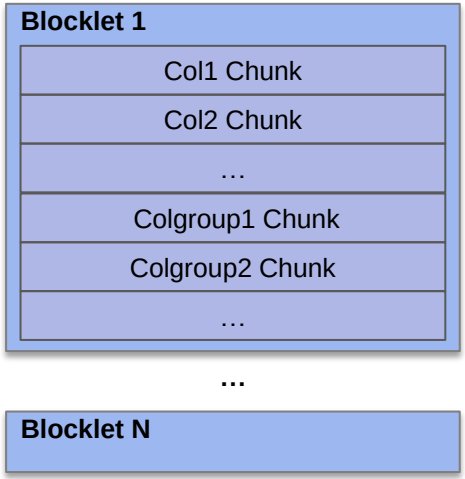
CarbonData File Format

Carbon File Structure

- **Blocklet : A set of rows in columnar format**
  - Default blocklet size: ~120k rows
  - Minimum size for predicate filtering
  - Large size for efficient reading and compression
- **Column chunk : Data for one column/column group in a Blocklet**
  - Column data stored as sorted index
  - Allow multiple columns forms a column group & stored as row-based
- **Footer : Metadata information**
  - File level statistics
  - Schema
  - Blocklet Index & Metadata



Blocklet

Column data are sorted along multi-dimensional key

Years	Quarters	Months	Territory	Country	Quantity	Sales
2003	QTR1	Jan	EMEA	Germany	142	11,432
2003	QTR1	Jan	APAC	China	541	54,702
2003	QTR1	Jan	EMEA	Spain	443	44,622
2003	QTR1	Feb	EMEA	Denmark	545	58,871
2003	QTR1	Feb	EMEA	Italy	675	56,181
2003	QTR1	Mar	APAC	India	52	9,749
2003	QTR1	Mar	EMEA	UK	570	51,018
2003	QTR1	Mar	Japan	Japan	561	55,245
2003	QTR2	Apr	APAC	Australia	525	50,398
2003	QTR2	Apr	EMEA	Germany	144	11,532

Dictionary Generation

Year	Quarters	Months	Territory	Country
2003 : 1	QTR1 : 1 QTR2 : 2	Jan : 1 Feb : 2 Mar : 3 Apr : 4	EMEA : 1 Japan : 2 APAC : 3	Germany : 1 China:2 Spain : 3 Denmark : 4 Italy : 5 India : 6 UK : 7 Japan : 8 Australia : 9

Dictionary Encoding

[1,1,1,1,1] : [142,11432]
[1,1,1,3,2] : [541,54702]
[1,1,1,1,3] : [443,44622]
[1,1,2,1,4] : [545,58871]
[1,1,2,1,5] : [675,56181]
[1,1,3,3,6] : [52,9749]
[1,1,3,1,7] : [570,51018]
[1,1,3,2,8] : [561,55245]
[1,2,4,3,9] : [525,50398]
[1,2,4,1,1] : [144,11532]

Sort
(MDK Index)

