CarbonData:
面向交互式分析的索引文件格式

Jacky Li / 李昆

Huawei BigData Platform Engineer

Apache CarbonData PMC & Committer

Previous experience:

L3/L4/L7 Network, Protocol, Security

Telecom Value Added Service Platform

Network Data Analytics

Big Data

企业数据量大，维度多，结构复杂，且在快速增长

Network

- 54B records per day
- 750TB per month
- Complex correlated data

Consumer

- 100 thousands of sensors
- >2 million events per second
- Time series, geospatial data

Enterprise

- 100 GB to TB per day
- Data across different domains

Typical Scenario

企业中包含多种数据应用，从商业智能到批处理到机器学习

Report & Dashboard

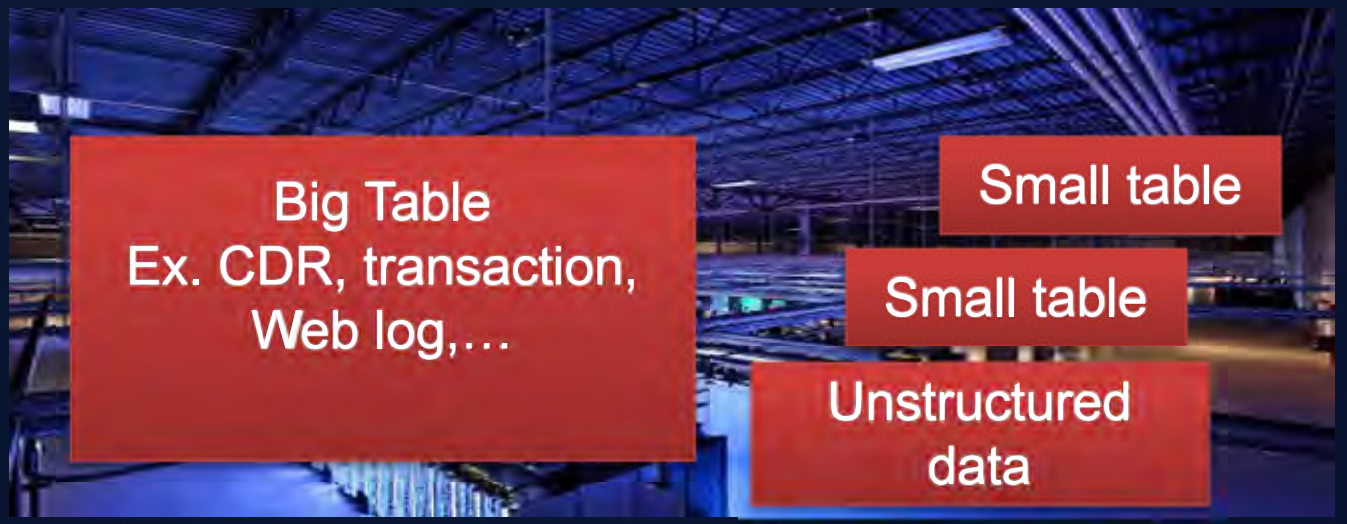
OLAP & Ad-hoc

Batch processing

Machine learning

Realtime Analytics

data



Analytic Examples

过去**1天**使用**Whatapp应用**的**终端按流量排名**情况？

过去**1天上海**市**每个小区**的**网络拥塞**统计？

Start Time	End Time	Procedure Type	MSISDN	LAC	CI	APN	Device Brand	Device Model	Device Type	Succeed Flag	Interface Type
2012/6/3 21:33	2012/6/3 21:33	CreatePDP	62878****3789	5.00E+39	794B	WWW.XLGPRS.NET	SAMSUNG	GT-C3222	3G Mobile Phone	succeed	Gn
2012/6/3 21:40	2012/6/3 21:40	DeletePDP	62878****3789	5.00E+39	794B	WWW.XLGPRS.NET	SAMSUNG	GT-C3222	3G Mobile Phone	succeed	Gn
2012/6/3 21:40	2012/6/3 21:40	CreatePDP	62878****3789	5.00E+39	794B	WWW.XLGPRS.NET	SAMSUNG	GT-C3222	3G Mobile Phone	succeed	Gn
2012/6/3 21:43	2012/6/3 21:43	DeletePDP	62878****3789	5.00E+39	794B	WWW.XLGPRS.NET	SAMSUNG	GT-C3222	3G Mobile Phone	succeed	Gn
2012/6/4 9:05	2012/6/4 9:05	CreatePDP	62878****3789	5.00E+39	84FB	WWW.XLMMS.NET	SAMSUNG	GT-C3222	3G Mobile Phone	succeed	Gn
2012/6/4 9:06	2012/6/4 9:06	DeletePDP	62878****3789	5.00E+39	84FB	WWW.XLMMS.NET	SAMSUNG	GT-C3222	3G Mobile Phone	succeed	Gn
2012/6/4 14:26	2012/6/4 14:26	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:26	2012/6/4 14:26	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:26	2012/6/4 14:26	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:26	2012/6/4 14:26	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:26	2012/6/4 14:26	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:26	2012/6/4 14:26	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:35	2012/6/4 14:35	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:38	2012/6/4 14:38	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:38	2012/6/4 14:38	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 14:38	2012/6/4 14:38	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 15:01	2012/6/4 15:01	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 15:02	2012/6/4 15:02	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 15:03	2012/6/4 15:03	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 16:24	2012/6/4 16:24	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 21:19	2012/6/4 21:19	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 21:22	2012/6/4 21:22	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn
2012/6/4 21:22	2012/6/4 21:22	CreatePDP	62878****3789	5.00E+39	956B	XLUNLIMITED	SAMSUNG	[illegible]	2G Mobile Phone	reject	Gn

Challenge - Data

- Data Size
  - Single Table >10 B
  - Fast growing

百亿级数据量

- Multi-dimensional
  - Every record > 100 dimension
  - Add new dimension occasionally

多维度

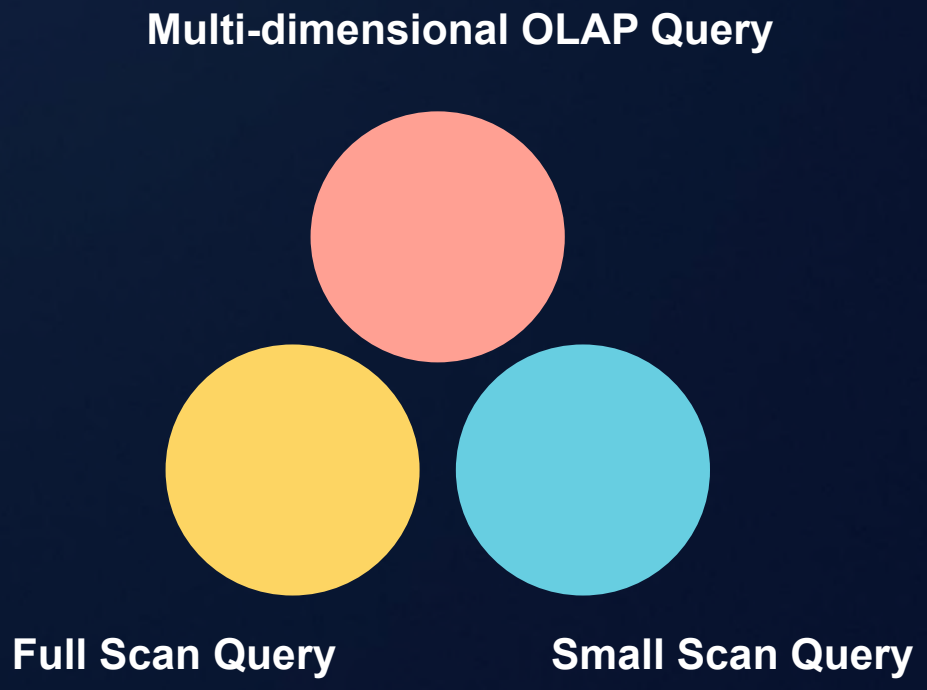
- Rich of Detail
  - Billion level high cardinality
  - 1B terminal * 200K cell * 1440 minutes = 28800 (万亿)

细粒度

Challenge - Application

- Enterprise Integration **企业应用集成**
  - SQL 2003 Standard Syntax
  - BI integration, JDBC/ODBC

- Flexible Query **灵活查询 无固定模式**
  - Any combination of dimensions
  - OLAP Vs Detail Record
  - Full scan Vs Small scan
  - Precise search & Fuzzy search



How to choose storage?

如何构建数据平台？

Option1: NoSQL Database

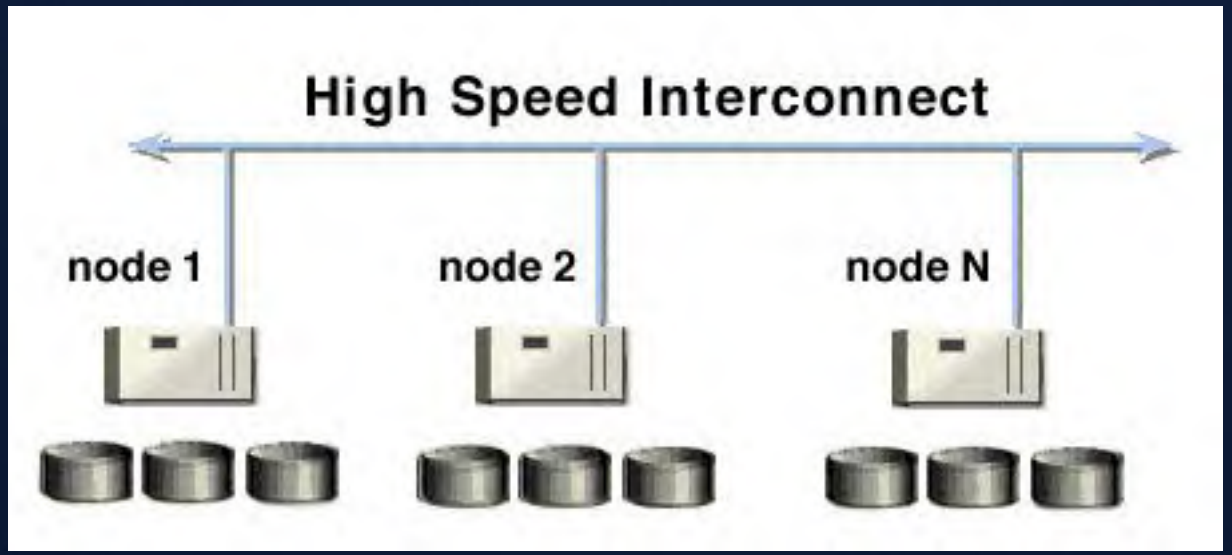
Key-Value store: low latency, <5ms

只能通过Key访问，一键一值
适合实时应用对接，不适合分析型应用

Type	Examples
Key-Value Store	redis riak
Wide Column Store	APACHE HBASE Cassandra

