INTRODUCTION TO COLUMNSTORE INDEXES

Patrick Keisler

Microsoft Corporation

PASS SQL saturday

#330 | CHARLOTTE, BI EDITION 2014

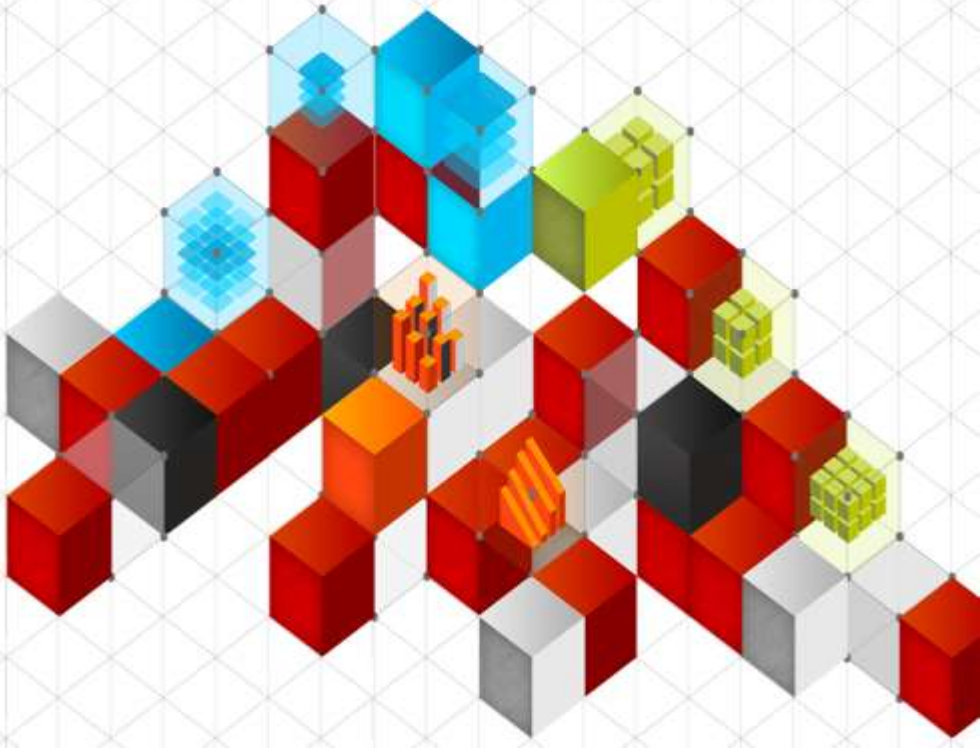
Track 3 – Room 3315
CPCC Levine Campus
12:45PM – 1:45PM

Microsoft

SQL Server

2014

Business Intelligence



ABOUT THE SPEAKER

Patrick Keisler

- Senior Premier Field Engineer
- Over 15 years of database experience
- Specialize in administration of high availability and disaster recovery environments

Email: pkeisler@microsoft.com
Blog: www.patrickkeisler.com
Twitter: @patrickkeisler

Microsoft

Microsoft CERTIFIED
IT Professional
Database Administrator 2008
Database Developer 2008

SESSION AGENDA

- Architecture
- 2012 vs 2014 Features
- Limitations
- Resources

Microsoft

WHAT IS A COLUMNSTORE INDEX?

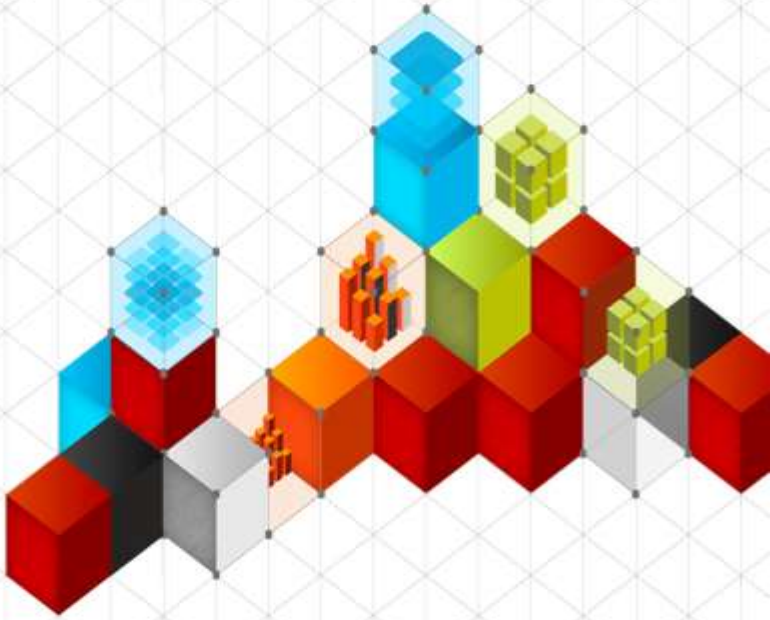
- New type of index for blazing fast query processing
- Intended for use by data warehouses
  - Star/snowflake schemas
  - Aggregations over large amounts of data
  - Assumes data is mostly read-only
- Can speed up query performance by 10x - 100x

Microsoft

WHAT'S THE SECRET SAUCE?

- Stores data in column-wise fashion instead of row-wise
- Uses a new type of compression algorithm
- Batch mode processing reduces CPU
- Segment elimination

