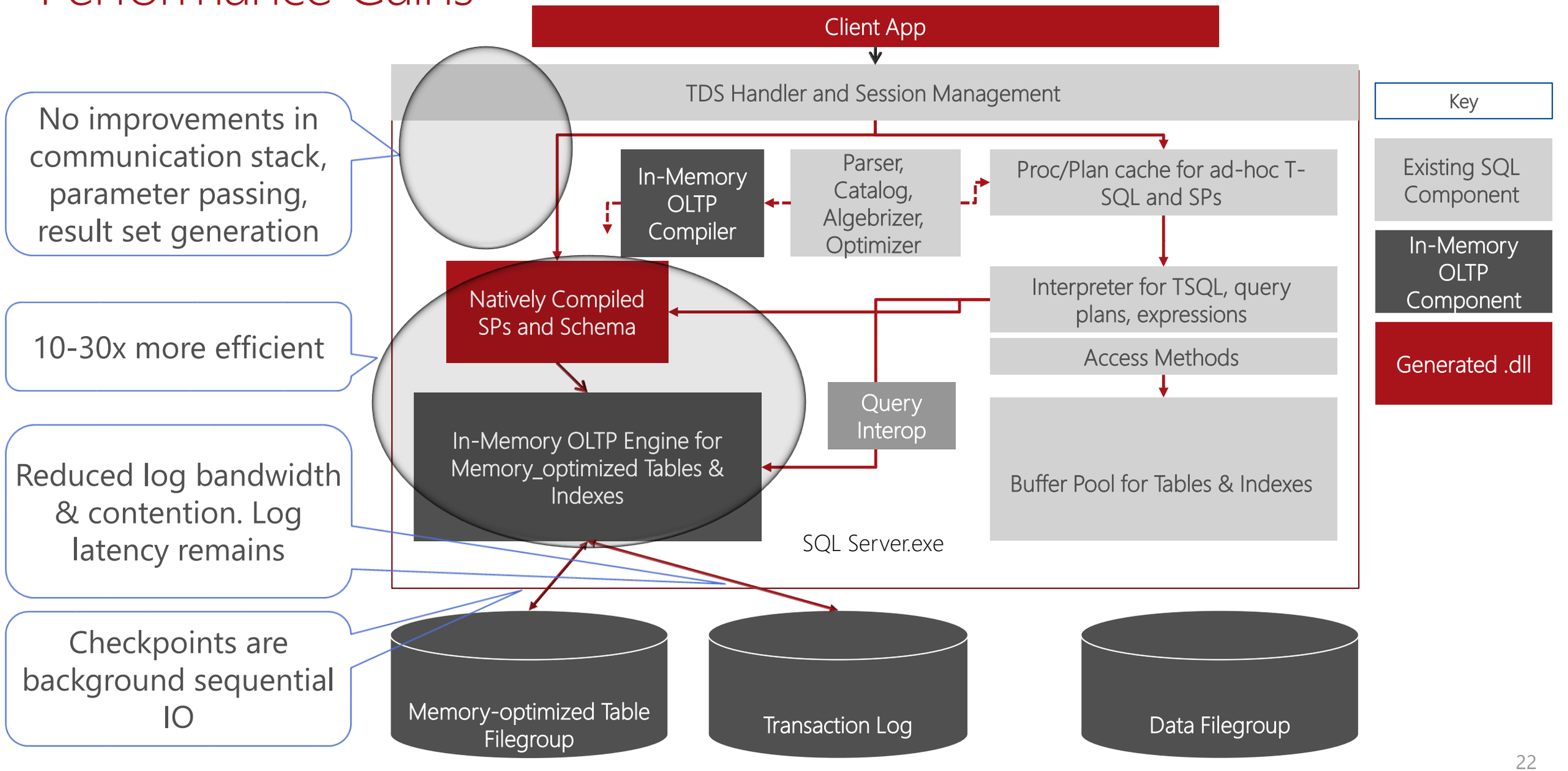
Drivers

Hardware trends

Stalling CPU clock rate

	Native compiled stored procedures	**Non-native compilation**
Performance	**High**. Significantly less instructions to go through	No different than T-SQL calls in SQL Server today
Migration strategy	Application changes; development overhead	Easier app migration as can still access memory-optimized tables
Access to objects	Can only interact with memory-optimized tables	All objects; access for transactions across memory optimized and B-tree tables
Support for T-SQL constructs	Limited	T-SQL surface area (limit on memory-optimized interaction)
Optimization, stats, and query plan	Statistics utilized at CREATE -> Compile time	Statistics updates can be used to modify plan at runtime
Flexibility	Limited (no ALTER procedure, compile-time isolation level)	Ad-hoc query patterns

Performance Gains



SQL Server Integration Design Drilldown

Benefits

Hybrid engine and integrated experience

In-Memory OLTP Tech Pillars

SQL Server integration

- Same manageability, administration, and development experience
- Integrated queries and transactions
- Integrated HA and backup/restore

Drivers

Business

TCO

In-Memory OLTP component	Integration with SQL Server
Memory management	Use Resource Governor Pool to control In-Memory OLTP memory
Query optimization	Same SQL Server optimizer
HA/DR	Integrate with AlwaysOn FCI/AG Backup/restore contains memory-optimized tables (and data if durable)
Monitoring and troubleshooting	Integrated catalog views, DMVs, performance monitor counters, extended events, and more
Interaction with non-In-Memory OLTP objects	Supported transaction interaction (insert…select, JOIN, and more) with non-In-Memory OLTP objects in database

Integrated Experience

Backup and restore

Full and log backup and restore is supported; piecemeal restore is supported

Failover clustering

Failover time depends on size of durable memory-optimized tables

AlwaysOn

Secondary has memory-optimized tables in memory
Failover time is not dependent on size of durable memory-optimized tables

DMVs, catalog views, performance monitor counters, XEvents

Monitoring memory, garbage collection activity, and transaction details

SQL Server Management Studio (SSMS)

Creating, managing, and monitoring tables; databases and server

In-Memory Data Structures

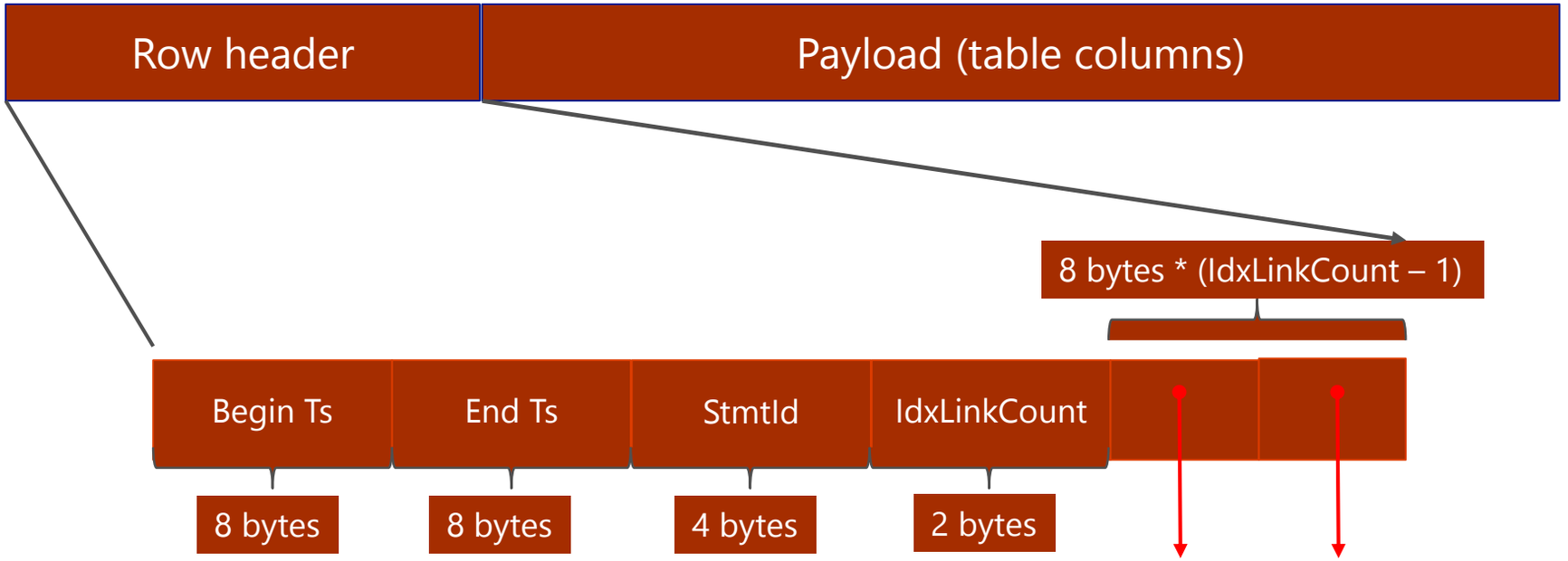
Rows

- New row format
  - Structure of the row is optimized for memory residency and access
- One copy of row
  - Indexes point to rows, they do not duplicate them

Indexes

- Hash index for point lookups
- Memory-optimized nonclustered index for range and ordered scans
- Do not exist on disk – recreated during recovery

Memory-optimized Table: Row Format



Key Points

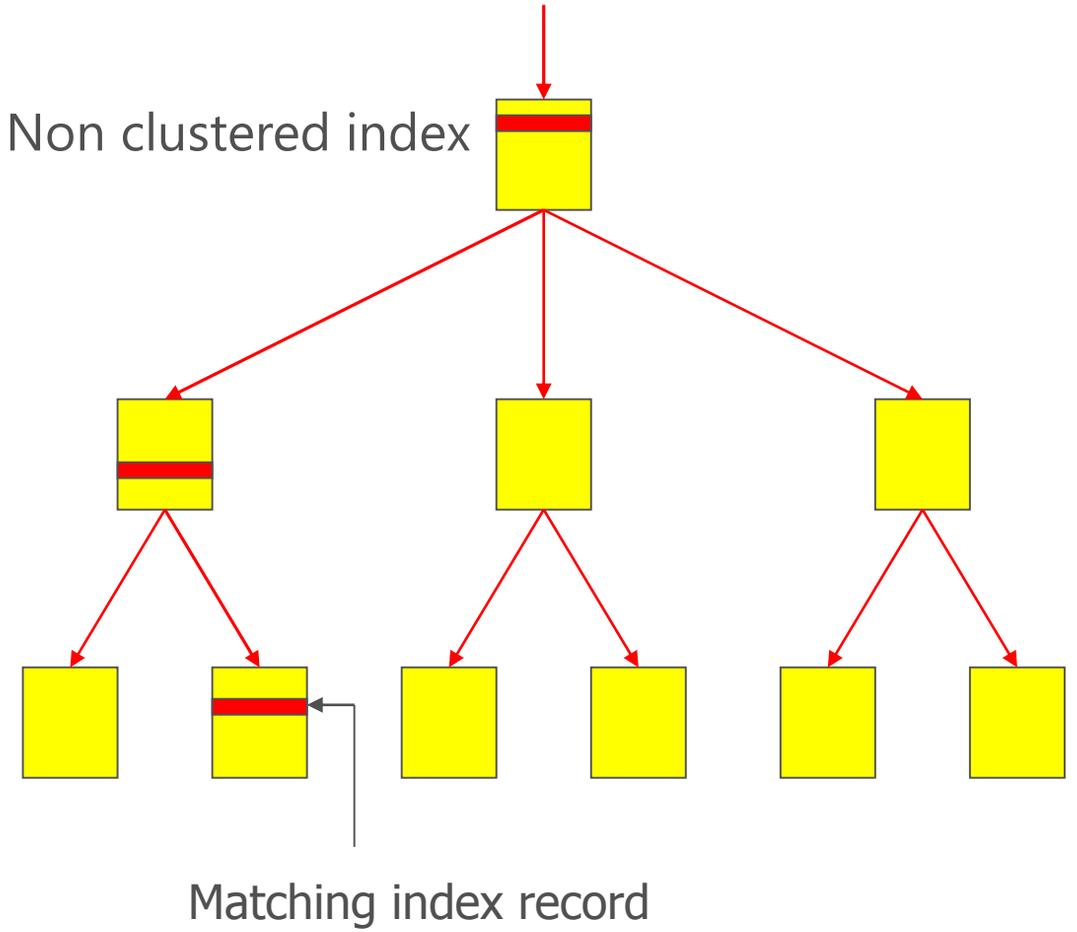
Begin/End timestamp determines row's validity

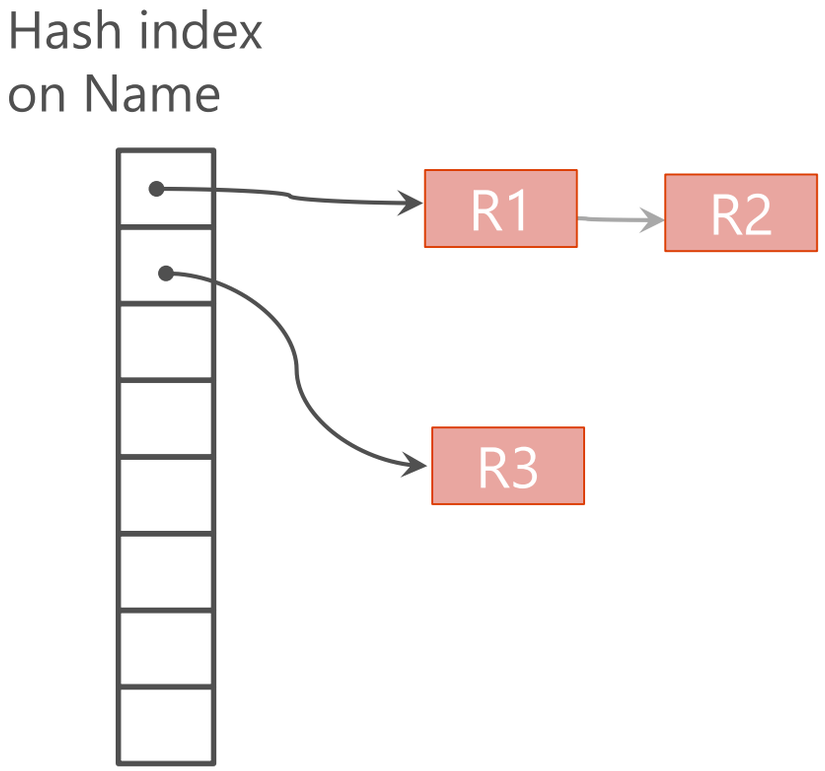
No data or index page; just rows

Row size limited to 8060 bytes to allow data to be moved to disk-based table

Not every SQL table schema is supported

Key lookup: B-tree vs. Memory-optimized Table





Memory Management

Data resides in memory at all times

- Must configure SQL server with sufficient memory to store memory-optimized tables
- Failure to allocate memory will fail transactional workload at run-time
- Integrated with SQL Server memory manager and reacts to memory pressure where possible

Integration with Resource Governor

- "Bind" a database to a resource pool
- Mem-opt tables in a database cannot exceed the limit of the resource pool
- Hard limit (80% of phys. memory) to ensure system remains stable under memory pressure

Garbage Collection

Stale Row Versions

- Updates, deletes, and aborted insert operations create row versions that (eventually) are no longer visible to any transaction
- Slow down scans of index structures
- Create unused memory that needs to be reclaimed (i.e. Garbage Collected)

Garbage Collection (GC)

- Analogous to version store cleanup task for disk-based tables to support Read Committed Snapshot (RCSI)
- System maintains 'oldest active transaction' hint

GC Design

- Non-blocking, Cooperative, Efficient, Responsive, Scalable
- A dedicated system thread for GC
- Active transactions work cooperatively and pick up parts of GC work

Cooperative Garbage Collection

Key Points

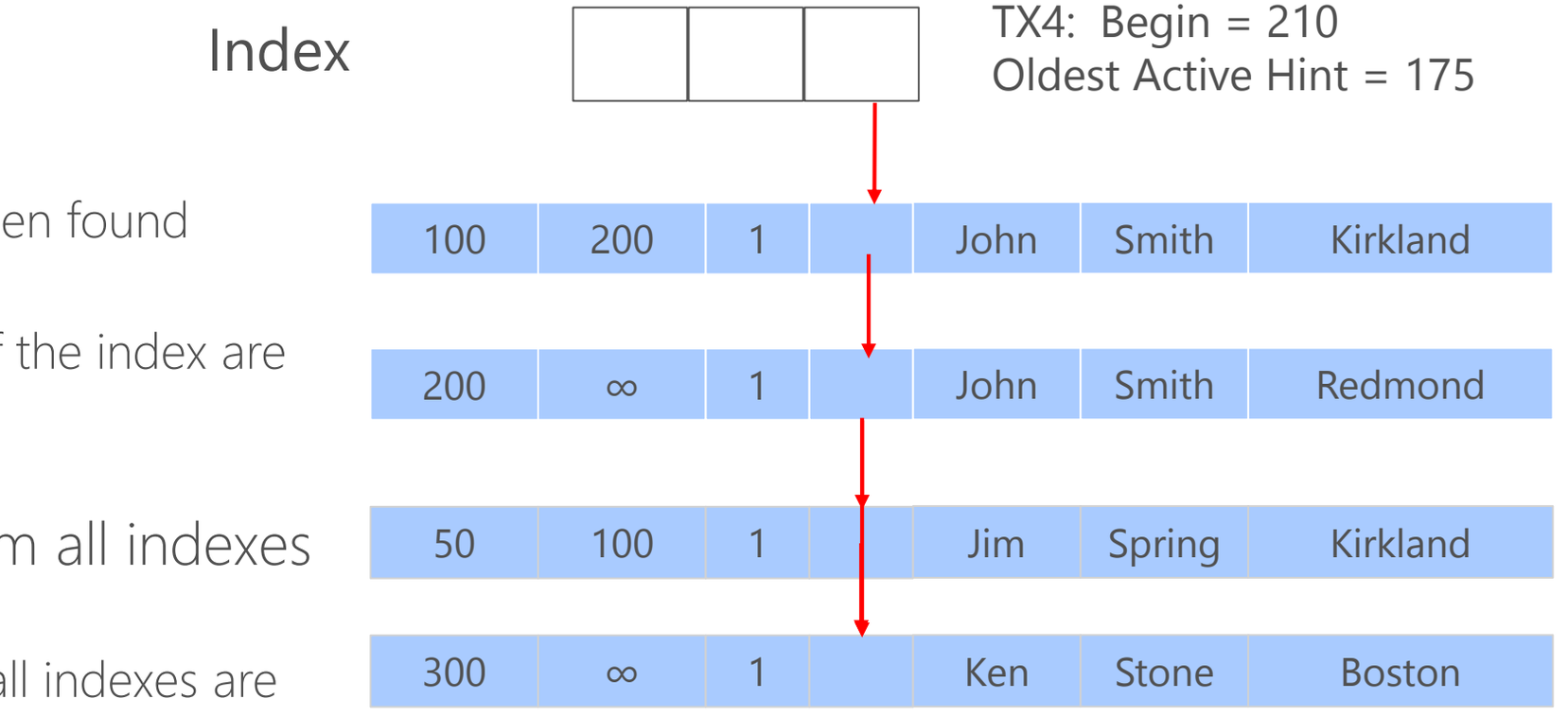
Scanners can remove expired rows when found

Offloads work from GC thread

Ensures that frequently visited areas of the index are cleaned regularly

A row needs to be removed from all indexes before memory can be freed

Garbage collection is most efficient if all indexes are frequently accesses



Durability

Memory-optimized tables can be durable or non-durable

- Default is ‘durable’
- Non-durable tables are useful for transient data

Durable tables are persisted in a single memory-optimized filegroup

- Storage used for memory-optimized has a different access pattern than for disk tables

Filegroup can have multiple containers (volumes)

- Additional containers aid in parallel recovery; recovery happens at the speed of IO

On-disk Storage

Filestream is the underlying storage mechanism

- Checksums and single-bit correcting ECC on files

Data files

- ~128MB in size, write 256KB chunks at a time
- Stores only the inserted rows (i.e. table content)
- Chronologically organized streams of row versions

Delta files

- File size is not constant, write 4KB chunks at a time.
- Stores IDs of deleted rows

Logging for Memory-Optimized Tables

Uses SQL transaction log to store content

Each HK log record contains a log record header followed by opaque memory optimized-specific log content

All logging for memory-optimized tables is logical

No log records for physical structure modifications.
No index-specific / index-maintenance log records.
No UNDO information is logged

Recovery Models

All three recovery models are supported

Backup for Memory-Optimized Tables

Integrated with SQL Database Backup

- Memory-Optimized file group is backed up as part SQL Server database backup
- Existing backup scripts work with minimal or no changes
- Transaction log backup includes memory-optimized log records

Not supported

- Differential backup

Recovery for Memory-Optimized Tables

Analysis Phase

Finds the last completed checkpoint

Data Load

Load from set of data/delta files from the last completed checkpoint
Parallel Load by reading data/delta files using 1 thread / file

Redo phase to apply tail of the log

Apply the transaction log from last checkpoint
Concurrent with REDO on disk-based tables

No UNDO phase for memory-optimized tables

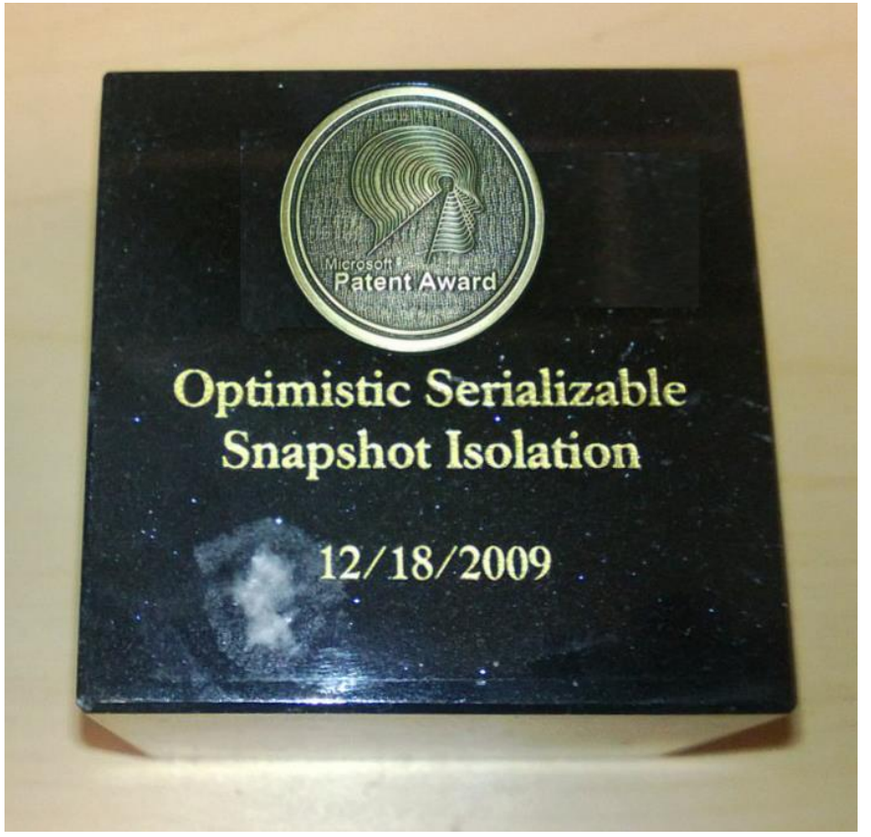
Only committed transactions are logged

Myth #1:

SQL Server In-Memory OLTP is a recent response to competitors' offerings

Reality

Project "Hekaton" was started around 4 years ago in response to business and hardware trends



Myth #2:

In-Memory OLTP is like DBCC PINTABLE

Reality

In-memory OLTP is completely new design to optimize for efficient In-Memory data operations.