Row Key	Timestamp	Customer		Sales	
Customer Id		Name	City	Product	Amount
101	T1	Suresh	Hyderabad		300
101	T2	Suresh Reddy		Books	
102	T1	Lavya Gavshinde	Indore	Fan	600
102	T2	Lavya			570
102	T3		Bhopal		
103	T1	Anurag	Raipur	Laptop	40000
104	T1	Deepesh	Delhi	Bike	32000

Option2：Parallel database



- Parallel scan + Fast compute

细粒度控制并行计算，适合中小规模数据分析（数据集市）

- Questionable scalability and fault-tolerance
  - Cluster size < 100 data node
  - Not suitable for big batch job

扩展能力有上限
查询内容错能力弱
不适合海量数据分析（企业级数仓）

Option3: Search engine

- All column indexed
- Fast searching
- Simple aggregation

适合多条件过滤，文本分析

- Designed for search but not OLAP
- Not for TopN, join, multi-level aggregation

无法完成复杂计算

- 3~4X data expansion in size

数据膨胀

- No SQL support

专用语法，难以迁移

Option4: SQL on Hadoop

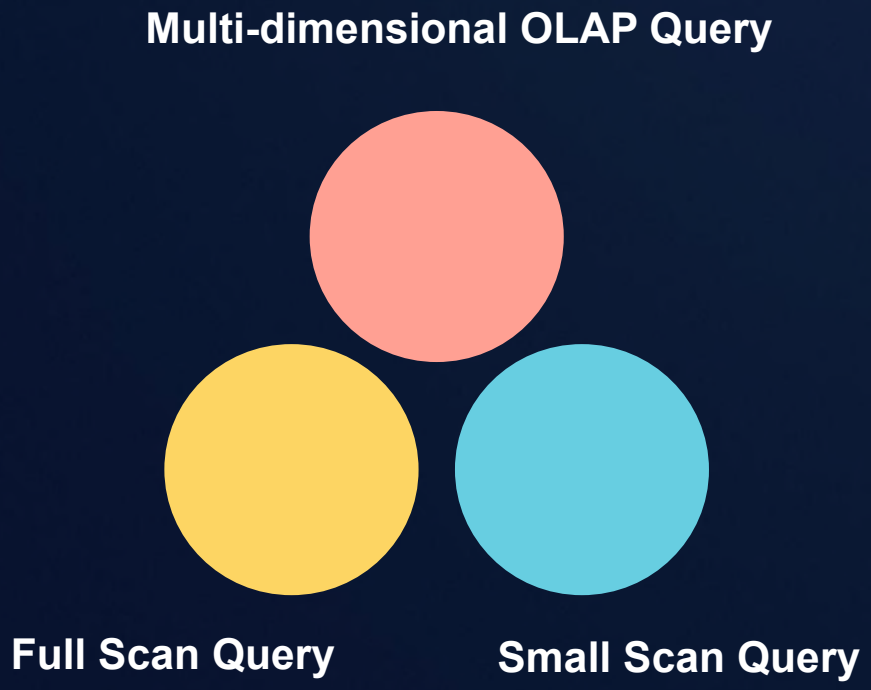
- Modern distributed architecture, scale well in computation.
  - Pipeline based: Impala, Drill, Flink, …
  - BSP based: Hive, SparkSQL

并行扫描+并行计算
适合海量数据计算

- BUT, still using file format designed for batch job
  - Focus on scan only
  - No index support, not suitable for point or small scan queries

仍然使用为批处理设计的存储，场景受限

Capability Matrix



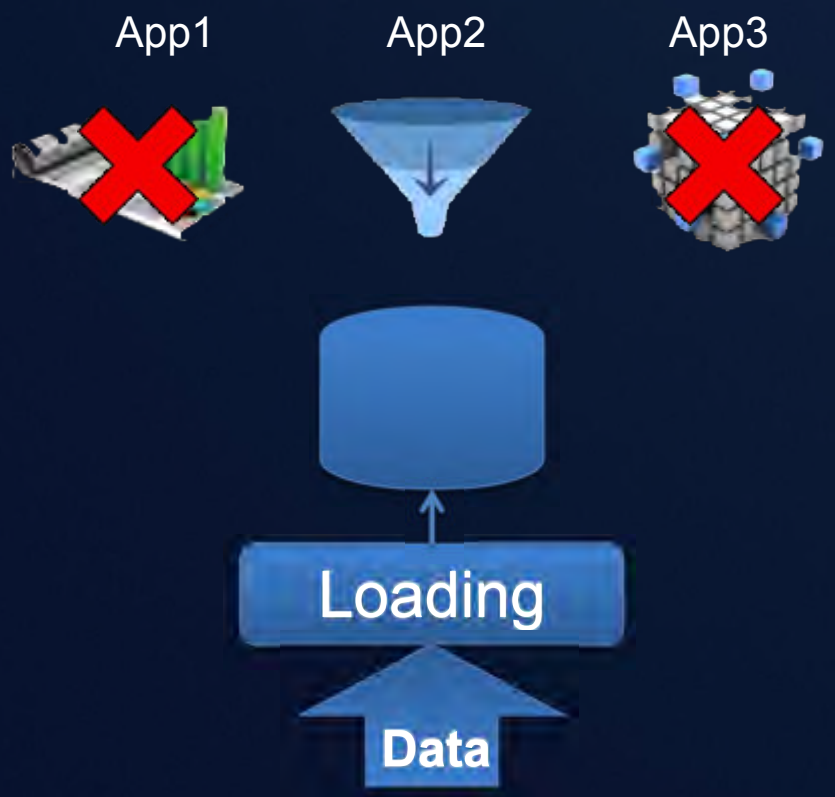
只针对某个场景设计，解决一部分问题

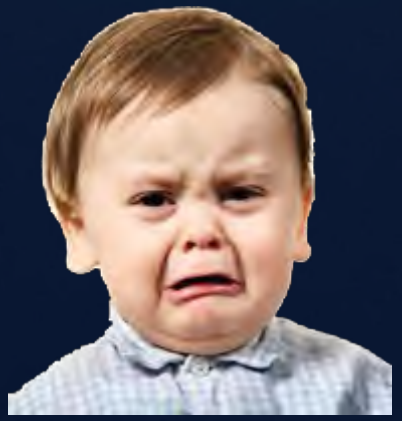
Option	Store	Good	Bad
KV Store	HBase, Cassandra, …		
Parallel database	Greenplum, Vertica, …		
Search engine	Solr, ElasticSearch, …		
SQL on Hadoop - Pipeline	Impala, HAWQ, Drill, …		
SQL on Hadoop - BSP	Hive, SparkSQL		

