

# SQLServerFast.com

## Execution Plan Video Training

Block 2: Reading data

Level: Advanced

Chapter 1: Columnstore indexes

# Columnstore indexes

## Columnstore index

Also called “columnstore”

Special index type

Optimized for large scale analytic work

Recommended reading:

Microsoft documentation: <https://docs.microsoft.com/en-us/sql/relational-databases/indexes/columnstore-indexes-overview>

Stairway to columnstore indexes: <https://www.sqlservercentral.com/stairways/stairway-to-columnstore-indexes>

Niko Neugebauer’s 131-part blog series: <https://www.nikoport.com/columnstore/>

# Columnstore indexes

## Columnstore index

- Also called “columnstore”

- Special index type

- Storage structure

- Effect on scan, seek, and lookup operators

# Columnstore indexes

## Storage structure

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       ProductName,  
       Amt,  
       NetPrice,  
       TotalPrice  
FROM   dbo.SalesTable  
WHERE  Saledate = '20210327';
```

Storage structure (row based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       ProductName,  
       Amt,  
       NetPrice,  
       TotalPrice  
FROM   dbo.SalesTable  
WHERE  Saledate = '20210327';
```

Storage structure (row based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       ProductName,  
       Amt,  
       NetPrice,  
       TotalPrice  
FROM   dbo.SalesTable  
WHERE  Saledate = '20210320';
```

Storage structure (row based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       SUM(Amt),  
       AVG(NetPrice)  
FROM   dbo.SalesTable  
GROUP BY SalesPerson;
```

## Storage structure (row based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	<i>...</i>
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       SUM(Amt),  
       AVG(NetPrice)  
FROM   dbo.SalesTable  
GROUP BY SalesPerson;
```

Storage structure (row based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,
       SUM(Amt),
       AVG(NetPrice)
FROM   dbo.SalesTable
GROUP BY SalesPerson;
```

Storage structure (row based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

Storage structure (column based)

<i>SaleDate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT *
FROM   dbo.SalesTable
WHERE  ProductName = 'Pocket knife';
```

Storage structure (column based)

<i>SaleDate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       SUM(Amt),  
       AVG(NetPrice)  
FROM   dbo.SalesTable  
GROUP BY SalesPerson;
```

## Storage structure (column based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	<i>...</i>
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes

```
SELECT SalesPerson,  
       SUM(Amt),  
       AVG(NetPrice)  
FROM   dbo.SalesTable  
GROUP BY SalesPerson;
```

Storage structure (column based)

<i>Saledate</i>	<i>ProductName</i>	<i>Amt</i>	<i>GrossPrice</i>	<i>SalesTax</i>	<i>NetPrice</i>	<i>TotalPrice</i>	<i>SalesPerson</i>	...
2021-03-08	Candy bar	50	1.50	0.32	1.82	91.00	John	...
2021-03-10	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-11	Apple (bag)	7	4.51	0.95	5.46	38.22	Angie	...
2021-03-12	Smart phone	1	349.50	73.40	422.90	422.90	Angie	...
2021-03-19	Chair	1	599.50	125.90	725.40	725.40	John	...
2021-03-20	Toy car	3	29.97	6.29	36.26	108.78	Caroline	...
2021-03-20	Chair	3	599.50	125.90	725.40	2,176.20	Angie	...
2021-03-21	Laptop	2	2,860.00	600.60	3,460.60	6,921.20	John	...
2021-03-21	Apple (bag)	14	4.51	0.95	5.46	76.44	Caroline	...
2021-03-27	Apple (bag)	2	4.51	0.95	5.46	10.92	Angie	...
2021-03-27	Chair	5	599.50	125.90	725.40	3,627.00	John	...
2021-03-28	Pocket knife	1	12.95	2.72	15.67	15.67	Caroline	...
2021-04-02	Smart phone	1	349.50	73.40	422.90	422.90	Caroline	...
...	...	...	...	...	...	...	...	...