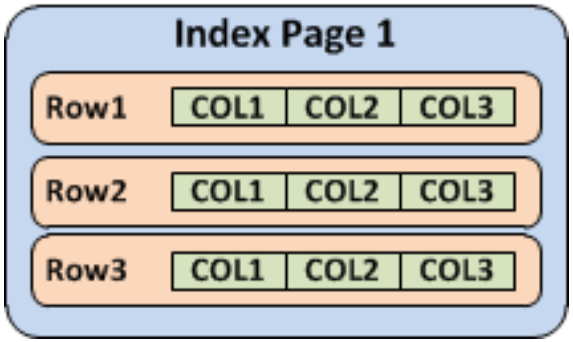
DEMO 1



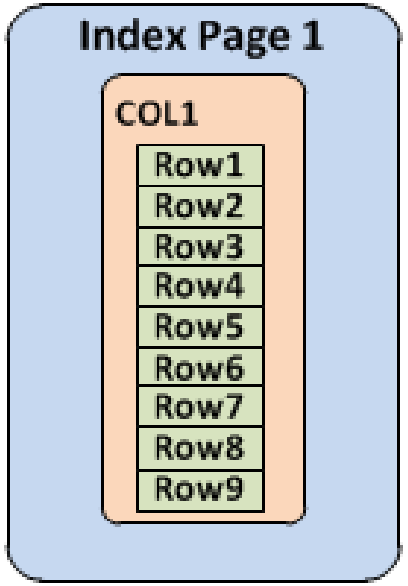
Microsoft

ROWSTORE VS COLUMNSTORE

- Rowstore
  - Stores data from all columns for each row together on a page



- Columnstore
  - Stores data from all rows for each column together on a page

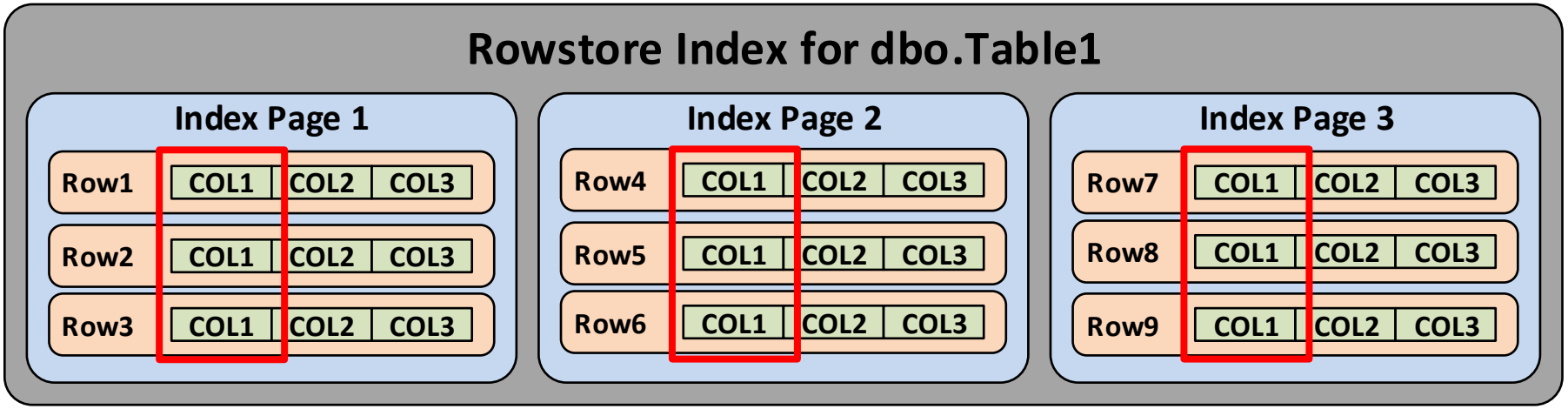


Microsoft

ROWSTORE INDEX

```
CREATE TABLE dbo.Table1 (COL1 CHAR(10) ,COL2 CHAR(10) ,COL3 CHAR(10));
GO

INSERT dbo.Table1 VALUES
   ('Patrick','ENGINEER','NC')
  ,('David','LEAD','NC')
  ,('Chris','DBA','NC')
  ,('Oscar','DBA','NC')
  ,('Larry','DBA','LV')
  ,('Sree','DBA','NY')
  ,('Krishna','DBA','CA')
  ,('Ryan','DBA','NC')
  ,('Chad','DBA','NC');
GO

CREATE NONCLUSTERED INDEX Table1_nci ON dbo.Table1(COL1,COL2,COL3);
GO
```

ROWSTORE INDEX



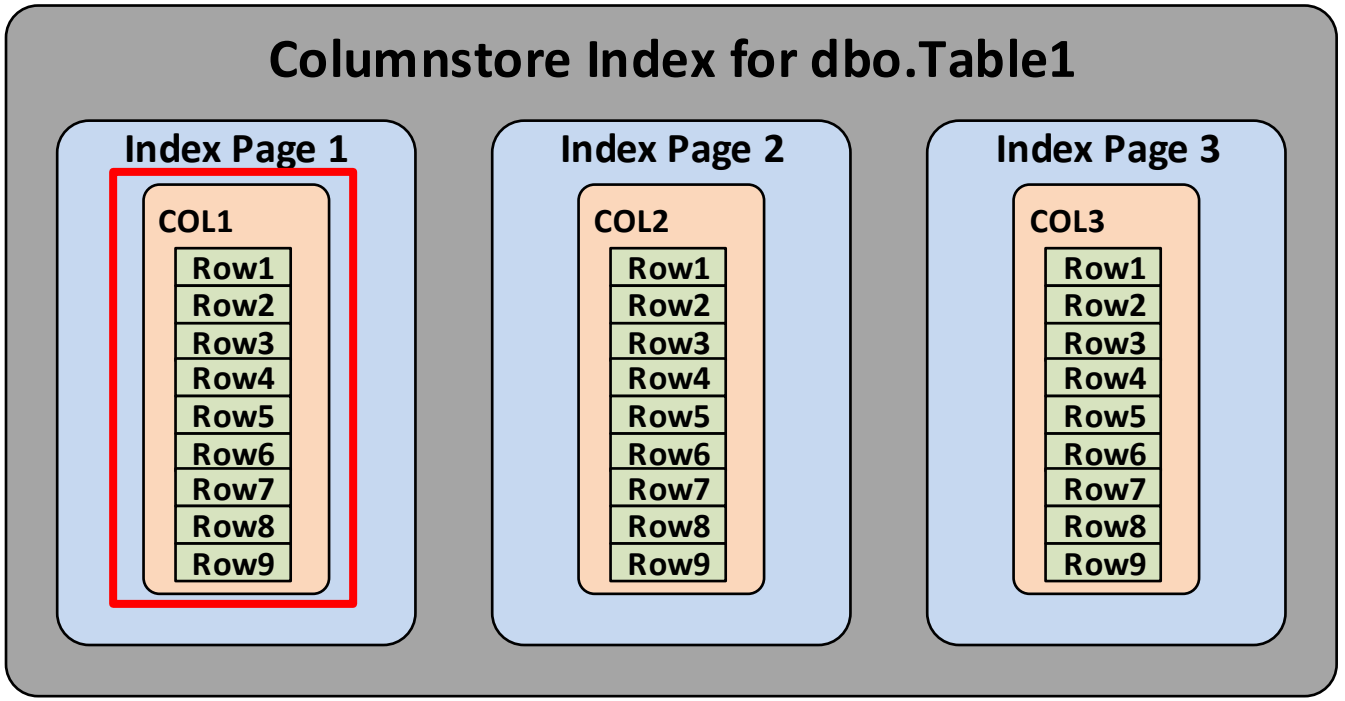

```
SELECT COL1 FROM dbo.Table1;
GO

I/O Statistics: logical reads = 3
```