Architect's choice

Choice 1: Compromising

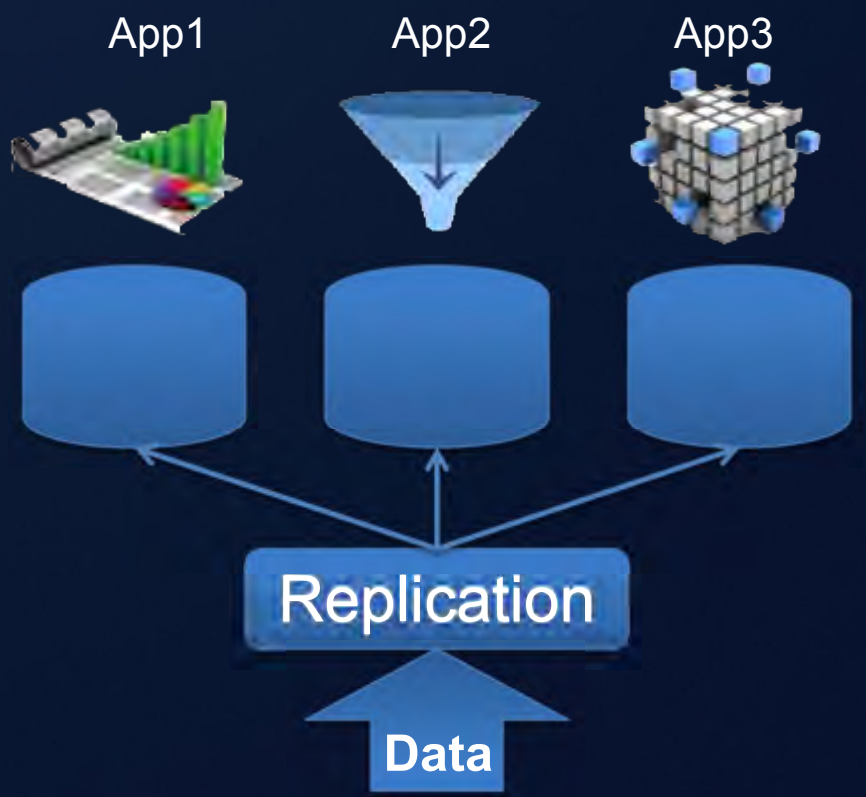
做出妥协，只满足部分应用





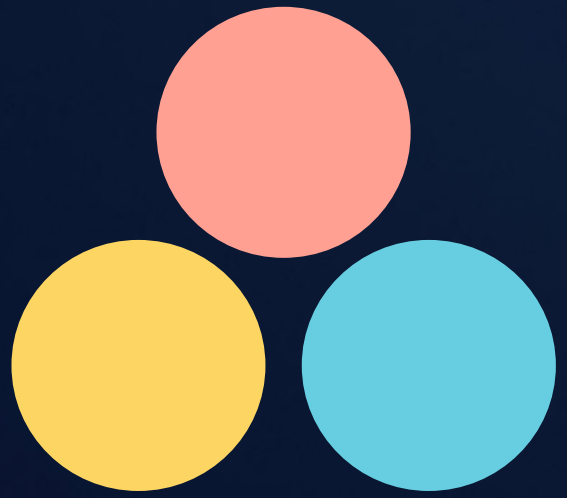
Choice 2: Replicating of data

复制多份数据，满足所有应用



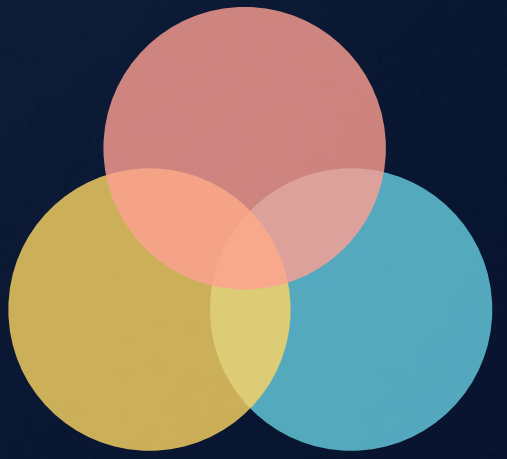
Motivation

Multi-dimensional OLAP Query



Full Scan Query　　Small Scan Query

CarbonData: Unified Storage

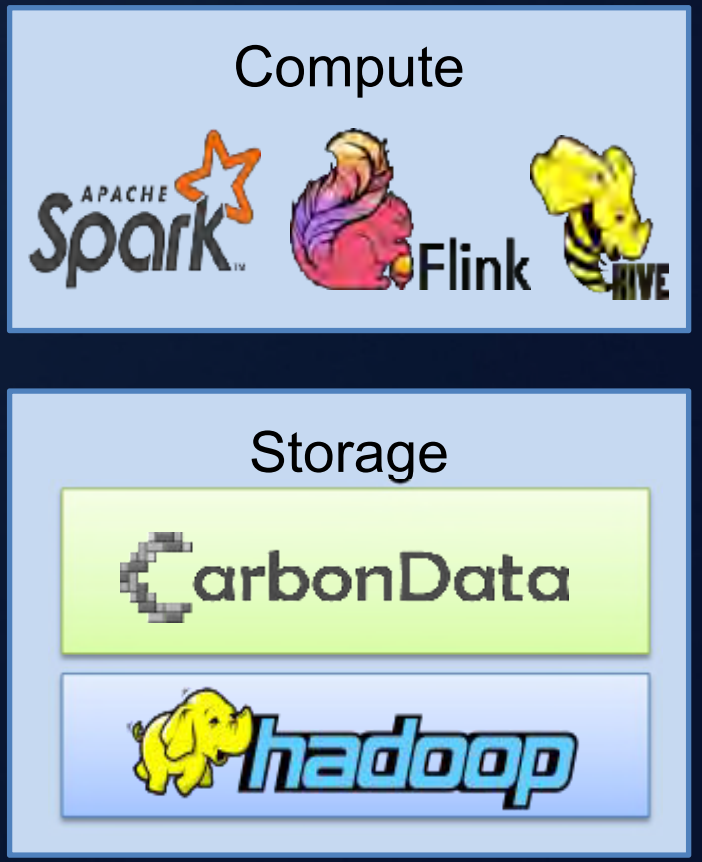


一份数据满足多种分析场景
详单过滤，海量数仓，数据集市，…

Spark + CarbonData:
打造大数据交互式分析引擎

Apache CarbonData社区介绍

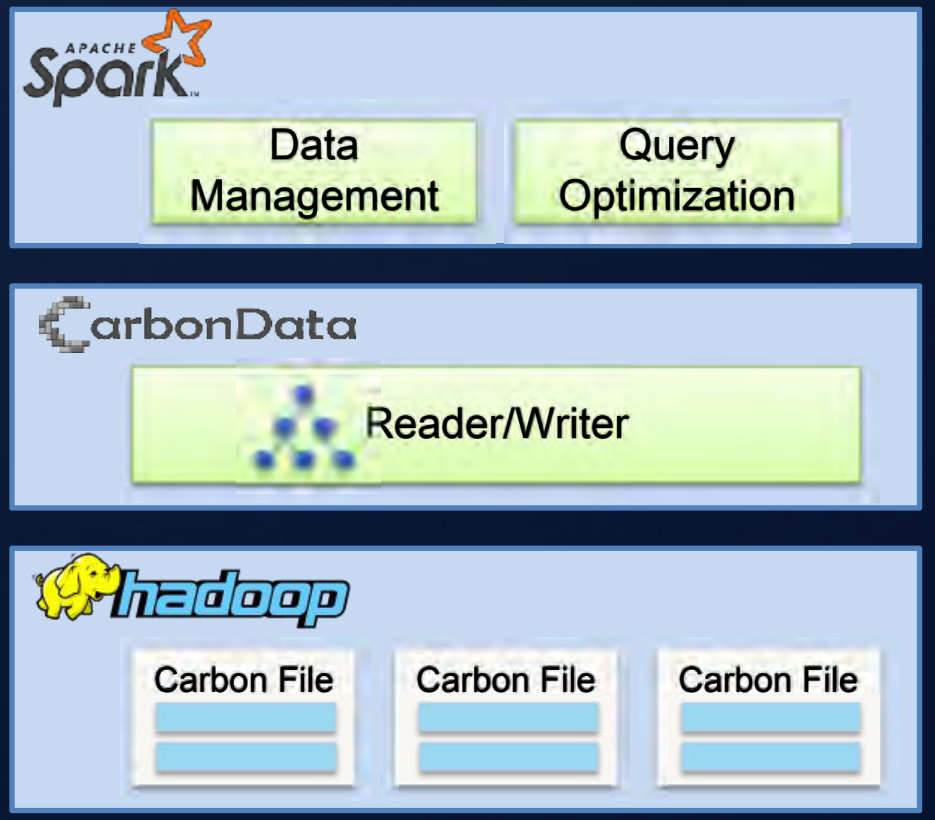
- CarbonData 2016年6月全票通过正式进入Apache孵化器。
- 目标：
  - 更易用，一份存储覆盖更多场景
  - 更高的分析性能，面向用户提供交互式分析
- 已发布了3个Apache稳定版本
- 欢迎订阅邮件列表和贡献:
  - Code: https://github.com/apache/incubator-carbondata
  - JIRA: https://issues.apache.org/jira/browse/CARBONDATA
  - Maillist: dev@carbondata.incubator.apache.org



- 贡献者来自: Huawei, Talend, Intel, eBay, Inmobi, 美团, 阿里, 乐视，Hulu

Carbon-Spark Integration

- Built-in Spark integration
  - Spark 1.5, 1.6 , 2.0
- Interface
  - SQL
  - DataFrame API
- Data Management
  - Bulk load/Incremental load
  - Delete load
  - Compaction



Integration with Spark

- Query CarbonData Table
  - DataFrame API

```
carbonContext.read
             .format("carbondata")
             .option("tableName", "table1")
             .load()
```

With late decode optimization and carbon-specific SQL command support

```
sqlContext.read
          .format("carbondata")
          .load("path_to_carbon_file")
```

  - SQL

```
CREATE TABLE IF NOT EXISTS T1 (name String, PhoneNumber String) STORED BY
"carbondata"

LOAD DATA LOCAL INPATH 'path/to/data' INTO TABLE T1
```

Data Ingestion

- Bulk Data Ingestion
  - Load from CSV file
  - Load from other table

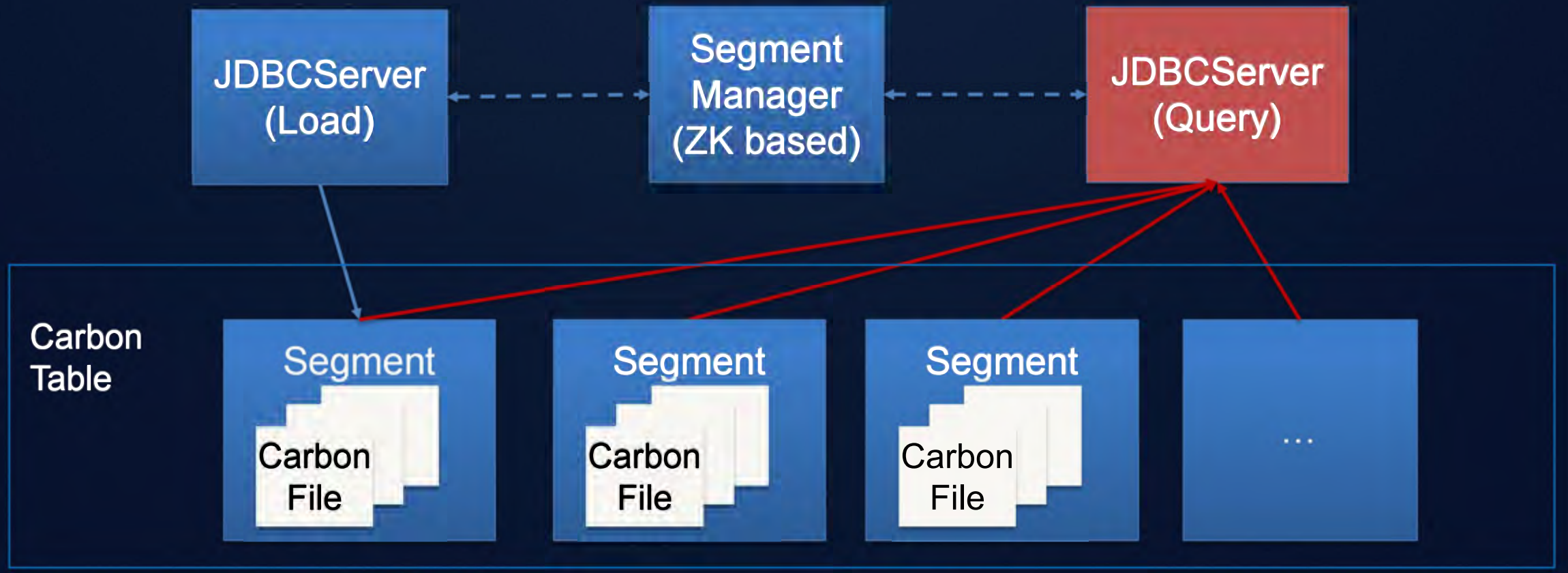
```
LOAD DATA [LOCAL] INPATH 'folder path' [OVERWRITE] INTO TABLE tablename
OPTIONS(property_name=property_value, ...)

INSERT INTO TABLE tablennme  select_statement1 FROM table1;
```