Sorted MDK Index

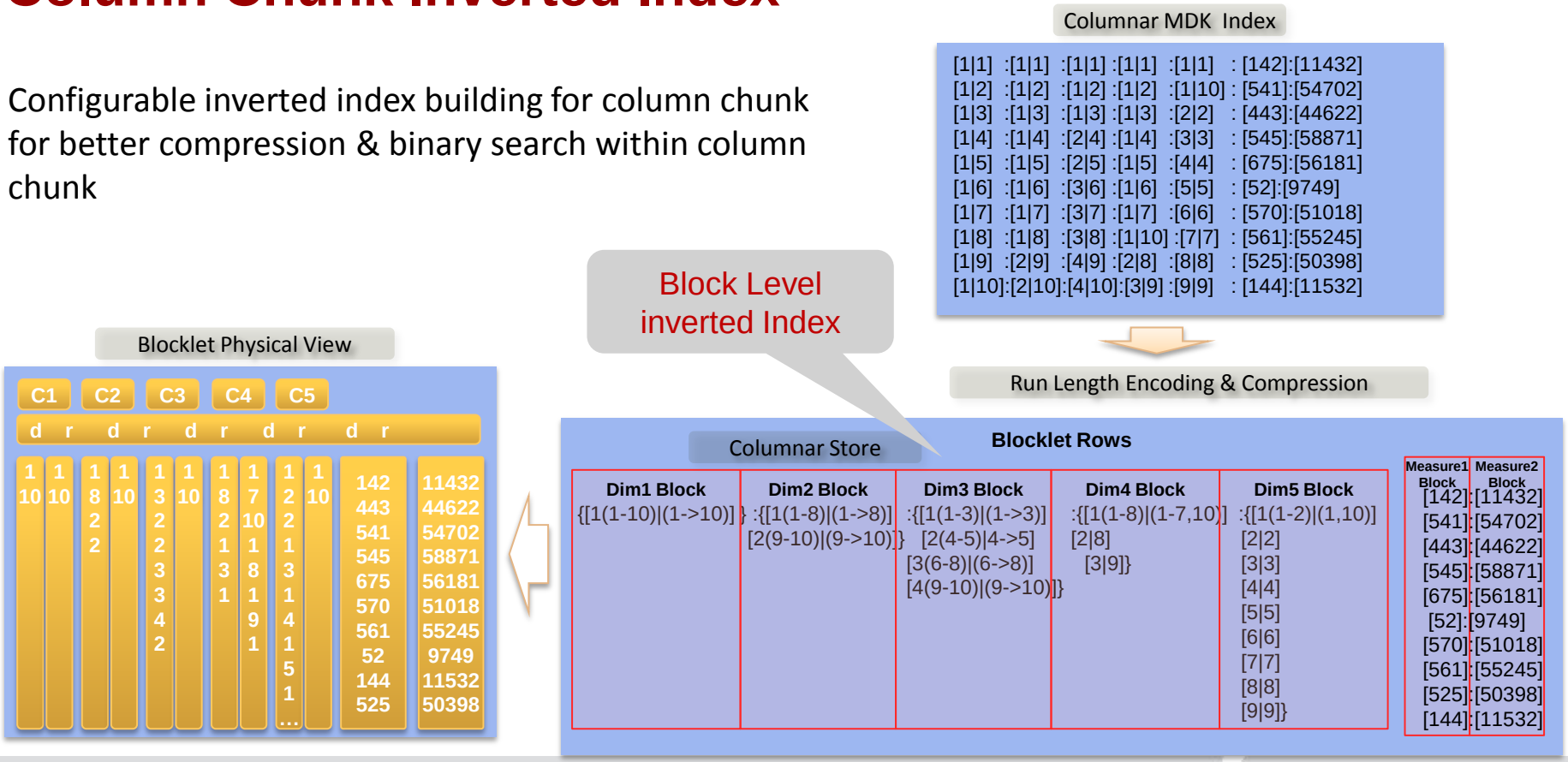
[1,1,1,1,1] : [142,11432]
[1,1,1,1,3] : [443,44622]
[1,1,1,3,2] : [541,54702]
[1,1,2,1,4] : [545,58871]
[1,1,2,1,5] : [675,56181]
[1,1,3,1,7] : [570,51018]
[1,1,3,2,8] : [561,55245]
[1,1,3,3,6] : [52,9749]
[1,2,4,1,1] : [144,11532]
[1,2,4,3,9] : [525,50398]

Blocklet Logical View

C1	C2	C3	C4	C5	C6	C7
1	1	1	1	1	142	11432
1	1	1	1	3	443	44622
1	1	1	3	2	541	54702
1	1	2	1	4	545	58871
1	1	2	1	5	675	56181
1	1	3	1	7	570	51018
1	1	3	2	8	561	55245
1	1	3	3	6	52	9749
1	2	4	1	1	144	11532
1	2	4	3	9	525	50398

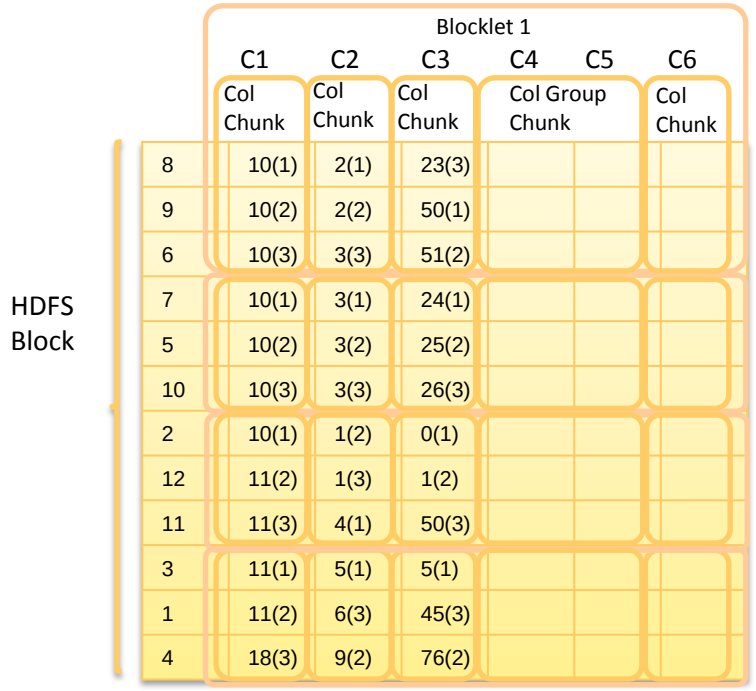
Column Chunk Inverted Index

Configurable inverted index building for column chunk for better compression & binary search within column chunk