# Columnstore indexes in SQL Server

[illegible]

# Columnstore indexes

# Columnstore indexes in SQL Server

## Partition 1

[illegible]

## Partition 2

7743	0	2013-06-20 00:00:00.000	S	1	504748	HAL	10-02-000202	2020	HAL	9	2794	2794	1	5622	1130904246	8596	70.19	42.632	15.748
7744	0	2013-06-20 00:00:00.000	S	501	501775	POW10013380	10-02-000386	2020	HAL	9	501	501	1	5623	128203010	8754	120.00	10.00	10.00
7745	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7746	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7747	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7748	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7749	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7750	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7751	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7752	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7753	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7754	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7755	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7756	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7757	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7758	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7759	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7760	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7761	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7762	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7763	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7764	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7765	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7766	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7767	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7768	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7769	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7770	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7771	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7772	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7773	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7774	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7775	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7776	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7777	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7778	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7779	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7780	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7781	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7782	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7783	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7784	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7785	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7786	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7787	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7788	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7789	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7790	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7791	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7792	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7793	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7794	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7795	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7796	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7797	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7798	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7799	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	0.740
7800	0	2013-06-20 00:00:00.000	S	504791	HAL	10-02-014464	10-02-000144	2020	HAL	9	11500	11500	1	2400	5368001214	276	357.27	28.216	

# Columnstore indexes in SQL Server

## Partition 2

Rowgroup 2: Rowgroup 1:

[illegible]

