

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>	...
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)

Saledate	ProductName	Amt	GrossPrice	SalesTax	NetPrice	TotalPrice	SalesPerson	...
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

```
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

Columnstore indexes in SQL Server

Partition 1

SegmentID	PartitionNumber	ObjectID	IndexID	ColumnOrder	LeafLevel	ParentColumnOrder	ColumnOrder	SubPartition	Terminal	MFIndexID	BigTableIndex	LeafLevel	LeafCount	LeafKeyPageCount	ColumnOrder	SubLeaf	Index	Page	
42781	0	2013-01-01-00:00:00	5	1	SC42781	NALL	10-00-000000	2100	NALL	10	2000	2000	1773	1120000000	10704	170148	1381004	41137	
43339	0	2013-01-01-00:00:00	5	0	SC43339	PC1722184C2	10-00-00001	2900	277	4	618	618	5	237	3502019830	NALL	4081805	45170	141000
72270	0	2013-01-01-00:00:00	5	1	SC72270	NALL	10-00-00002	2002	NALL	9	21880	21880	1	106	8421100000	NALL	53819	41192	114000
68827	0	2013-01-01-00:00:00	5	1	SC68827	NALL	10-00-01428	1428	NALL	7	1273	1273	1	1108	8420700000	NALL	2419106	1031040	60100
44421	0	2013-01-01-00:00:00	5	1	SC44421	NALL	10-00-01428	1428	NALL	10	11700	11700	1	107	8310700000	NALL	307207	2062070	891000
62884	0	2013-01-01-00:00:00	5	1	SC62884	NALL	10-00-01211	2111	NALL	6	2042	2042	1	2087	3080000000	NALL	1100148	921004	28100
48871	0	2013-02-01-00:00:00	5	1	SC48871	PC4030185E8	10-00-00002	2970	209	6	618	618	5	1184	4700000000	NALL	6200101	9047428	157120
58913	0	2013-01-01-00:00:00	5	1	SC58913	NALL	10-00-00256	2526	NALL	31	10000	10000	1	1040	3170000000	NALL	10215	30100	10000
71171	0	2013-01-01-00:00:00	5	1	SC71171	NALL	10-00-00268	2608	NALL	4	2002	2002	1	1031	6200000000	NALL	94146	73000	2100
55516	0	2013-01-01-00:00:00	5	1	SC55516	NALL	10-00-00276	2676	NALL	4	1520	1520	1	1103	7100000000	NALL	63107	51176	15000
40123	0	2013-01-01-00:00:00	5	0	SC40123	PC0000779C2	10-00-00000	2910	270	6	940	940	5	616	8020000000	NALL	141002	1001000	1000
56789	0	2013-01-01-00:00:00	5	1	SC56789	NALL	10-00-00002	2002	NALL	1	13670	13670	1	4041	10417000000	NALL	1102149	891000	201120
47162	0	2013-01-01-00:00:00	5	1	SC47162	NALL	10-00-01024	1024	NALL	1	20407	20407	1	1190	5320000000	NALL	244130	199140	610000
69306	0	2013-01-01-00:00:00	5	1	SC69306	NALL	10-00-00294	2947	NALL	6	20176	20176	1	4800	1020000000	NALL	10110	60000	10000
49966	0	2013-01-01-00:00:00	5	1	SC49966	NALL	10-00-01295	1295	NALL	10	13000	13000	1	303	1120000000	NALL	2071406	1007100	913000
52006	0	2013-01-01-00:00:00	5	1	SC52006	NALL	10-00-00000	2000	NALL	6	10070	10070	1	216	1240700000	NALL	9010	47000	14000
62623	0	2013-01-01-00:00:00	5	1	SC62623	NALL	10-00-00207	2007	NALL	9	20000	20000	1	800	4170000000	NALL	49107	33010	10000
47463	0	2013-01-01-00:00:00	5	0	SC47463	PC0001070E8	10-00-00000	29407	275	2	818	818	5	800	6077000000	NALL	230000	223000	60000
58770	0	2013-01-01-00:00:00	5	1	SC58770	NALL	10-00-00000	2000	NALL	4	23463	23463	1	760	1210000000	NALL	12100	6100	1000
56206	0	2013-01-01-00:00:00	5	1	SC56206	NALL	10-00-00104	2004	NALL	1	10110	10110	1	1004	5000000000	NALL	10310	9100	30000
48108	0	2013-01-01-00:00:00	5	1	SC48108	NALL	10-00-00000	2000	NALL	8	2000	2000	1	3000	3000000000	NALL	10000000	10000	20000
60004	0	2013-01-01-00:00:00	5	1	SC60004	NALL	10-00-01196	1196	NALL	1	17001	17001	1	1021	8000000000	NALL	56107	40070	14000
