SACC 2016 第八届中国系统架构师大会
SYSTEM ARCHITECT CONFERENCE CHINA 2016

架构创新之路

THE APACHE® SOFTWARE FOUNDATION

CarbonData

实现大数据即席查询秒级响应

Email:chenliang613@apache.org

Liang Chen / 陈 亮

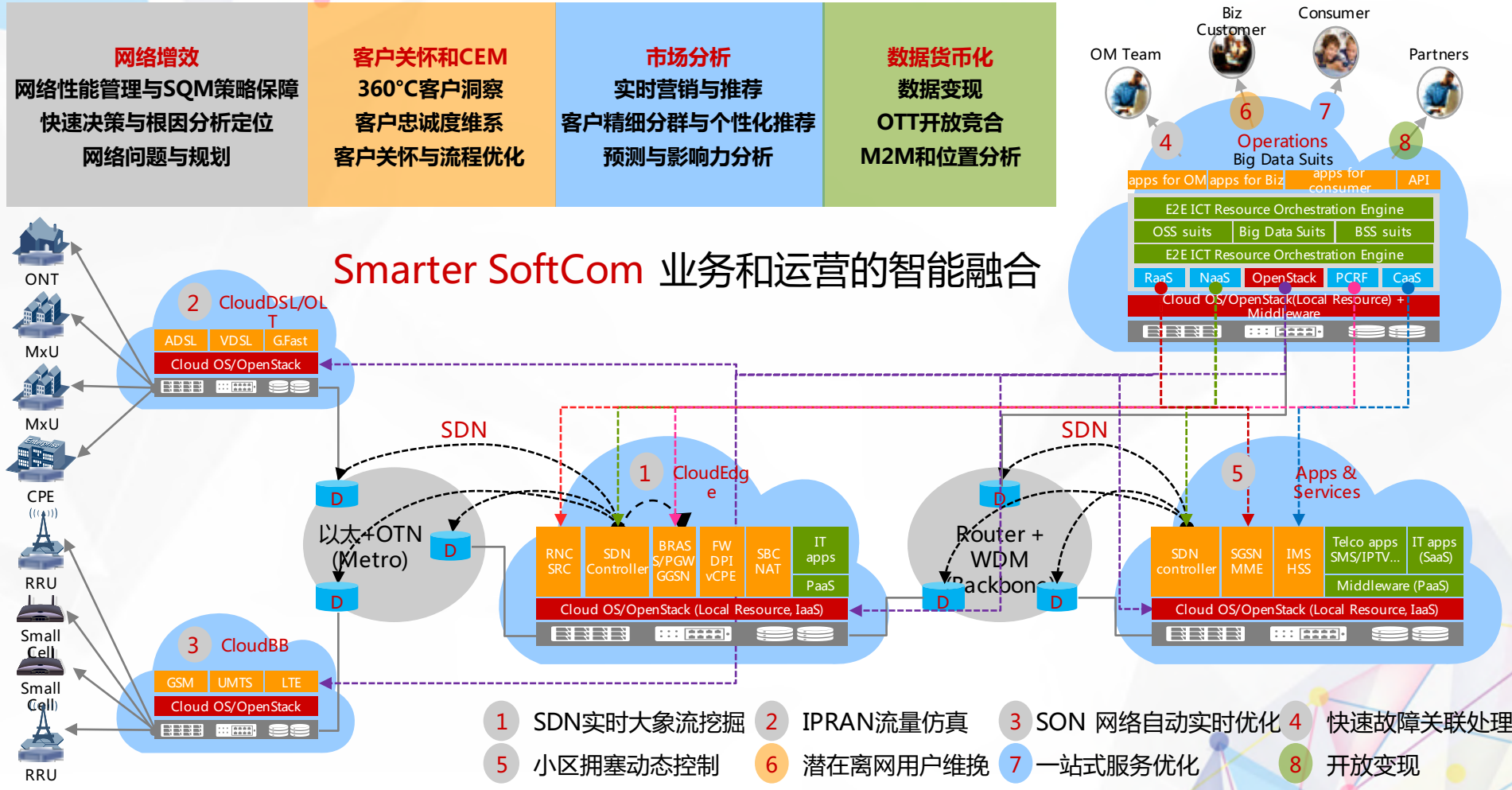
华为大数据开源开发部Leader

Apache CarbonData PMC & Committer

10多年大数据和BI项目开发和实践经验，对大数据开源技术(Hadoop,Spark,CarbonData等)有深入理解.

