Aas Trailblazers

Unleashing insights

Index Options

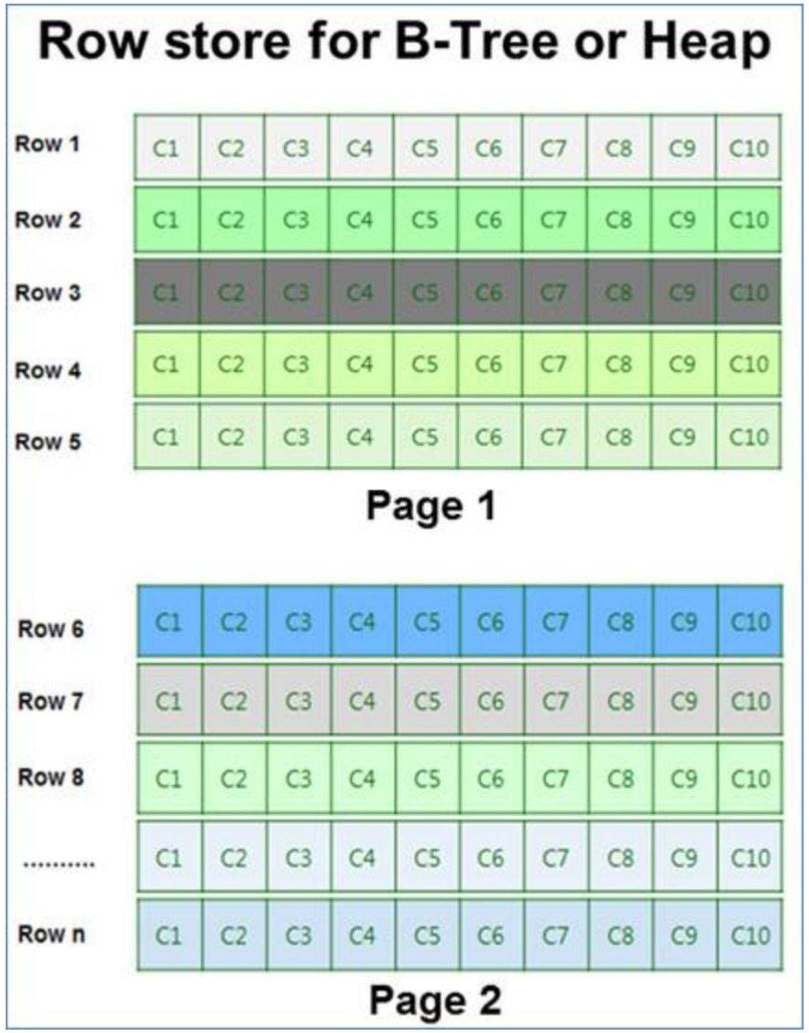
Row Store

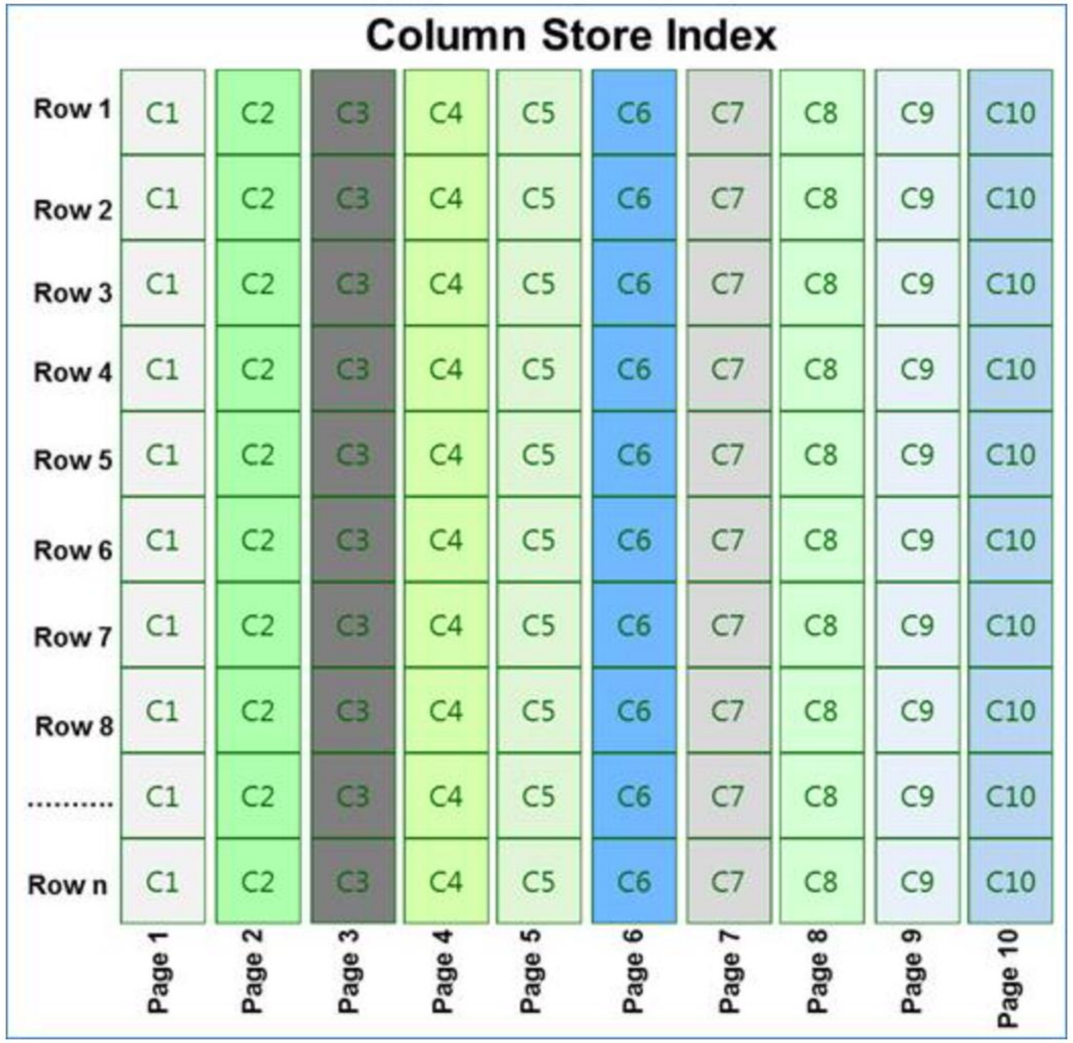
- Heap
- Clustered Index
- Non-Clustered Index

Clustered Columnstore Index

- Default if not defined in CREATE TABLE

Row store vs. Column store





https://www.databasejournal.com/features/mssql/understanding-new-column-store-index-of-sql-server-2012.html

Row store: Heaps

- Just a collection of pages
  - Append only
  - No ordering
  - Minimal fragmentation
- Fast target for loading
- Much slower when used for small range scans & single selects

Understanding Heaps

Deletes are not physically removed from a heap until it is rebuilt. As data is added SQL allocates more pages to the table.

Row store: Indexes (clustered and non-clustered)

- Use a Balanced Tree (b-tree) to organize data
- Get fragmented over time
    - Require more maintenance
- Leaf level
    - Clustered index: data
    - Non-clustered index: Row ID (RID) – may require lookup of data
- Ideal for limited range scans & singleton selects
- Slower for table scans / partition scans / loading