# Columnstore indexes in SQL Server

Segment

Rowgroup 1: 1048576 rows

Rowgroup 2:  
< 1048576 rows

Rowgroup 1: 1048576 rows

Rowgroup 2:  
< 1048576 rows

62781	9	2015-12-01 00:00:00	5	1	5062761	NAIL	10450-021580	21600	NAIL	10	22055	22055	1	17773	11925052118	10704	17548	106304	43137		47348	0	10420-021580	26020	NAIL	10450-021580	26020	NAIL	10	27460	27460	1	9622	11388064204	5088	702189	612492	151470	
62782	9	2015-12-01 00:00:00	5	1	5062762	NAIL	10450-021581	21600	NAIL	10	22055	22055	1	17773	11925052118	10704	17548	106304	43137		47348	0	10420-021580	26020	NAIL	10450-021580	26020	NAIL	10	27460	27460	1	9622	11388064204	5088	702189	612492	151470	
72275	9	2015-08-05 00:00:00	5	0	5072170	NAIL	10450-020602	26022	NAIL	9	11908	11908	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-020602	26022	NAIL	10450-020602	26022	NAIL	9	11908	11908	1	10086	64331130560	12140	10476	431902	134308	
68807	9	2015-02-03 00:00:00	5	0	5068887	NAIL	10450-011409	11409	NAIL	9	21073	21073	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011409	11409	NAIL	10450-011409	11409	NAIL	10	20110	20110	1	7014	77724004469	9403	128933	101264	32233	
68808	9	2015-02-03 00:00:00	5	0	5068888	NAIL	10450-011409	11409	NAIL	9	21073	21073	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011409	11409	NAIL	10450-011409	11409	NAIL	10	20110	20110	1	7014	77724004469	9403	128933	101264	32233	
6904	9	2015-02-03 00:00:00	5	0	5069054	NAIL	10450-011409	11409	NAIL	9	21073	21073	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011409	11409	NAIL	10450-011409	11409	NAIL	10	20110	20110	1	7014	77724004469	9403	128933	101264	32233	
6904	9	2015-02-03 00:00:00	5	0	5069054	NAIL	10450-011409	11409	NAIL	9	21073	21073	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011409	11409	NAIL	10450-011409	11409	NAIL	10	20110	20110	1	7014	77724004469	9403	128933	101264	32233	
59913	9	2015-11-17 00:00:00	5	0	5089813	NAIL	10450-021628	21628	NAIL	9	26042	26042	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-021628	21628	NAIL	10450-021628	21628	NAIL	10	26042	26042	1	11980	12451741006	1000	4937	139971	1200	
59913	9	2015-11-17 00:00:00	5	0	5089813	NAIL	10450-021628	21628	NAIL	9	26042	26042	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-021628	21628	NAIL	10450-021628	21628	NAIL	10	26042	26042	1	11980	12451741006	1000	4937	139971	1200	
71717	9	2015-04-30 00:00:00	5	0	5071717	NAIL	10450-043008	21608	NAIL	9	28802	28802	1	13321	622600490004	NAIL	9444	75808	28802		11675	0	10450-043008	21608	NAIL	10450-043008	21608	NAIL	7	11509	11509	1	140	62016170	NAIL	2443135	184460	61300	
46173	9	2015-04-30 00:00:00	5	0	5046173	POD3PT17423	10450-020808	20810	270	5	940	940	5	4008	622600490004	NAIL	9444	75808	28802		11675	0	10450-020808	20810	270	5	940	940	5	4008	622600490004	NAIL	9444	75808	28802				
46173	9	2015-04-30 00:00:00	5	0	5046173	POD3PT17423	10450-020808	20810	270	5	940	940	5	4008	622600490004	NAIL	9444	75808	28802		11675	0	10450-020808	20810	270	5	940	940	5	4008	622600490004	NAIL	9444	75808	28802				
47102	9	2015-02-03 00:00:00	5	0	5047102	NAIL	10450-011424	11424	NAIL	9	20476	20476	1	10086	620551054910	NAIL	2443135	184460	61300		11675	0	10450-011424	11424	NAIL	10450-011424	11424	NAIL	10	20476	20476	1	10086	620551054910	NAIL	2443135	184460	61300	
9306	9	2015-11-03 00:00:00	5	0	5093066	NAIL	10450-020447	20447	NAIL	6	26176	26176	1	10086	620551054910	NAIL	2443135	184460	61300		11675	0	10450-020447	20447	NAIL	10450-020447	20447	NAIL	6	26176	26176	1	10086	620551054910	NAIL	2443135	184460	61300	
9306	9	2015-11-03 00:00:00	5	0	5093066	NAIL	10450-020447	20447	NAIL	6	26176	26176	1	10086	620551054910	NAIL	2443135	184460	61300		11675	0	10450-020447	20447	NAIL	10450-020447	20447	NAIL	6	26176	26176	1	10086	620551054910	NAIL	2443135	184460	61300	
5205	9	2015-07-01 00:00:00	5	0	5062026	NAIL	10450-021038	20338	NAIL	6	12873	12873	1	10086	620551054910	NAIL	2443135	184460	61300		11675	0	10450-021038	20338	NAIL	10450-021038	20338	NAIL	6	12873	12873	1	10086	620551054910	NAIL	2443135	184460	61300	