Microsoft

COLUMNSTORE INDEX

```
CREATE TABLE dbo.Table1 (COL1 CHAR(10) ,COL2 CHAR(10) ,COL3 CHAR(10));
GO

INSERT dbo.Table1 VALUES
   ('Patrick','ENGINEER','NC')
  ,('David','LEAD','NC')
  ,('Chris','DBA','NC')
  ,('Oscar','DBA','NC')
  ,('Larry','DBA','LV')
  ,('Sree','DBA','NY')
  ,('Krishna','DBA','CA')
  ,('Ryan','DBA','NC')
  ,('Chad','DBA','NC');
GO

CREATE NONCLUSTERED COLUMNSTORE INDEX Table1_csi
  ON dbo.Table1(COL1,COL2,COL3);
GO
```

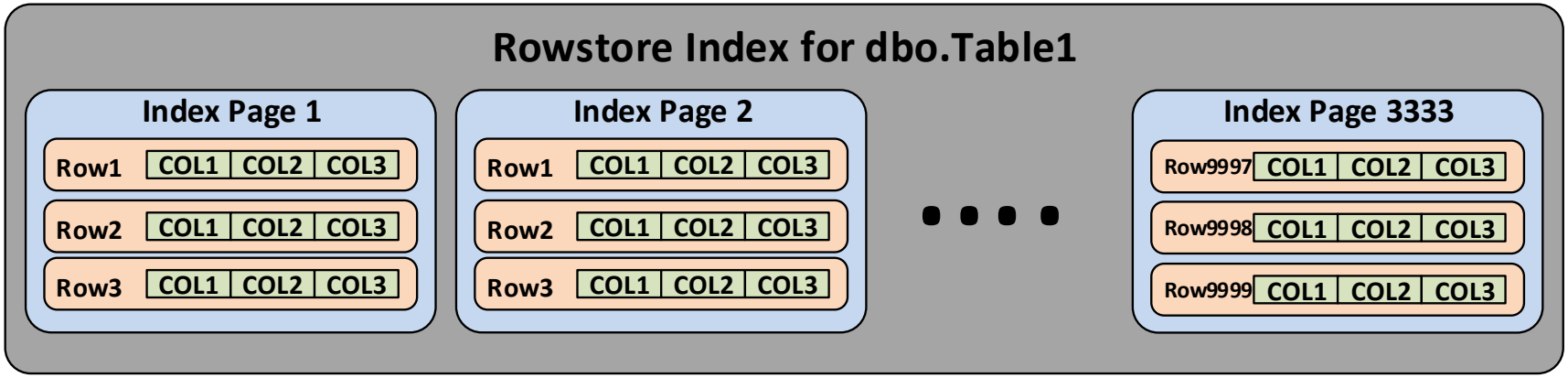
COLUMNSTORE INDEX



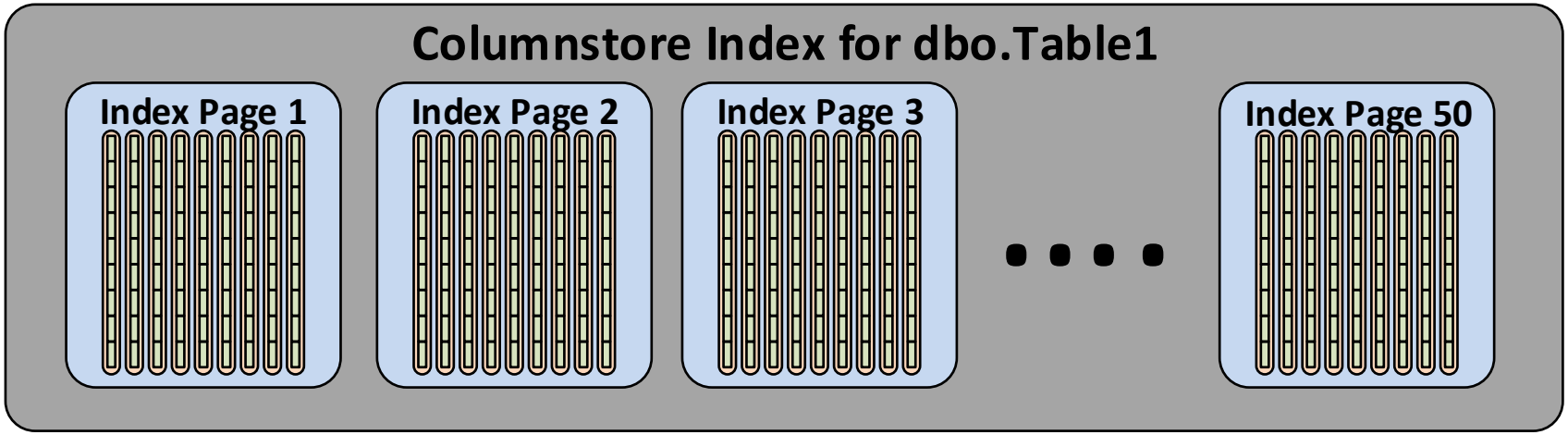

```
SELECT COL1 FROM dbo.Table1;
GO

I/O Statistics: logical reads = 1
```

Microsoft

COLUMNSTORE COMPRESSION



I/O Statistics: **logical reads ~ 3333**



I/O Statistics: **logical reads ~ 50**

Microsoft

COLUMNSTORE STORAGE & COMPRESSION

```
CREATE NONCLUSTERED COLUMNSTORE INDEX Table1_csi
  ON dbo.Table1 (COL1,COL2,COL3);
GO
```

Col1	Col2	Col3
Chad	DBA	NC
Chris	DBA	NC
David	LEAD	NC
Krishna	DBA	CA
Larry	DBA	LV
Oscar	DBA	NC
Patrick	ENGINEER	NC
Ryan	DBA	NC
Sree	DBA	NY

COLUMNSTORE STORAGE & COMPRESSION

- Horizontally partition into 1 million row “Row Groups”

Col1	Col2	Col3
Chad	DBA	NC
Chris	DBA	NC
David	LEAD	NC
Krishna	DBA	CA
Larry	DBA	LV