There are no pages or buffer pool for memory-optimized tables.

Myth #3:

In-Memory Databases are new separate products

Reality

In-Memory OLTP is a feature fully integrated into SQL Server 2014

Myth #4:

You can use In-Memory OLTP in an existing SQL Server app with NO changes whatsoever

Reality

There are at least some changes, at minimum changing some of the schema

Myth #5:

Since tables are in memory, the data is not Durable or Highly Available – I will lose it after server crash

Reality

In-Memory OLTP is fully durable, and includes several HA features, including AlwaysOn

Data is persisted on disk, and will survive server crash

In-Memory OLTP summary

What's being delivered

High-performance, memory-optimized OLTP engine integrated into SQL Server and architected for modern hardware trends

Main benefits

- Optimized for in-memory data up to 20–30 times throughput
  - Indexes (hash and range) exist only in memory; no buffer pool, B-trees
  - T-SQL compiled to machine code via C code generator and Visual C compiler
  - Core engine uses lock-free algorithms; no lock manager, latches, or spinlocks
- Multiversion optimistic concurrency control with full ACID support
- On-ramp existing applications
- Integrated experience with same manageability, administration, and development experience

Microsoft

In-Memory DW

Microsoft

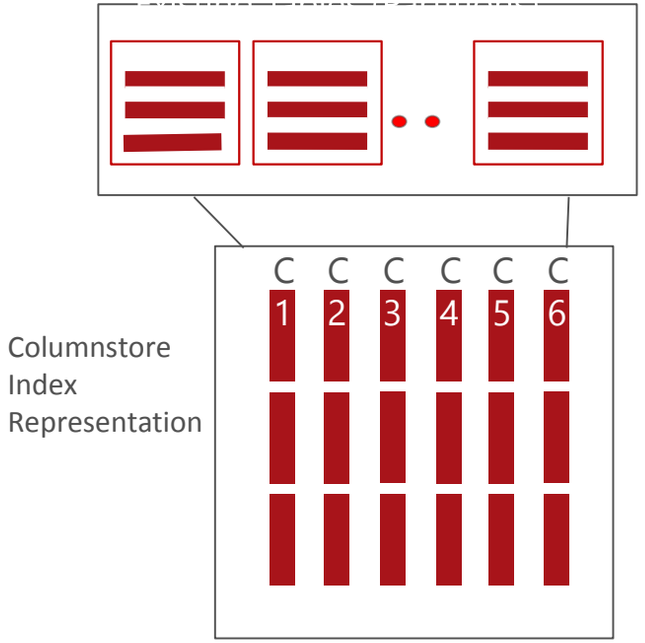
In-Memory DW

In-Memory In the Data Warehouse
Data Stored Row-Wise: Heaps, b-Trees, Key-Value

- In-Memory ColumnStore
- Both memory and disk
- Built-in to core RDBMS engine
- Customer Benefits:
  - 10-100x faster
  - Reduced design effort
  - Work on customers' existing hardware
  - Easy upgrade; Easy deployment

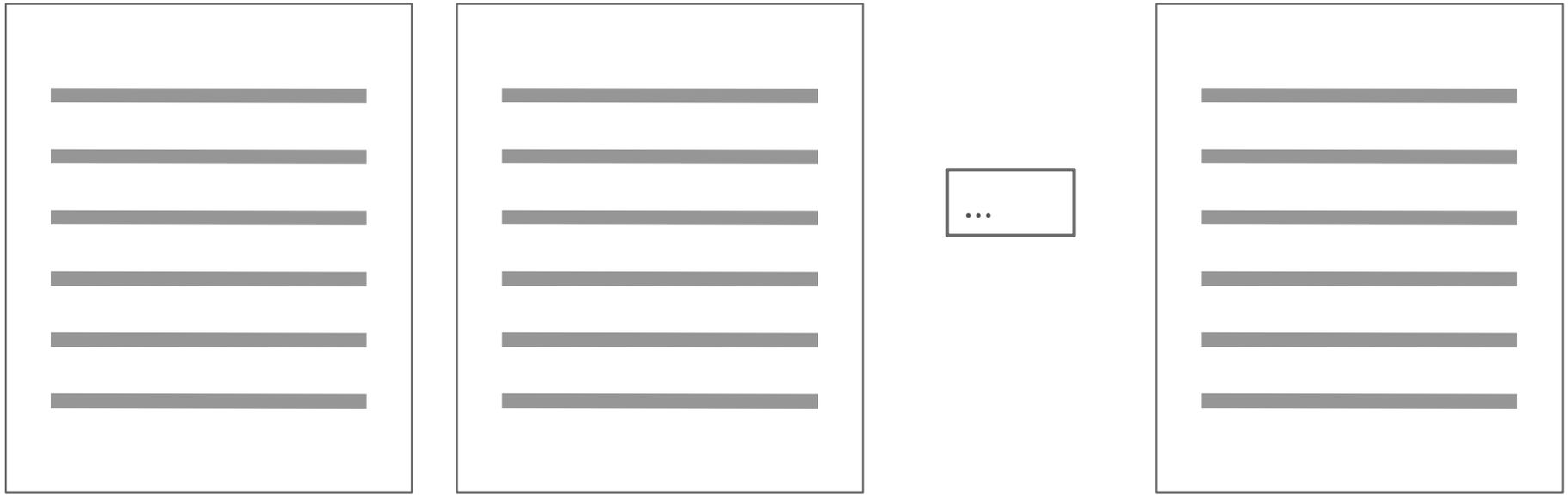
"By using SQL Server 2012 In-Memory ColumnStore, we were able to extract about 100 million records in **2 or 3 seconds** versus the **30 minutes required** previously. "

- Atsuo Nakajima Asst Director, Bank of Nagoya



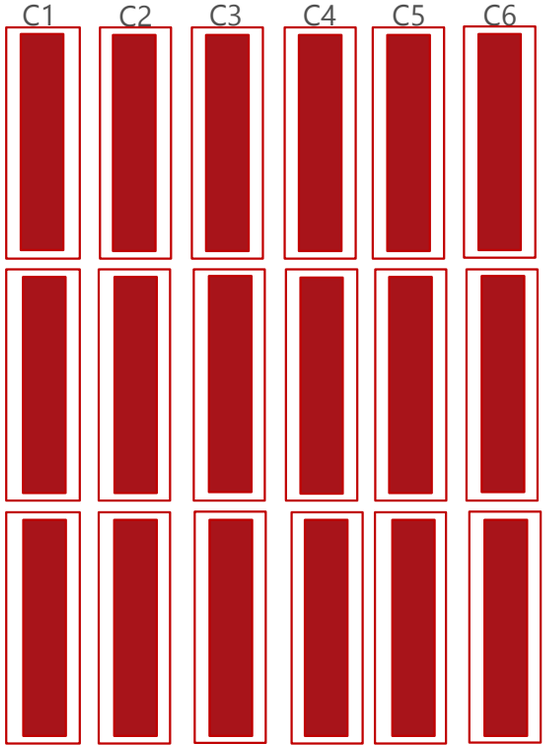
Traditional Storage Models
Data Stored Row-Wise: Heaps, b-Trees, Key-Value

- Relational, dimensional, map reduce



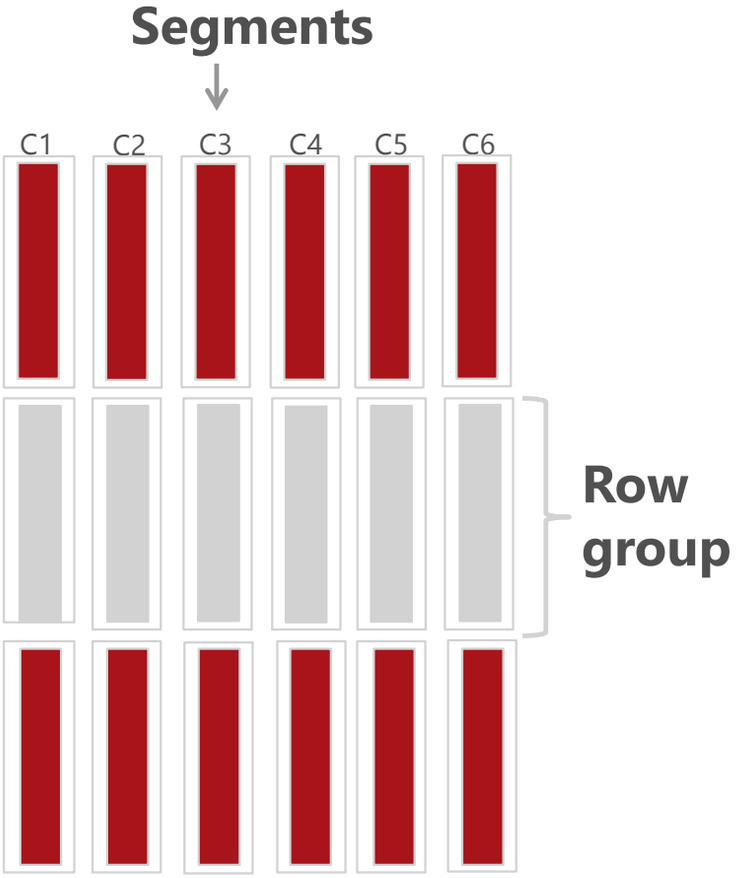
In-Memory DW Storage Model
Data Stored Column-wise

- Each page stores data from a single column
- Highly compressed
  - More data fits in memory
- Each column can be accessed independently
  - Fetch only columns needed
  - Can dramatically decrease I/O



In-Memory DW Index Structure
Row Groups & Segments

- A segment contains values for one column for a set of rows
- Segments for the same set of rows comprise a row group
- Segments are compressed
- Each segment stored in a separate LOB
- Segment is unit of transfer between disk and memory



In-Memory DW Index Processing an Example

OrderDateKey	ProductKey	StoreKey	RegionKey	Quantity	SalesAmount
20101107	106	01	1	6	30.00
20101107	103	04	2	1	17.00
20101107	109	04	2	2	20.00
20101107	103	03	2	1	17.00
20101107	106	05	3	4	20.00
20101108	106	02	1	5	25.00
20101108	102	02	1	1	14.00
20101108	106	03	2	5	25.00
20101108	109	01	1	1	10.00
20101109	106	04	2	4	20.00
20101109	106	04	2	5	25.00
20101109	103	01	1	1	17.00

Horizontally Partition Row Groups

OrderDateKey	ProductKey	StoreKey	RegionKey	Quantity	SalesAmount
20101107	106	01	1	6	30.00
20101107	103	04	2	1	17.00
20101107	109	04	2	2	20.00
20101107	103	03	2	1	17.00
20101107	106	05	3	4	20.00
20101108	106	02	1	5	25.00

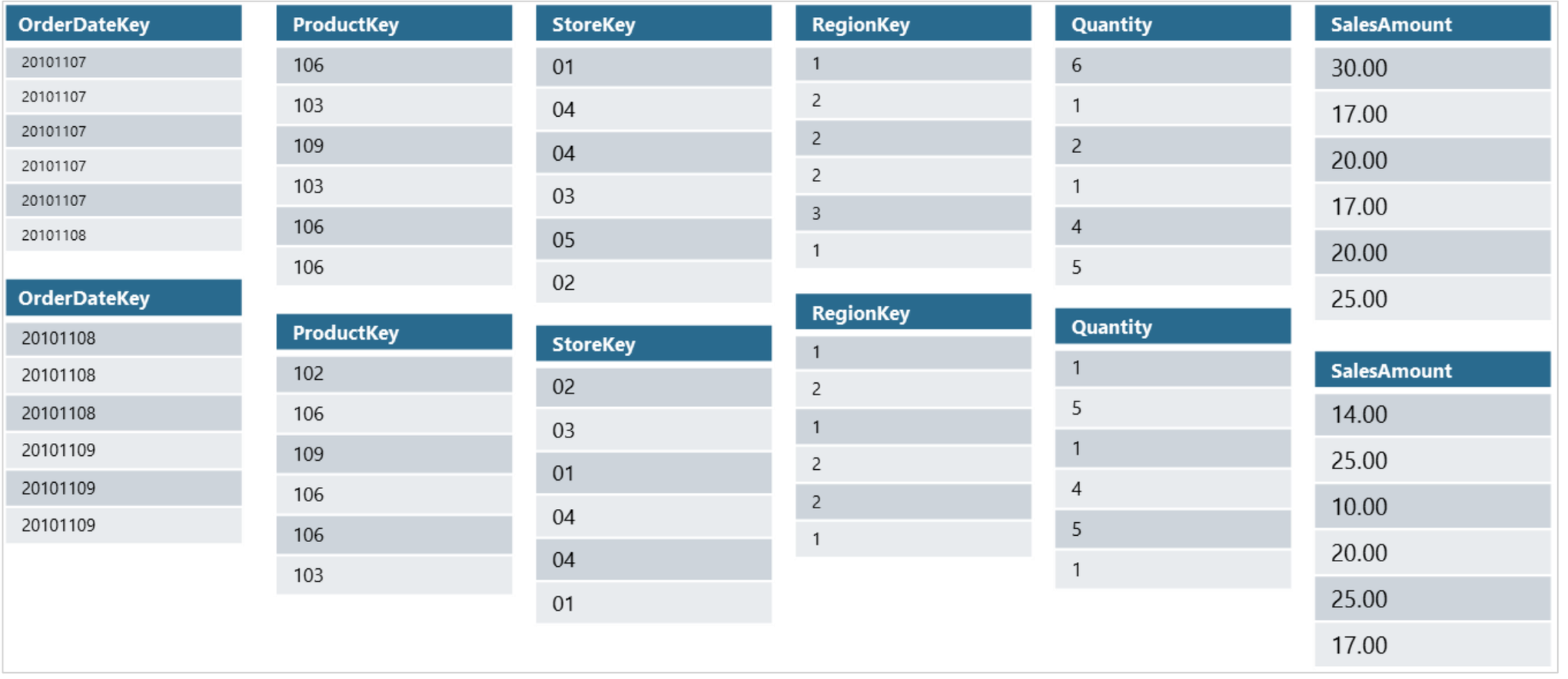
OrderDateKey	ProductKey	StoreKey	RegionKey	Quantity	SalesAmount
20101108	102	02	1	1	14.00
20101108	106	03	2	5	25.00
20101108	109	01	1	1	10.00
20101109	106	04	2	4	20.00
20101109	106	04	2	5	25.00
20101109	103	01	1	1	17.00

Vertical Partition Segments

OrderDateKey	ProductKey	StoreKey	RegionKey	Quantity	SalesAmount
20101107	106	01	1	6	30.00
20101107	103	04	2	1	17.00
20101107	109	04	2	2	20.00
20101107	103	03	2	1	17.00
20101107	106	05	3	4	20.00
20101108	106	02	1	5	25.00

OrderDateKey	ProductKey	StoreKey	RegionKey	Quantity	SalesAmount
20101108	102	02	1	1	14.00
20101108	106	03	2	5	25.00
20101108	109	01	1	1	10.00
20101109	106	04	2	4	20.00
20101109	106	04	2	5	25.00
20101109	103	01	1	1	17.00

Compress Each Segment*
Some Compress More than Others



OrderDateKey
20101107
20101107
20101107
20101107
20101107
20101108

OrderDateKey
20101108
20101108
20101108
20101109
20101109
20101109

ProductKey
106
103
109
103
106
106

ProductKey
102
106
109
106
106
103

StoreKey
01
04
04
03
05
02

StoreKey
02
03
01
04
04
01

RegionKey
1
2
2
2
3
1

RegionKey
1
2
1
2
2
1

Quantity
6
1
2
1
4
5

Quantity
1
5
1
4
5
1

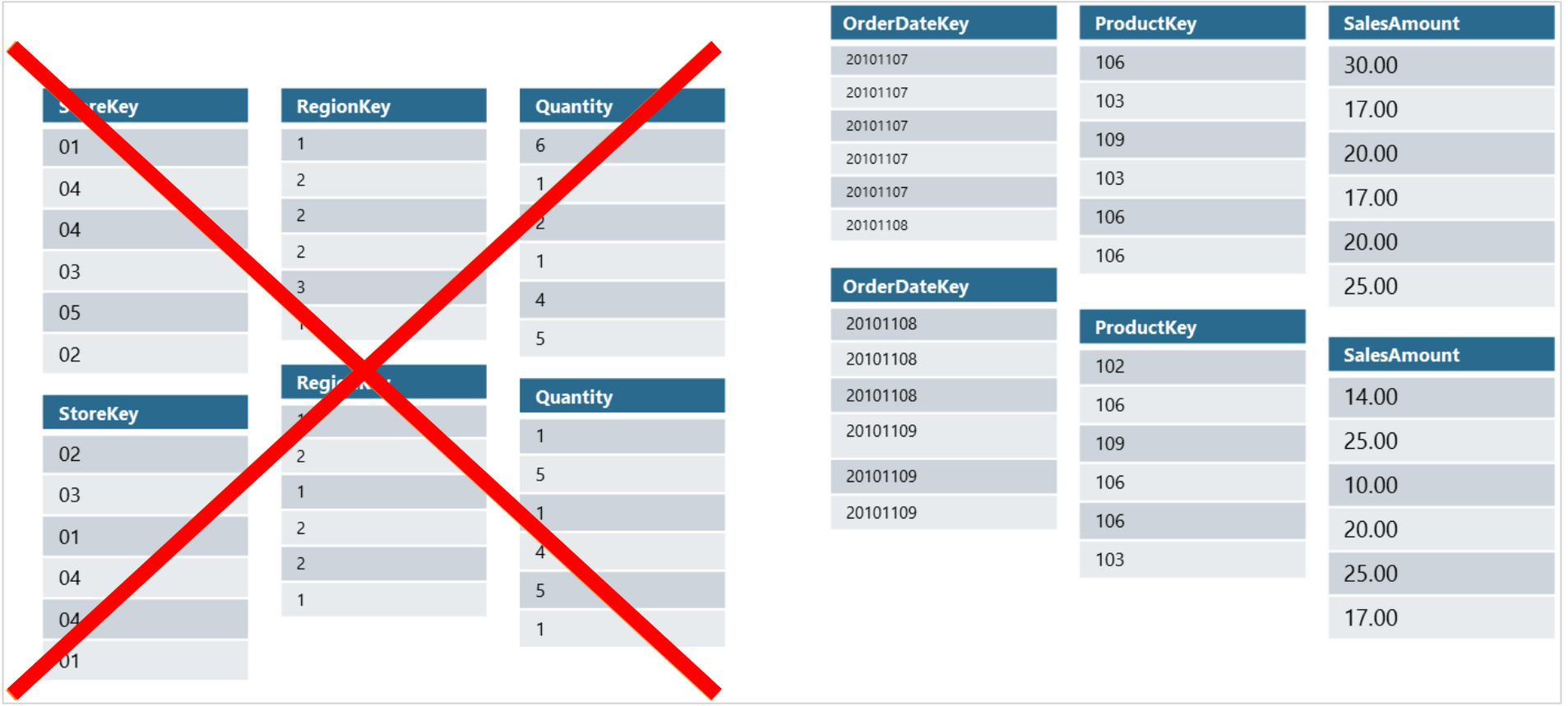
SalesAmount
30.00
17.00
20.00
17.00
20.00
25.00

SalesAmount
14.00
25.00
10.00
20.00
25.00
17.00

*Encoding and reordering not shown

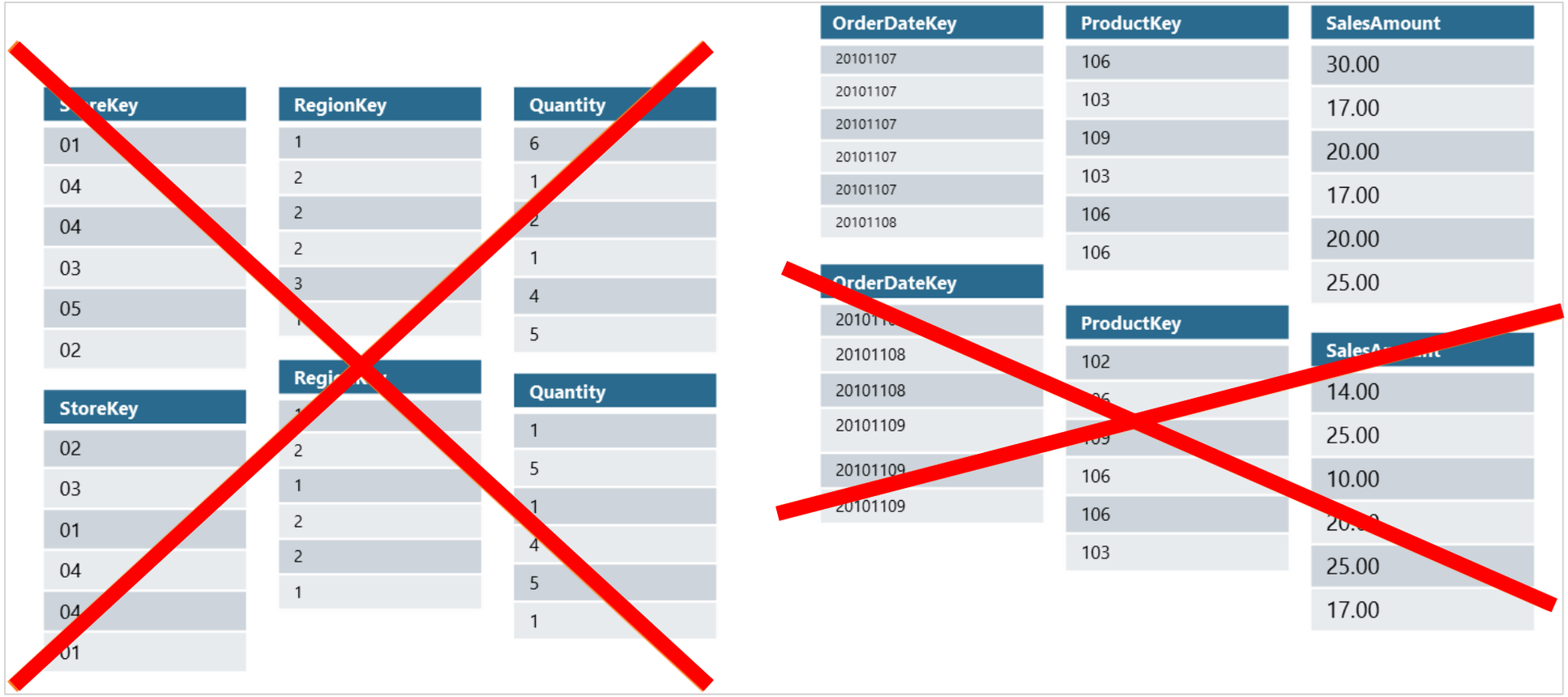
Fetch Only Needed Columns
Segment Elimination

```
SELECT ProductKey, SUM (SalesAmount)
FROM SalesTable
WHERE OrderDateKey < 20101108
```