47063	9	2015-07-01 00:00:00	5	0	5064763	NAIL	10450-020076	20076	270	2	818	818	5	4008	620551054910	NAIL	2443135	184460	61300		11675	0	10450-020076	20076	270	2	818	818	5	4008	620551054910	NAIL	2443135	184460	61300				
47063	9	2015-07-01 00:00:00	5	0	5064763	POD61973485	10450-020076	20076	270	2	818	818	5	4008	620551054910	NAIL	2443135	184460	61300		11675	0	10450-020076	20076	270	2	818	818	5	4008	620551054910	NAIL	2443135	184460	61300				
59770	9	2015-11-09 00:00:00	5	0	5093770	NAIL	10450-020006	20006	NAIL	9	23463	23463	1	10086	620551054910	NAIL	2443135	184460	61300		11675	0	10450-020006	20006	NAIL	10450-020006	20006	NAIL	9	23463	23463	1	10086	620551054910	NAIL	2443135	184460	61300	
48109	9	2015-02-03 00:00:00	5	0	5048109	NAIL	10450-011206	11206	NAIL	9	21073	21073	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011206	11206	NAIL	10450-011206	11206	NAIL	10	20110	20110	1	7014	77724004469	9403	128933	101264	32233	
48109	9	2015-02-03 00:00:00	5	0	5048109	NAIL	10450-011206	11206	NAIL	9	21073	21073	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011206	11206	NAIL	10450-011206	11206	NAIL	10	20110	20110	1	7014	77724004469	9403	128933	101264	32233	
61448	9	2015-12-01 00:00:00	5	0	5061448	NAIL	10450-011331	11331	NAIL	9	21086	21086	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011331	11331	NAIL	10450-011331	11331	NAIL	10	21086	21086	1	10086	64331130560	12140	10476	431902	134308	
61448	9	2015-12-01 00:00:00	5	0	5061448	NAIL	10450-011331	11331	NAIL	9	21086	21086	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011331	11331	NAIL	10450-011331	11331	NAIL	10	21086	21086	1	10086	64331130560	12140	10476	431902	134308	
59889	9	2015-09-01 00:00:00	5	1	5065886	NAIL	10450-020801	20801	NAIL	9	21900	21900	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-020801	20801	NAIL	10450-020801	20801	NAIL	10	21900	21900	1	10086	64331130560	12140	10476	431902	134308	
48110	9	2015-01-01 00:00:00	5	1	5048110	NAIL	10450-011027	10227	NAIL	9	20183	20183	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-011027	10227	NAIL	10450-011027	10227	NAIL	10	20183	20183	1	10086	64331130560	12140	10476	431902	134308	
62581	9	2015-12-01 00:00:00	5	1	5062581	NAIL	10450-021582	21600	NAIL	10	22055	22055	1	17773	11925052118	10704	17548	106304	43137		47348	0	10420-021582	26020	NAIL	10450-021582	26020	NAIL	10	27460	27460	1	9622	11388064204	5088	702189	612492	151470	
62582	9	2015-12-01 00:00:00	5	1	5062582	NAIL	10450-021583	21600	NAIL	10	22055	22055	1	17773	11925052118	10704	17548	106304	43137		47348	0	10420-021583	26020	NAIL	10450-021583	26020	NAIL	10	27460	27460	1	9622	11388064204	5088	702189	612492	151470	
62583	9	2015-12-01 00:00:00	5	1	5062583	NAIL	10450-021584	21600	NAIL	10	22055	22055	1	17773	11925052118	10704	17548	106304	43137		47348	0	10420-021584	26020	NAIL	10450-021584	26020	NAIL	10	27460	27460	1	9622	11388064204	5088	702189	612492	151470	
62584	9	2015-12-01 00:00:00	5	1	5062584	NAIL	10450-021585	21600	NAIL	10	22055	22055	1	17773	11925052118	10704	17548	106304	43137		47348	0	10420-021585	26020	NAIL	10450-021585	26020	NAIL	10	27460	27460	1	9622	11388064204	5088	702189	612492	151470	
50909	9	2015-09-01 00:00:00	5	1	5090909	NAIL	10450-021900	21900	NAIL	10	22400	22400	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-021900	21900	NAIL	10450-021900	21900	NAIL	10	22400	22400	1	10086	64331130560	12140	10476	431902	134308	
50909	9	2015-09-01 00:00:00	5	1	5090909	NAIL	10450-021900	21900	NAIL	10	22400	22400	1	10086	64331130560	12140	10476	431902	134308		42791	0	10450-021900	21900	NAIL	10450-021900	21900	NAIL	10	22400	22400	1	10086	64331130560	12140	10476	431902	134308	
60882	9	2015-03-01 00:00:00	5	0	5060882	NAIL	10450-011764	11764	NAIL	7	22087	22087	1	10086	620551054910	NAIL	2443135	184460	61300		11675	0	10450-011764	11764	NAIL	10450-011764	11764												