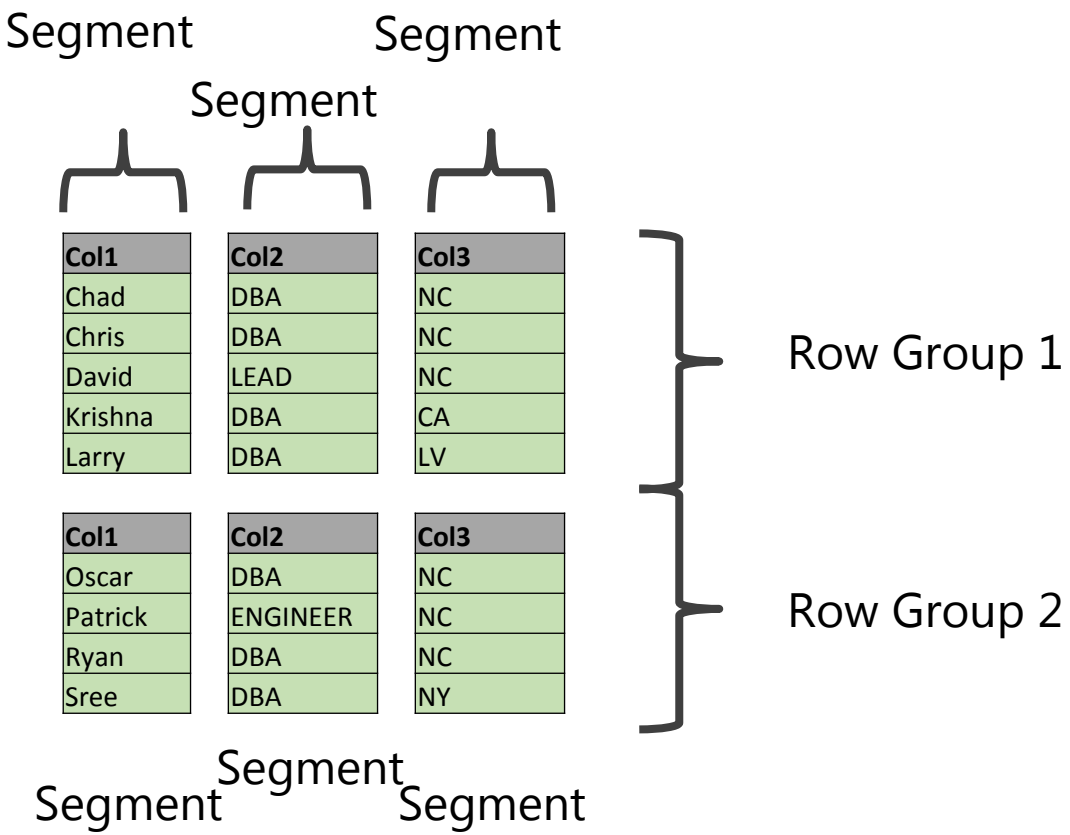
Row Group 1

Col1	Col2	Col3
Oscar	DBA	NC
Patrick	ENGINEER	NC
Ryan	DBA	NC
Sree	DBA	NY

Row Group 2

COLUMNSTORE STORAGE & COMPRESSION

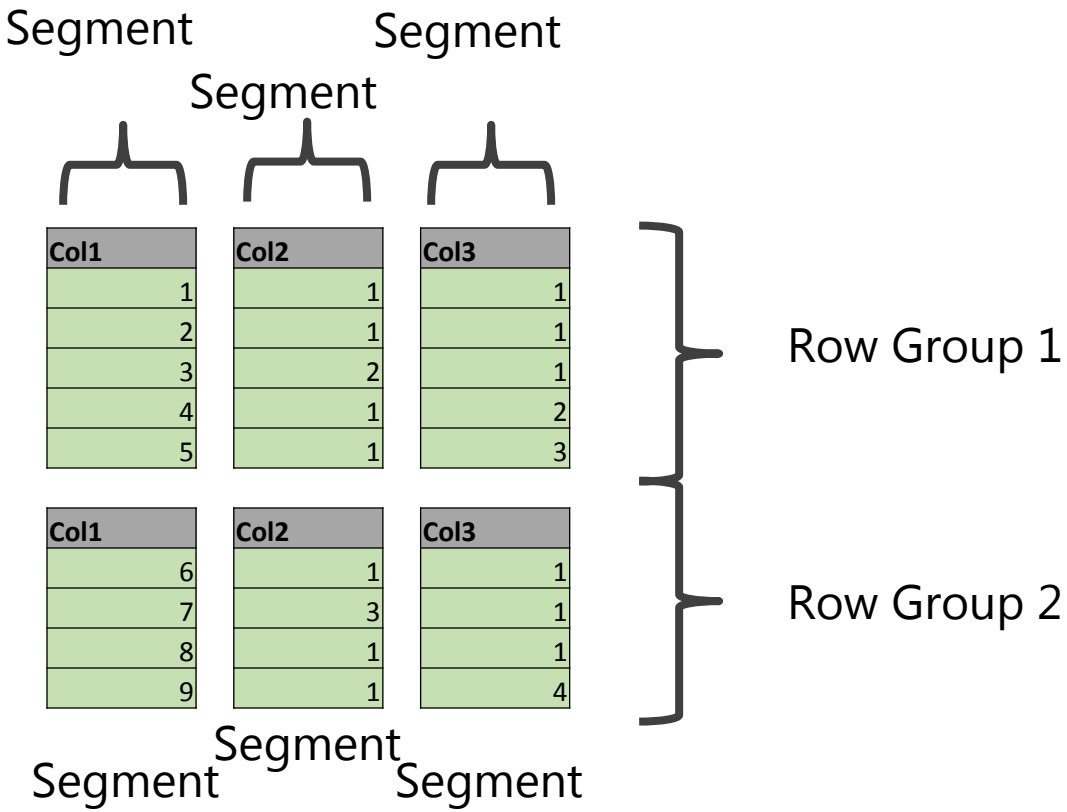
- Vertically partition into segments



Microsoft

COLUMNSTORE STORAGE & COMPRESSION

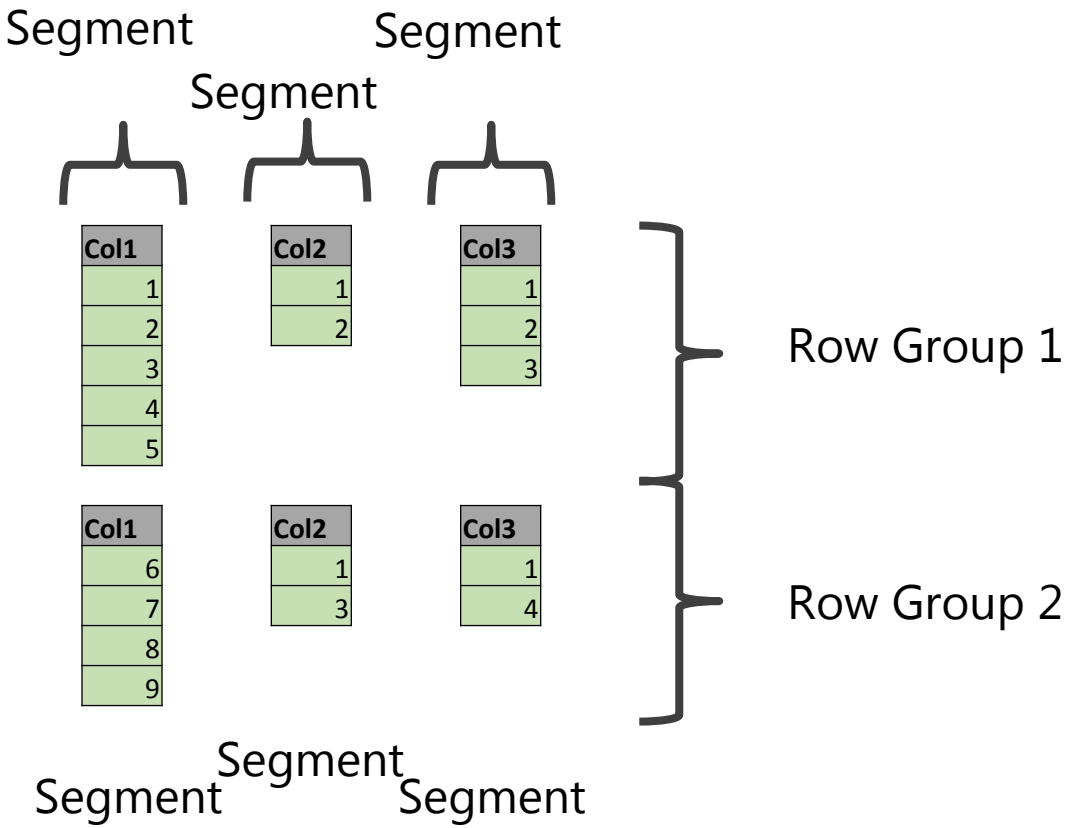
- Encode the data values
  - Dictionary based
  - Value based