Fetch Only Needed Segments
Segment Elimination

```
SELECT ProductKey, SUM (SalesAmount)
FROM SalesTable
WHERE OrderDateKey < 20101108
```

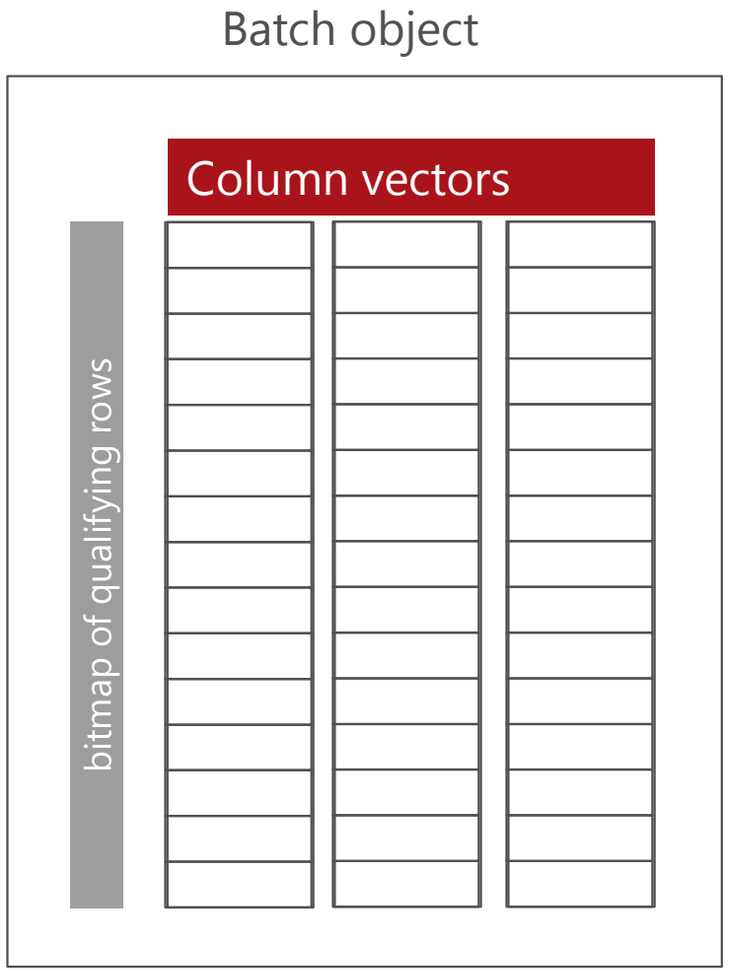


Batch Mode
Improving CPU Utilization

- Biggest advancement in query processing in years!
- Data moves in batch through query plan operators
  - Minimizes instructions per row
  - Takes advantage of cache structures
- Highly efficient algorithms
- Better parallelism

Batch Mode Processing
QP Vector Operators

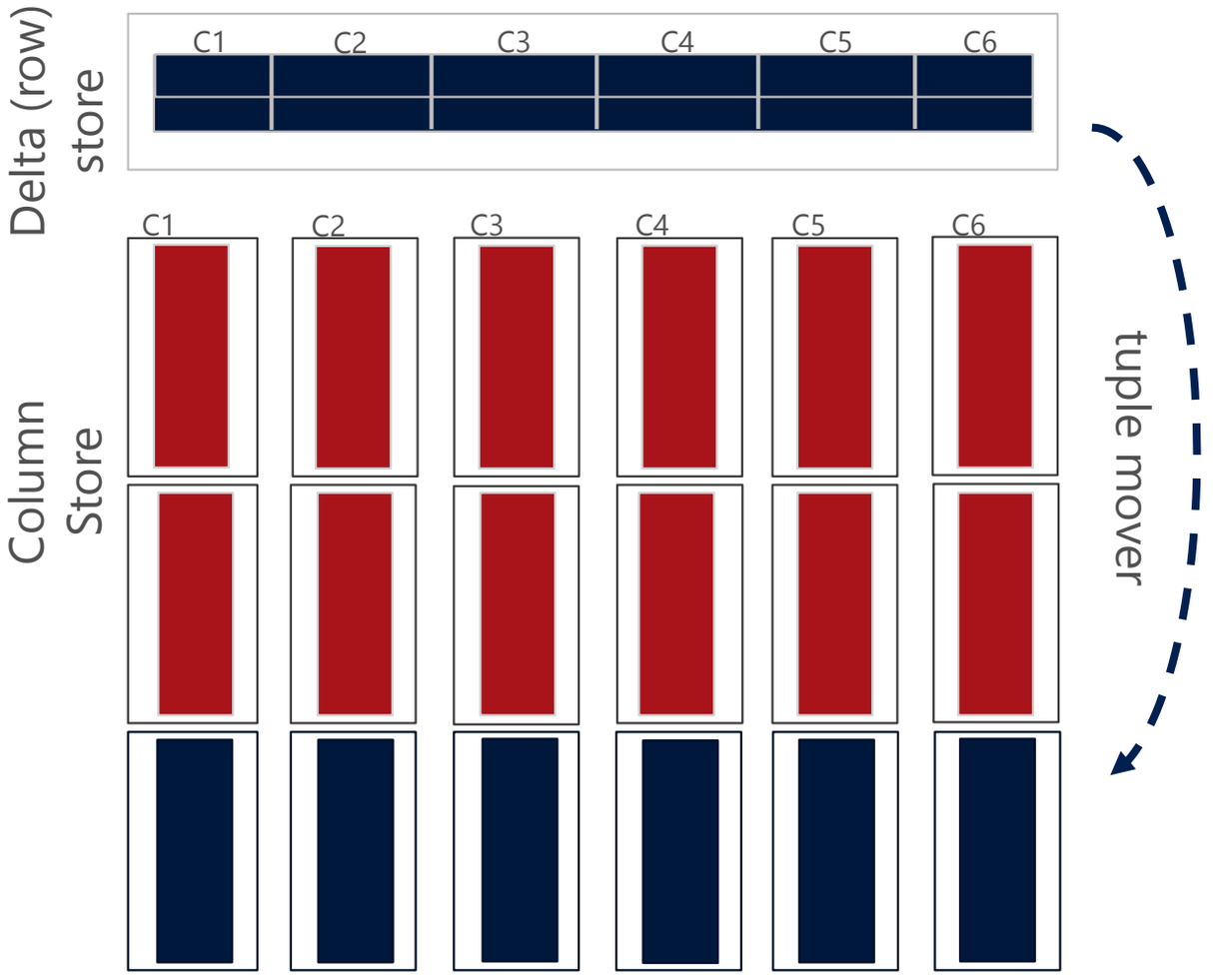
- Process ~1000 rows at a time
- Batch stored in vector form
- Optimized to fit in cache
- Vector operators implemented
- Filter, hash join, hash aggregation, …
- Greatly reduced CPU time (7 to 40X)



In-Memory DW: Clustered & Updatable

- Fast execution for data warehouse queries
  - Speedups of 10x and more
- No need for separate base table
  - Save space
- Data can be inserted, updated or deleted
  - Simpler management
- Eliminate need for other indexes
  - Save space and simpler management
- More data types supported

Updatable Columnstore Index

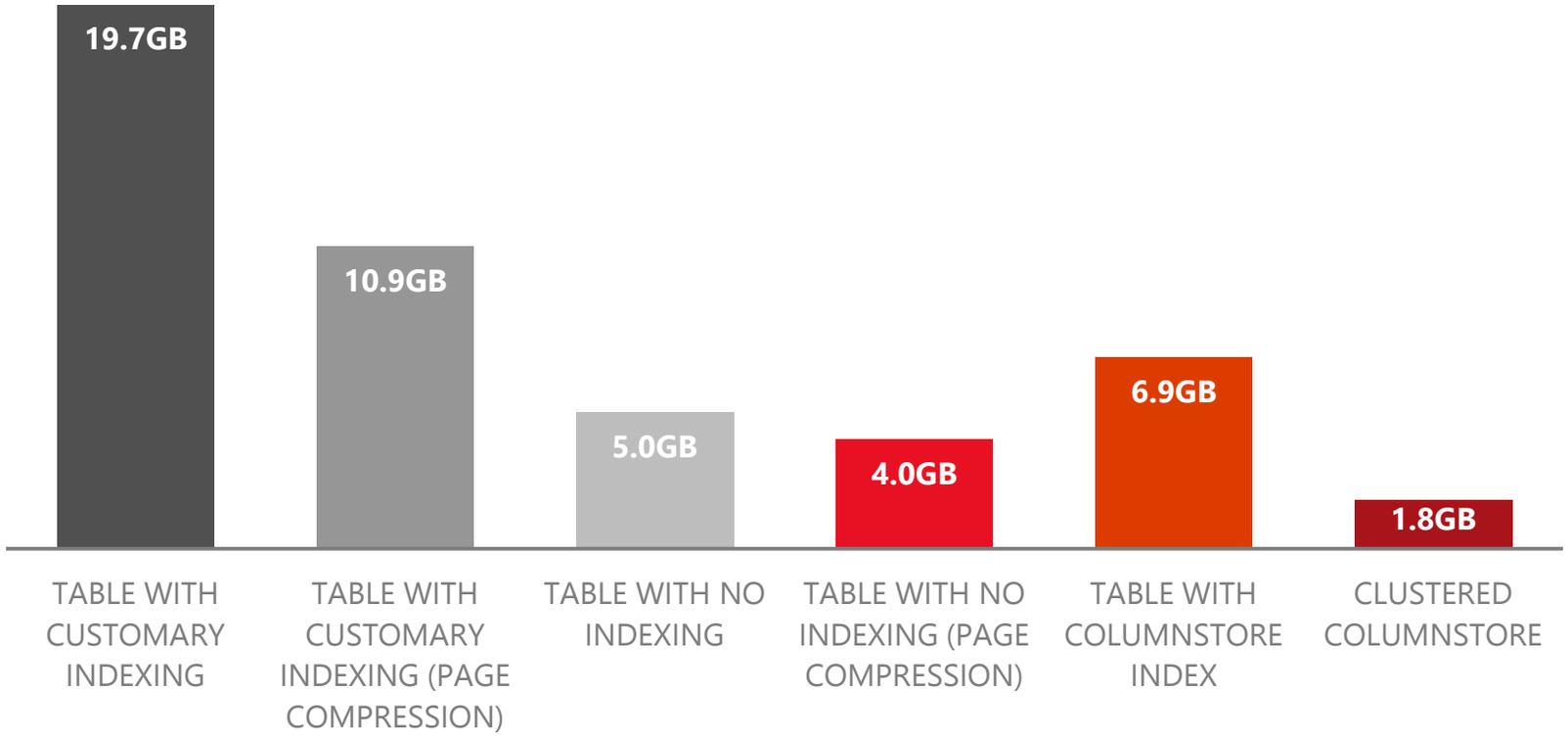


- Table consists of column store and row store
- DML (update, delete, insert) operations leverage delta store

- INSERT Values
  - Always lands into delta store
- DELETE
  - Logical operation
  - Data physically remove after REBUILD operation is performed.
- UPDATE
  - DELETE followed by INSERT.
- BULK INSERT
  - if batch < 100k, inserts go into delta store, otherwise columnstore
- SELECT
  - Unifies data from Column and Row stores - internal UNION operation.

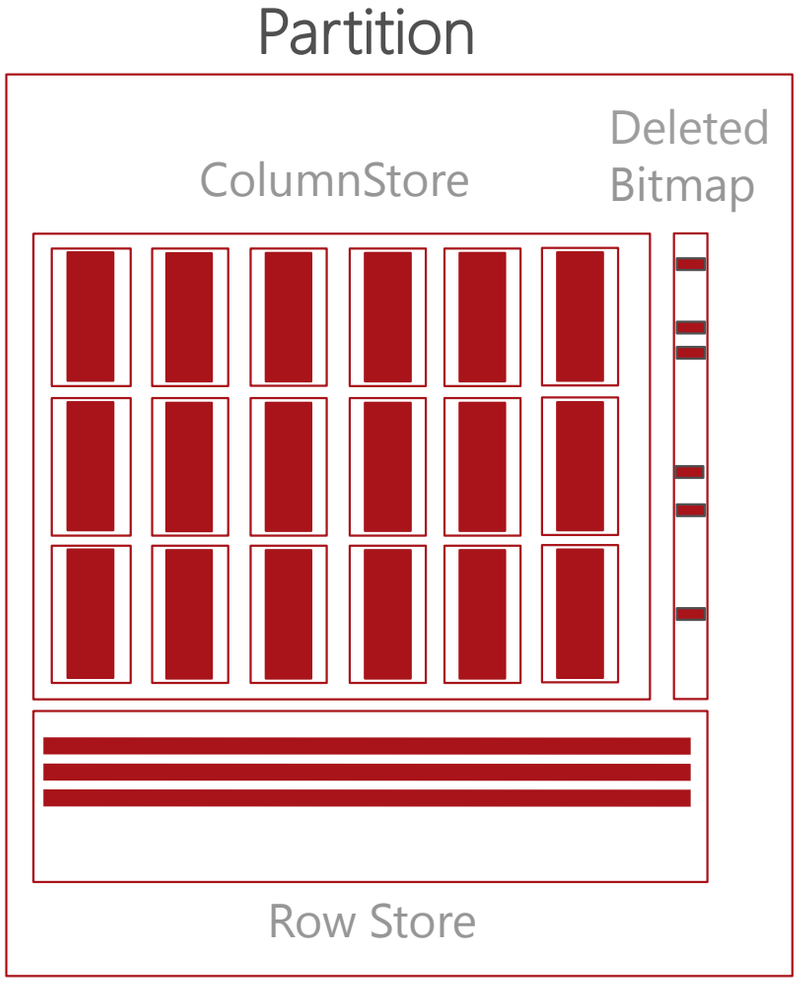
- "Tuple mover" converts data into columnar format once segment is full (1M of rows)
- REORGANIZE statement forces tuple mover to start.

Comparing Space Savings
101 Million Row Table + Index Space



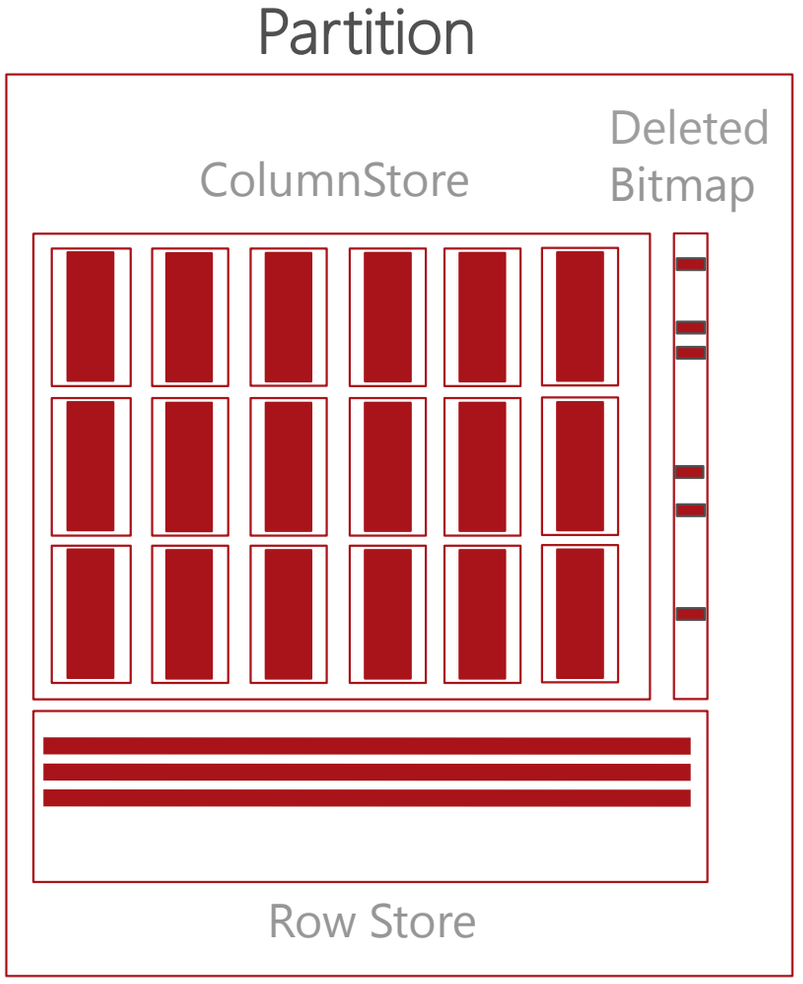
Structure of In-Memory DW
How It Works

- CREATE CLUSTERED COLUMNSTORE
  - Organizes and compresses data into columnstore
- BULK INSERT
  - Creates new columnstore row groups
- INSERT
  - Rows are placed in the row store (heap)
  - When row store is big enough, a new columnstore row group is created



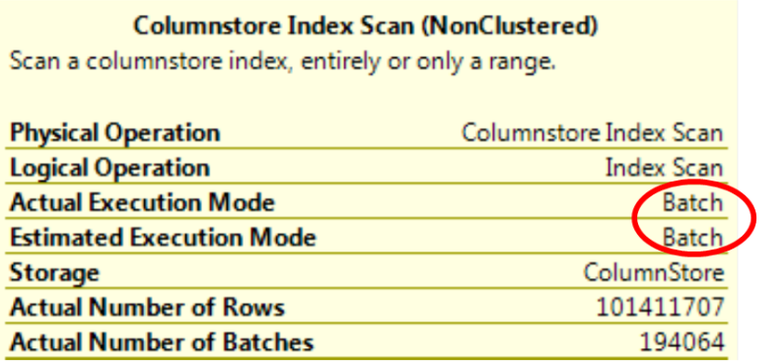
Structure of In-Memory DW
How It Works (cont'd)

- DELETE
  - Rows are marked in the deleted bitmap
- UPDATE
  - Delete plus insert
- Most data is in columnstore format



Batch Mode Processing
What's New?

- SQL Server 2014
  - Support for all flavors of JOINs
    - OUTER JOIN
    - Semi-join: IN, NOT IN
  - UNION ALL
  - Scalar aggregates
  - Mixed mode plans
  - Improvements in bitmaps, spill support, …



Global Batch Aggregation
What's New?

- Replaces a set of three operators in the query plan
  - Local (partial) batch aggregation
  - Row aggregation, somewhere above it
  - Repartition exchanges, somewhere between them
- Improves scenarios with large aggregation output
  - Process the same data with less memory than local batch aggregation
  - Better performance than local batch aggregation, example big hash tables
  - Removes the need for row mode aggregation in mostly batch query plans, resulting in less data conversion and better management of granted memory

Archival Compression
What's New?