# Columnstore indexes

(metadata in system table)

Minimum: 2011-06-21 00:00:00.000

Maximum: 2014-06-01 00:00:00.000

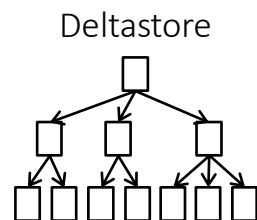
# Columnstore indexes in SQL Server

### Partition 1

[illegible]

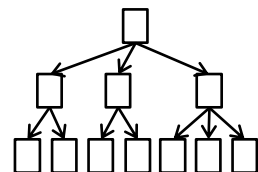
Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows



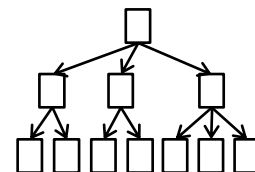
## Deltastore

Deleted "bitmap"



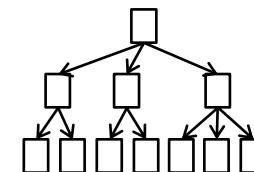
Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows



Deltastore

## Partition 2

[illegible]

Deleted "bitmap"

# Columnstore indexes

## Clustered vs nonclustered

Same structure

Maximum one columnstore index per table

Clustered columnstore index

- All columns included

- No other clustered index allowed

Nonclustered columnstore index

- Specify columns to include

- Best practice: include all columns

- Table may have clustered index, or may be a heap

# Columnstore indexes

## Columnstore Index Scan

Shown as distinct operator

Actually just an Index Scan or Clustered Index Scan in the XML

Property *Storage* = ColumnStore



Columnstore Index Scan (NonClustere...

# Columnstore indexes

## Columnstore Index Scan Properties (popup)

## Rowgroup elimination

Also (incorrectly) called segment elimination

Uses *Predicate* property

Checked against min/max in metadata

Entire rowgroup may be skipped

Columnstore

Columnstore Index Scan (Clustered)	
Scan a columnstore index, entirely or only a range.	
Physical Operation	Columnstore Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Batch
Estimated Execution Mode	Batch
Storage	ColumnStore
Actual Number of Rows for All Executions	90
Actual Number of Batches	1
Estimated I/O Cost	0,0490509
Estimated Operator Cost	0,267114 (71%)
Estimated CPU Cost	0,218063
Estimated Subtree Cost	0,267114
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows for All Executions	20933,4
Estimated Number of Rows Per Execution	20933,4
Estimated Number of Rows to be Read	1982250
Estimated Row Size	31 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Actual Number of Locally Aggregated Rows	15708
Node ID	1
Predicate	[Playground].[dbo].[Fact_Sales].[ReturnAmount] as [fs].[ReturnAmount]> (\$10.0000)
Object	[Playground].[dbo].[Fact_Sales].[ccix_Fact_Sales] [fs]
Output List	[Playground].[dbo].[Fact_Sales].StoreKey; [Playground].[dbo].[Fact_Sales].SalesAmount; [Playground].[dbo].[Fact_Sales].ReturnQuantity

# Columnstore indexes

## Columnstore Index Scan Properties (popup)

## Column elimination

Read segments for columns in Output List

Read segments for columns in Predicate

Segments for all other columns are not read

Columnstore

Columnstore Index Scan (Clustered)	
Scan a columnstore index, entirely or only a range.	
Physical Operation	Columnstore Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Batch
Estimated Execution Mode	Batch
Storage	ColumnStore
Actual Number of Rows for All Executions	90
Actual Number of Batches	1
Estimated I/O Cost	0,0490509
Estimated Operator Cost	0,267114 (71%)
Estimated CPU Cost	0,218063
Estimated Subtree Cost	0,267114
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows for All Executions	20933,4
Estimated Number of Rows Per Execution	20933,4
Estimated Number of Rows to be Read	1982250
Estimated Row Size	31 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Actual Number of Locally Aggregated Rows	15708
Node ID	1
Predicate	[Playground].[dbo].[Fact_Sales].[ReturnAmount] as [fs].[ReturnAmount]> (\$10.0000)
Object	[Playground].[dbo].[Fact_Sales].[ccix Fact_Sales] [fs]
Output List	[Playground].[dbo].[Fact_Sales].StoreKey; [Playground].[dbo].[Fact_Sales].SalesAmount; [Playground].[dbo].[Fact_Sales].ReturnQuantity