- Save Spark Dataframe as Carbon data file

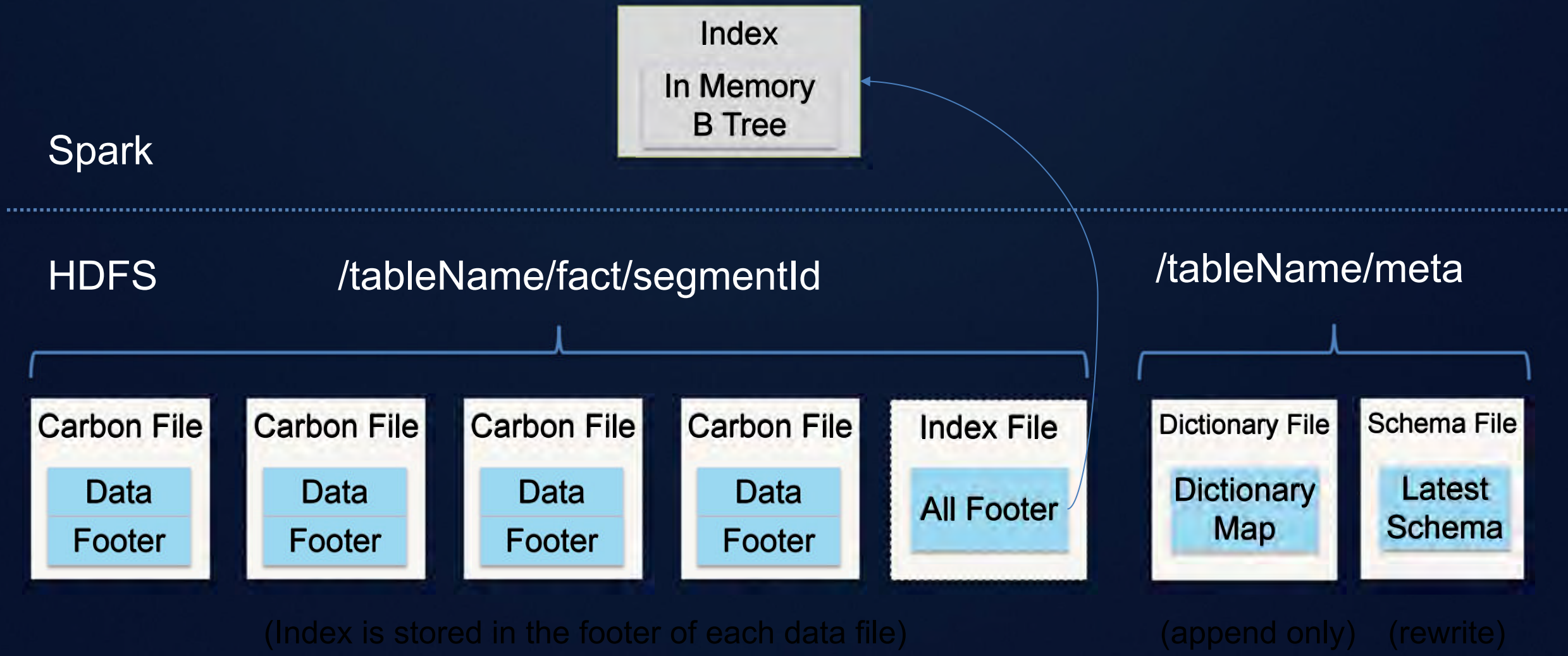
```
df.write
   .format("carbondata")
   .options("tableName", "tbl1"))
   .mode(SaveMode.Overwrite)
   .save()
```

Segment Introduction

Every data load becomes one segment in CarbonData table, data is **sorted** within one segment.



CarbonData Table Organization



Data Compaction

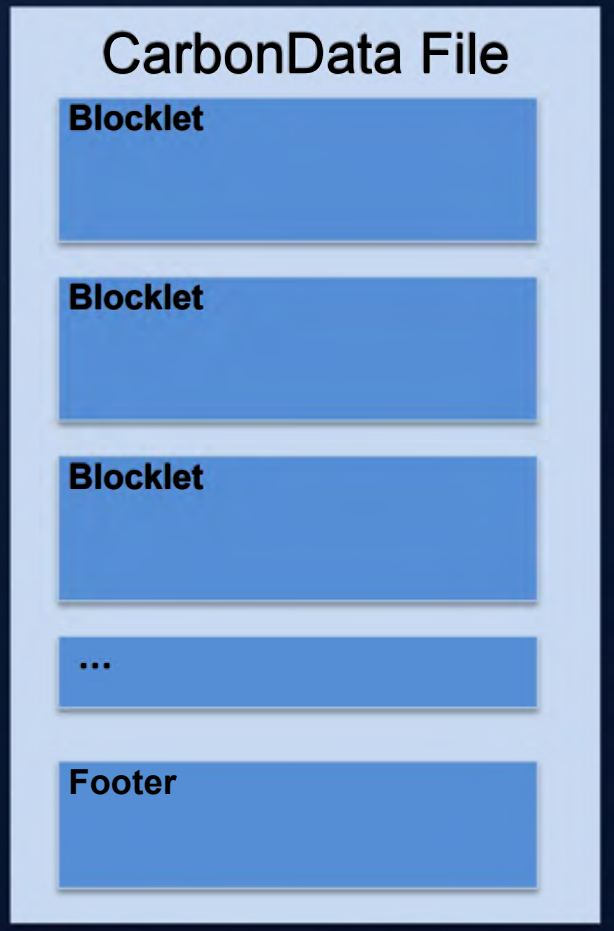
- Data compaction is used to merge small files
  - Re-clustering across loads for better performance

- Two types of compactions supported
  - Minor compaction
    - Compact adjacent segment based on number of segment
    - Automatically trigger
  - Major compaction
    - Compact segments based on size
    - Manually trigger

```
ALTER TABLE [db_name.]table_name COMPACT 'MINOR/MAJOR'
```

CarbonData File Structure

- **Built-in Columnar & Index**
  - Store index and data in the same file, co-located in HDFS
  - Balance between batch and point query
- **Index support:**
  - Multi-dimensional Index (B+ Tree)
  - Min/Max index
  - Inverted index
- **Encoding:**
  - Dictionary, RLE, Delta
  - Snappy for compression
- **Data Type:**
  - Primitive type and nested type
- **Schema Evolution:**
  - Add, Remove, Rename columns



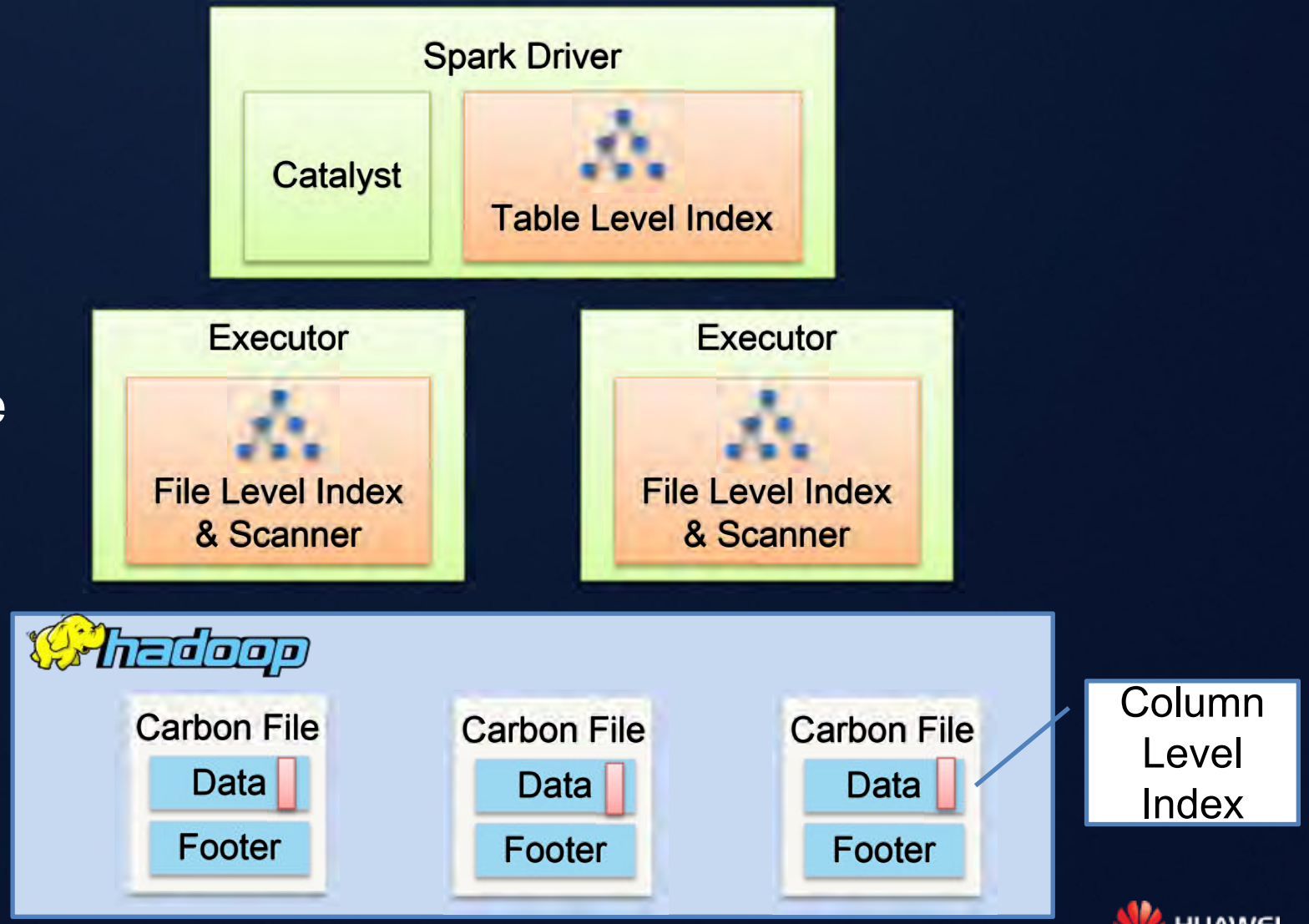
Index Introduction

Multi-level indexes:

•Table level index: global B+ tree index, used to filter blocks

•File level index: local B+ tree index, used to filter blocklet

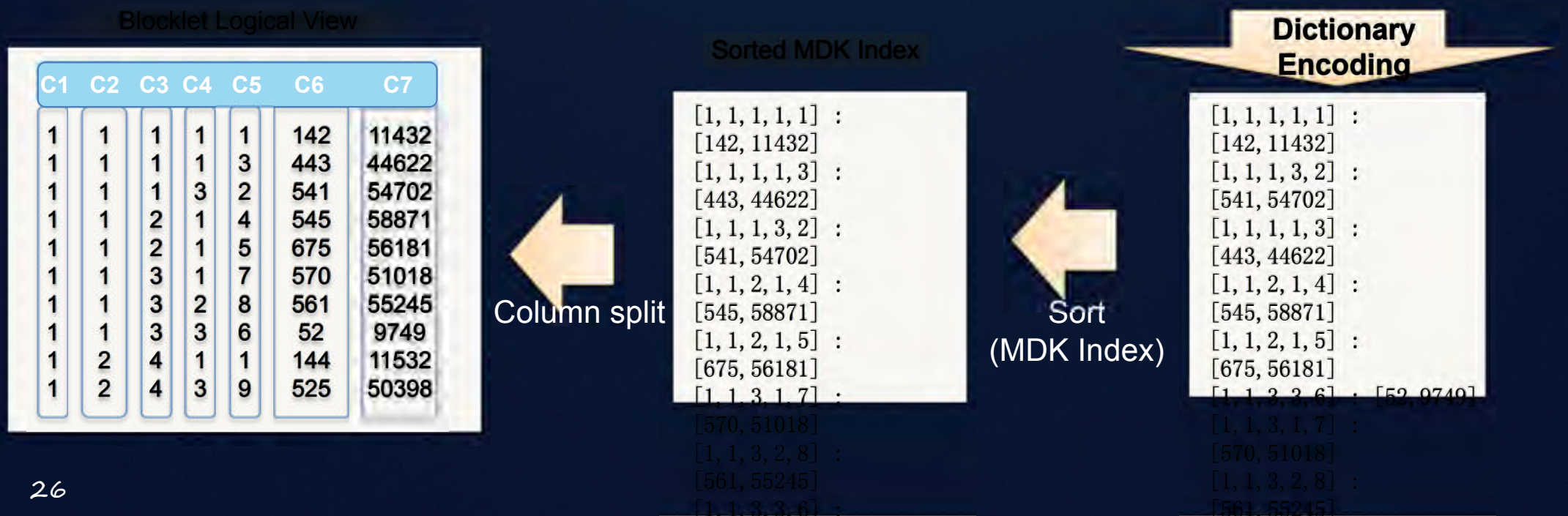
•Column level index: inverted index within column chunk