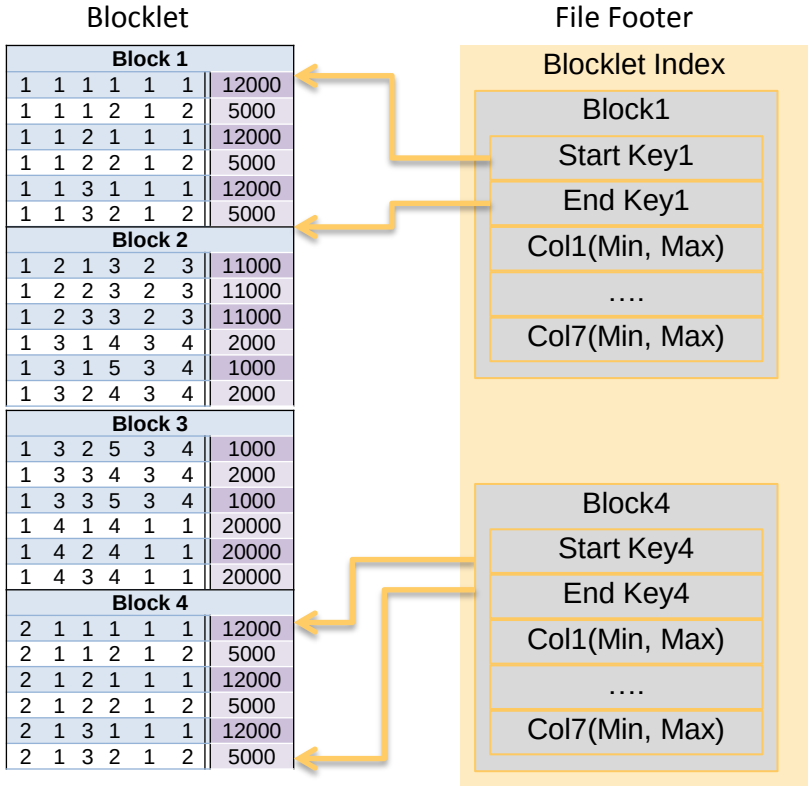


Column Group

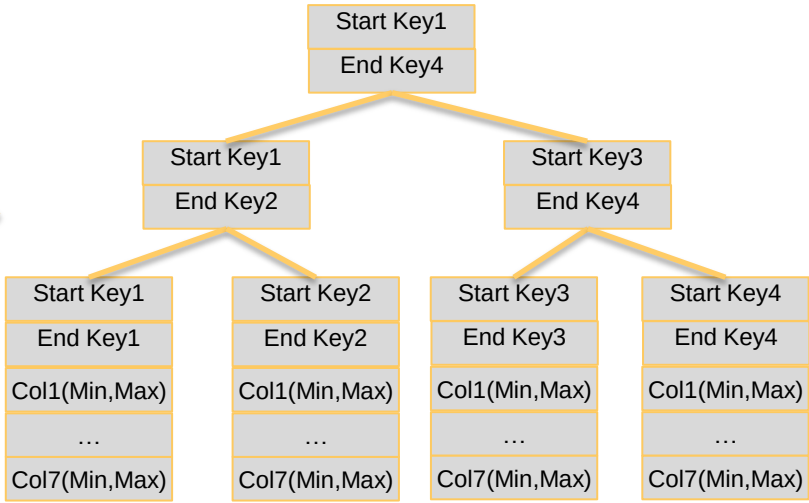
Allow multiple columns form a column group
- stored as a single column chunk in row-based format
- suitable to set of columns frequently fetched together
- saving stitching cost for reconstructing row



Blocklet Index



Build in-memory file level MDK index tree for filtering Blocklet



Encoding & Compression

- Supported Encoding Scheme:
  - DELTA Encoding:
  - Run Length Encoding:
  - BIT_PACKED:
  - Dictionary Encoding:
    - low cardinality: global dictionary (table level)
      - speedup aggregation
      - less run-time memory usage
      - Lazy decoding
      - Fast distinct counts
    - medium high cardinality: local dictionary (file level)
  - CUSTOM
- Compression Scheme: Snappy

Thank you