Columnstore: Index

Group rows into batches of ~1M rows

- This is called a row group

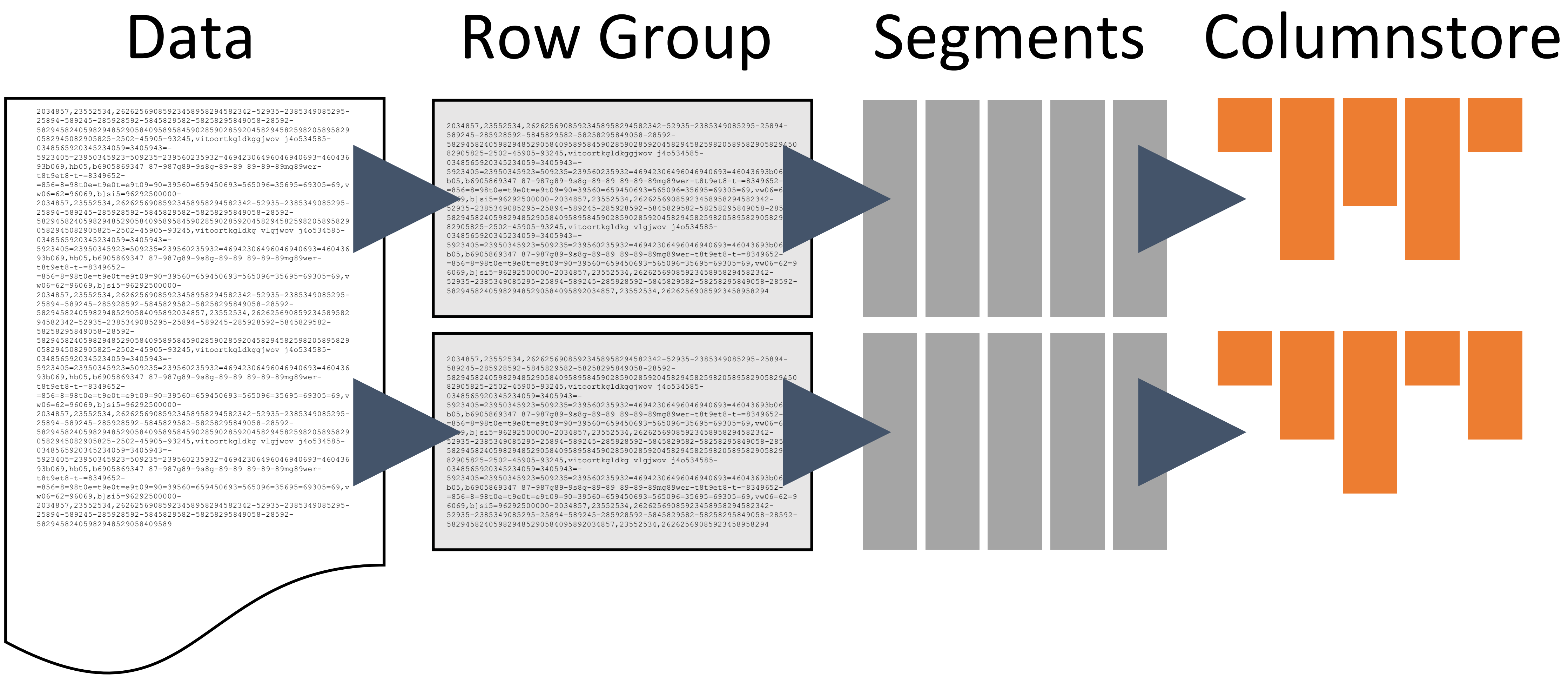
Compress/Encode each column

- Lower selectivity compresses more efficiently
- Primitive data types (int, date, short string, etc.) compresses efficiently
- Long string fields do NOT compress well