Microsoft

COLUMNSTORE STORAGE & COMPRESSION

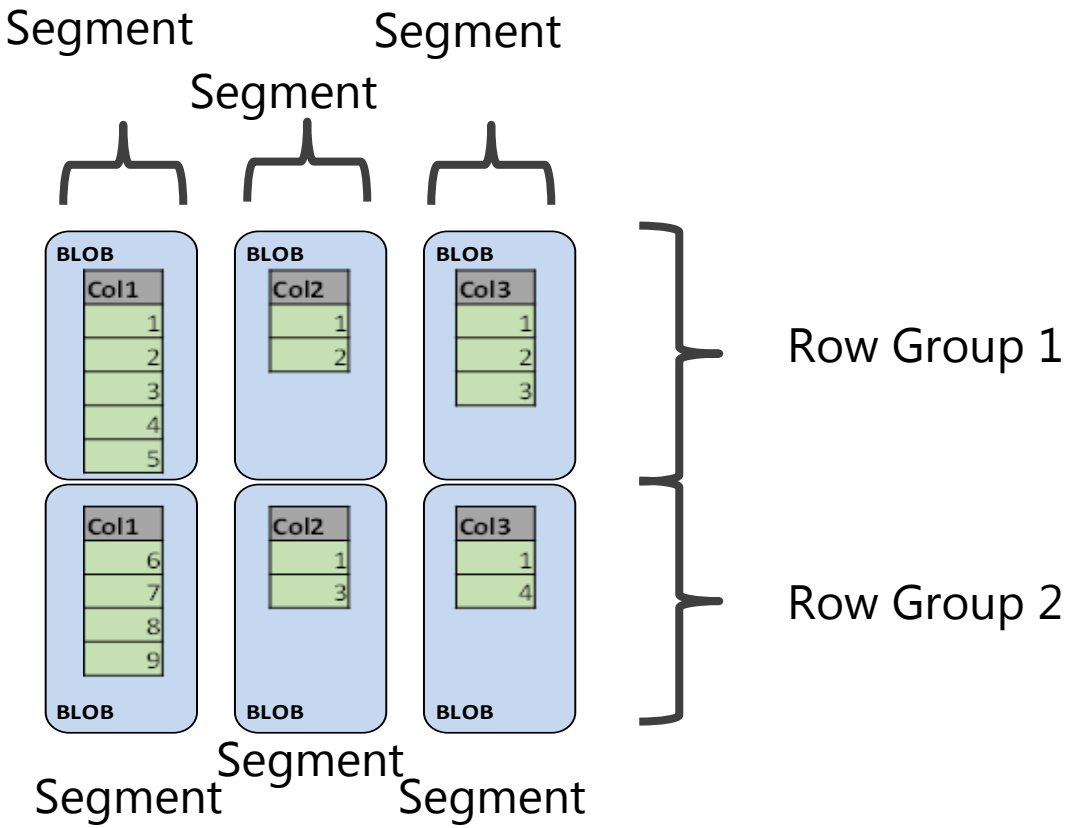
- Compress the data using RLE



Microsoft

COLUMNSTORE STORAGE & COMPRESSION

- Segments are stored as BLOBs on disk



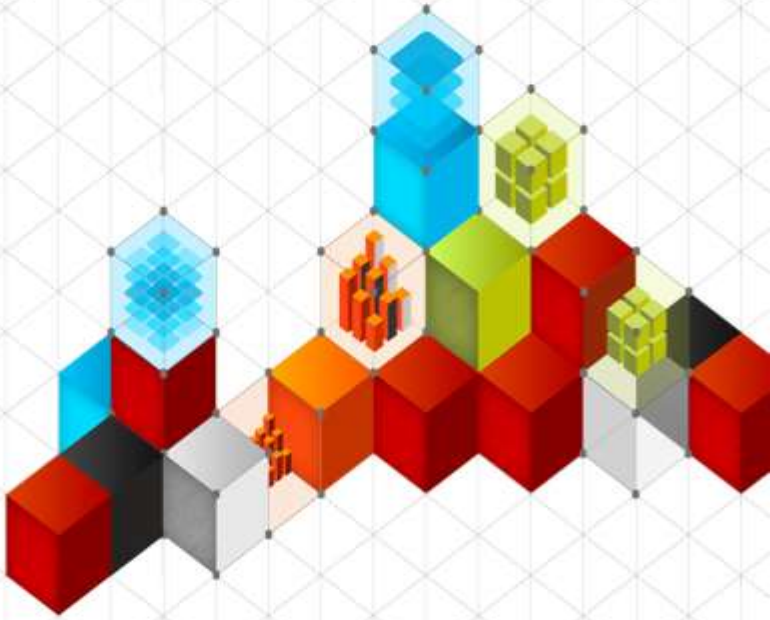
Microsoft

BATCH MODE PROCESSING

- New execution mode for certain operators
  - Batch Mode vs Row Mode
- Processes ~1000 rows at a time
- Faster processing of queries
- Lower CPU utilization

BATCH MODE PROCESSING

DEMO 2



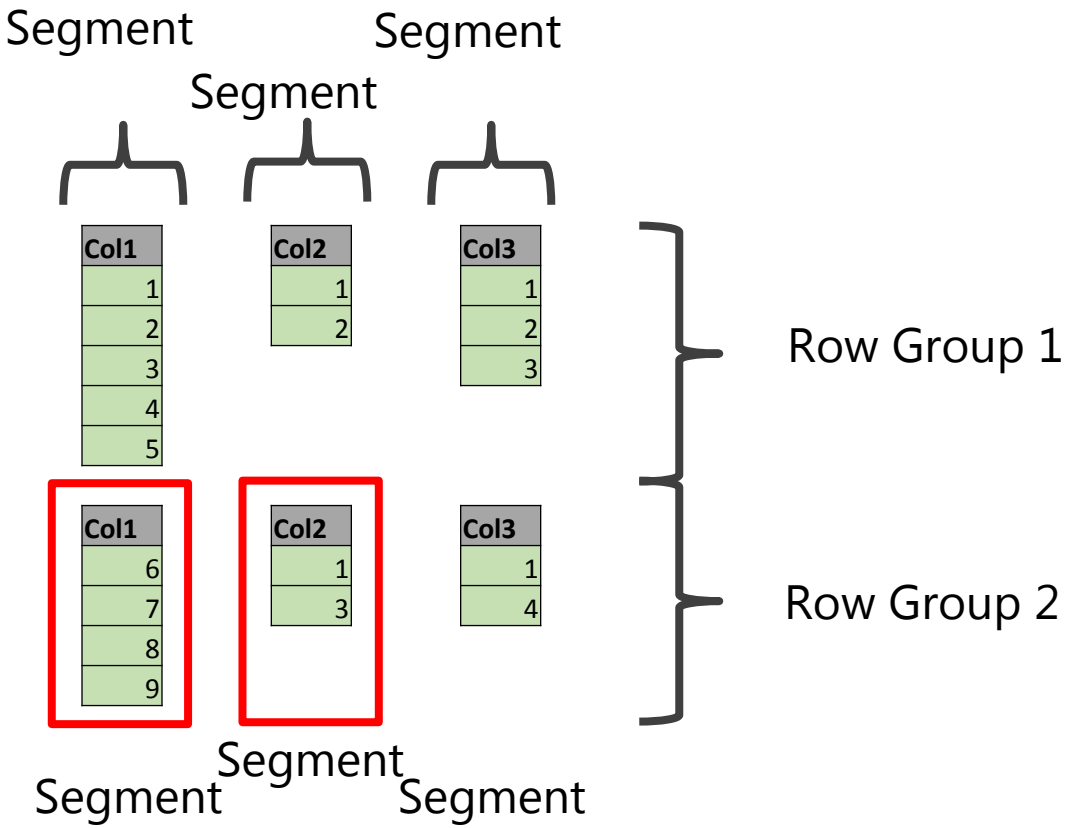
Microsoft

SEGMENT ELIMINATION

- Uses internal metadata to exclude segments based on the predicate
- Reduces I/O
- Same principle as partition elimination

SEGMENT ELIMINATION

```
SELECT COL1 FROM dbo.Table1
WHERE COL2 = 'ENGINEER';
GO
```



Microsoft

SQL SERVER 2012 NON-CLUSTERED COLUMNSTORE INDEX LIMITATIONS

- READ-ONLY
  - Applies to the Columnstore index as well as the base table
- The index cannot be altered; only dropped & recreated
- Certain data types cannot be used
- Columnstore indexes are stored in memory, but could be paged out under memory pressure
- No more than 1024 columns
- Cannot act as primary or foreign key
- Cannot include sparse or filestream data
- Cannot be combined with replication, change tracking, or change data capture
- Rewrite your queries to take advantage of batch mode
- Enterprise Edition only

SQL SERVER 2012 NON-CLUSTERED COLUMNSTORE INDEX LIMITATIONS

Prohibited data types:

- Decimal with precision greater than 18 digits
- binary and varbinary
- ntext , text, and image
- varchar(max) and nvarchar(max)
- uniqueidentifier
- rowversion (and timestamp)
- sql_variant
- datetimeoffset with scale greater than 2
- CLR types (hierarchyid and spatial types)
- xml