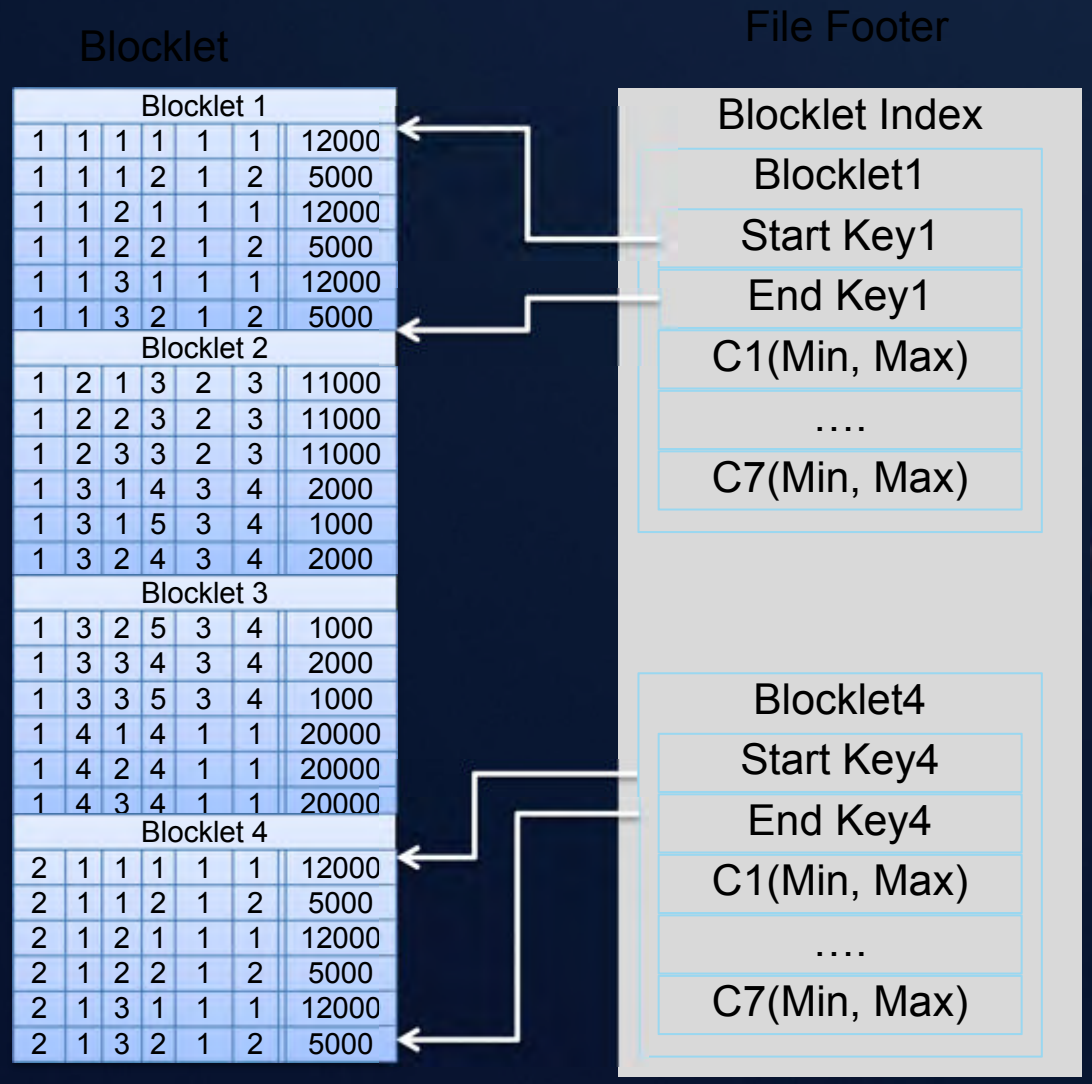
Encoding Example

- Data are sorted along MDK (multi-dimensional keys)
- Data stored as index in columnar format

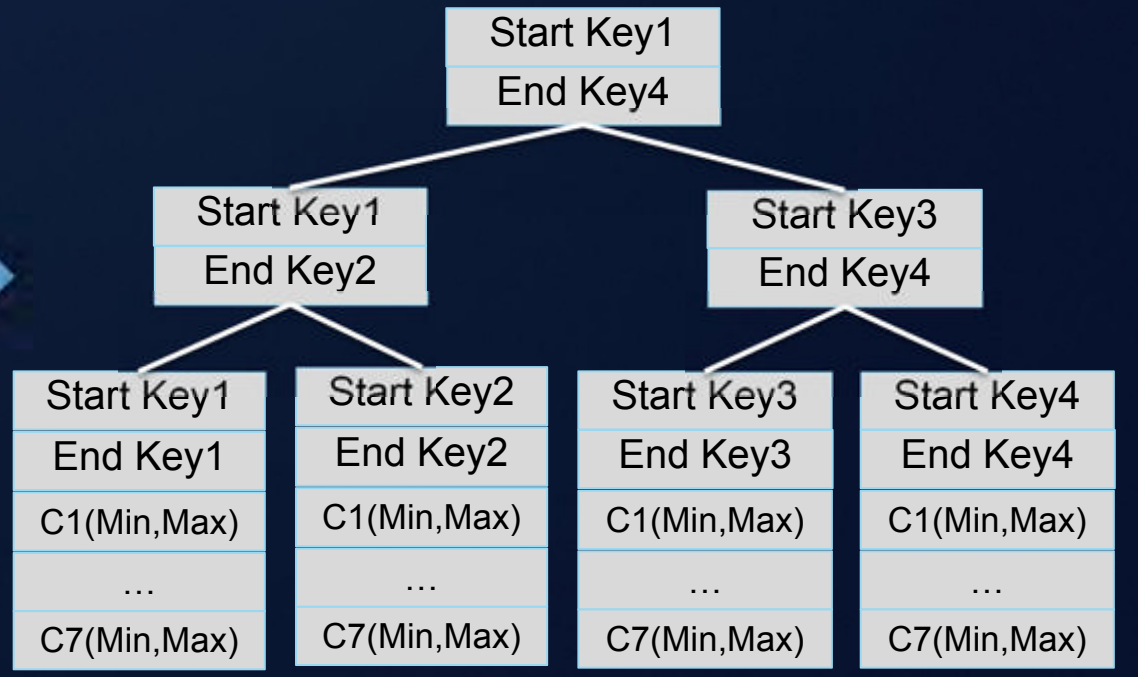
Years	Quarters	Months	Territory	Country	Quantity	Sales
2003	QTR1	Jan	EMEA	Germany	142	11,432
2003	QTR1	Jan	APAC	China	541	54,702
2003	QTR1	Jan	EMEA	Spain	443	44,622
2003	QTR1	Feb	EMEA	Denmark	545	58,871
2003	QTR1	Feb	EMEA	Italy	675	56,181
2003	QTR1	Mar	APAC	India	52	9,749
2003	QTR1	Mar	EMEA	UK	570	51,018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



File Level Blocklet Index

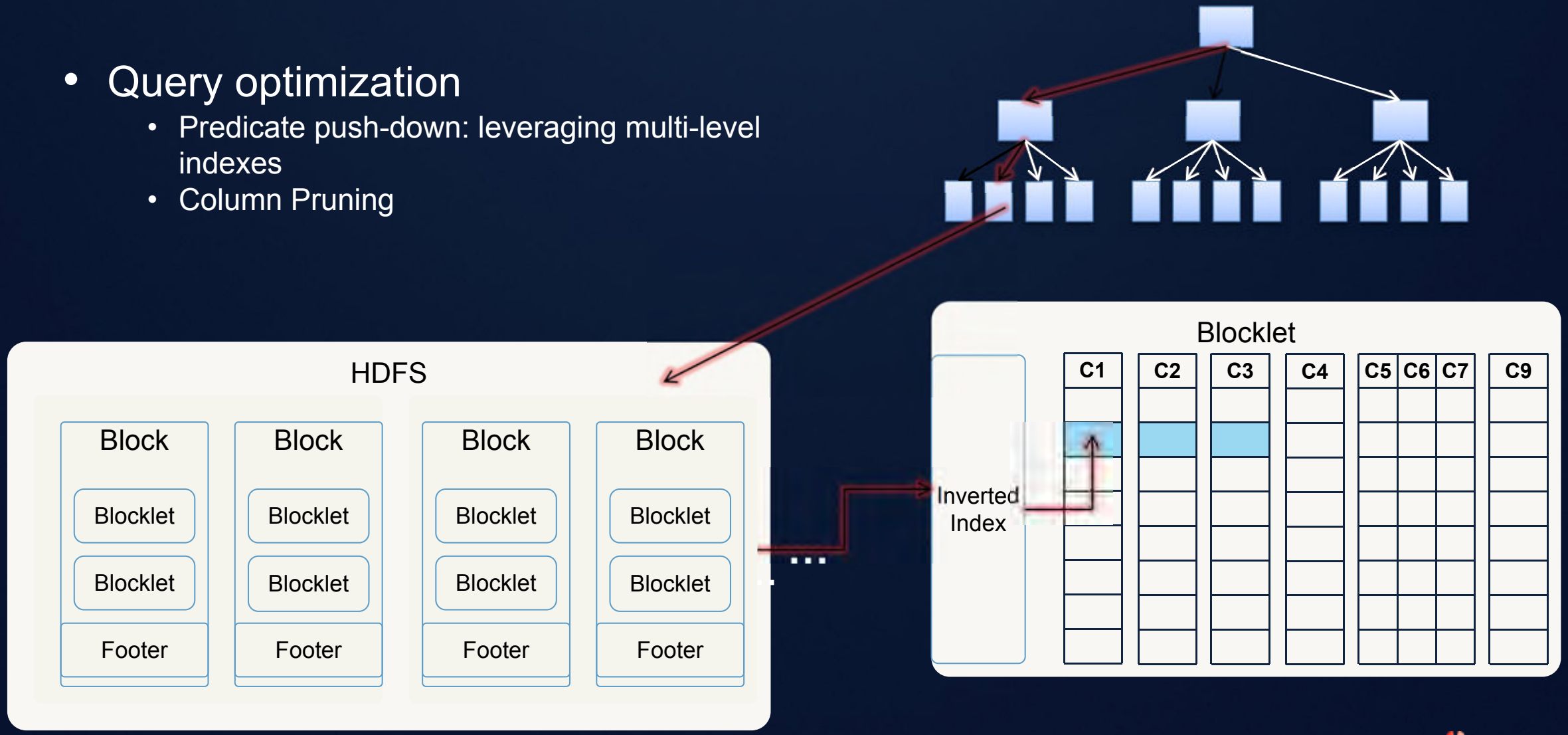


- Build in-memory file level MDK index tree for filtering
- Major optimization for efficient scan