41460	0	2013-01-01-00:00:00	5	1	SC41460	NALL	10-00-01031	1031	NALL	6	21000	21000	1	1412	7220000000	NALL	10000	10000	10000
54005	0	2013-01-01-00:00:00	5	1	SC54005	NALL	10-00-00202	2021	NALL	10	20000	20000	1	NALL	NALL	NALL	8320	10014	4100
49310	0	2013-01-01-00:00:00	5	1	SC49310	NALL	10-00-01007	1007	NALL	4	20100	20100	1	1102	7000000000	NALL	10000	60000	10000
70000	0	2013-01-01-00:00:00	5	1	SC70000	NALL	10-00-00000	2000	NALL	1	10170	10170	1	7007	1020000000	NALL	14100	10000	10000
62308	0	2013-01-01-00:00:00	5	1	SC62308	NALL	10-00-00204	2004	NALL	4	11000	11000	1	1304	1200000000	NALL	45100	30170	11000
66460	0	2013-01-01-00:00:00	5	1	SC66460	NALL	10-00-00000	2000	NALL	7	10010	10010	1	8148	6400000000	NALL	11000	10000	20000
66043	0	2013-01-01-00:00:00	5	1	SC66043	NALL	10-00-00000	2000	NALL	4	27000	27000	1	1700	4200000000	NALL	7000	6100	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	7	27000	27000	1	1000	3000000000	NALL	21000	10000	10000
63007	0	2013-01-01-00:00:00	5	1	SC63007	NALL	10-00-00000	2000	NALL	10	20010	20010	1	1000	2000000000	NALL	10000	10000	10000
63002	0	2013-01-01-00:00:00	5	1	SC63002	NALL	10-00-00000	2000	NALL	7	20007	20007	1	6400	1270000000	NALL	77000	62100	10000
40306	0	2013-01-01-00:00:00	5	1	SC40306	NALL	10-00-00000	2000	NALL	4	17000	17000	1	1700	4200000000	NALL	30107	20010	10000
68027	0	2013-01-01-00:00:00	5	1	SC68027	NALL	10-00-01070	1070	NALL	10	20004	20004	1	2100	5000000000	NALL	64007	51000	10000
61000	0	2013-01-01-00:00:00	5	1	SC61000	NALL	10-00-00000	2000	NALL	9	23000	23000	1	7100	3000000000	NALL	20010	10000	10000
61003	0	2013-01-01-00:00:00	5	1	SC61003	NALL	10-00-00000	2000	NALL	4	10000	10000	1	1000	1000000000	NALL	20010	10000	10000
51403	0	2013-01-01-00:00:00	5	1	SC51403	NALL	10-00-01001	1001	NALL	9	14000	14000	1	800	2000000000	NALL	20010	10000	10000
61000	0	2013-01-01-00:00:00	5	1	SC61000	NALL	10-00-00000	2000	NALL	7	20000	20000	1	1000	1000000000	NALL	10000	10000	10000
58532	0	2013-01-01-00:00:00	5	1	SC58532	NALL	10-00-01104	2104	NALL	9	14000	14000	1	1433	2000000000	NALL	10000	10000	10000

Partition 2

SegmentID	PartitionNumber	ObjectID	IndexID	ColumnOrder	LeafLevel	ParentColumnOrder	ColumnOrder	SubPartition	Terminal	MFIndexID	BigTableIndex	LeafLevel	LeafCount	LeafKeyPageCount	ColumnOrder	SubLeaf	Index	Page	
47040	0	2013-01-01-00:00:00	5	1	SC47040	NALL	10-00-00000	2000	NALL	5	27000	27000	1	5020	1100000000	NALL	9000	10000	10000
51700	0	2013-01-01-00:00:00	5	0	SC51700	PC102018000	10-00-00000	2970	278	6	501	501	5	10010	1000000000	NALL	10000	10000	10000
43701	0	2013-01-01-00:00:00	5	1	SC43701	NALL	10-00-01004	1004	NALL	9	11000	11000	1	2400	5000000000	NALL	270	10000	10000
51703	0	2013-01-01-00:00:00	5	1	SC51703	NALL	10-00-00000	2000	NALL	10	20110	20110	1	7014	7000000000	NALL	10000	10000	10000
64411	0	2013-01-01-00:00:00	5	1	SC64411	NALL	10-00-01007	1007	NALL	10	10000	10000	1	1001	7000000000	NALL	10000	10000	10000
10000	0	2013-01-01-00:00:00	5	1	SC10000	NALL	10-00-00000	2000	NALL	4	10000	10000	1	10000	1000000000	NALL	10000	10000	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	4	20000	20000	1	400	8000000000	NALL	10000	10000	10000
51010	0	2013-01-01-00:00:00	5	0	SC51010	PC0000770E8	10-00-00000	2970	275	4	900	900	5	240	2700000000	NALL	90000	10000	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	10	10000	10000	1	10000	1000000000	NALL	10000	10000	10000