- Adds an additional layer of compression on top of the inherent compression used by columnstore
- Shrink on-disk database sizes by up to 27%
  - Compression applies per partition and can be set either during index creation or during rebuild

Enhanced Compression

- Table compression options:

  DATA_COMPRESSION = { NONE | ROW | PAGE | **COLUMNSTORE** | **COLUMNSTORE_ARCHIVE** }

1. COLUMNSTORE Compression

- Default compression when creating a table with Clustered Columnstore Index
- Typical customer workloads gets 5-7x compression ratios

TPCH	3.1X
TPCDS	2.8X
Customer 1	3.9X
Customer 2	4.3X

** compression measured against raw data file

2. ARCHIVAL Compression

- Enables additional **30%** compression for whole table and/or chosen partitions.
- Going back and forth between columnstore and columnstore_archive compressions.
- **sys.partitions** exposes compression info (**3 – columnstore**, **4 – columnstore_archive**)

Partitioning ColumnStores
The Basic Mechanism

- The motivation is manageability over performance

CREATE TABLE <table> (…) *As usual*

CREATE CLUSTERED COLUMNSTORE INDEX <name> on <table>
Converts entire table to Columnstore format

BULK INSERT, SELECT INTO

INSERT
UPDATE
DELETE

Insert & Updating Data
Load Sizes

- Bulk insert
  - Creates row groups of 1Million rows, last row group is probably not full
  - But if <100K rows, will be left in Row Store
- Insert/Update
  - Collects rows in Row Store
- Tuple Mover
  - When Row Store reaches 1Million rows, convert to a ColumnStore Row Group
  - Runs every 5 minutes by default
  - Started explicitly by ALTER INDEX <name> ON <table> REORGANIZE

Building Index in ColumnStore Making Them Fast

- Memory resource intensive
  - Memory requirement related to number of columns, data, DOP
- Unit of parallelism is the segment
  - Lots of segments, lots of potential parallelism
- Low memory throttles parallelism
  - Increase the max server memory option
  - Set REQUEST_MAX_MEMORY_GRANT_PERCENT to 50
  - Add physical memory to the system
  - Reduce parallelism: (MAXDOP = 1);

Columnstore enhancements summary

- What's being delivered
  - Clustered and updateable columnstore index
  - Columnstore archive option for data compression
  - Global batch aggregation
- Main benefits
  - Real-time super fast data warehouse engine
    - Ability to continue queries while updating without the need to drop and recreate index or partition switching
  - Huge disk space saving due to compression
    - Ability to compress data 5–15x using archival per-partition compression
  - Better performance and more efficient (less memory) batch query processing using batch mode rather than row mode

Microsoft

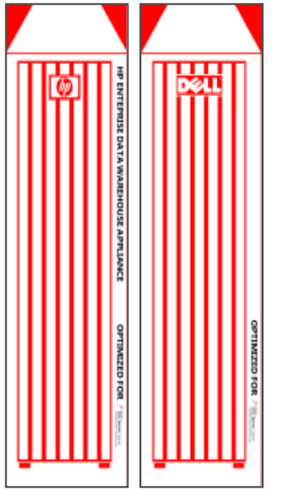
PDW V2

Microsoft

PDW V2

In-Memory Columnstore in PDW V2 & SQL Server 2014

- xVelocity in-memory columnstore in PDW columnstore index as primary data store in a scale-out MPP Data Warehouse - PDW V2 Appliance
  - Updateable clustered index
  - Support for bulk load and insert/update/delete
  - Extended data types – decimal/numeric for all precision and scale
  - Query processing enhancements for more batch mode processing (for example, Outer/Semi/Antisemi joins, union all, scalar aggregation)

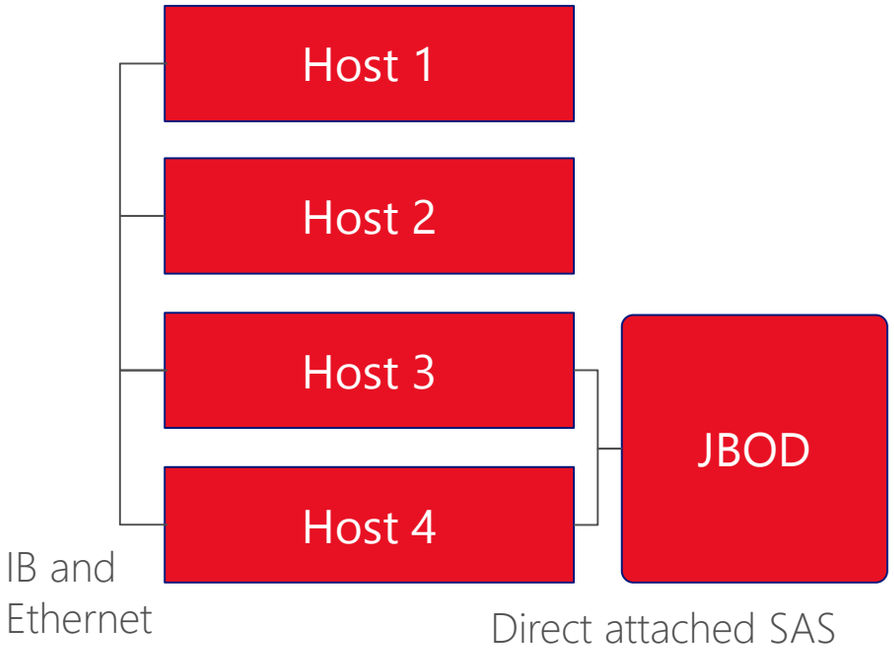


Customer benefits

- Outstanding query performance from in-memory columnstore index
  - 600 GB per hour for a single 12-core server
- Significant hardware cost savings due to high compression
  - 4–15x compression ratio
- Improved productivity through updateable index
- Ships in PDW V2 appliance and SQL Server 2014

Hardware architecture

Overview



One standard node type

- Doubled memory to 256 GB
- Connected via FDR Infiniband

Moving from SAN to JBODs

- Significant reduction in costs
- Use Windows Server 2012 technologies to achieve the same level of reliability and robustness
- Easy path to other DAS architectures and potentially different types

Backup and Landing Zone are now reference architectures, and not in the basic topology

- Customers can use their own hardware and customize to their needs
- Customers can use more than 1 Backup Unit or Landing Zone

Scale unit concept

- Capacity scale unit: adding 2/3 compute nodes and related storage
- Spare scale unit
- Base scale unit: min populated rack with networking

Hardware architecture

Storage details

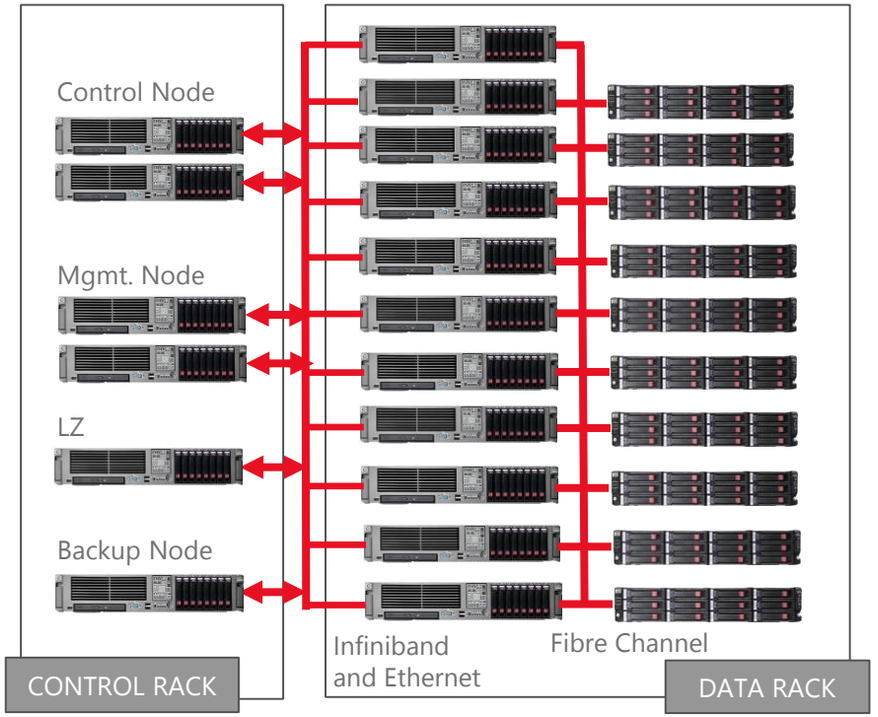
To take advantage of the fact that we have ~2x the number of spindles, we use more files per filegroup to better align SQL Server activity to actual disks available and provide better parallelization of disk IO.

	V1	V2
Replicated user data	1 FG per DB, 8 files per FG	1 FG per DB, 16 files per FG
Distributed user data	1 FG per distribution, 1 file per FG	1 FG per distribution, 2 files per FG
TempDB and Log	1 FG per DB, 1 file per FG	1 FG per DB, 16 file per FG

Overall, we expect to see 70 percent higher I/O bandwidth.

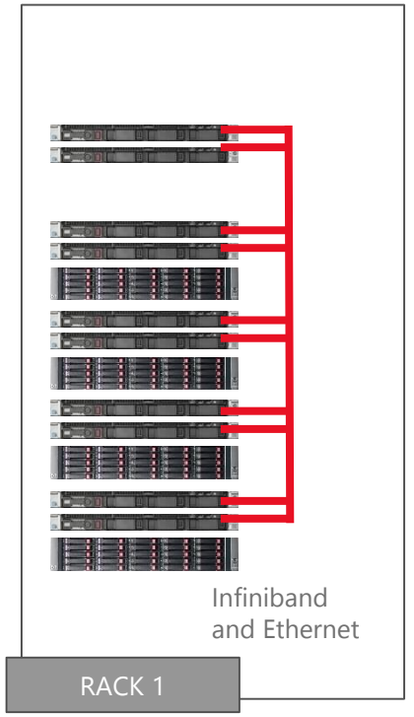
Hardware architecture

Comparison with V1: the basic 1-Rack



- 160 cores on 10 compute nodes
- 1.28 TB of RAM on compute
- Up to 30 TB of TempDB
- Up to 150 TB of user data

$ Estimated total hardware component list price: $1 million



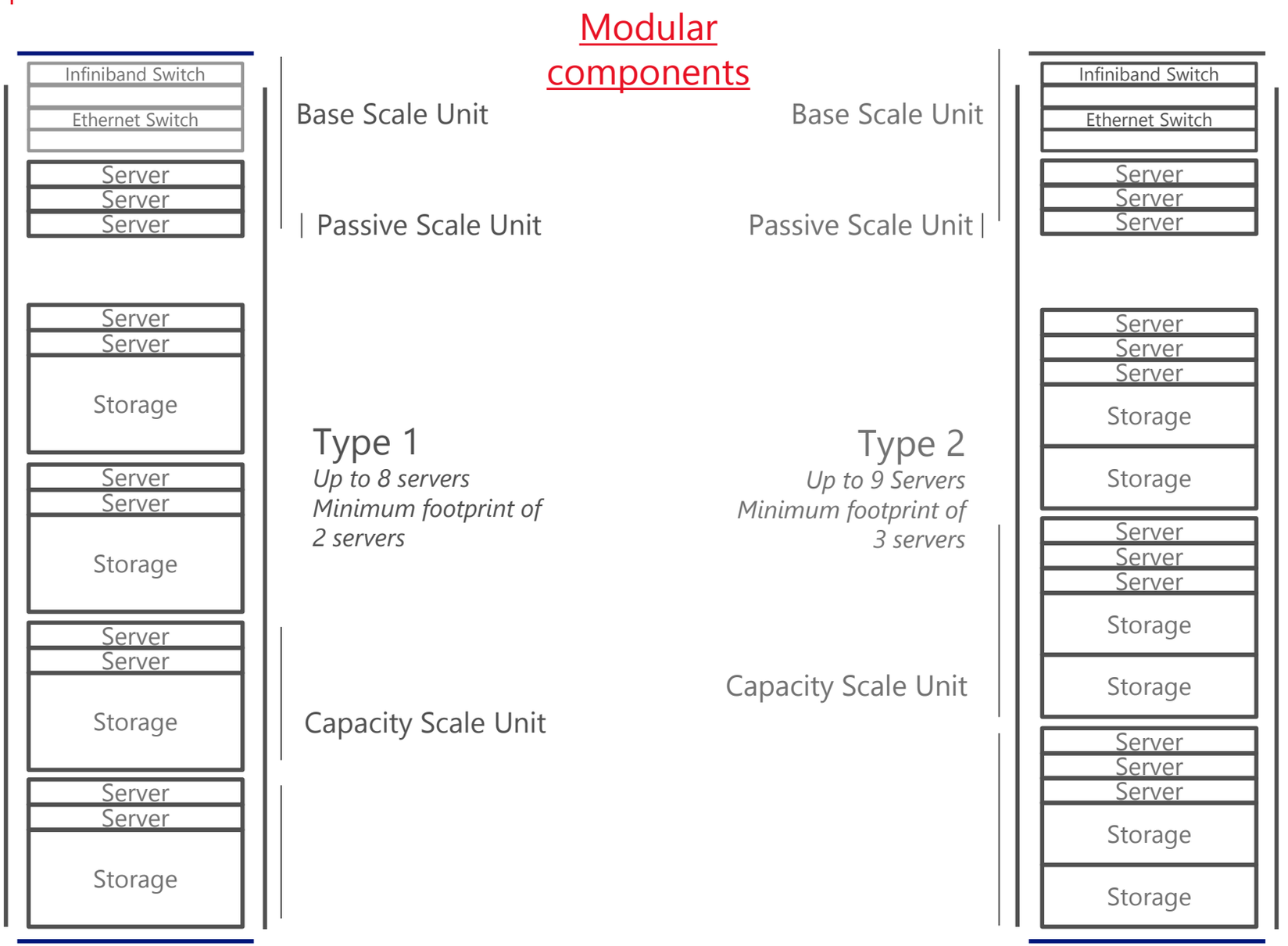
- 128 cores on 8 compute nodes
- 2 TB of RAM on compute
- Up to 168 TB of TempDB
- Up to 1 PB of user data

$ Estimated total hardware component list price: $500,000

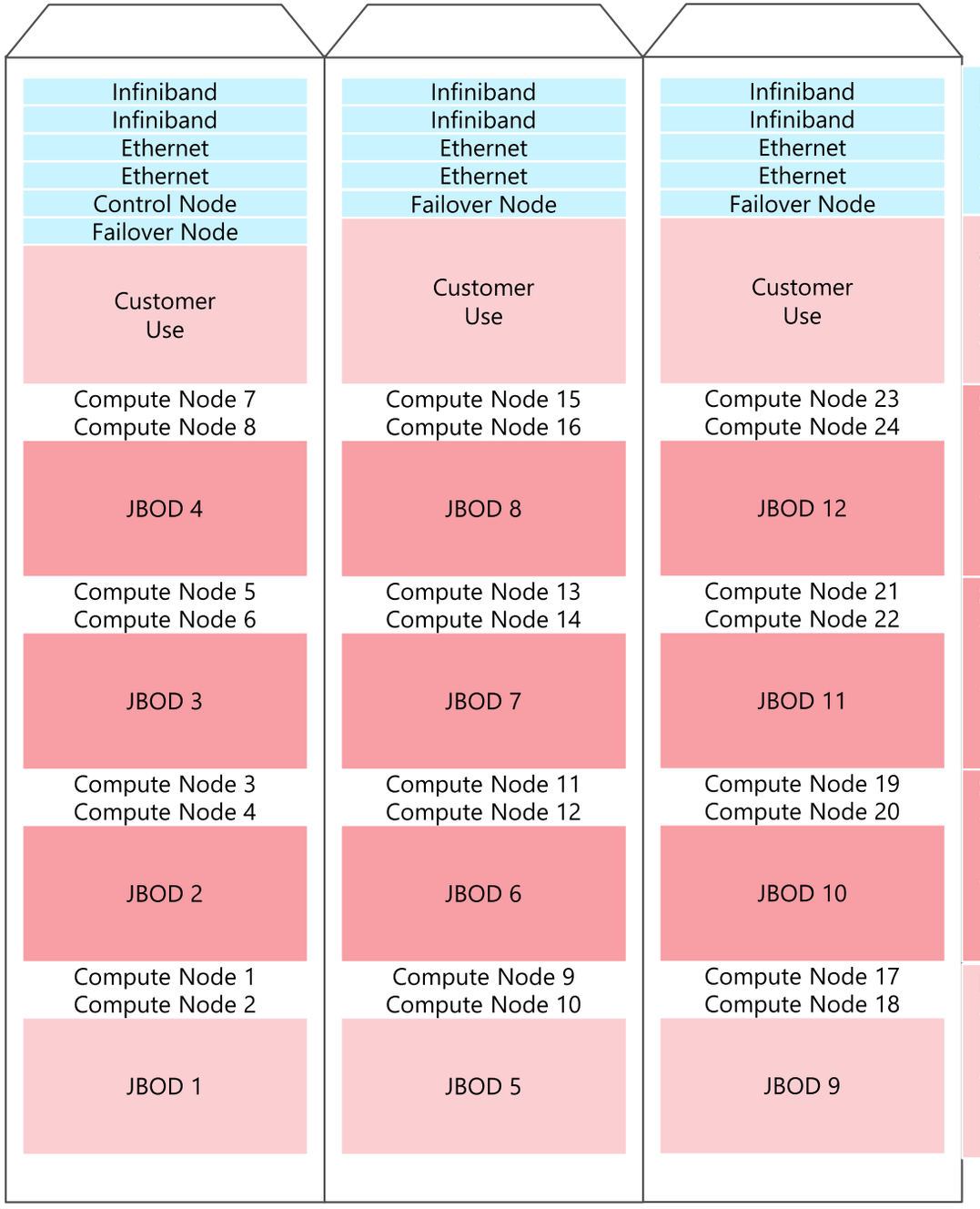
- Pure hardware costs are ~50 percent lower
- Price close to 70 percent lower due to higher capacity
- 70 percent more disk I/O bandwidth

Hardware architecture

Modular design



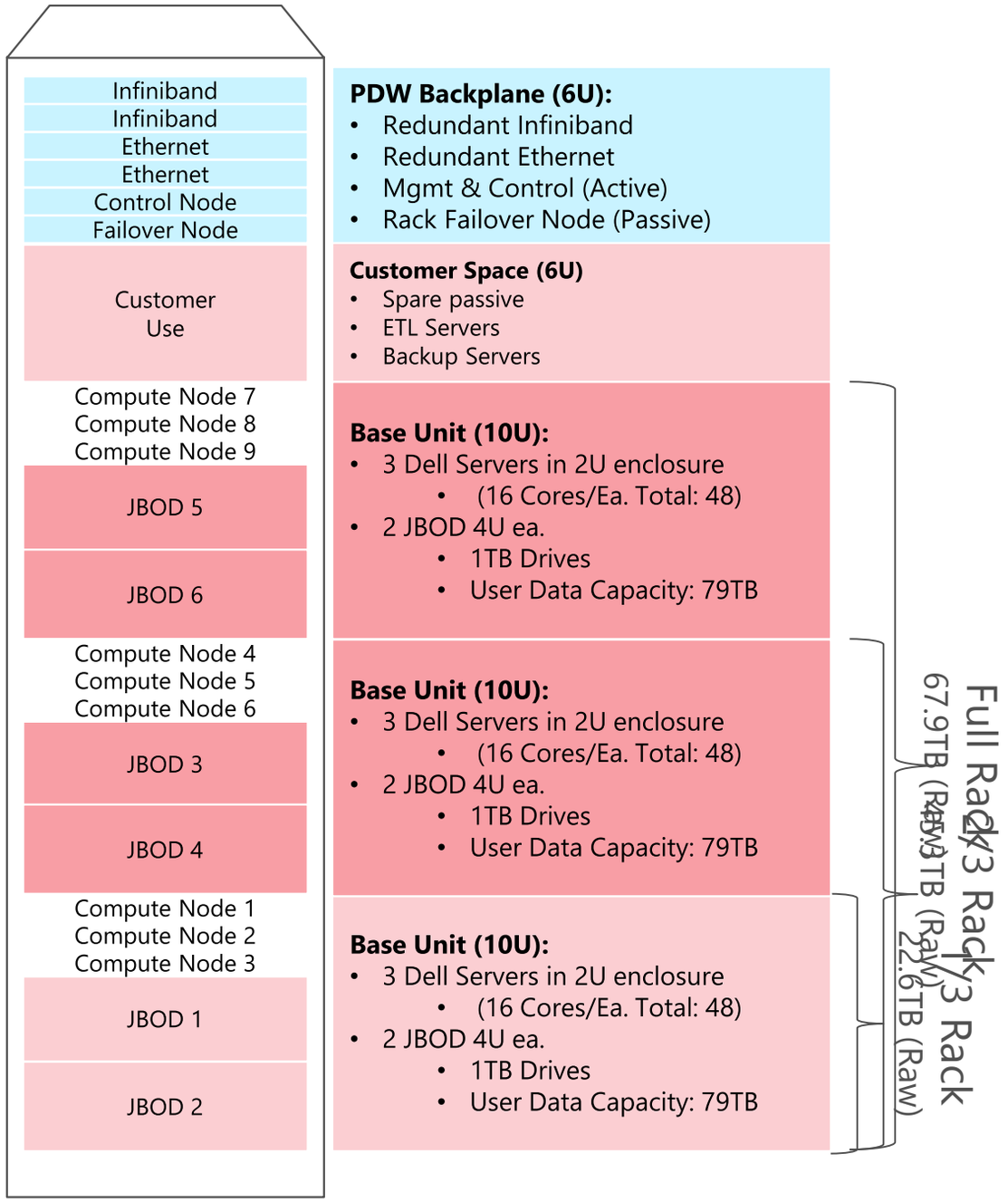
HP Configuration



Details

- 2 – 56 compute nodes
- 1 – 7 racks
- 1, 2, or 3 TB drives
- 15.1 – 1268.4 TB raw
- 53 – 6342 TB User data
- Up to 7 spare nodes available across the entire appliance

Dell Configuration



Details

- 2 – 54 compute nodes
- 1 – 6 racks
- 1, 2, or 3 TB drives
- 22.65 – 1223.1 TB raw
- 79 – 6116 TB User data
- Up to 6 spare nodes available across the entire appliance

Hardware architecture

Supported topologies

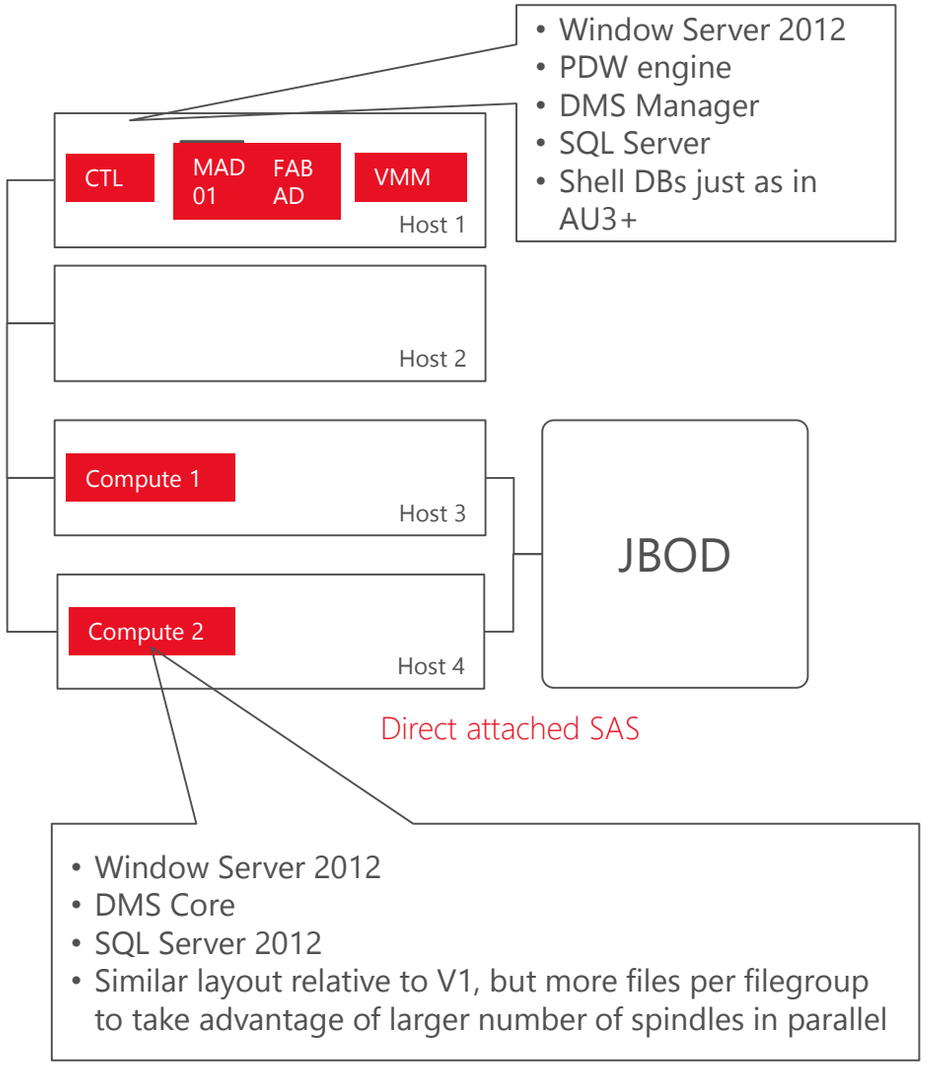
HP	Base	Active	Compute	Incr.	Spare	Total	Raw disk: 1TB	Raw disk: 3TB	Capacity
Quarter-rack	1	0	2	N/A	1	4	15.1	45.3	53-227 TB
Half	1	1	4	100%	1	6	30.2	90.6	106-453 TB
Three-quarters	1	2	6	50%	1	8	45.3	135.9	159-680 TB
Full rack	1	3	8	33%	1	10	60.4	181.2	211-906 TB
One-&-quarter	2	3	10	25%	2	13	75.5	226.5	264-1133 TB
One-&-half	2	4	12	20%	2	15	90.6	271.8	317-1359 TB
Two racks	2	6	16	33%	2	19	120.8	362.4	423-1812 TB
Two and a half	3	7	20	25%	3	24	151	453	529-2265 TB
Three racks	3	9	24	20%	3	28	181.2	543.6	634-2718 TB
Four racks	4	12	32	33%	4	37	241.6	724.8	846-3624 TB
Five racks	5	15	40	25%	5	46	302	906	1057-4530 TB
Six racks	6	18	48	20%	6	55	362.4	1087.2	1268-5436 TB
Seven racks	7	21	56	17%	7	64	422.8	1268.4	1480-6342 TB