Block Pruning

- Query optimization
  - Predicate push-down: leveraging multi-level indexes
  - Column Pruning



Column Chunk Inverted Index

- Optionally store column data as inverted index within column chunk
  - suitable to low cardinality column
  - better compression & fast predicate filtering

Blocklet
(sort column within column chunk)

[1|1]	[1|1]	[1|1]	[1|1]	[1|1]	: [142]:[11432]
[1|2]	[1|2]	[1|2]	[1|2]	[1|9]	: [443]:[44622]
[1|3]	[1|3]	[1|3]	[1|4]	[2|3]	: [541]:[54702]
[1|4]	[1|4]	[2|4]	[1|5]	[3|2]	: [545]:[58871]
[1|5]	[1|5]	[2|5]	[1|6]	[4|4]	: [675]:[56181]
[1|6]	[1|6]	[3|6]	[1|9]	[5|5]	: [570]:[51018]
[1|7]	[1|7]	[3|7]	[2|7]	[6|8]	: [561]:[55245]
[1|8]	[1|8]	[3|8]	[3|3]	[7|6]	: [52]:[9749]
[1|9]	[2|9]	[4|9]	[3|8]	[8|7]	: [144]:[11532]
[1|10]	[2|10]	[4|10]	[3|10]	[9|10]	: [525]:[50398]

Run Length Encoding & Compression

Column chunk Level inverted Index

Blocklet Physical View

C1		C2		C3		C4		C5		C6	C7
d	r	d	r	d	r	d	r	d	r	d	r
1	1	1	1	1	1	1	1	1	1	1	142	11432
10	10	8	10	3	10	6	2	2	9	4	443	44622
		2		2		2	4	2	1	1	541	54702
		2		2		1	3	1	3	5	545	58871
				3		3	9	3	1	4	675	56181
				3		3	1	1	2	3	570	51018
				4			7	4	1	2	561	55245
				2			1	1	4	1	52	9749
							3	5	1	1	144	11532
							1	1		1	525	50398
							…	…		…		

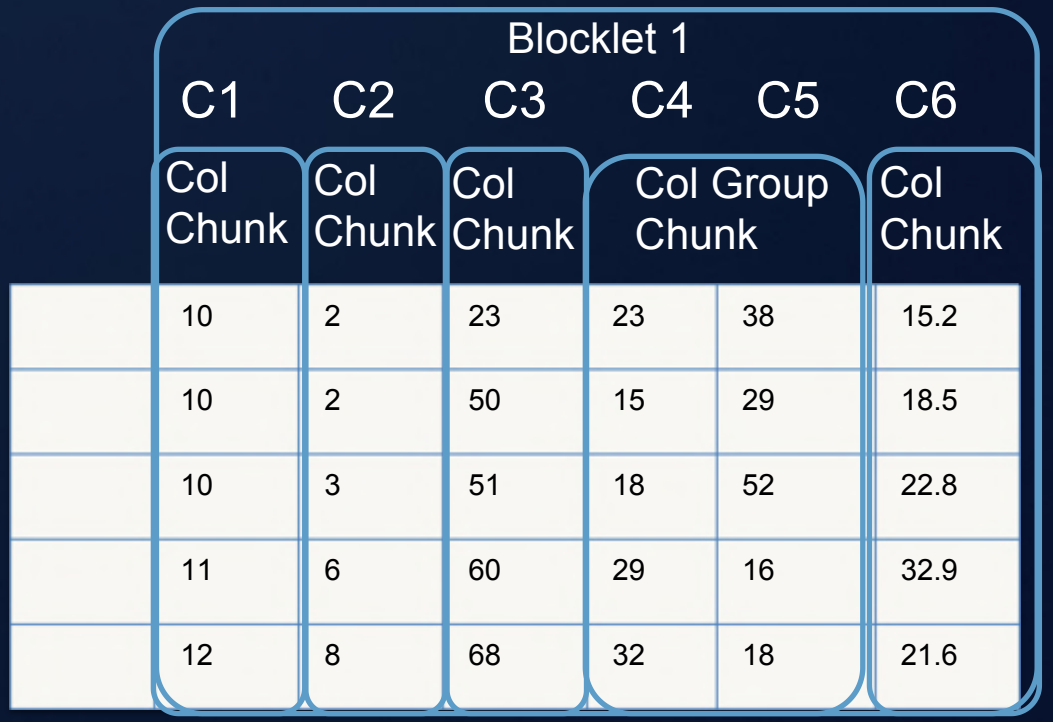
Columnar Store

Blocklet Rows

Dim1 Block	Dim2 Block	Dim3 Block	Dim4 Block	Dim5 Block	Measure1 Block	Measure2 Block
1(1-10)	1(1-8)	1(1-3)	1(1-2,4-6,9)	1(1,9)	[142]:[11432]	
	2(9-10)	2(4-5)	2(7)	2(3)	[443]:[44622]	
		3(6-8)	3(3,8,10)	3(2)	[541]:[54702]	
		4(9-10)		4(4)	[545]:[58871]	
				5(5)	[675]:[56181]	
				6(8)	[570]:[51018]	
				7(6)	[561]:[55245]	
				8(7)	[52]:[9749]	
				9(10)	[144]:[11532]	
					[525]:[50398]	

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



Nested Data Type Representation

Arrays

- Represented as a composite of two columns
- One column for the element value
- One column for start index & length of Array

Struts

- Represented as a composite of finite number of columns
- Each struct element is a separate column

Name	Array<Ph_Number>
John	[192,191]
Sam	[121,345,333]
Bob	[198,787]

Name	Array [start,len]	Ph_Number
John	0,2	192
Sam	2,3	191
Bob	5,2	121
		345
		333
		198
		787

Name	Info Strut<age,gender>
John	[31,M]
Sam	[45,F]
Bob	[16,M]

Name	Info.age	Info.gender
John	31	M
Sam	45	F
Bob	16	M

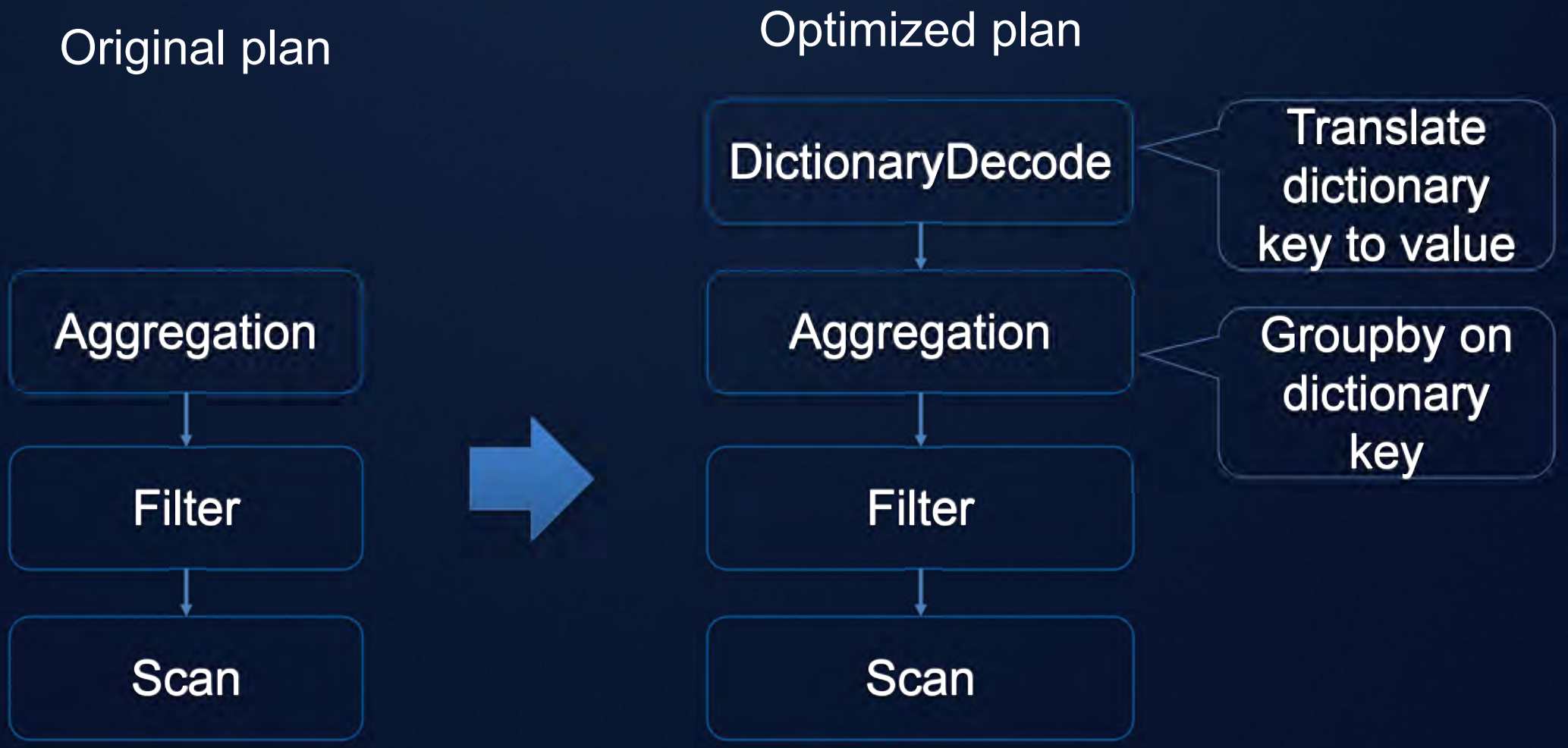
Encoding & Compression

- Efficient encoding scheme supported:
  - DELTA, RLE, BIT_PACKED
  - Dictionary:
    - table level global dictionary -> Enable Lazy Decode optimization
- Compression:
  - Column data compression: Snappy
  - Adaptive Data Type Compression

Big Win:

- Speedup Aggregation
- Reduce run-time memory footprint
- Enable fast distinct count