Typically, up to 10x compression and query performance up to 10x*

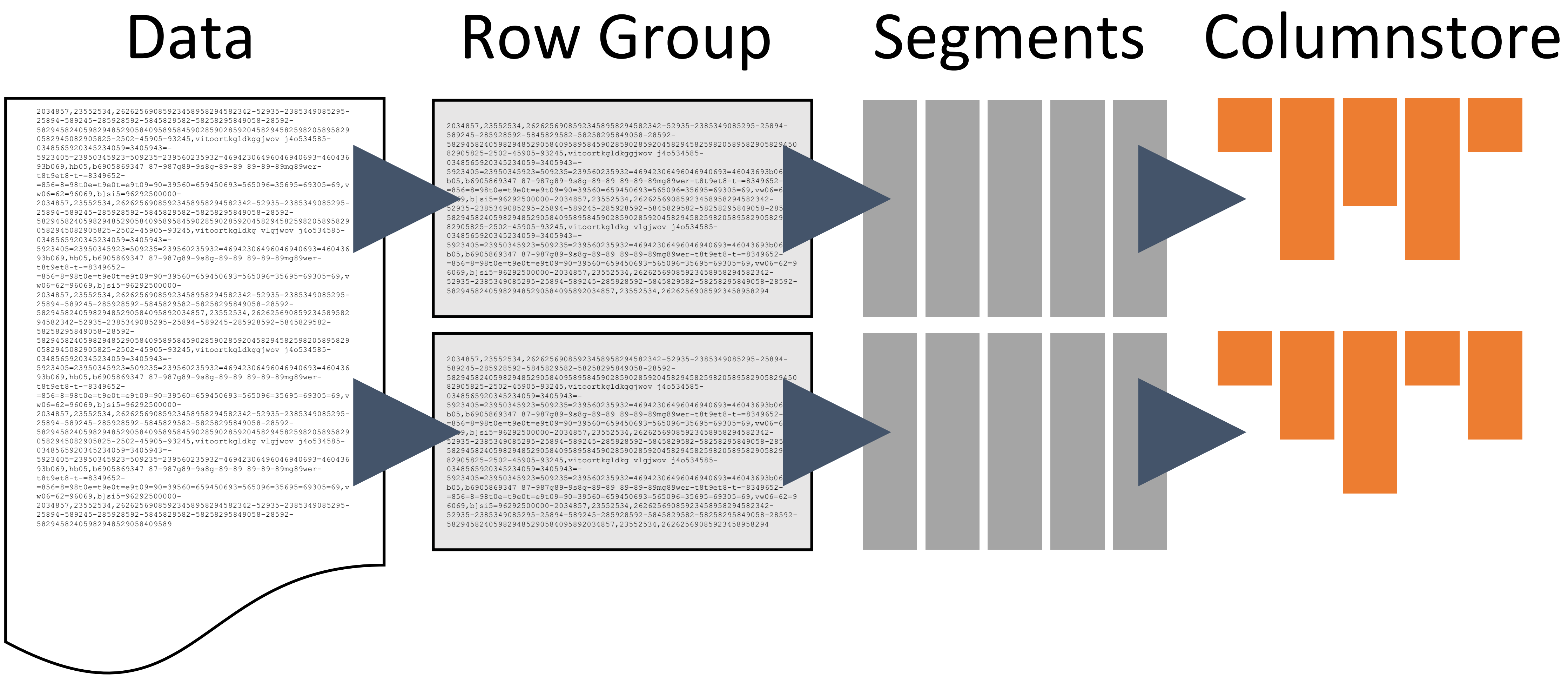
* https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-overview?view=sql-server-ver15