DELL	Base	Active	Compute	Incr.	Spare	Total	Raw disk: 1TB	Raw disk: 3TB	Capacity
Quarter-rack	1	0	3	N/A	1	5	22.65	67.95	79-340 TB
2 thirds	1	1	6	100%	1	8	45.3	135.9	159-680 TB
Full rack	1	2	9	50%	1	11	67.95	203.85	238-1019 TB
One and third	2	2	12	33%	2	15	90.6	271.8	317-1359 TB
One and 2 third	2	3	15	25%	2	18	113.25	339.75	396-1699 TB
2 racks	2	4	18	20%	2	21	135.9	407.7	476-2039 TB
2 and a third	3	4	21	17%	3	25	158.55	475.65	555-2378 TB
2 and 2 thirds	3	5	24	14%	3	28	181.2	543.6	634-2718 TB
Three racks	3	6	27	13%	3	31	203.85	611.55	713-3058 TB
Four racks	4	8	36	33%	4	41	271.8	815.4	951-4077 TB
Five racks	5	10	45	25%	5	51	339.75	1019.25	1189-5096 TB
Six racks	6	12	54	20%	6	61	407.7	1223.1	1427-6116 TB

- 2–56 nodes
- 15 TB–1.3 PB raw
- Up to 6PB user data
- 2–3 node increments for small topologies

Software architecture

Overview



General details

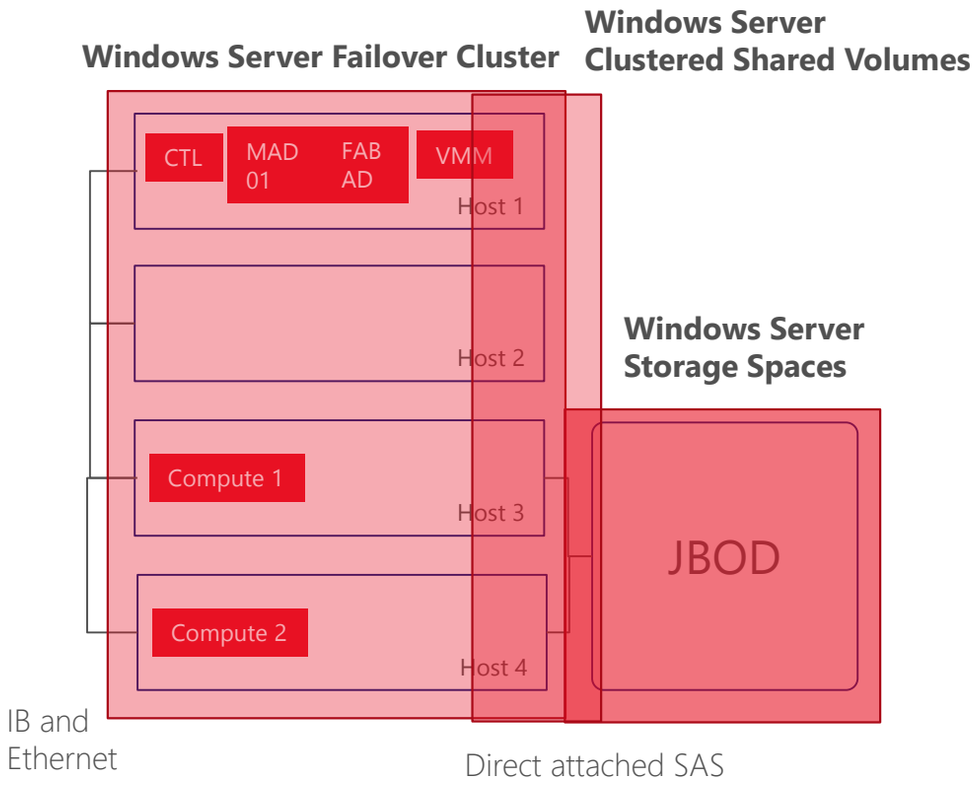
- All hosts run Windows Server 2012 Standard
- All virtual machines run Windows Server 2012 Standard as a guest operating system
- All fabric and workload activity happens in Hyper-V virtual machines
- Fabric virtual machines, MAD01, and CTL share one server; lower overhead costs especially for small topologies
- PDW Agent runs on all hosts and all virtual machines; collects appliance health data on fabric and workload
- DWConfig and Admin Console continue to exist mostly unchanged; minor extensions to expose host level information
- Windows Storage Spaces handles mirroring and spares; enables use of lower cost DAS (JBODs) rather than SAN
- Provisioning based on virtual machines cuts down time and complexity for setup and other maintenance tasks

PDW workload details

- SQL Server 2012 Enterprise Edition (PDW build) is used on control node and compute nodes for PDW workload

Software architecture

High availability changes



Two high-level changes in the high-availability/failover story for SQL Server PDW

- We are more directly involved in maintaining HA for storage
- We use virtual machine migration to move workload nodes to new hosts after hardware failure

Storage details

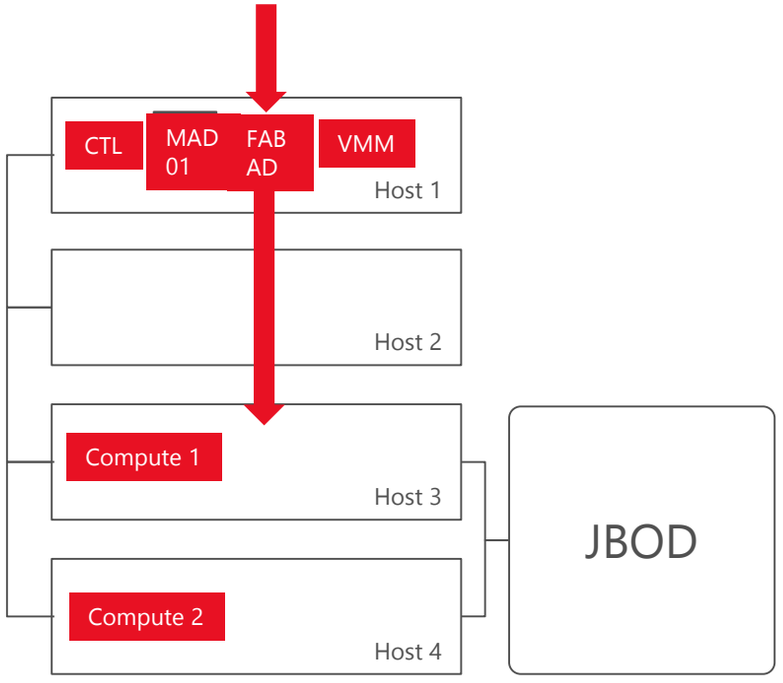
- Storage Spaces manages the physical disks on the JBOD(s)
  - 33 logical mirrored drives
  - 4 hot spares
- CSV allows all nodes to access the LUNs on the JBOD as long as at least one of the two nodes attached to the JBOD is active

Failover details

- One cluster across the whole appliance
- Virtual machines are automatically migrated on host failure
- Affinity and anti-affinity maps enforce rules
- Failback continues to be through CSS use of Windows Failover Cluster Manager

Software architecture

Replacing nodes

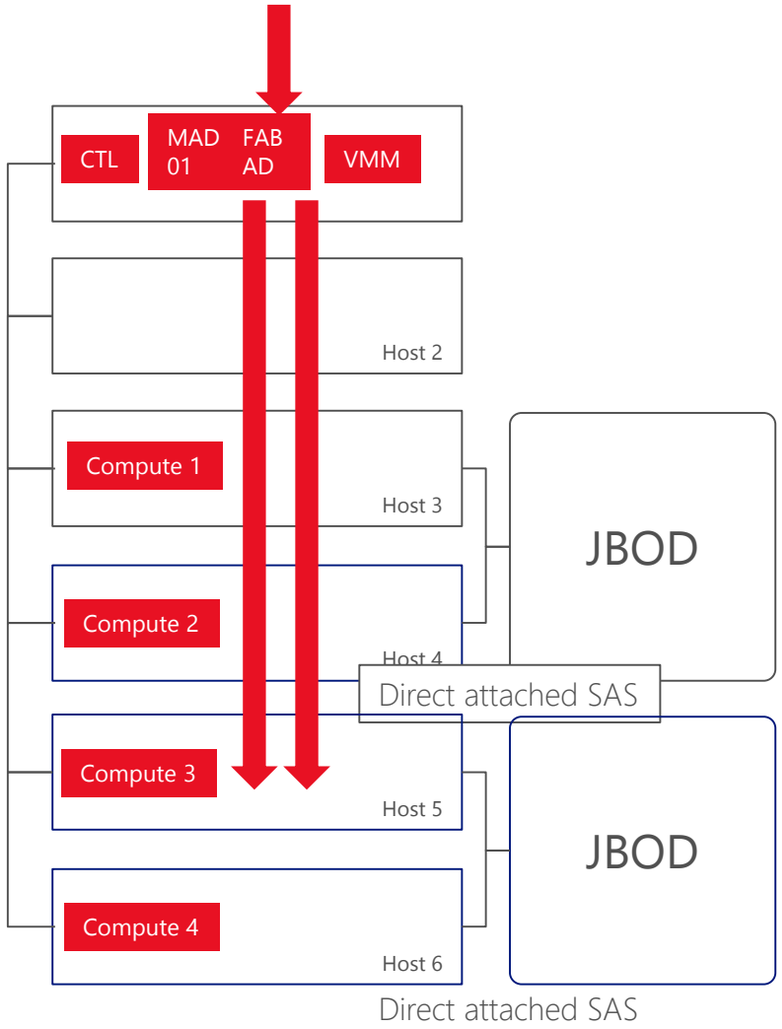


Direct attached SAS

- Single type of node, with the sole differentiating factor being storage attached vs. storage unattached
  - Execution has massive commonality regardless of which specific host is being replaced
- Workloads migrate with the virtual machines
  - Replace node follows a subset of the bare metal provisioning phase of provisioning
  - Workload virtual machines do not have to be re-provisioned
- CSS logs into fabric AD node and executes Setup.exe with the replace node action specified, along with the necessary information targeting the replacement node
- Workload virtual machines are failed back using Windows Failover Cluster Manager (V1 parity)
- At this time, we are not using Live Migration for failover and failback
  - Failback still incurs small downtime
  - There may be a small performance penalty for failed over compute nodes; documentation will suggest failback

Software architecture

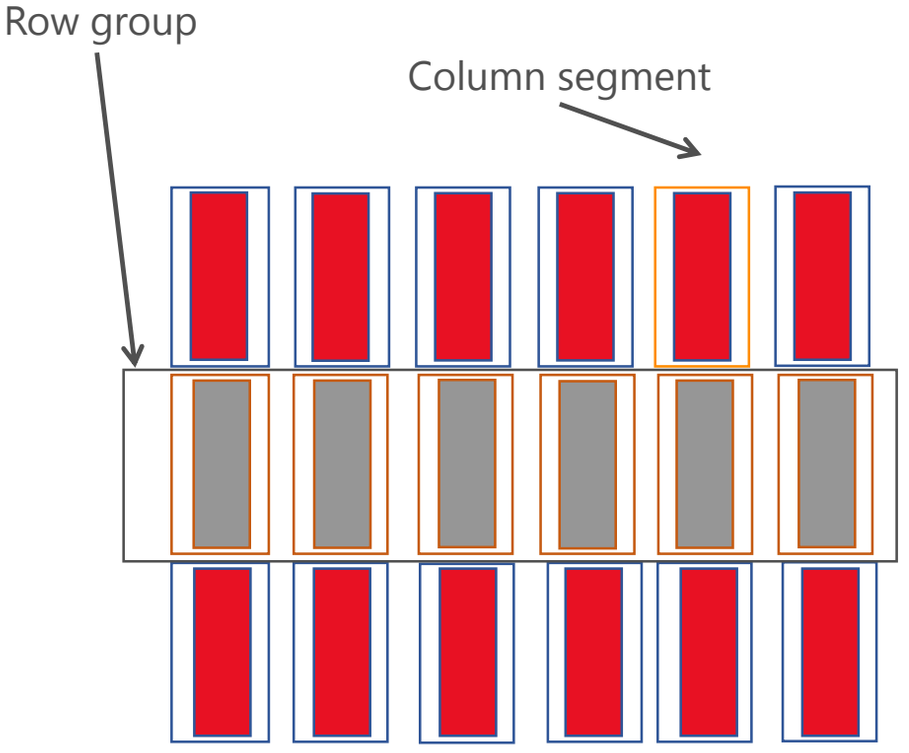
Adding capacity: scaling from 2–56 nodes



- Any addition to the appliance has to be in the form of one or more standard scale units
- IHV owns installation and cabling of new scale units
- Software provisioning consists of three phases:
  - Bare metal provisioning of new nodes
  - Provisioning of workload virtual machines and “hooking up” to other workload virtual machines
  - Redistribution of data
- CSS logs into PDW AD virtual machine and kicks off Add Unit action, which kicks off bare metal provisioning (which is online)
- When BMP completes, the next step takes the appliance offline and completes steps 2 and 3
- Data redistribution cannot be guaranteed to be workable in every situation
  - Tool to test ahead of time whether it will work
  - Step 2 will block if it cannot guarantee success
  - CSS may have to help prepare user data
    - Deleting old data
    - Partition switching from largest tables
    - CRTAS to move data off appliance temporarily

In-Memory DW in PDW

Columnstore terminology



Column segment

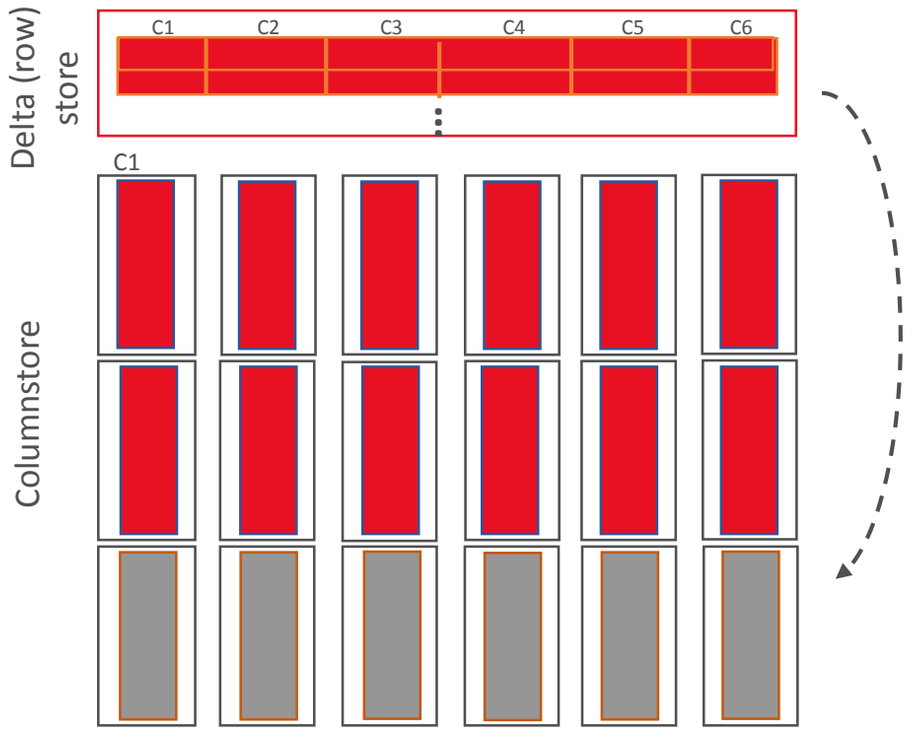
- A **segment** contains values from one column for a set of rows.
- Segments for the same set of rows comprise a **row group**.
- Segments are compressed.
- Each segment is stored in a separate LOB.
- A segment is a unit of transfer between disk and memory.

Tables are stored as segments of **columns rather than rows**.

For data warehouse queries, this often has significant **performance benefits**.

In-Memory DW in PDW

New design: Delta Store



- Table consists of columnstore and row store
- DML (update, delete, insert) operations done against delta store
- "Tuple Mover" converts data into columnar format once segment is full (1M of rows)
- Tuple Mover can be forced by executing REORGANIZE statement
- INSERT Values
  - Always lands into delta store
- DELETE
  - logical operation does not physically remove row until REBUILD is performed
- UPDATE
  - DELETE followed by INSERT
- BULK INSERT
  - if batch is less than 1M, inserts go into delta store (otherwise columnstore)
- SELECT
  - Unifies data from column and row stores for internal UNION operation

In-Memory DW in PDW
Supported functionality

Supporting CLUSTERED COLUMNSTORE indexes

- Columnstore is the **preferred** storage engine in PDW
- No secondary (columnstore or rowstore) indexes are supported

```
CREATE TABLE user_db.dbo.user_table (C1 int, C2 varchar(20))
WITH (DISTRIBUTION = HASH (id), CLUSTERED
COLUMNSTORE INDEX)
```

Functionality supported

- Create < *permanent_or_temp* > Table
- CTAS < *permanent_or_temp* > Table
- Alter Table ADD/DROP/ALTER COLUMN
- Alter Table REBUILD/REORGANIZE, partition switching
- **Full DML** (Insert, Update, Delete, Select)
- Truncate table
- Same functionality as row store, such as create/update statistics, PDW cost model, etc.
- All existing PDW data types supported fully

```
DROP INDEX index_name ON [ database_name . [ schema ]
. | schema . ] table_name [;]

Dropping a columnstore index creates a row store table.
```

Note: non-clustered columnstore indexes (e.g. Apollo2) not supported

In-Memory DW in PDW

Break-through performance

Dramatic performance increases

- 5–10x on customer workloads
- Confirmed with TAP customer workloads

Improved compression on disk and in backups

- 2–3x better compression relative to row store

Preserved appliance model

- Few tuning knobs

Better memory management

- Run-time memory management respects Resource Governor; batch processing can now spill
- Use Resource Classes to increase memory available to improve quality

T-SQL compatibility improvements

Better support for Microsoft and third-party tools

First-class support for Microsoft BI

- Full support for SSAS, SSRS, PowerPivot, Power View (including DirectQuery)

Improved third-party support

- Supports Tableau, Cognos (BI), BOBJ (BI), QlikView, and MicroStrategy

Much broader compatibility with existing ETL and reporting scripts

- Enhanced support for built-in functions for metadata and security
- Named and mixed parameter support for stored procedures
- Output parameter support for system and user-defined stored procedures
- Support for bitwise operators

Looks just like normal SQL Server

- Connectivity through SQL Server clients
- SQL Server Data Tools as the primary query tool

Microsoft

SSD Bufferpool Extension

Microsoft

SSD Bufferpool Extension

SSD Buffer Pool Extension and scale up

- What's being delivered
  - Use of non-volatile drives (SSD) to extend buffer pool
  - NUMA-Aware large page and BUF array allocation
- Main benefits
  - BP extension for SSDs
    - Improve OLTP query performance with no application changes
    - No risk of data loss (using clean pages only)
    - Easy configuration optimized for OLTP workloads on commodity servers (32 GB RAM)
      Example:
      ALTER SERVER CONFIGURATION
      SET BUFFER POOL EXTENSION ON
      (FILENAME = 'F:\SSDCACHE\EXAMPLE.BPE',
      SIZE = 50 GB)
  - Scalability improvements for systems with more than eight sockets

Ease of Use

```
-- View Buffer Pool Extension Details to see if it is enabled or not
SELECT * FROM sys.dm_os_buffer_pool_extension_configuration
GO

-- Monitor Buffer Pool Extension usage to see if any data or index page(s) are in Buffer Pool or not
-- (last column of the query result)
SELECT * FROM sys.dm_os_buffer_descriptors
GO

-- Disable Buffer Pool Extension is very easy
ALTER SERVER CONFIGURATION SET BUFFER POOL EXTENSION OFF
GO
```