Lazy Decode



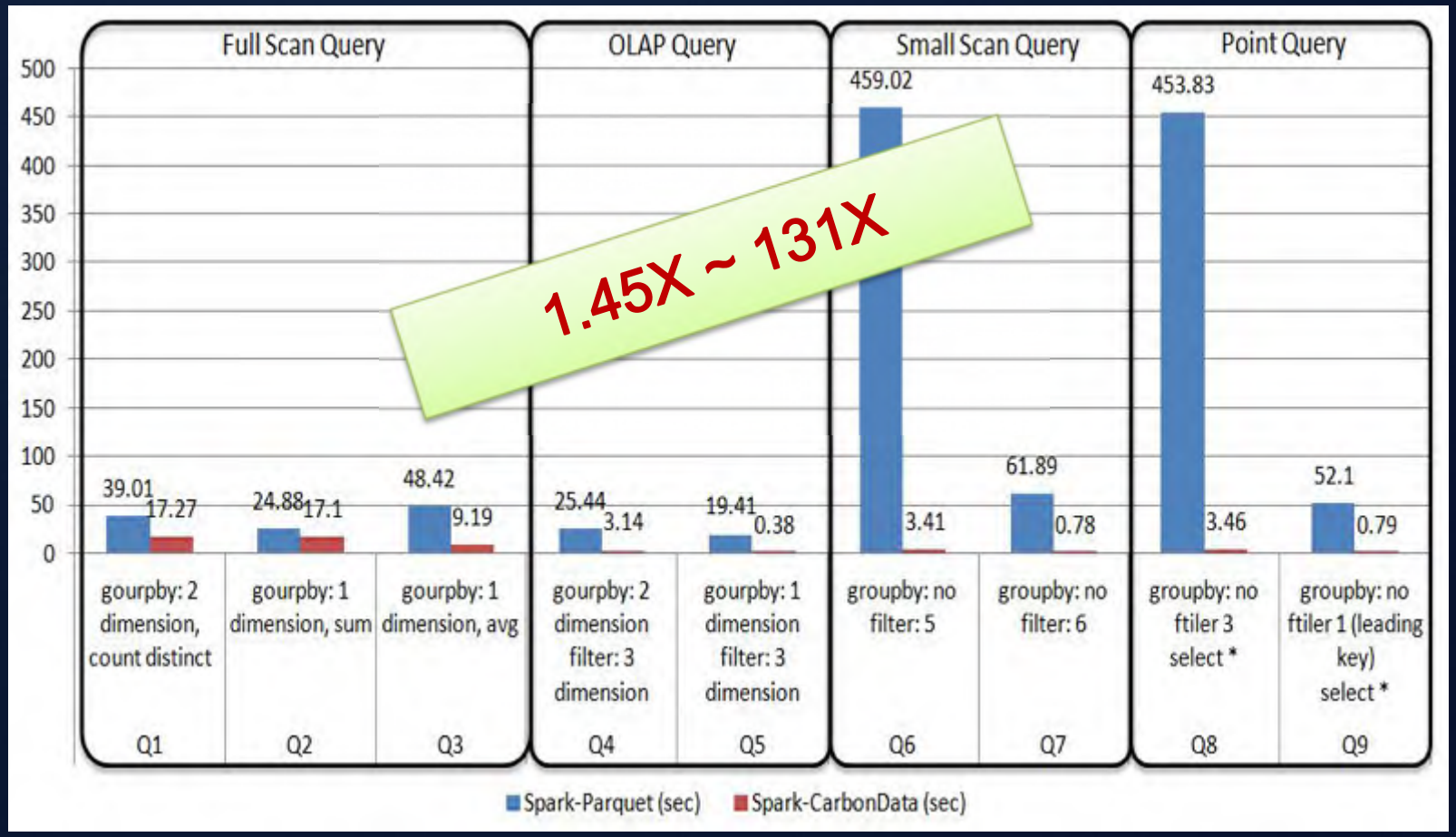
CarbonData性能对比（Spark-Parquet）

测试环境

- 集群：3(Worker)+1(Master)，40核, 384GB, 10G网络带宽
- 软件：Hadoop 2.7.2，Spark 1.5.2
- 数据：10亿记录，300列，原始数据1.9TB

查询特点

- Point Query：基于主key过滤
- Small Scan：包含多个列过滤
- Full Scan：复杂聚合、Join，无过滤条件
- OLAP Query：同时带过滤，聚合



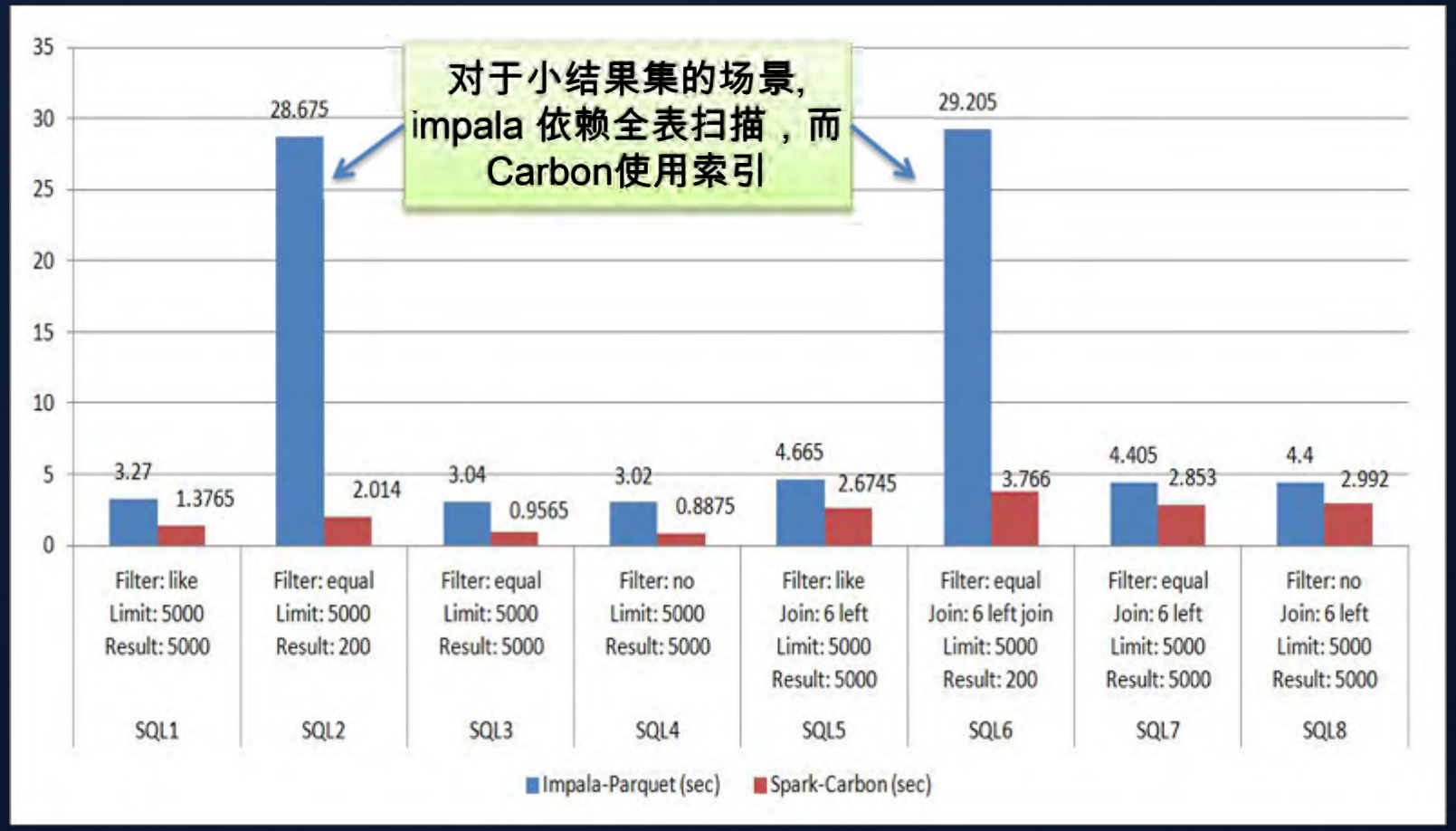
CarbonData性能对比（Impala）

测试环境

- 集群：3(Worker)+1(Master)，40核, 384GB, 10G网络带宽
- 软件：Hadoop 2.7.2，Spark 1.5.2，Impala 2.6
- 数据：10亿记录，300列，原始数据830G

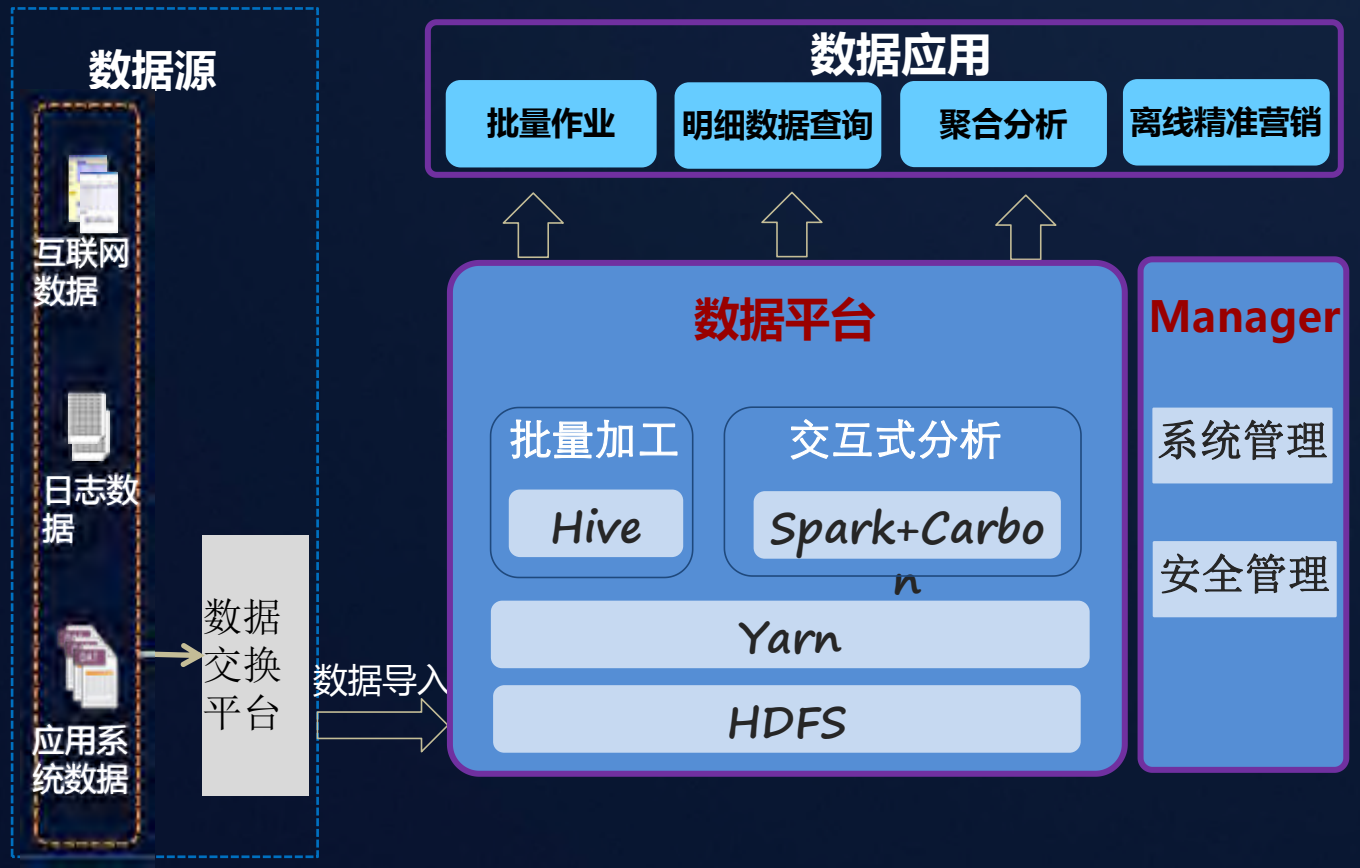
查询特点

- 多维度过滤
- 多个Join：与多个维表



Success Case
应用案例

金融交互式分析：提供高性能交互式分析体验



老系统：开源Hive+Impala+Parquet，35台服务器

客户挑战

- Impala查询**性能**（>700s,且极易查询失败）
- 交互查询**稳定性**问题（impala存在挂死情况）
- impala**资源不能统一管理，无法共享**