Columnstore: Design and storage



Demo

Columnstore: Design and storage



Columnstore: Row Groups

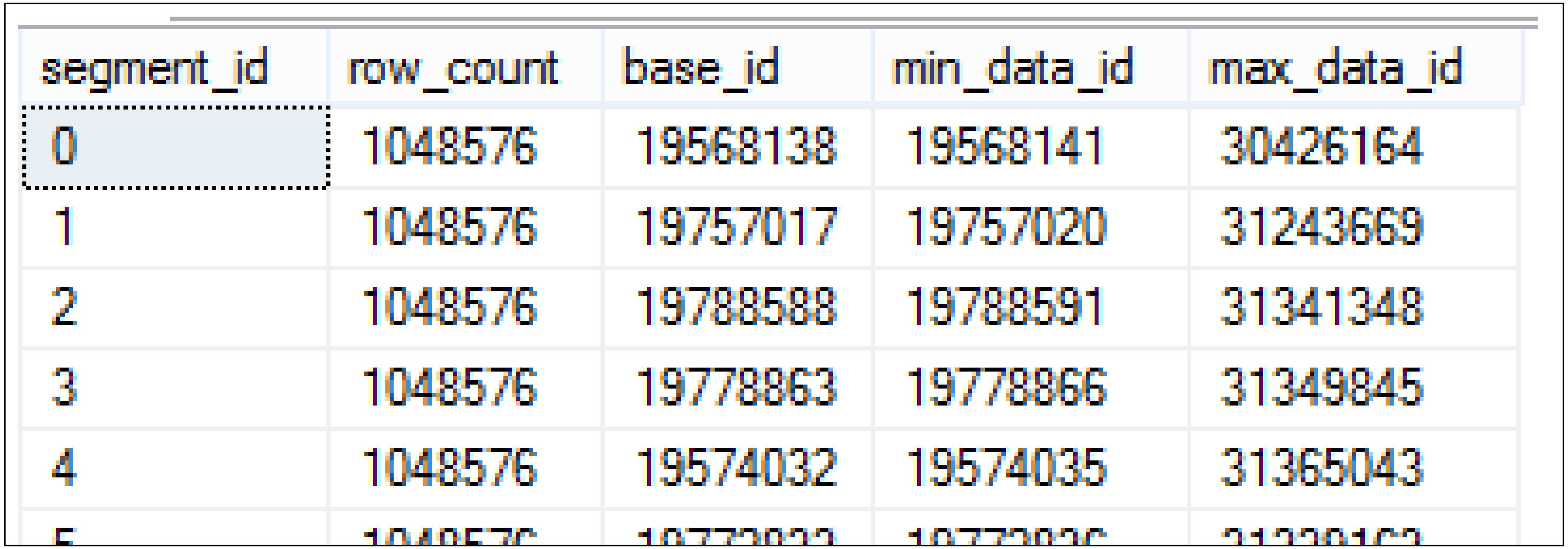
Exist in one of these states

- **Open**: In delta store; accepting new rows, data is not compresses or indexed
- **Closed**: In delta store; not accepting new rows
- **Compressed**: In Column Store format
- **Tombstone**: a rowgroup with all the data deleted

System View: **sys.pdw_nodes_column_store_row_groups**

Columnstore: Column Segments

Example of a segment from **sys.pdw_nodes_column_store_segments**



segment_id	row_count	base_id	min_data_id	max_data_id
0	1048576	19568138	19568141	30426164
1	1048576	19757017	19757020	31243669
2	1048576	19788588	19788591	31341348
3	1048576	19778863	19778866	31349845
4	1048576	19574032	19574035	31365043
5	1048576	19773833	19773836	31338163

Demo

Columnstore: Updating

CCIs perform **large block compression**

- Maximises compression
- Expensive to update

CCIs use two mechanisms to **handle updates**

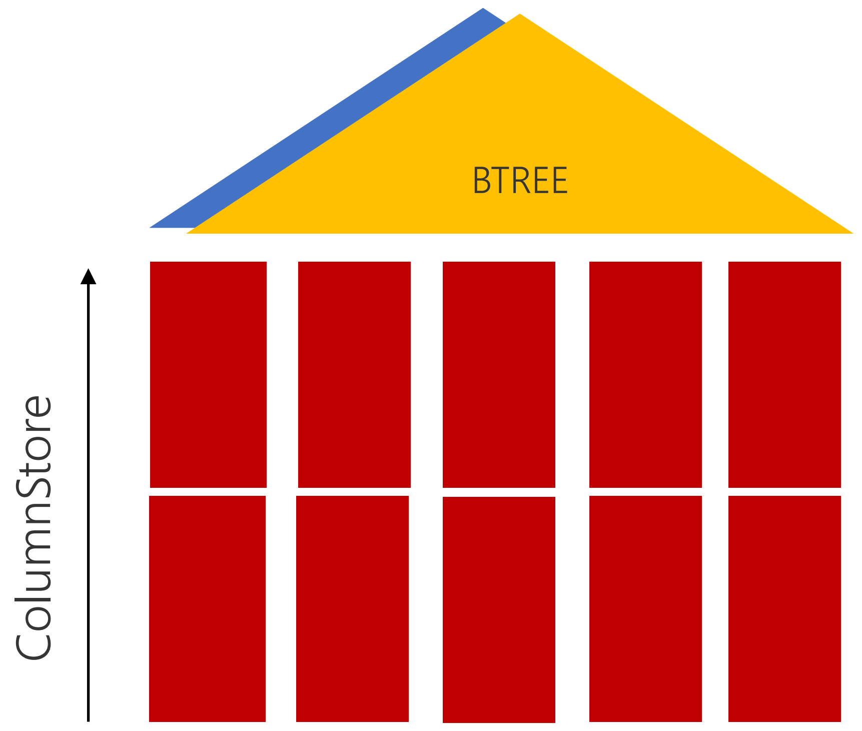
- Delete bitmap
- Delta stores

Columnstore: Delta Store

Page compressed b-tree i.e. a row store

- B-tree on unique integer row ID
- Matches user columns defined in the CCI
- Aligned to underlying CCI partition
- Created only if needed

Max # rows
delta store
1,048,576



Columnstore: Delta Store to Columnstore

Once a row group has closed it can be converted to column store format:

As an **online operation**

- Tuple Mover: a background process
- ALTER INDEX..REORGANIZE WITH (COMPRESS_ALL_ROW_GROUPS = ON);

As an **offline operation**

- ALTER INDEX..REBUILD

Columnstore: Small DML

Inserts

- Small inserts are written first to the delta store
- Bulk-load based on some internal threshold writes as column store