# Columnstore indexes

## Columnstore Index Scan Properties (popup)

Data stored unordered  
Property *Ordered* is always false  
No “free” ordered data

Columnstore

Columnstore Index Scan (Clustered)	
Scan a columnstore index, entirely or only a range.	
Physical Operation	Columnstore Index Scan
Logical Operation	Clustered Index Scan
Actual Execution Mode	Batch
Estimated Execution Mode	Batch
Storage	ColumnStore
Actual Number of Rows for All Executions	90
Actual Number of Batches	1
Estimated I/O Cost	0,0490509
Estimated Operator Cost	0,267114 (71%)
Estimated CPU Cost	0,218063
Estimated Subtree Cost	0,267114
Estimated Number of Executions	1
Number of Executions	1
Estimated Number of Rows for All Executions	20933,4
Estimated Number of Rows Per Execution	20933,4
Estimated Number of Rows to be Read	1982250
Estimated Row Size	31 B
Actual Rebinds	0
Actual Rewinds	0
Ordered	False
Actual Number of Locally Aggregated Rows	15708
Node ID	1
Predicate	[Playground].[dbo].[Fact_Sales].[ReturnAmount] as [fs].[ReturnAmount] > (\$10.0000)
Object	[Playground].[dbo].[Fact_Sales].[ccix_Fact_Sales] [fs]
Output List	[Playground].[dbo].[Fact_Sales].StoreKey; [Playground].[dbo].[Fact_Sales].SalesAmount; [Playground].[dbo].[Fact_Sales].ReturnQuantity

# Columnstore indexes

## Columnstore Index Scan

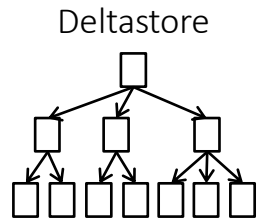
Minimum: 1

Maximum: 5

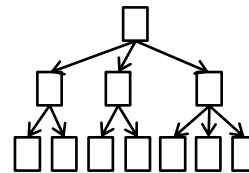
Partition 1

Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows



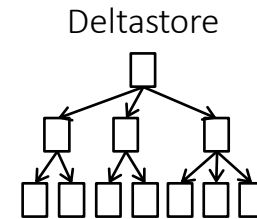
Deleted "bitmap"



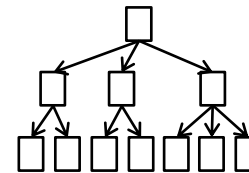
Partition 2

Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows



Deleted "bitmap"



Predicate: ShipMethodID = 6

Output List: TerritoryID, SubTotal

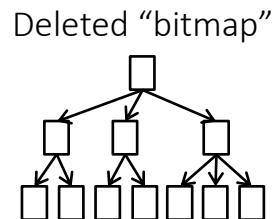
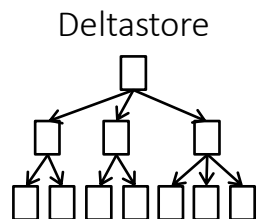
# Columnstore indexes

# Columnstore Index Scan

## Partition 1

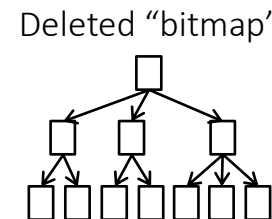
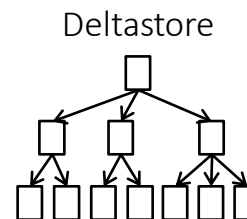
Minimum: 1  
Maximum: 8

Rowgroup 2:  
1048576 rows



Rowgroup 2:	Rowgroup 1:
< 1048576 rows	1048576 rows

## Partition 2

[illegible]