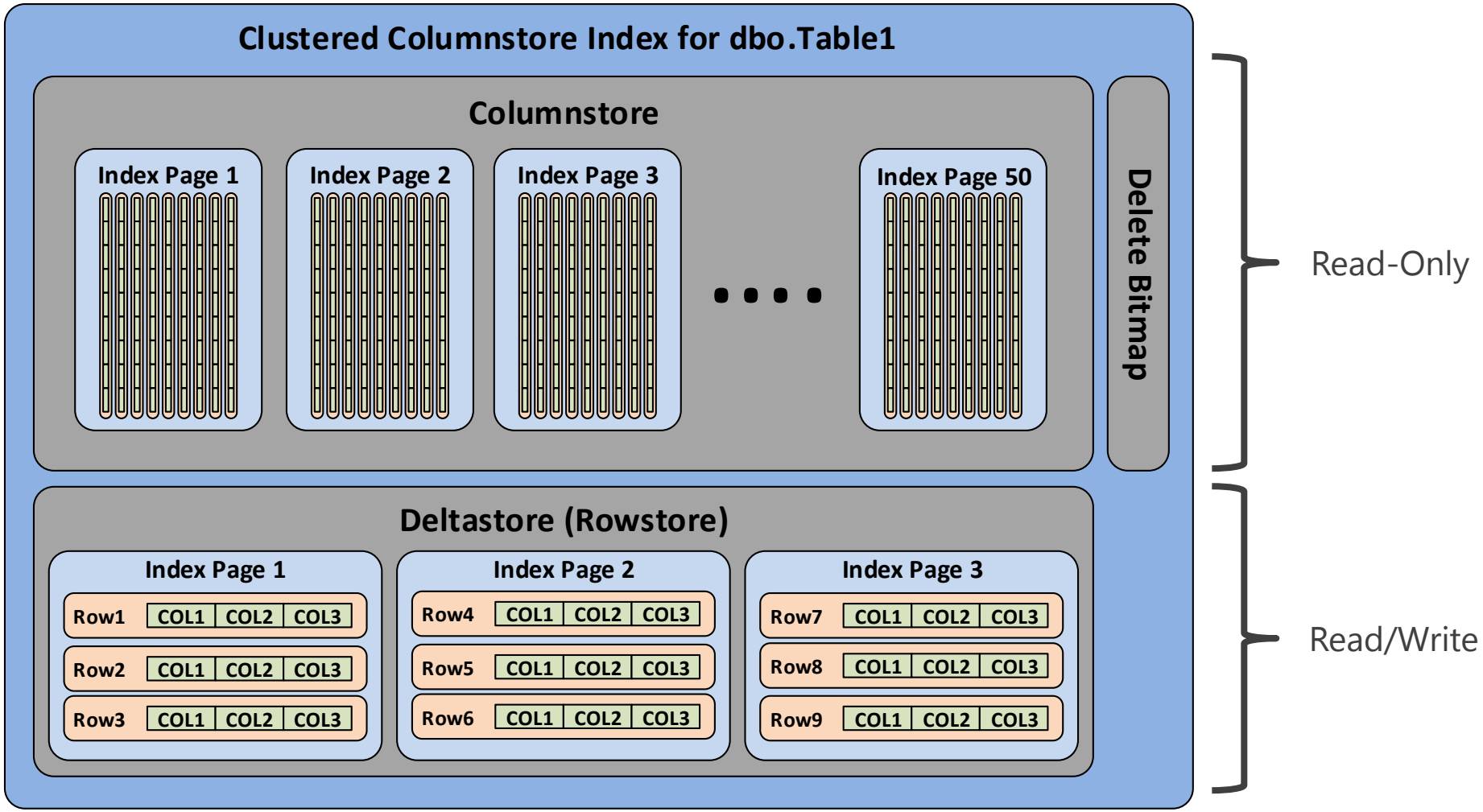
SQL SERVER 2014 IMPROVEMENTS

- *Writeable* clustered Columnstore indexes
- More operators can use Batch Mode processing
- Most data type restrictions are gone
- Archival compression

CLUSTERED COLUMNSTORE INDEX

```
CREATE TABLE dbo.Table1 (COL1 CHAR(10) ,COL2 CHAR(10) ,COL3 CHAR(10));
GO

INSERT dbo.Table1 VALUES
   ('Patrick','ENGINEER','NC')
  ,('David','LEAD','NC')
  ,('Chris','DBA','NC')
  ,('Oscar','DBA','NC')
  ,('Larry','DBA','LV')
  ,('Sree','DBA','NY')
  ,('Krishna','DBA','CA')
  ,('Ryan','DBA','NC')
  ,('Chad','DBA','NC');
GO

CREATE CLUSTERED COLUMNSTORE INDEX Table1_csi ON dbo.Table1;
GO
```

CLUSTERED COLUMNSTORE INDEX



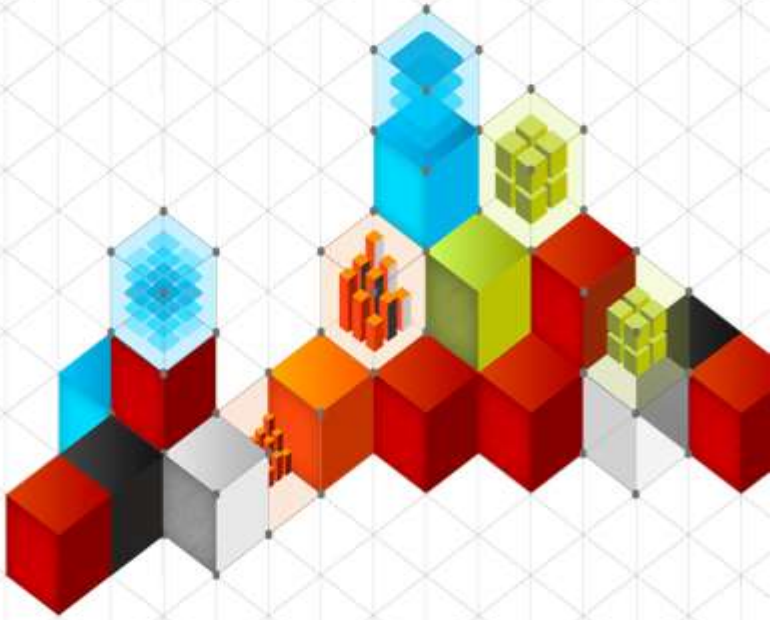
Microsoft

HOW DOES A CLUSTERED COLUMNSTORE INDEX WORK?

- Inserts (new data) are added to the Deltastore
  - Once it reaches about ~1 million rows, it's compressed into a new columnstore row group
  - However, batches of 100k+ rows will generate a new row group
  - Ideal insert size = < 100k or > 1 million
  - Multiple BULK INSERTs will each create their own Deltastore
- Deletes are only marked as "deleted"
  - Similar to Ghost Records in the Row Store
  - "deleted" rows will actually be deleted once the index is rebuilt
- Updates are treated as a delete + insert
  - Updated rows are marked as "deleted"
  - "updated" values are added to the Deltastore
- For both Deleted & Updates
  - Monitor the deleted_rows value in sys.column_store_row_groups

CLUSTERED COLUMNSTORE INDEX

DEMO 3



Microsoft

COLUMNSTORE ARCHIVAL COMPRESSION

- Compresses the columnstore data even more
  - ~25% more
- Can be used on both
  - Clustered Columnstore Indexes
  - Non-clustered Columnstore Indexes
- Compressed at the partition level

```
ALTER TABLE dbo.Table1 REBUILD
  WITH (DATA_COMPRESSION = COLUMNSTORE_ARCHIVE);
GO
```