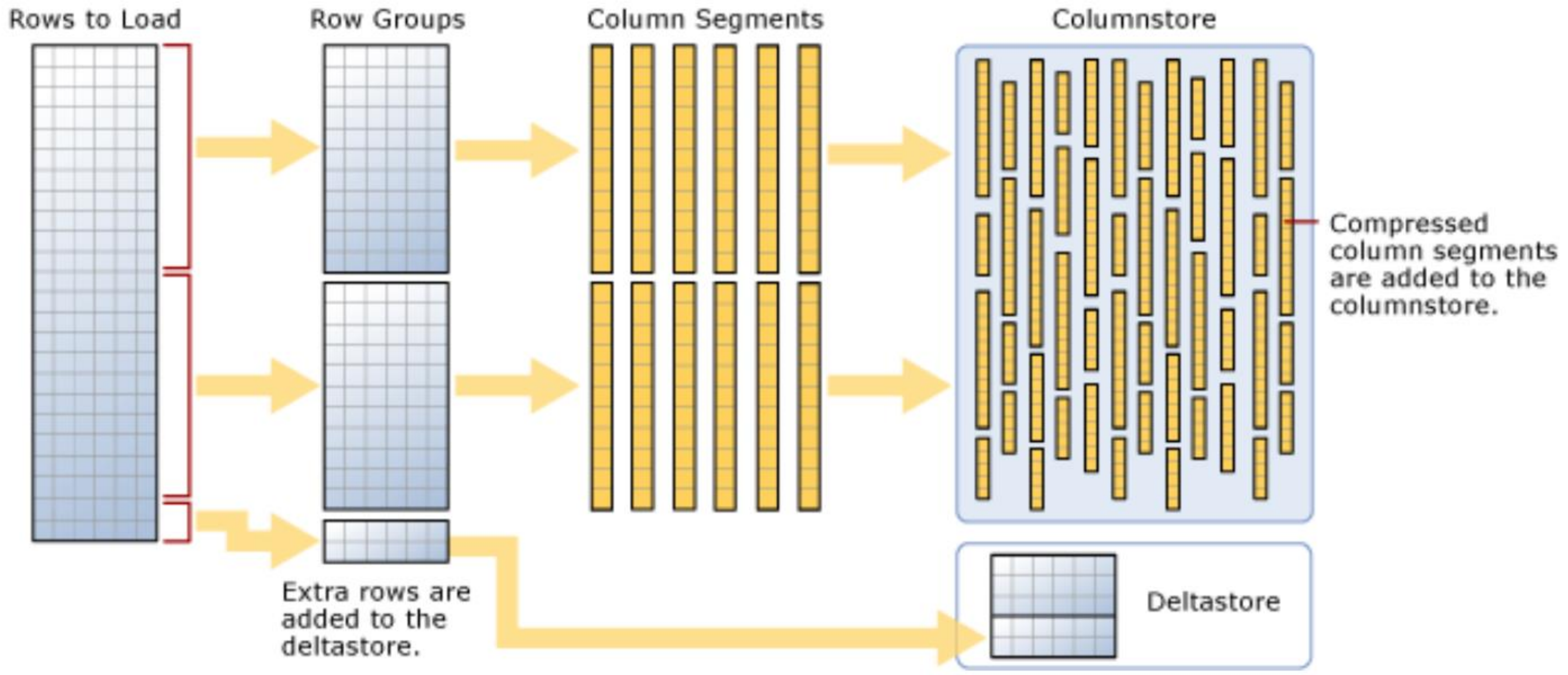
Deletes

- Logical against rows in column store
- Physically against rows in the delta store

Updates

- Converted to a logical delete and an insert

Columnstore: Design and storage


Rows to Load
Row Groups
Column Segments
Columnstore
Compressed column segments are added to the columnstore.
Extra rows are added to the deltastore.
Deltastore

Demo

Optimizing with Indexes: Choose right type

Clustered Columnstore
(Default)

- Optimal choice for large tables
- Limits scans to columns in the query
- Optimal compression
- Slower to load than Heap
- Keep partitions large enough to compress (> 1 million rows)

Heap

- Optimal choice for temporary or staging tables
- Fastest load performance

Clustered Index

- Optimal for tables < 60M rows
- Sorting operation slows-down load

Non-clustered Index

- **Use sparingly**
- Optimize single row lookups
- Will slow-down load