# Columnstore indexes

# Columnstore Index Scan

## Partition 1

Predicate: ShipMethodID = 6

Output List: TerritoryID, SubTotal

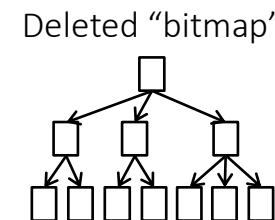
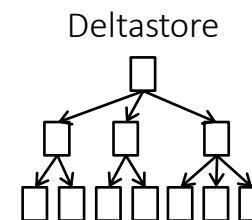
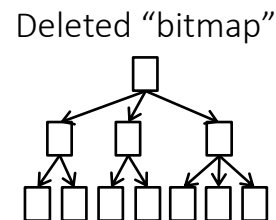
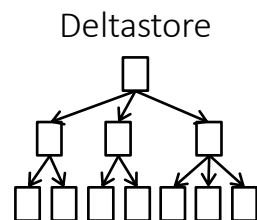
## Partition 2

Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows

[illegible][illegible]

Rowgroup 2:  
< 1048576 rows

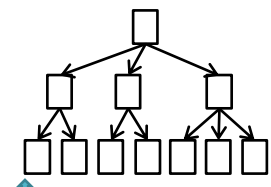
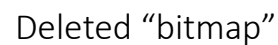
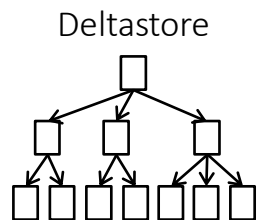


# Columnstore indexes

# Columnstore Index Scan

## Partition 1

Rowgroup 2:  
1048576 rows



Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows

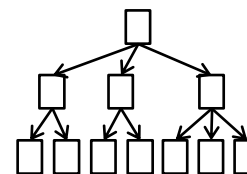
Predicate: ShipMethodID = 6

Output List: TerritoryID, SubTotal

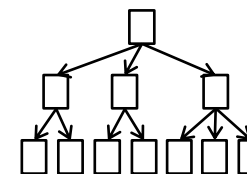
## Partition 2

[illegible][illegible]

Deltastore



Deleted "bitmap"



# Columnstore indexes

# Columnstore Index Scan

## Partition 1

Predicate: ShipMethodID = 6

Output List: TerritoryID, SubTotal

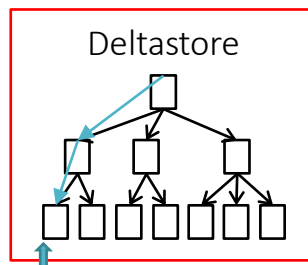
## Partition 2

Rowgroup 1:  
1048576 rows

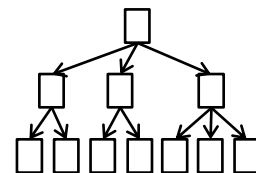
Rowgroup 2:  
< 1048576 rows

[illegible][illegible]

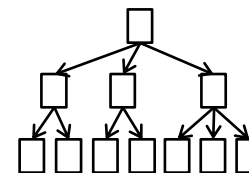
Rowgroup 2:  
< 1048576 rows



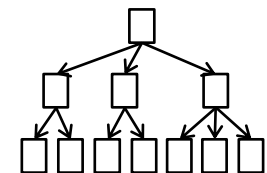
Deleted "bitmap"



Deltastore



Deleted "bitmap"