51015	0	2013-01-01-00:00:00	5	1	SC51015	NALL	10-00-00000	2000	NALL	4	20000	20000	1	10000	1000000000	NALL	10000	10000	10000
70100	0	2013-01-01-00:00:00	5	1	SC70100	NALL	10-00-00000	2000	NALL	4	20000	20000	1	10000	1000000000	NALL	10000	10000	10000
47007	0	2013-01-01-00:00:00	5	1	SC47007	NALL	10-00-00000	2000	NALL	1	10000	10000	1	10000	1000000000	NALL	10000	10000	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	10	10000	10000	1	10000	1000000000	NALL	10000	10000	10000
51004	0	2013-01-01-00:00:00	5	1	SC51004	NALL	10-00-01000	1000	NALL	10	10000	10000	1	10000	1000000000	NALL	10000	10000	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	6	14000	14000	1	11000	1000000000	NALL	10000	10000	10000
62001	0	2013-01-01-00:00:00	5	1	SC62001	NALL	10-00-01000	1000	NALL	8	20000	20000	1	20000	1000000000	NALL	10000	10000	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	8	20000	20000	1	20000	1000000000	NALL	10000	10000	10000
44001	0	2013-01-01-00:00:00	5	1	SC44001	NALL	10-00-00000	2000	NALL	6	24000	24000	1	8000	1000000000	NALL	10000	10000	10000
60000	0	2013-01-01-00:00:00	5	1	SC60000	NALL	10-00-00000	2000	NALL	6	14000	14000	1	8000	1000000000	NALL	10000	10000	10000
59001	0	2013-01-01-00:00:00	5	1	SC59001	NALL	10-00-00000	2000	NALL	6	21000	21000	1	20000	1000000000	NALL	10000	10000	10000
61001	0	2013-01-01-00:00:00	5	1	SC61001	NALL	10-00-00000	2000	NALL	1	10000	10000	1	10000	1000000000	NALL	10000	10000	10000
60000	0	2013-01-01-00:00:00	5																

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

Partition 2

Rowgroup 1:
1048576 rows

StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey
80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000
80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000

Rowgroup 1:
1048576 rows

StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey
80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000	80170	1	2013-12-20 00:00:00.000		12013122000000000000
80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000	80170	1048576	2013-12-20 00:00:00.000		12013122000000000000

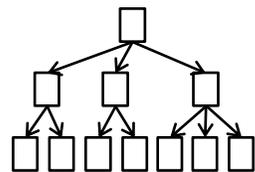
Rowgroup 2:
< 1048576 rows

StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey
80170	1	2014-04-13 00:00:00.000		12014041300000000000	80170	1	2014-04-13 00:00:00.000		12014041300000000000	80170	1	2014-04-13 00:00:00.000		12014041300000000000	80170	1	2014-04-13 00:00:00.000		12014041300000000000	80170	1	2014-04-13 00:00:00.000		12014041300000000000
80170	< 1048576	2014-04-13 00:00:00.000		12014041300000000000	80170	< 1048576	2014-04-13 00:00:00.000		12014041300000000000	80170	< 1048576	2014-04-13 00:00:00.000		12014041300000000000	80170	< 1048576	2014-04-13 00:00:00.000		12014041300000000000	80170	< 1048576	2014-04-13 00:00:00.000		12014041300000000000

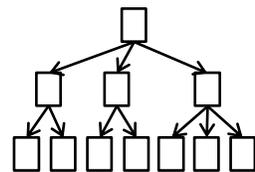
Rowgroup 2:
< 1048576 rows

StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey	StoreNumber	RowNumber	OrderDate	Status	OrderDateKey
80170	1	2013-09-04 00:00:00.000		12013090400000000000	80170	1	2013-09-04 00:00:00.000		12013090400000000000	80170	1	2013-09-04 00:00:00.000		12013090400000000000	80170	1	2013-09-04 00:00:00.000		12013090400000000000	80170	1	2013-09-04 00:00:00.000		12013090400000000000