Troubleshooting options

DMVs

- sys.dm_os_buffer_pool_extension_configuration
- sys.dm_os_buffer_descriptors

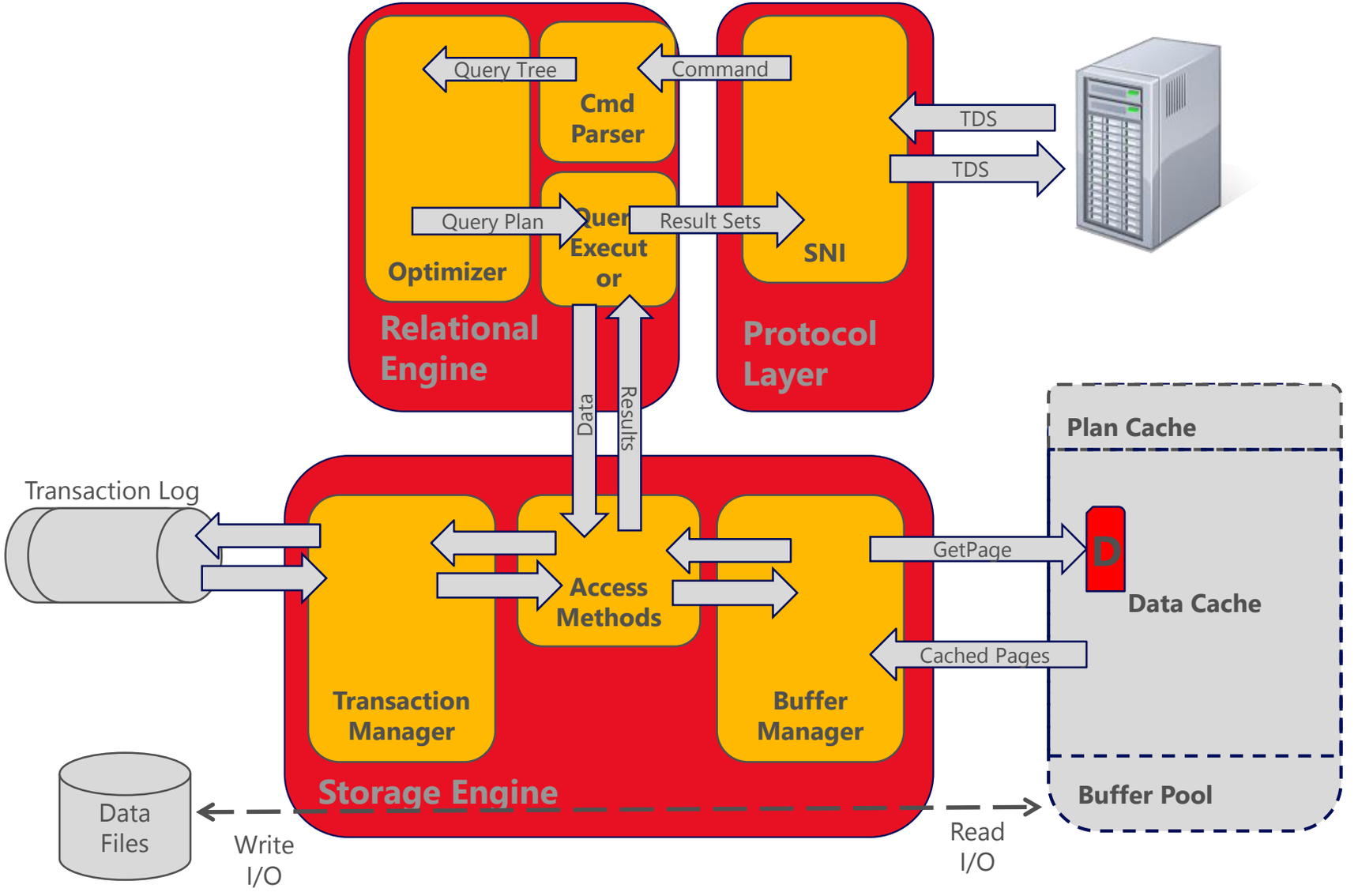
XEvents

- sqlserver.buffer_pool_extension_pages_written
- sqlserver.buffer_pool_extension_pages_read
- sqlserver.buffer_pool_extension_pages_evicted
- sqlserver.buffer_pool_page_threshold_recalculated

Performance counters

- Extension page writes/sec
- Extension page reads/sec
- Extension outstanding IO counter
- Extension page evictions/sec
- Extension allocated pages
- Extension free pages
- Extension page unreferenced time
- Extension in use as percentage on buffer pool level

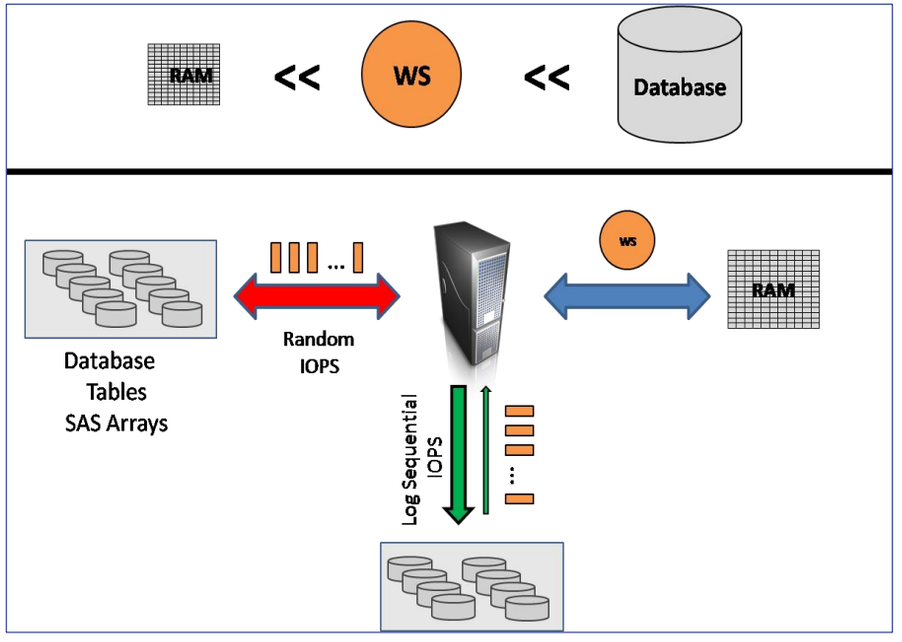
Buffer Pool Manager



IOPS offload to Storage Class Memory (SCM) in memory hierarchy

RANDOM IOPS to mechanical drives

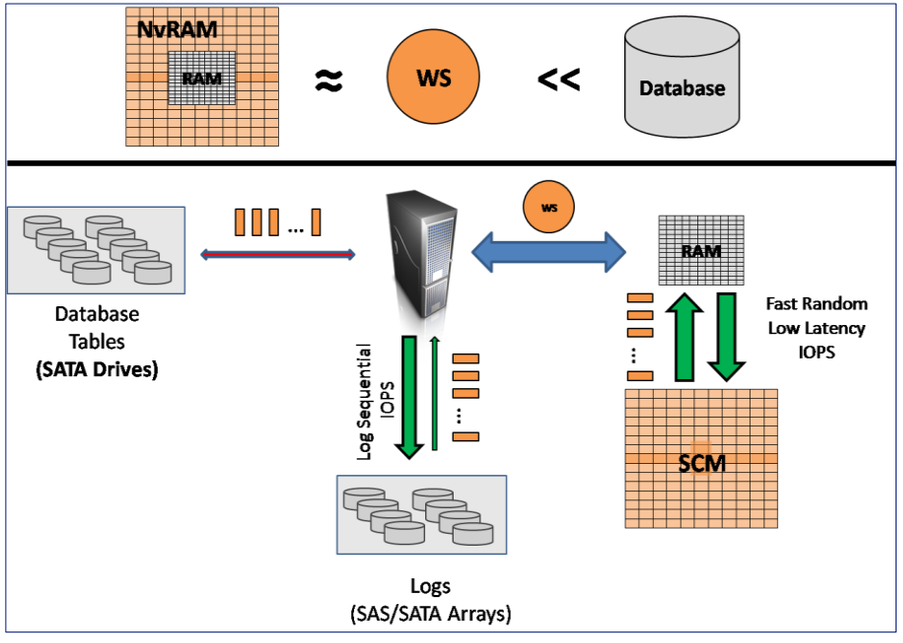
(before Buffer Pool Extension)



More traffic into disk drives

IOPS offload to SCM

(after Buffer Pool Extension)



Less traffic into disk drives

Microsoft

Enhanced Query Processing

Microsoft

Enhanced Query Processing

Query processing enhancements

- What's being delivered
  - New cardinality estimator
  - Incremental statistics for partition
  - Parallel SELECT INTO
- Main benefits
  - Better query performance:
    - Better choice of query plans
    - Faster and more frequent statistics refresh on partition level
  - Consistent query performance
  - Better supportability using two steps (decision making and execution) to enable better query plan troubleshooting
  - Loading speed into table improved significantly using parallel operation

Secure

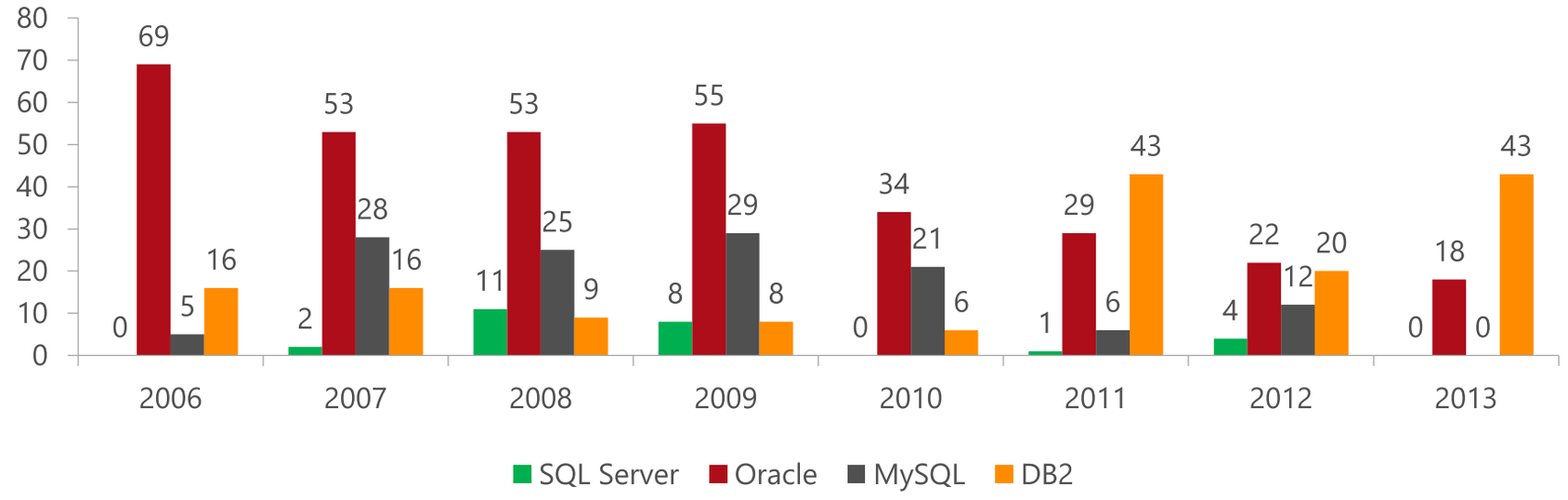
Least vulnerable database

5 years in a row

Most used database in the world

46% market share

Database Vulnerabilities



Key features

Redefined Engineering Security Processes

CC Certification at High Assurance Level

Enhanced Separation of Duty

Transparent Data Encryption

Encryption Key Management

Support for Windows Server Core

Sources: National Institute of Standards and Technology Comprehensive Vulnerability Database 4/17/2013, Market share from IDC 2013

Microsoft

Security Enhancements

Microsoft

Security Enhancements

Separation of duties enhancement

- Four new permissions
  - CONNECT ANY DATABASE (server scope)
  - IMPERSONATE ANY LOGIN (server scope)
  - SELECT ALL USER SECURABLES (server scope)
  - ALTER ANY DATABASE EVEN SESSION (database scope)
- Main benefit
  - Greater role separation to restrict multiple DBA roles
  - Ability to create new roles for database administrators who are not sysadmin (super user)
  - Ability to create new roles for users or apps with specific purposes

Best Practices for Separation of Duties

- Eliminate the use of superusers (SA login, SYSADMIN server role, DBO database user)
- Use permission system rather than superuser
- Use CONTROL SERVER (server level) and CONTROL DATABASE (database level) instead and use DENY for specifics
- Always document the user of ownership chains

Example Roles for Separation of Duties

- Software Installer:
  - Specific Active Directory Windows user account enabled during installation and disabled when not doing installation
- Instance Identity Manager:
  - Can be designated as server role (SQL Server 2012 or above)
  - Enable ALTER ANY LOGIN, ALTER ANY SERVER ROLE permissions
- Highest-level DBA:
  - Can be designated as server role (SQL Server 2012 or above)
  - Enable CONTROL SERVER permission but disable ALTER ANY LOGIN, ALTER ANY SERVER ROLE, IMPERSONATE ANY LOGIN permissions
  - Disable EXECUTE/SELECT/INSERT/UPDATE/DELETE on a per-database schema or table level
  - Or Disable SELECT ALL USER SECURABLES permission combined with disabling EXECUTE permission because of ownership chaining

Example (cont'd)

- Lower-level DBA:
  - Enable CONTROL SERVER permission but …
  - Disable DENY ALTER ANY CERTIFICATE or DENY ALTER ANY ASYMMETRIC KEY to prevent creation of signed modules
  - Disable CONNECT to specific user databases
  - DENY permissions on certain instance-level securables (e.g. DENY ALTER ANY LINKED SERVER)
- Troubleshooter:
  - Enable specific permission such as ALTER ANY EVENT SESSION, ALTER ANY DATABASE EVENT SESSION, ALTER TRACE, ALTER ANY SERVER EVENT NOTIFICATION, ALTER ANY DATABASE EVENT NOTIFICATION
- Auditing:
  - Enable CONNECT ANY DATABASE and VIEW ANY DEFINITION permissions
  - Enable ALTER ANY SERVER AUDIT and ALTER ANY DATABASE AUDIT (if using SQL Server built-in auditing facility)
  - Enable SELECT ALL USER SECURABLES (if user table auditing is needed)

Backup Encryption

- Increase security of backups stored separate from the instance (another environment such as the Cloud)
- Encryption keys can be stored on-prem while backup files in the cloud
- Support non-encrypted databases (don’t need to turn on Transparent Data Encryption anymore)
- Different policies for databases and their backups

T-SQL BACKUP/RESTORE

```
BACKUP DATABASE <dbname> TO <device> = <path to device>
WITH
ENCRYPTION
(
  ALGORITHM = <Algorithm_name> ,
 { SERVER  CERTIFICATE = <Encryptor_Name> |
   SERVER  ASYMMETRIC KEY = <Encryptor_Name> }
);
```

No changes to RESTORE

T-SQL Views

- msdb.dbo.backupset

backup_set_id	name	key_algorithm	encryptor_thumbprint	encryptor_type
3	Full Backup	NULL	NULL	NULL
4	Full Backup	aes_256	0x00B1BD62DAA0196	CERTIFICATE

- msdb.dbo.backupmediaset

media_set_id	is_password_protected	is_compressed	is_encrypted
3	0	1	0
4	0	1	1

Additional Details

- AES 128, AES 192, AES 256, and Triple DES
- Unique backup key is generated for each backup
- Certificate
- Asymmetric key from an EKM provider only
- All operations require certificate or key
- Appended backup is **not** supported
- Compression has no effect on pre-encrypted databases

Scalable

Scale compute

Up 640 logical processors

64 vCPUs/virtual machine

1 TB of memory/virtual machine

64 nodes/cluster

Scale networking

Network virtualization provides flexibility and isolation

Assign minimum and maximum bandwidth

Scale storage

Storage virtualization

Enterprise-class network storage on standard hardware

Storage tiering for higher performance

SQL Server 2014 Resource Governance

	CPU	Memory	IO	
Pool 1	20%	10%	30%	App 1
Pool 2	30%	50%	30%	App 2, App 3
Pool 3	50%	40%	40%	App 4, App 5, App 6
	100%	100%	100%	

Key features

SQL Server 2014

- Resource Governor adds IO governance
- Sysprep at cluster level

Windows Server 2012 R2

- Hyper-V
- Storage Spaces
- NIC Teaming
- Online VHDX Resize
- Network and Storage QoS

Microsoft

Better Together

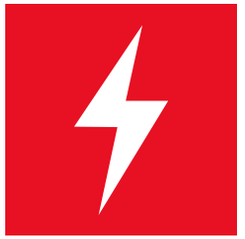
Microsoft

Better Together

Better together

SQL Server, Windows Server, System Center

Scaling



Performance

Availability

Storage

Management

Better scaling with Windows Server

Technical benefits

Increased virtual processor and memory

Enables SQL Server virtual machine to use up to 64 virtual processors and 1 TB of memory

16x
increase over previous versions

Increased logical processor and memory

Enables SQL Server to use up to 640 logical processors and 4 TB of memory

Up to 640
logical processors

Increased cluster node scalability

Supports SQL Server clusters up to 64 nodes

4X
increase over previous versions

Increased virtual machine density

Up to 8,000 SQL Server virtual machines per cluster

Support for up to 320 logical processors and 4 TB of memory

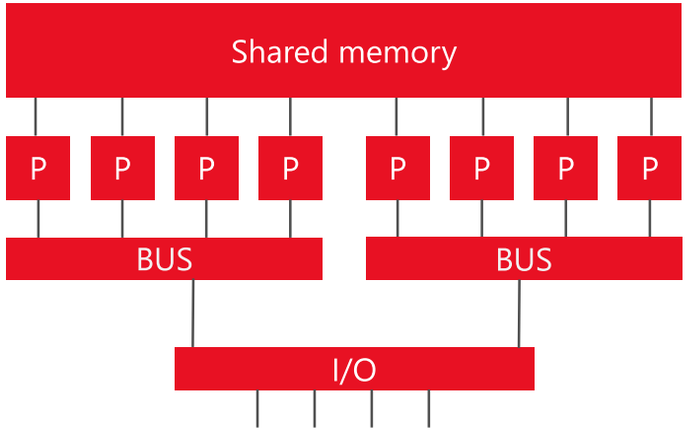
8x
increase over previous versions

Better performance with Windows Server

Technical benefits

Support for NUMA

Optimization in SQL Server virtual machine, including thread scheduler and memory allocation



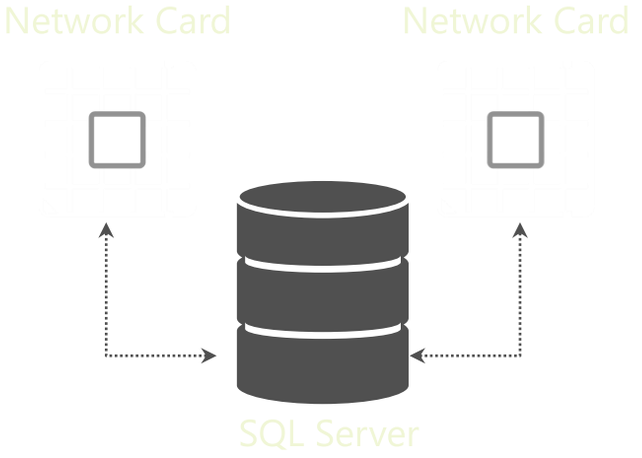
QoS – Network Bandwidth Enforcing

Bandwidth management for multiple SQL Server services (Virtual Machine, Storage, Live Migration, CSV)

Windows NIC Teaming

- Load-balancing to increase bandwidth for SQL Server network traffic by configuring multiple NICs
- Failover helps maintain availability of SQL Server by configuring multiple NICs for hardware failover

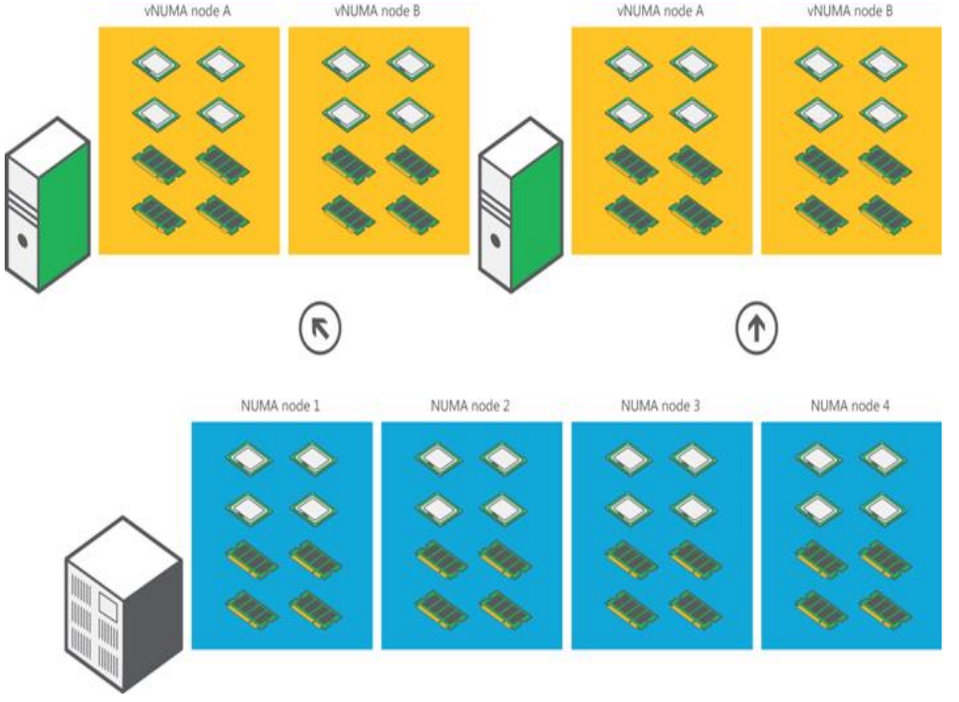
FAILOVER AND LOAD BALANCING



NUMA: Hyper-V host support

Non-Uniform Memory Access (NUMA) support in a virtual machine

- Projects NUMA topology onto a virtual machine
- Enables guest operating systems and applications to make intelligent NUMA decisions
- Aligns guest NUMA nodes with host resources



Guest NUMA topology by default matches host NUMA topology

Quality of Service

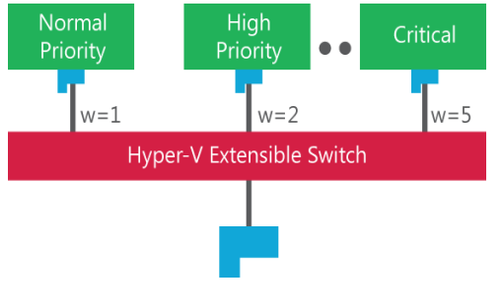
Features

- Establishes a bandwidth floor
- Assigns specified bandwidth for each type of traffic
- Helps to ensure fair sharing when there is no congestion
- Can exceed quota when there is no congestion

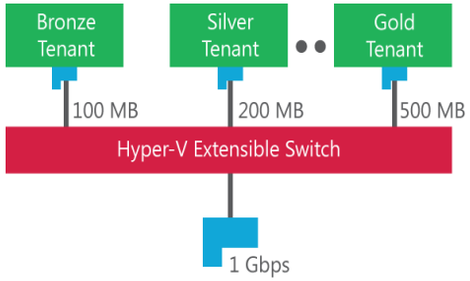
Two mechanisms

- Enhanced packet scheduler (software)
- Network adapter with DCB support (hardware)

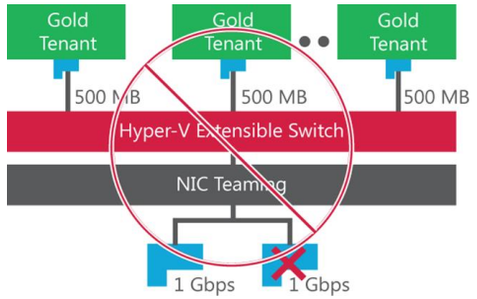
Relative minimum bandwidth



Strict minimum bandwidth



Bandwidth oversubscription



Better availability with Windows Server

Technical benefits

Cluster-Aware Updating (CAU)