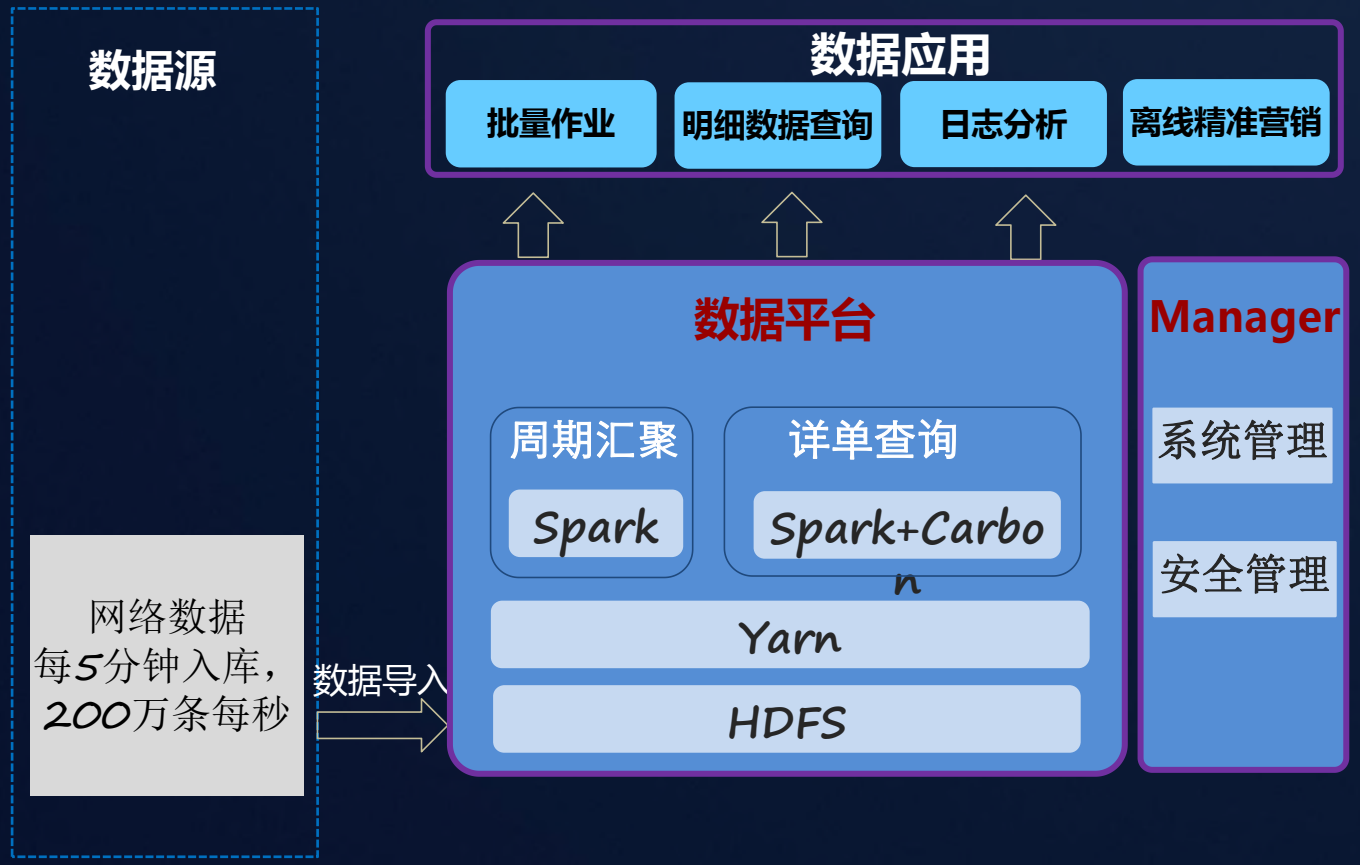
应对方案

- 批量加工： Hive
- 交互式分析： SparkSQL+CarbonData

客户价值

- 资源采用Yarn统一管理，用户可配，可调
- 百亿-千亿数据，秒级响应（半年查询<10秒，1年查询<20秒）

电信详单分析：开源系统无法满足业务性能和稳定性要求



老系统：开源Hive+Impala+Parquet，98台服务器

客户挑战

- 按手机号码做分区，分区多，导致小文件问题
- 非手机号码查询时，查询**性能较差**，并发下响应慢
- Impala**资源不能统一管理，无法共享**

应对方案

- 周期汇聚： Spark
- 详单查询： SparkSQL+CarbonData（每5分钟入库一批数据，每X批自动compaction）

客户价值

- 资源采用Yarn统一管理，用户可配，可调
- 百亿-千亿数据，任意维度过滤查询，秒级响应

What's Next
下一步计划

What's coming next: functionality

- Batch Update:
  - Support daily update scenario for OLAP, Not OLTP!
- Streaming Ingest:
  - Introduce row-based format for fast ingestion
  - Compaction from row to columnar format
- Broader Integration across Hadoop-ecosystem: Flink, Kafka, Kylin

What's coming next: performance

- Better query performance by Spark 2.X integration
  - Whole stage codegen
  - Vectorized reader
- Better join performance by bucket table
- Better loading performance
  - Offheap sort
  - Single-pass loading (on-the-fly dictionary generation)

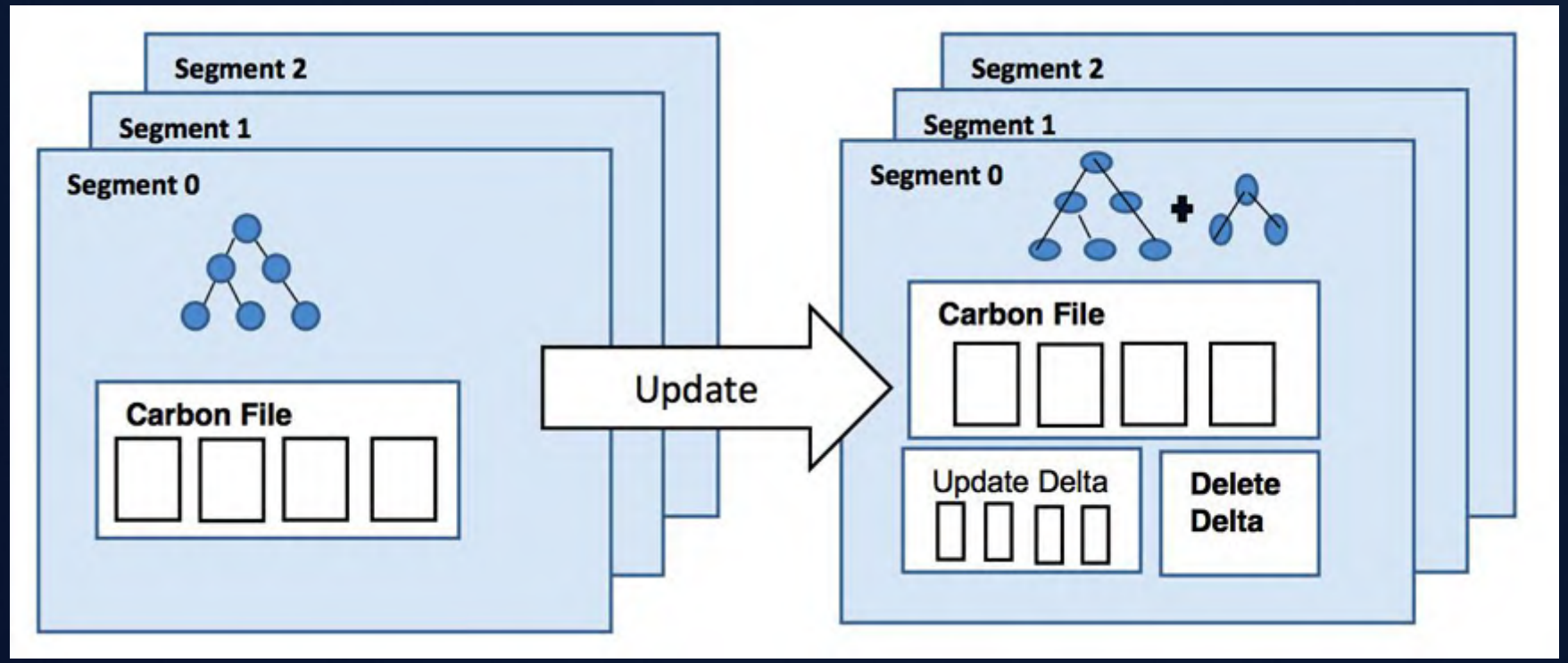
数据更新（预览）

Data Update/Delete（beta）

- 场景：
  - 缓慢变化的维表，如刷新用户信息
  - 批量更新的事实表，如更新某些度量值，ID字符串

- ACID属性：
  - 支持更新语句原子性（要么成功要么失败），一致性（更新成功后马上可查）
  - 更新过程中不影响查询
  - 不支持并行更新，不支持OLTP类应用

Data Update/Delete（beta）



Delete Delta文件:
记录要删除的行号，bitmap文件。

Update Delta文件：
记录要插入的记录，Carbon文件。

更新流程：

1. 找到要更新的Segment和目标文件
2. 找到要更新的行
3. 写入Delete Delta和Update Delta文件

读取流程：

1. 读取Base文件
2. 排除Delete记录（RowID）
3. 加入Update Delta（新记录）

- 计划在1月份CarbonData 1.0中发布

- JIRA: CARBONDATA-440
- issues.apache.org/jira/browse/CARBONDATA-440
- github.com/apache/incubator-carbondata