80170	< 1048576	2013-09-04 00:00:00.000		12013090400000000000	80170	< 1048576	2013-09-04 00:00:00.000		12013090400000000000	80170	< 1048576	2013-09-04 00:00:00.000		12013090400000000000	80170	< 1048576	2013-09-04 00:00:00.000		12013090400000000000	80170	< 1048576	2013-09-04 00:00:00.000		12013090400000000000

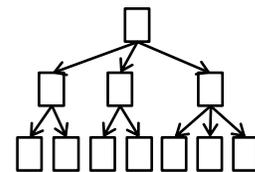
Deltastore



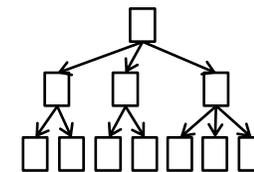
Deleted "bitmap"



Deltastore



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

Partition 1

Partition 2

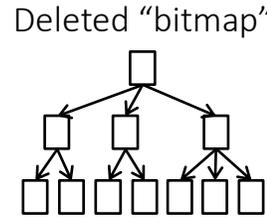
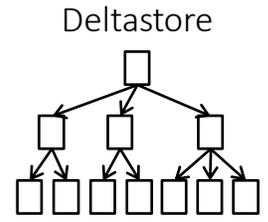
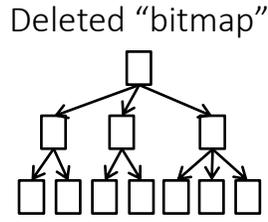
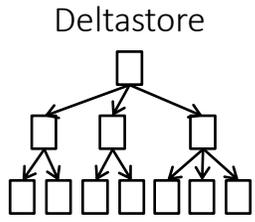
RowGroupID	PartitionNumber	OrderDate	Status	OrderDateKey	ShipOrderNumber	ProductOrderNumber	AccountNumber	CustomerID	ShipProductID	ProductID	ShipMethodID	ShipMethodCode	ShipMethodLabel	SubTotal	TaxRate	freight
47186	1	2013-07-01 00:00:00.000	T	20130701	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	761.98	0.0000	133.708
47191	1	2013-08-01 00:00:00.000	T	20130801	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	4700.908	0.0000	1348.8
47192	1	2013-08-01 00:00:00.000	T	20130801	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	3075.27	0.0000	69.458
48411	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	108.16	0.0000	50.270
48412	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	38.88	0.0000	0.748
48413	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	48.27	0.0000	1.040
48414	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	24.59	0.0000	0.524
48415	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	1073.948	0.0000	288.478
48416	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	2003.36	0.0000	518.88
48417	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	188.40	0.0000	47.808
48418	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	88.27	0.0000	2.408
48419	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	4.708	0.0000	1.040
48420	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	88.27	0.0000	21.608
48421	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	17.98	0.0000	4.708
48422	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	2084.87	0.0000	563.618
48423	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	803.27	0.0000	208.478
48424	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	108.16	0.0000	30.0
48425	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	4.708	0.0000	1.240
48426	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	28.88	0.0000	7.608
48427	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	42.28	0.0000	1.040
48428	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	1020.978	0.0000	269.88
48429	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	2927.298	0.0000	761.98
48430	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	2483.30	0.0000	618.88
48431	1	2013-09-01 00:00:00.000	T	20130901	104000000000	104000000000	104000000000	104000000000	104000000000	104000000000	6	104000000000	Ship	152.40	0.0000	41.6

Rowgroup 1:
1048576 rows

Rowgroup 2:
< 1048576 rows

Rowgroup 2:
< 1048576 rows

TerritoryID	SubTotal
1	14.23
4	48.22
7	105.48
10	78.88
13	2181.825
16	176.33
19	637.27
22	2871.096
25	2281.96
28	2473.93
31	1223.84
34	758.46



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