Enables updates to be applied automatically to the host operating system in clustered SQL Server environments while maintaining availability during the update process

Dynamic Quorum

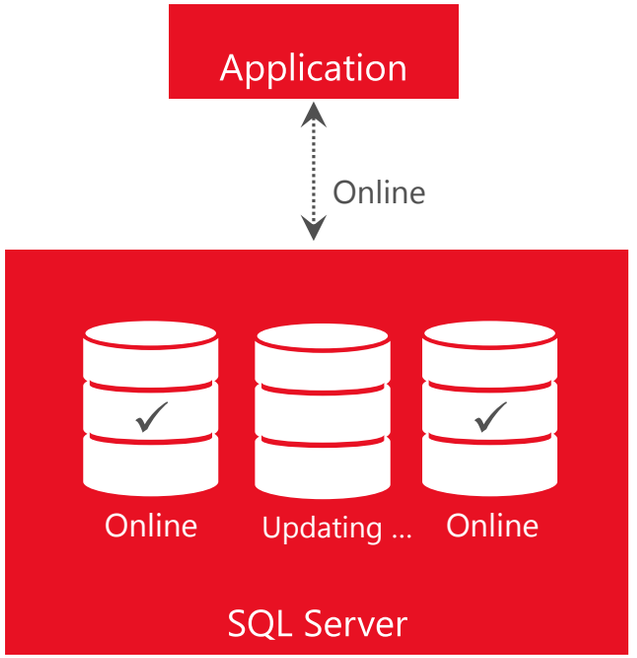
Enables SQL Server AlwaysOn cluster to dynamically adjust the number of required quorum votes

Windows Server Core

Eliminate 50–60 percent of OS-level patches and reduce OS reboot while enabling easy conversion between full GUI mode and Server Code mode

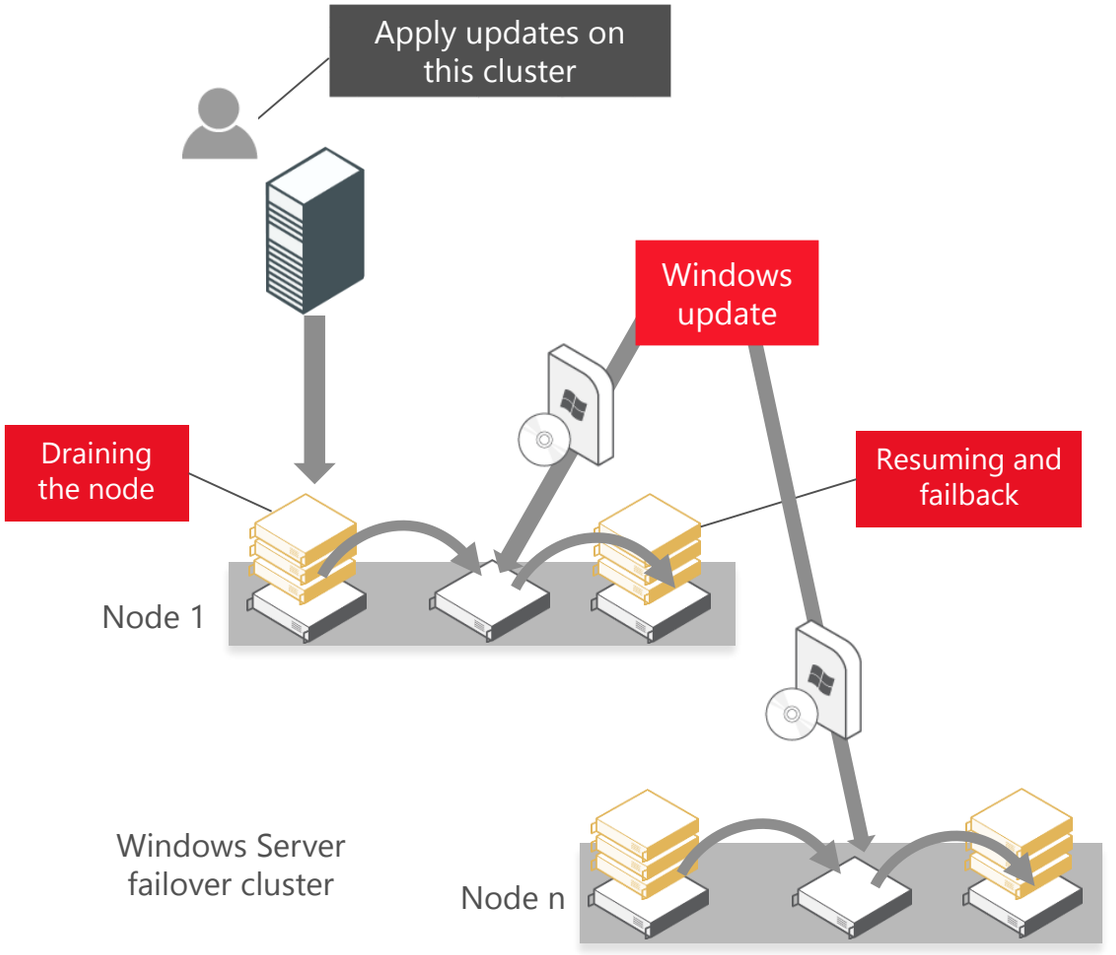
Online VHDX resize (Windows Server 2012 R2)

Scale SQL Server virtual machines without downtime



CLUSTER-AWARE UPDATING

Cluster-Aware Updating



- Eliminates downtime associated with cluster updating
- Simplifies cluster updates through configurable automation
- Transparent to users and hosted applications
- Extensible to install even non-Windows software updates through custom plug-ins

Better storage with Windows Server

Technical benefits

SMB support

Enables SQL Server to store data files with remote shared folders that use SMB (continuously available file server)

Fibre Channel support

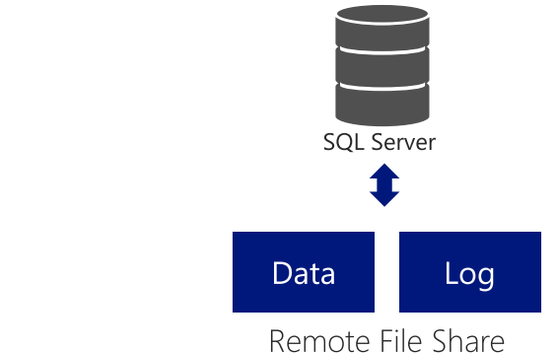
Enables SQL Server virtual machine to directly connect to Fibre Channel to support N_Port ID Virtualization (NPIV), virtual SAN, and multipath IO (MPIO)

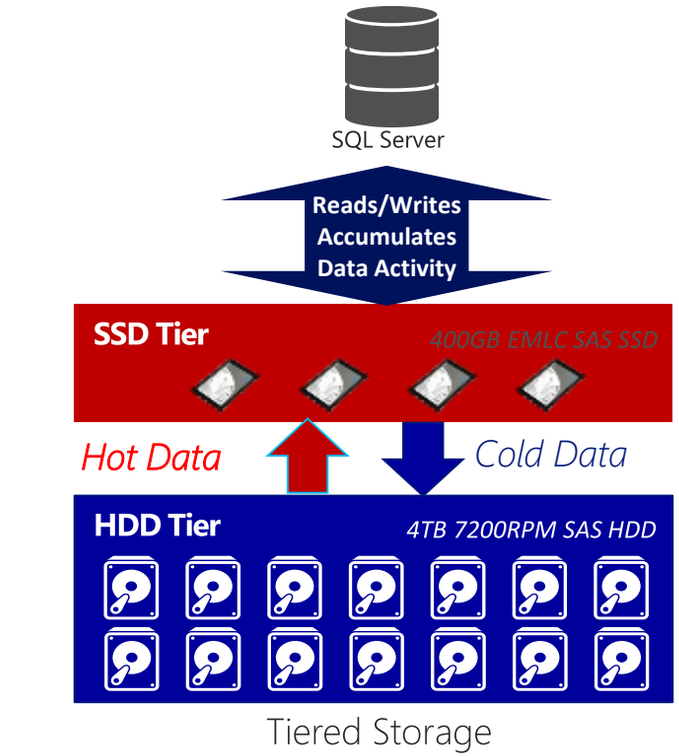
ReFS support (SQL Server 2014)

Better protection for SQL Server data files with background checksum processes in resilient file systems (backward compatible with NTFS)

Tiered Storage Spaces (Windows Server 2012 R2)

- Improves the flexibility of SQL Server storage through Storage Pools with resilient storage (mirroring and parity) and multitenancy isolation (ACLs)
- Optimizes performance by automatically moving SQL Server data blocks from/to faster disk (SSD) to/from regular disk (HDD) based on access patterns (data activity)



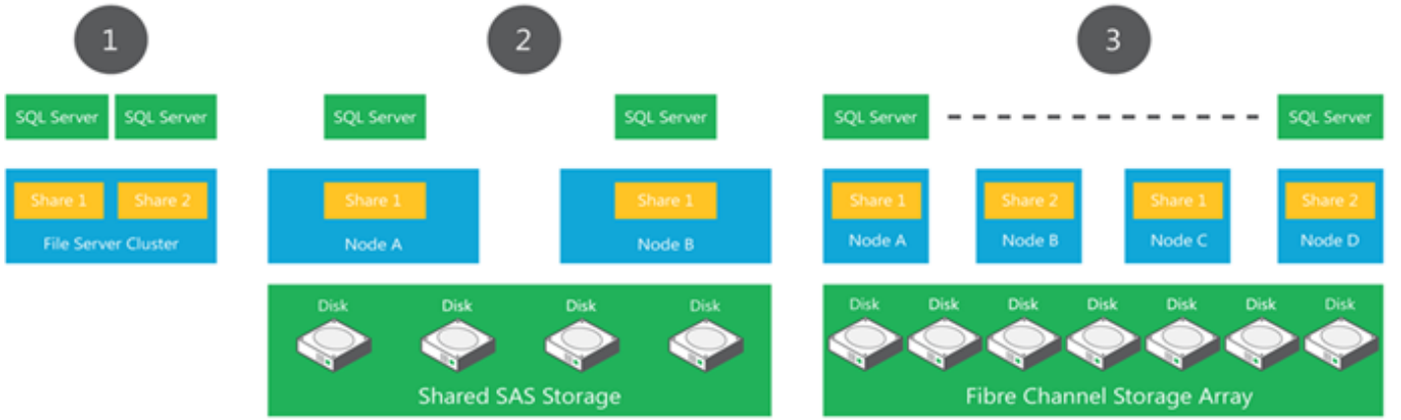


http://download.microsoft.com/download/8/0/F/80FCCBEF-BC4D-4B84-950B-07FBE31022B4/ESG-Lab-Validation-Windows-Server-Storage.pdf

SMB support for SQL Server and Hyper-V

- Hyper-V can now take advantage of file share storage for virtual storage
- Application-specific capabilities for SQL Server and Hyper-V
- Take advantage of new high-performance and high-availability SMB features for applications
- CAU for applications

Single-Node File Server (D1)	Dual-Node Cluster File Server (D2)	Multi-Node Clustered File Server (D3)
Low cost	Medium cost	Higher cost
Shares not continuously available	Shares continuously available	Shares continuously available
Limited scalability (~100 spindles)	Medium scalability (~200 spindles)	Highest scalability (~1,000 spindles)



Better management with System Center

Technical benefits

Virtual Machine Manager and App Controller

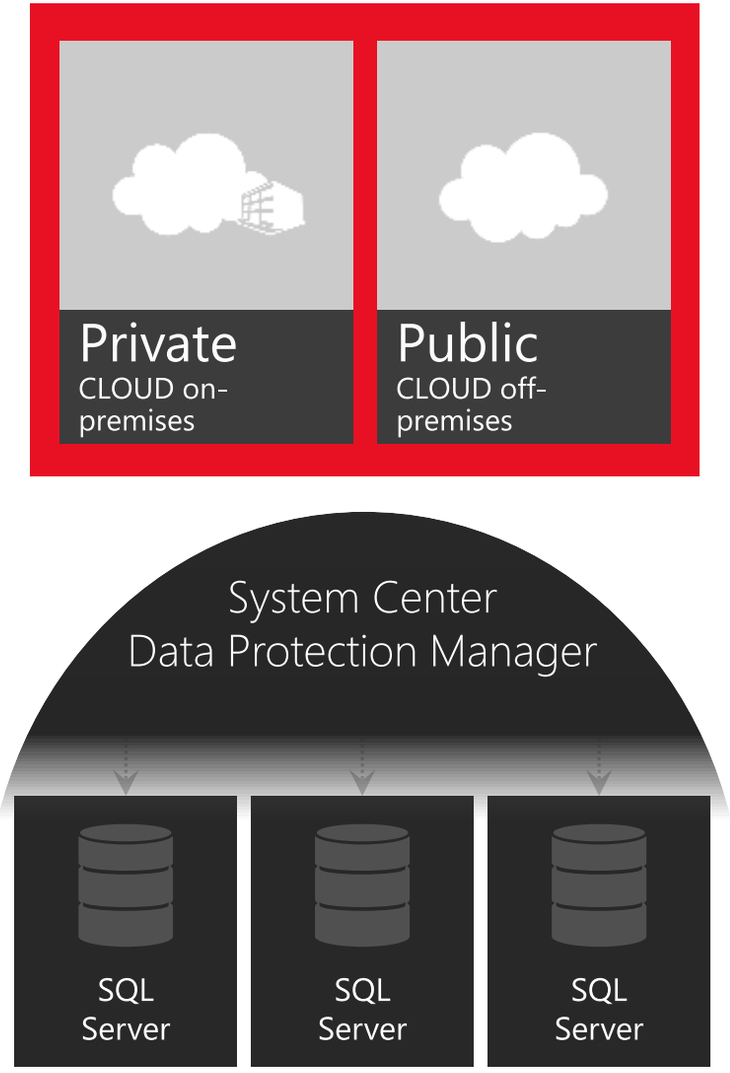
- Creation and management of a private cloud based on SQL Server virtual machines
- Deployment of SQL Server virtual machine across private and public cloud environments

Operations Manager and Advisor

- Proactive and reactive monitoring of SQL Server instances
- Early detection and problem resolution of SQL Server issues using agent-based operations that perform continuous server scanning

Data Protection Manager

Backup and restore of SQL Server databases for multiple configurations (single instance, clustered instances, and mirrored instances)



SQL Server 2014 for Private Cloud

Resource Pooling
Consolidate databases

BENEFITS
- **Reduce capital expenses**
  - Standardize and consolidate
- **Reduce operational expenses**
  - Improve hardware utilization
  - Manage IT infrastructure efficiently
- **Green IT**
  - Reduce space and power needs

CONSOLIDATE

STEPS	TOOLS OR FEATURES
Discover database sprawl and capacity planning	Microsoft® Assessment and Planning (MAP) Toolkit
Consolidation options	SQL Server Upgrade Advisor SQL Server Migration Assistant (SSMA)

Physical to virtual migration	System Center Virtual Machine Manager
Virtualize and manage instances	Windows Server Hyper-V™ System Center Virtual Machine Manager and App Controller

Elasticity
Scale resources efficiently

BENEFITS
- **Greater agility**
  - Handle peak load scenarios faster
  - Scale to mission-critical workloads
- **Dynamic infrastructure**
  - Scale to efficiently meet demand
  - High availability across multiple data centers

SCALE

STEPS	TOOLS OR FEATURES
Setup for high availability	SQL Server AlwaysOn Failover Clustering and Availability Group Windows Server Core, Dynamic Quorum, ReFS support

Setup for disaster recovery	SQL Server AlwaysOn Hyper-V Live Migration

Scale virtual machines (CPU, memory, storage, network)	In-Memory OLTP, SSD buffer pool extension, Columnstore, Resource Governor, Hyper-V scale, Dynamic Memory, online VHDX, Tiered Storage, SMB support, NIC teaming & QoS

Load balance virtual machines	System Center Virtual Machine Manager

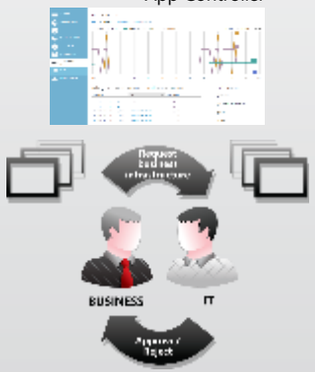
Self Service
Deploy resources on demand

BENEFITS
- **Faster time to market**
  - Automation without compromising control
- **Reduce administration overhead**
- **Business units can request resources on demand**

DEPLOY

STEPS	TOOLS OR FEATURES
Create database virtual machine templates	SQL Server Sysprep System Center Virtual Machine Manager

Build automation	Windows Azure Pack for Windows Server, System Center Service Manager, Orchestrator, App Controller



Usage Based
Optimize IT to business priorities

BENEFITS
- **Make IT strategic by mapping consumption to business priorities**
- **Get a centralized view of total resource consumption across the enterprise**

DRIVE

STEPS	TOOLS OR FEATURES
Measure usage - Assign cost - Map usage to business units	Windows Azure Pack for Windows Server, System Center Service Manager
Charge back and reporting	Windows Azure Pack for Windows Server, SQL Server Analysis Services OLAP cubes and Excel PowerPivot, Power View

Call to Action

Learn more at
http://www.microsoft.com/SQLServerPrivateCloud

MANAGE THROUGH A SINGLE PANE OF GLASS

Microsoft

Resource Governor

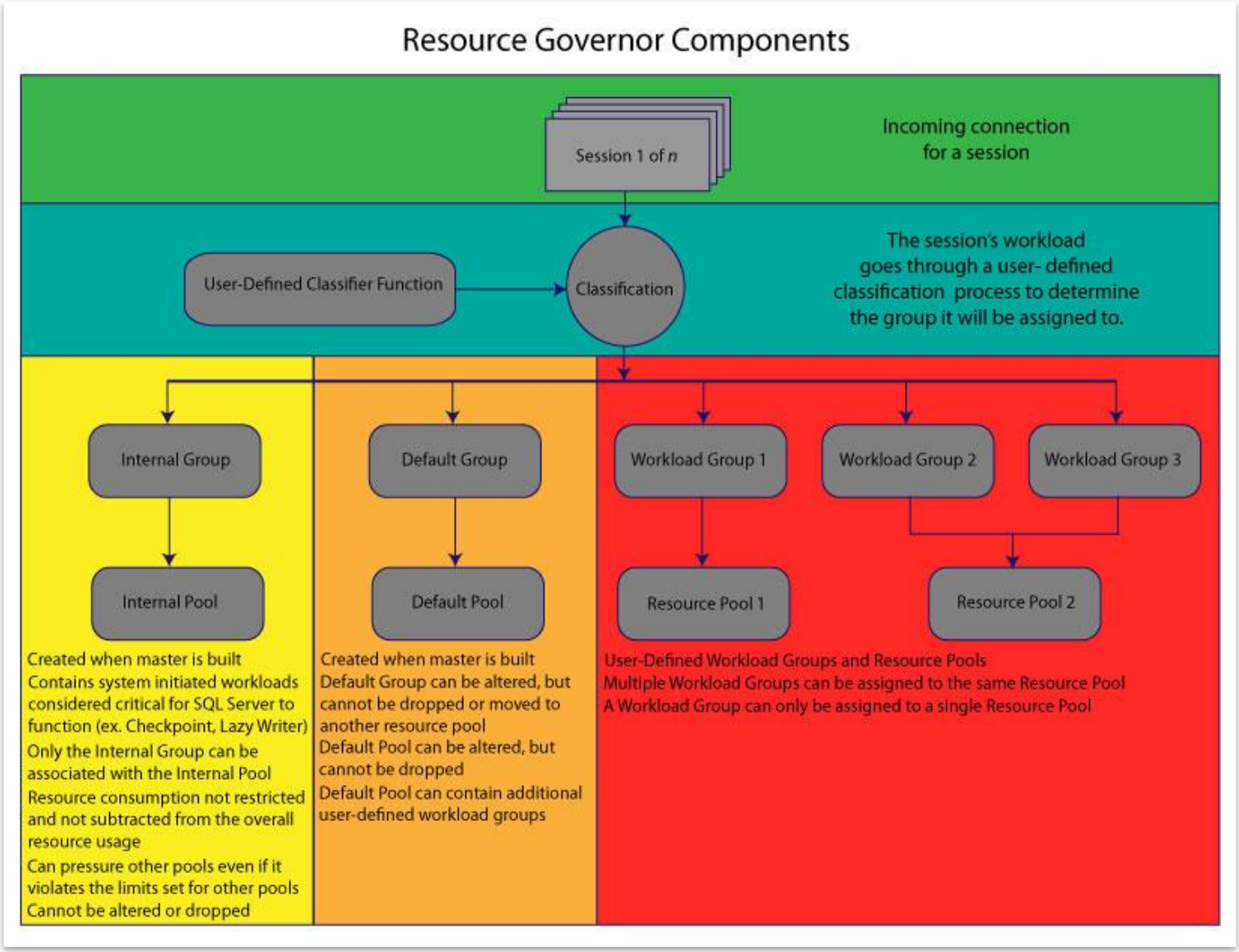
Microsoft

Resource Governor

Resource Governor goals

- Ability to differentiate workloads
- Ability to monitor resource usage per group
- Limit controls to enable throttled execution or prevent/minimize probability of “run-aways”
- Prioritize workloads
- Provide predictable execution of workloads
- Specify resource boundaries between workloads

Resource Governor components



Complete Resource Governance

- What's being delivered
  - Add max/min IOPS per volume to Resource Governor pools
  - Add DMVs and perfcounters for IO statistics per pool per volume
  - Update SSMS Intellisense for new T-SQL
  - Update SMO and DOM for new T-SQL and objects
- Main benefits
  - Better isolation (CPU, memory, and IO) for multitenant workloads
  - Guarantee performance in private cloud and hosters scenario

Resource Pools

- Represents physical resources of server
- Can have one or more workloads assigned to pool
- Pool divided into shared and non-shared
- Pools control min/max for CPU/memory and now IOPS

```
CREATE RESOURCE POOL pool_name
[ WITH
    ( [ MIN_CPU_PERCENT = value ]
    [ [ , ] MAX_CPU_PERCENT = value ]
     [ [ , ] CAP_CPU_PERCENT = value ]
     [ [ , ] AFFINITY {SCHEDULER = AUTO |
(Scheduler_range_spec) | NUMANODE =
(NUMA_node_range_spec)} ]
    [ [ , ] MIN_MEMORY_PERCENT = value ]
    [ [ , ] MAX_MEMORY_PERCENT = value ]
    [ [ , ] MIN_IOPS_PER_VOLUME = value ]
    [ [ , ] MAX_IOPS_PER_VOLUME = value ])
]
```

Resource Pools

- Minimums across all resource pools can not exceed 100 percent
- Non-shared portion provides minimums
- Shared portion provides maximums
- Pools can define min/max for CPU/Memory/IOPS
  - Mins defined non-shared
  - Max defined shared

Steps to implement Resource Governor

- Create workload groups
- Create function to classify requests into workload group
- Register the classification function in the previous step with the Resource Governor
- Enable Resource Governor
- Monitor resource consumption for each workload group
- Use monitor to establish pools
- Assign workload group to pool

Resource Governor scenarios

- **Scenario 1:** I just got a new version of SQL Server and would like to make use of resource governor. How can I use it in my environment?
- **Scenario 2 (based on Scenario 1):** Based on monitoring results I would like to see an event any time a query in the ad-hoc group (groupAdhoc) runs longer than 30 seconds.
- **Scenario 3 (based on Scenario 2):** I want to further restrict the ad-hoc group so it does not exceed 50 percent of CPU usage when all requests are cumulated.