# Columnstore indexes

# Columnstore Index Scan

### Partition 1

Predicate: ShipMethodID = 6

Output List: TerritoryID, SubTotal

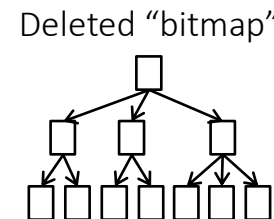
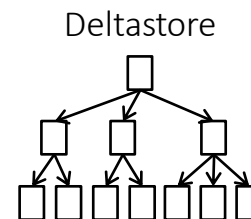
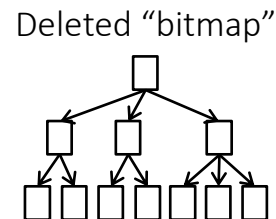
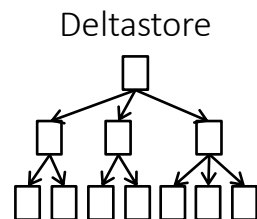
## Partition 2

Rowgroup 1:  
1048576 rows

Rowgroup 2:  
< 1048576 rows

[illegible][illegible]

Rowgroup 2:  
< 1048576 rows



# Columnstore indexes

## Columnstore Index Scan

Processes one rowgroup at a time

Columnstore compressed rowgroups:

- Read metadata, try rowgroup elimination

- Read only segments for required columns in memory

- Decompress segment data

- Read, in sync, data from all segments



Columnstore Index Scan (NonClustere...

# Columnstore indexes

## Columnstore Index Scan

Processes one rowgroup at a time

Columnstore compressed rowgroups:

- Read metadata, try rowgroup elimination

- Read only segments for required columns in memory

- Decompress segment data

- Read, in sync, data from all segments and deleted bitmap

Deltastore rowgroups:

- Effectively a B-tree

- Regular index scan process



Columnstore Index Scan (NonClustere...

# Columnstore indexes

## Columnstore Index Scan

### Batch mode execution

Very common for columnstore operations

Returns batches of several hundred rows at a time

Format somewhat similar to columnstore storage on disk

Conceptually, read and process rows until batch full

Actual implementation probably optimized

This is not documented



Columnstore Index Scan (NonClustere...

# Columnstore indexes

## Columnstore Index Scan

Functionally the same as index scans on a rowstore index  
Differences based on other storage format



Columnstore Index Scan (NonClustere...

# Columnstore indexes

## Columnstore Index Seek

- Does not exist

- Data structure not related to data

  - Rowgroups based on order of data in the existing table / order of inserts

  - Data within rowgroup reordered to optimize compression

  - Can't be used to pinpoint specific rows

# Columnstore indexes

## Key Lookup

Supports columnstore indexes since SQL Server 2016

Based on *Columnstore Locator* column

Similar to RID on heap tables

Stores rowgroup number and ordinal position within rowgroup



Key Lookup (Clustered)

# Columnstore indexes

## Key Lookup

Supports columnstore indexes since SQL Server 2016

Based on *Columnstore Locator* column

Steps to read data:

- Access rowgroup

- Attempt rowgroup elimination

- Read required segments

- Decompress segments

- Process from row 1 until specified row

- Combine values and return row



Key Lookup (Clustered)

# Columnstore indexes

## Key Lookup

Supports columnstore indexes since SQL Server 2016

Based on *Columnstore Locator* column

Steps to read data

Special case: data in deltastore rowgroup

Stored as clustered index on position

Effectively a seek on the position



Key Lookup (Clustered)

# Columnstore indexes

## Key Lookup

Supports columnstore indexes since SQL Server 2016

Based on *Columnstore Locator* column

Steps to read data

Special case: data in deltastore rowgroup

Dangerous when estimates are wrong!

No support for batch mode execution



Key Lookup (Clustered)

# Summary

## Columnstore indexes

### Structure

- Per rowgroups

- Per column

- Deltastore rowgroups

- Deleted bitmap

# Summary

## Columnstore indexes

- Structure

- Columnstore Index Scan

  - Rowgroup elimination

  - Decompress and read in sync, start to finish

- Key Lookup on columnstore index

  - Very high cost per execution

  - Sensitive to estimation errors

# Next chapters

Chapter 2: Memory-optimized indexes

- Storage structure

- Scans, seeks, and lookups

Chapter 3: Special index types

Chapter 4: Reading data in parallel or batch mode

Chapter 5: Assorted read optimizations