SQL SERVER 2014 NON-CLUSTERED COLUMNSTORE LIMITATIONS

- READ-ONLY
  - Applies to the Columnstore index as well as the base table
- The index cannot be altered; only dropped & recreated
- Certain data types cannot be used
- Columnstore indexes are stored in memory, but could be paged out under memory pressure
- No more than 1024 columns
- Cannot act as primary or foreign key
- Cannot include sparse or filestream data
- Cannot be combined with replication, change tracking, or change data capture
- ~~Rewrite your queries to take advantage of batch mode~~
- Enterprise Edition only

SQL SERVER 2014 NON-CLUSTERED COLUMNSTORE INDEX LIMITATIONS

Prohibited data types:

- ~~Decimal with precision greater than 18 digits~~
- ~~binary and varbinary~~
- ntext , text, and image
- varchar(max) and nvarchar(max)
- ~~uniqueidentifier~~
- rowversion (and timestamp)
- sql_variant
- ~~datetimeoffset with scale greater than 2~~
- CLR types (hierarchyid and spatial types)
- xml

Microsoft

SQL SERVER 2014 CLUSTERED COLUMNSTORE INDEX LIMITATIONS

- It is the only index on the table
  - No constraints allowed
  - PK, FK, DEFAULT, UNIQUE, CHECK
- If needed, then use non-clustered Columnstore index
- Deleted record counts
- The index cannot be altered; only dropped & recreated
- Certain data types cannot be used
- Columnstore indexes are stored in memory, but could be paged out under memory pressure
- No more than 1024 columns
- Cannot include sparse or filestream data
- Cannot be combined with replication, change tracking, or change data capture
- Enterprise Edition only

2012 & 2014 COLUMNSTORE INDEX CAVEATS

Issue:

- Non-clustered Columnstore Index Read Only

Workarounds:

- Disable the index, change the data, rebuild the index
- Trickle loading
  - Union between a columnstore table and a rowstore table
  - During a nightly batch, move the data from the rowstore table into the columnmstore table
- Use table partitioning
  - Switch partitions in/out

2012 & 2014 COLUMNSTORE INDEX CAVEATS

Issue:

- Creating a columnstore index is very memory intensive

```
Msg 8658, Level 17, State 1, Line 2
Cannot start the columnstore index build because it requires at least 345520
KB, while the maximum memory grant is limited to 84008 KB per query in
workload group 'default' (2) and resource pool 'default' (2).
```

Workarounds:

- Use MAXDOP hint to reduce the memory grant requirement

```
CREATE NONCLUSTERED COLUMNSTORE INDEX Table1_csi
  ON dbo.Table1(Col1,Col2,Col3) WITH (MAXDOP = 1);
GO
```

- Change the default workload group

```
ALTER WORKLOAD GROUP [default] WITH (request_max_memory_grant_percent = 50);
GO
ALTER RESOURCE GOVERNOR RECONFIGURE;
GO
```

2012 & 2014 COLUMNSTORE INDEX CAVEATS

Issue:

- Using non-clustered Columnstore index could be slower than B-Tree index

Workaround:

- Use the follow query hint

```
OPTION (IGNORE_NONCLUSTERED_COLUMNSTORE_INDEX
```

Microsoft

2012 & 2014 COLUMNSTORE INDEX CAVEATS

Issue:

- Table contains an unsupported data type

Workarounds:

- Redesign the table to use a clustered Columnstore index

- Create a non-clustered Columnstore index and exclude those columns from the index definition

Microsoft

CATALOG VIEWS & DMOS

SQL Server 2012 & 2014:

- sys.column_store_segments
- sys.column_store_dictionaries
- sys.dm_db_index_usage_stats
- sys.dm_db_index_operational_stats

SQL Server 2014 (only):

- sys.column_store_row_groups

OTHER RESOURCES

Technet WIKI:

- http://social.technet.microsoft.com/wiki/contents/articles/3540.sql-server-columnstore-index-faq.aspx
- http://social.technet.microsoft.com/wiki/contents/articles/4995.sql-server-columnstore-performance-tuning.aspx

Books Online:

- http://msdn.microsoft.com/en-us/library/gg492088(v=SQL.110).aspx
- http://technet.microsoft.com/en-us/library/gg492088(v=sql.120).aspx

Jimmy May (SQL CAT Team) :

- http://sqlblog.com/blogs/jimmy_may/archive/2014/03/24/sql-server-2014-columnstore-indexes-the-big-deck.aspx

Questions?

RELATED MICROSOFT TEAM SESSIONS

Title	Speaker	Level	Time	Room
Your Best Interview Ever	Chris Skorlinski	Beginner	9:15 AM	Track 7 Room 3314
Introduction to SQL Server 2014 AlwaysOn Solutions	Uttam Parui	Intermediate	9:15 AM	Track 2 Room 2300
Introduction to Columnstore Indexes	Patrick Keisler	Beginner	12:45 AM	Track 3 Room 3315
Introduction to Azure Document DB	Dan Shaver	Beginner	12:45 PM	Track 6 Room 2316
SQL Server Data Tools	Prabhaker Potharaju	Beginner	12:45 PM	Track 2 Room 2300
All Your HBase Are Belong To Us- an intro to HBase	Jason Howell	Beginner	2 PM	Track 3 Room 3315
From Zero to Hero - Designing and Deploying Dynamic Reports Using SSRS	Josh Gerszewski	Beginner	2 PM	Track 5 Room 2302
Micro Management at its Best (Microsoft Management That Is!)	Matt Papes	Non-Technical	3:15 PM	Track 7 Room 3314
PowerBI	Scott Holmes	Beginner	3:15 PM	Track 5 Room 2302
Introduction to Microsoft Azure Machine Learning	Bill Carroll	Beginner	3:15 PM	Track 6 Room 2316

PLEASE EVALUATE THIS SESSION

PASS SQL saturday

#330 | CHARLOTTE, BI EDITION 2014

There will be a raffle drawing held at the end of the day.

Microsoft

INTRODUCTION TO COLUMNSTORE INDEXES

Patrick Keisler

Microsoft Corporation

PASS SQL saturday

#330 | CHARLOTTE, BI EDITION 2014

Track 3 – Room 3315
CPCC Levine Campus
12:45PM – 1:45PM

Microsoft

SQL Server

2014

Business Intelligence