Monitoring Resource Governor

- System views
  - sys.resource_governor_resource_pools
  - sys.resource_governor_configuration
- DMVs
  - sys.dm_resource_governor_resource_pools
  - sys.dm_resource_governor_resource_pool_volumes
  - sys.dm_resource_governor_configuration
- New performance counters
  - SQL Server: Resource Pool
  - SQL Server: Workload group
- XEvents
  - file_read_enqueued
  - file_write_enqueued

Monitoring Resource Governor

- System views
  - sys.resource_governor_resource_pools
  - sys.resource_governor_configuration
- DMVs
  - sys.dm_resource_governor_resource_pools
  - sys.dm_resource_governor_resource_pool_volumes
  - sys.dm_resource_governor_configuration
- New performance counters
  - SQL Server:Resource Pool Stats
  - SQL Server:Workload group
- XEvents
  - file_read_enqueued
  - file_write_enqueued

Microsoft

Sysprep

Microsoft

Sysprep

Sysprep: Why invest?

Support SQL Server images in Azure Gallery

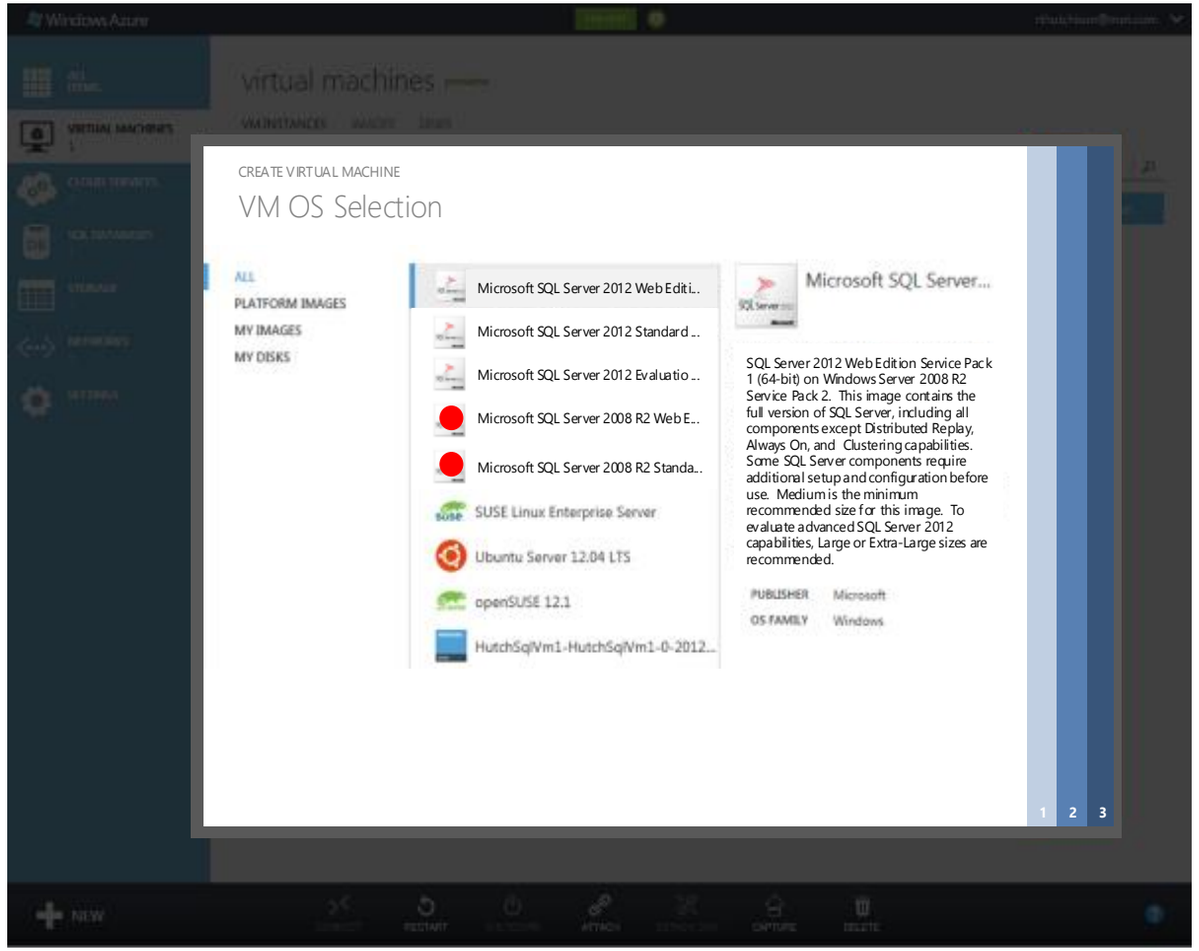
Provide quick and flexible SQL Server provisioning for IaaS scenarios

Support SQL Server configuration as part of the provisioning process

Need to be faster than full installation

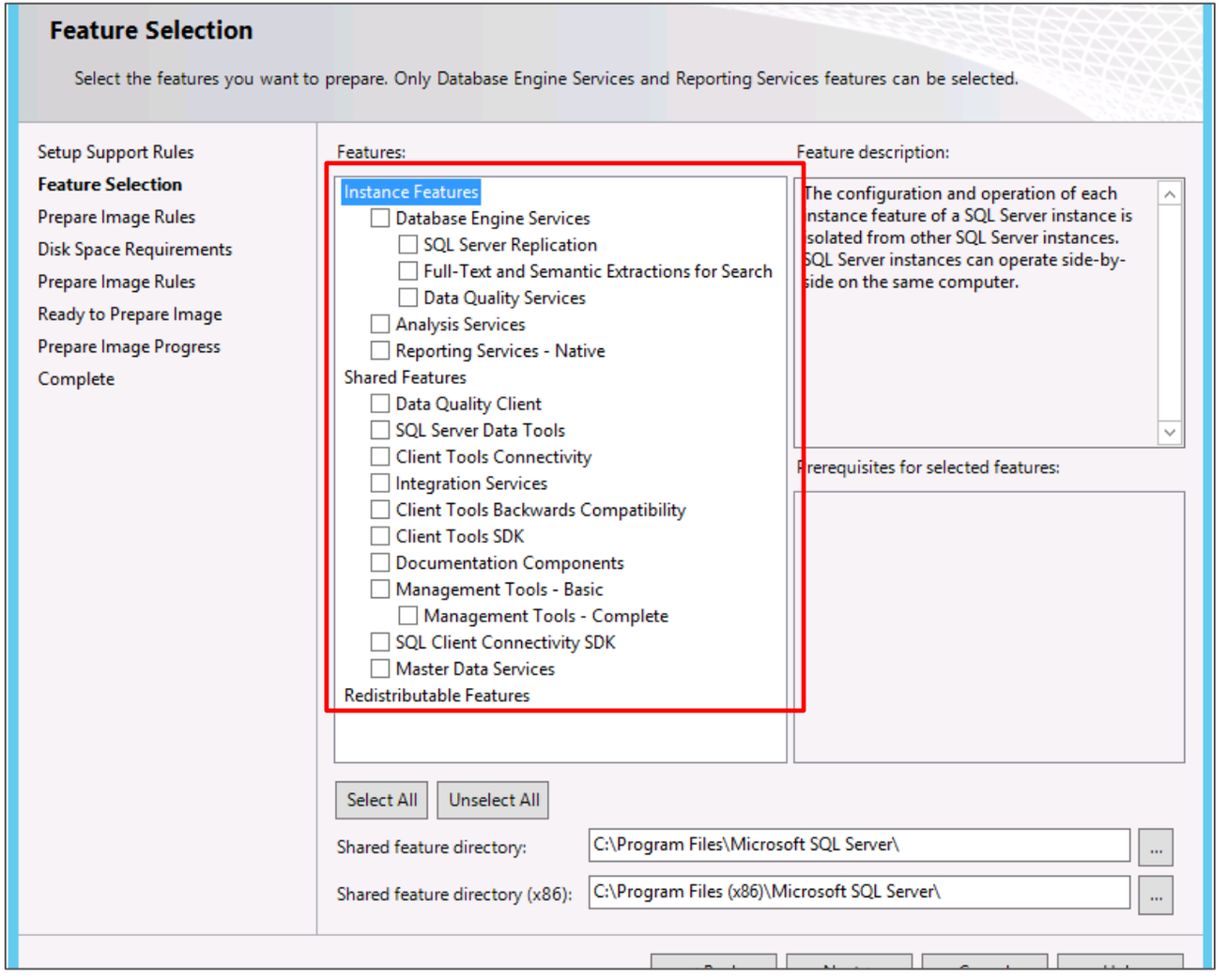
Remove limitations that currently exist

This has been long requested by customers



Sysprep enhancements

- Sysprep support for:
  - Database engine
  - Reporting Services
  - Analysis Services
  - Integration Services
  - Management Tools (SSMS)
  - Other shared features
  - Performance improvements
- Delivered in SQL Server 2012 SP1 CU2



Sysprep for SQL Server cluster

- What's being delivered
  - Extensions to SQL Server Sysprep functionality to support image-based deployment of clustered SQL Server instances
- Main benefit
  - Supports full automation of SQL Server Failover Cluster deployment scenarios
  - Reduces deployment times for SQL Server Failover Clusters
  - Combined together, these features enable customers to automate the provisioning of SQL Server Failover Clusters both on-premises and through IaaS
  - Built on top of SQL Server 2012 SP1 CU2 Sysprep enhancements

Microsoft

AlwaysOn Enhancements

Microsoft

AlwaysOn Enhancements

AlwaysOn in SQL Server 2014

- What's being delivered
  - Increase number of secondaries from four to eight
  - Increase availability of readable secondaries
  - Support for Windows Server 2012 CSV
  - Enhanced diagnostics
- Main benefits
  - Further scale out read workloads across (possibly geo-distributed) replicas
  - Use readable secondaries despite network failures (important in geo-distributed environments)
  - Improve SAN storage utilization
    - Avoid drive letter limitation (max 24 drives) via CSV paths
    - Increase resiliency of storage failover
  - Ease troubleshooting

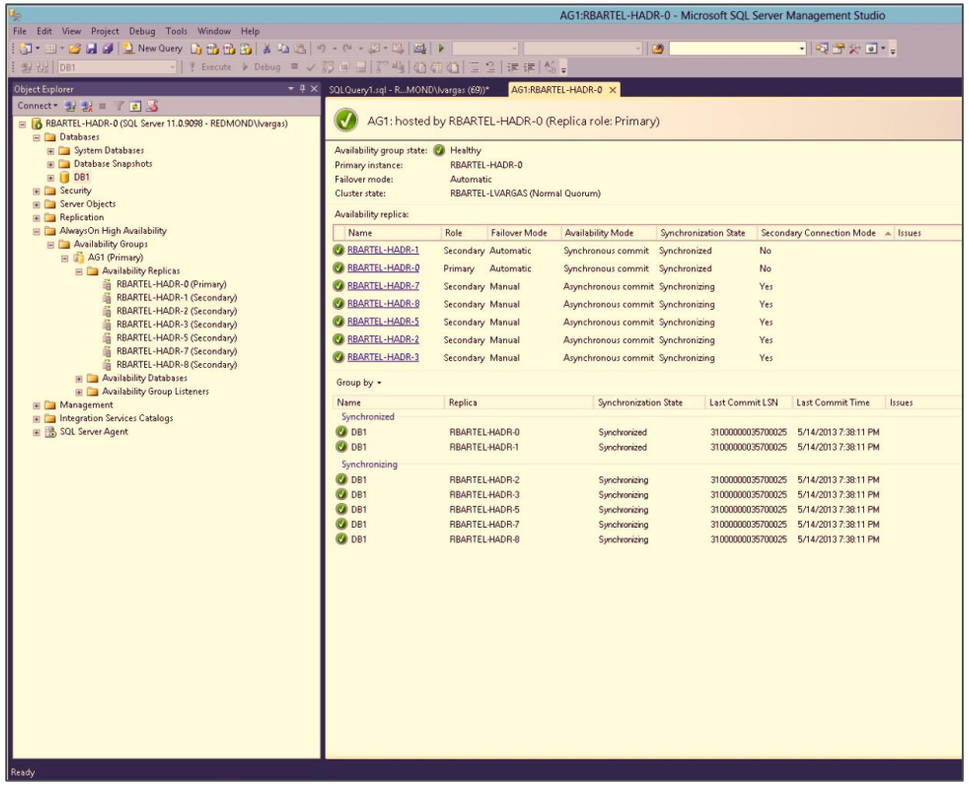
Increase number of Availability Group secondaries

Description

- Increase number of secondaries (4–8)
- Max number of sync secondaries is still two

Reason

- Customers want to use readable secondaries
  - One technology to configure and manage
  - Many times faster than replication
- Customers are asking for more database replicas (4–8)
  - To reduce query latency (large-scale environments)
  - To scale out read workloads



Support for Windows Server Cluster Shared Volumes

Description

Allow FCI customers to configure CSV paths for system and user databases

Reason

- Avoid drive letter limitation on SAN (max 24 drives)
- Improves SAN storage utilization and management
- Increased resiliency of storage failover (abstraction of temporary disk-level failures)
- Migration of SQL Server customers using PolyServe (to be discontinued in 2013)

Support for Windows Server Cluster Shared Volumes

Description

Allow FCI customers to configure CSV paths for system and user databases

Reason

- Avoid drive letter limitation on SAN (max 24 drives)
- Improves SAN storage utilization and management
- Increased resiliency of storage failover (abstraction of temporary disk-level failures)
- Migration of SQL Server customers using PolyServe (to be discontinued in 2013)

Microsoft

Managed Lock Priority

Microsoft

Managed Lock Priority

Manage Lock Priority

Blocking by online DDL operations	Concurrency and throughput affected
Partition SWITCH Short Sch-M lock on the source and target tables **Online Index Rebuild** Short table S and Sch-M lock	Blocking transactions need to be completed before DDL SWITCH/OIR will block new transactions Workload slow down or timeouts Impact to Tier1 mission-critical OLTP workloads

Managed Lock Priority Options

Kill all blockers	Switch to normal queue	Exit DDL after wait
Blocking user transactions killed Immediately or specified wait time MAX_DURATION* =n minutes]	Wait for blockers MAX_DURATION* Regular lock queue	Wait for blockers MAX_DURATION* Terminates DDL (SWITCH/OIR)

LOW PRIORITY LOCK QUEUE

*If no blockers, lock granted immediately and the DDL statement will complete successfully

Managed Lock Priority Details

PARTITION SWITCH

- Sch-M lock (source and destination)
- Blocking by user transactions
- Killed at source and destination tables

ONLINE INDEX REBUILD

- MAX_DURATION applies to every lock request
- Time reset for every S & Sch-M lock
- Only Sch-M lock conflict for read only workloads

Benefits

Managed by DBA for both partition switch and online index rebuild
Lock request placed in lower priority queue
Decision to wait or kill self or blockers
Executed immediately if no blockers

Managed Lock Priority Syntax

New clause in existing T-SQL DDL for ALTER TABLE and ALTER INDEX

<low_priority_lock_wait>::=
{
WAIT_AT_LOW_PRIORITY **(** MAX_DURATION = <time>[MINUTES],

ABORT_AFTER_WAIT = { NONE | SELF | BLOCKERS })
}

NONE - current behavior
SELF - abort DDL
BLOCKERS – abort user blockers

Syntax

ALTER TABLE stgtab SWITCH PARTITION 1 TO parttab PARTITION 1
WITH (WAIT_AT_LOW_PRIORITY (MAX_DURATION= 60 minutes,
ABORT_AFTER_WAIT=BLOCKERS))

ALTER INDEX clidx ON parttable REBUILD
WITH (ONLINE=ON **(WAIT_AT_LOW_PRIORITY (MAX_DURATION= 300,**
ABORT_AFTER_WAIT=SELF)))

Examples

Diagnostics

Errorlog

Abort session diagnostics

Deadlock diagnostics in deadlock graph

DMV extensions

sys.dm_tran_locks
"request_status" extensions

LOW_PRIORITY_CONVERT, LOW_PRIORITY_WAIT, or ABORT_BLOCKERS

sys.dm_os_wait_stats
"wait_type" extensions

...LOW_PRIORITY and ..ABORT_BLOCKERS

Extended Events

lock_request_priority_state

process_killed_by_abort_blockers

ddl_with_wait_at_low_priority

Microsoft

Single Partition Online Index Rebuild

Microsoft

Single Partition Online Index Rebuild

Single Partition Online Index Rebuild

Rebuild active partitions

Rebuild online - entire index for a partitioned table

Rebuild offline - a selected partition

- Table locked exclusively (with Sch-M lock) for the entire duration

Concurrency and throughput affected

Timeouts, Workload slow down affects availability

Heavy resource usage – CPU, Disk, Memory

Transaction log bloat

Impact to mission-critical workloads

Benefits

Granularity	• One or more partitions
Accessibility	• Table accessible for DML and query operations • Short term locks beginning and end of the index rebuild
Lock Priority	• Use Managed Lock Priority with SPOIR
Availability	• Reduced down time for mission critical workloads
Resource savings	• CPU, memory and disk space • Log space usage reduced

Syntax

New clause in existing T-SQL DDL for ALTER INDEX

```
<single_partition_rebuild_index_option> ::=
{
   ....  | ONLINE = { ON [ ( <low_priority_lock_wait> ) ] | OFF
}
 }
  <low_priority_lock_wait>::=
  {
    WAIT_AT_LOW_PRIORITY ( MAX_DURATION =
<time>[MINUTES],
    ABORT_AFTER_WAIT = { NONE | SELF | BLOCKERS } )
   }
```

Syntax

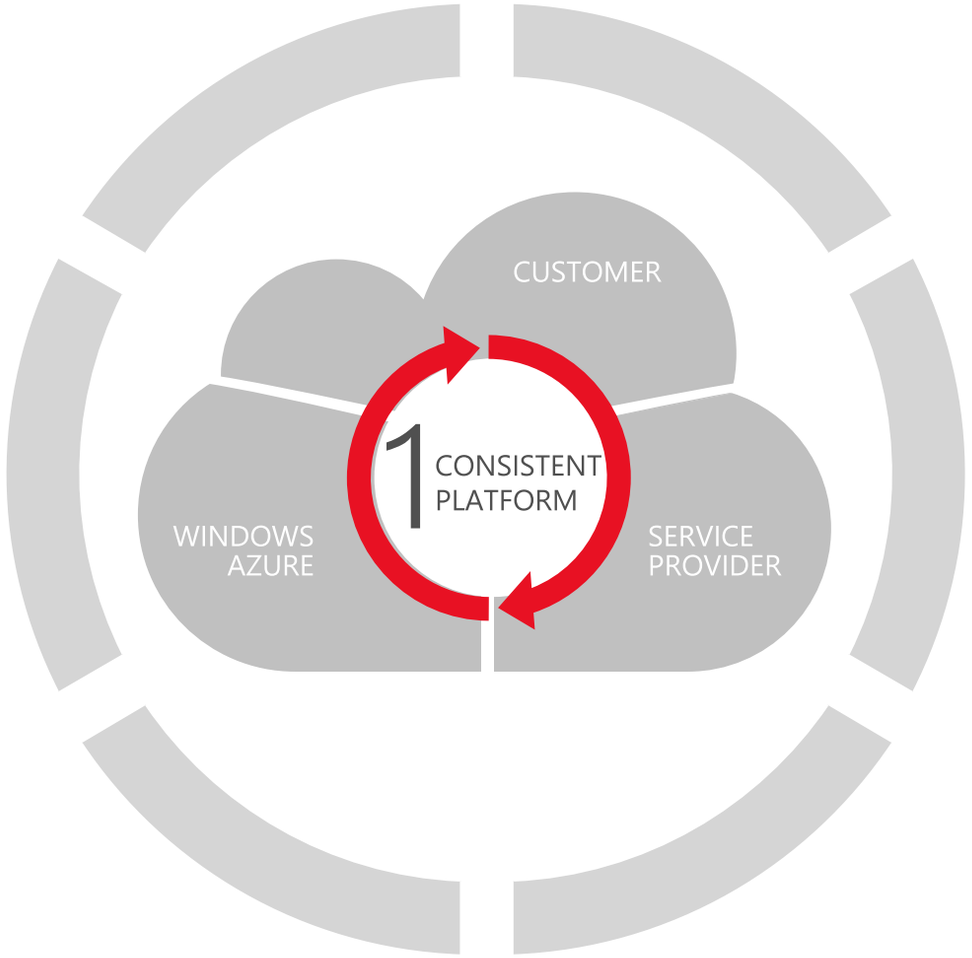
```
ALTER INDEX clidx ON part_table REBUILD
PARTITION= 3
      WITH ( ONLINE=ON
(WAIT_AT_LOW_PRIORITY
                  (MAX_DURATION= 300,
ABORT_AFTER_WAIT=NONE)) )
```

Example

Diagnostics

Error Message	Query Plan	Extended Event
Error 155 removed 'ONLINE' is now a recognized ALTER INDEX REBUILD PARTITION option	Shows partition info OIR for partition #**4** - OIR DDL plan shows -- Constant Scan(VALUES:((**4**)))	sqlserver.progress_report_online_index_operation Two new data fields partition_number: ordinary number of the partition being built partition_id : ID of the partition being built

Complete and consistent data platform



SQL Server 2014

Mission-critical performance

Faster insights from any data

Platform for hybrid cloud

Development | Management | Data | Identity | Virtualization

Call to action

Download SQL Server 2014 CTP2

www.microsoft.com/sqlserver