大数据现在和未来将深刻的改变运营商

网络增效	客户关怀和CEM	市场分析	数据货币化
网络性能管理与SQM策略保障	360°客户洞察	实时营销与推荐	数据变现
快速决策与根因分析定位	客户忠诚度维系	客户精细分群与个性化推荐	OTT开放竞合
网络问题与规划	客户关怀与流程优化	预测与影响力分析	M2M和位置分析



1 SDN实时大象流挖掘　2 IPRAN流量仿真　3 SON 网络自动实时优化　4 快速故障关联处理

5 小区拥塞动态控制　6 潜在离网用户维挽　7 一站式服务优化　8 开放变现

How to choose storage for complex big data requirements?

NoSQL Database

- Key-Value store: low latency, <5ms
- Can not support multi-dimension query

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

Key	Value
K1	AAA,BBB,CCC
K2	AAA,BBB
K3	AAA,DDD
K4	AAA,2,01/01/2015
K5	3,ZZZ,5623

Multi-dimensional problem

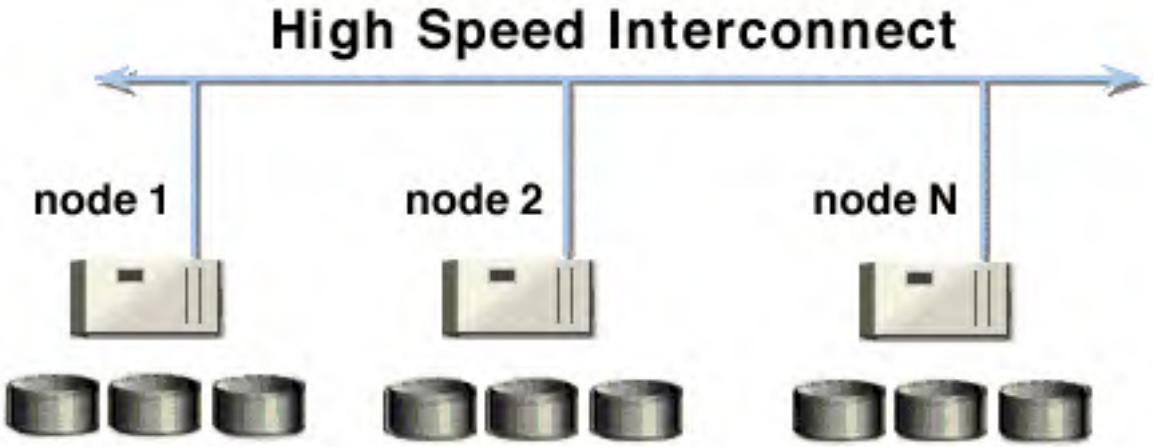
- Pre-compute all aggregation combinations

- Complexity: O(2^n)
- Dimension < 10
- Too much space
- Slow loading speed

- Row Keys: identify the rows in an HBase table.

	Row Key	CF1			CF2				...
		colA	colB	colC	colA	colB	colC	colD	
R1	axxx	val		val	val			val	
	...								
	gxxx	val			val	val	val		
R2	hxxx	val	val	val	val	val	val	val	
	...								
	jxxx	val							
R3	kxxx	val		val	val			val	
	...								
	rxxx	val	val	val	val	val	val		
...	sxxx	val						val	

Shared nothing database



- Parallel scan + distributed compute
- Questionable scalability and fault-tolerance
  - Cluster size < 100 data node
  - Not suitable for big batch job
  - Can not integrate with Hadoop ecosystem

Search engine

- All column indexed
- Fast searching
- Simple aggregation

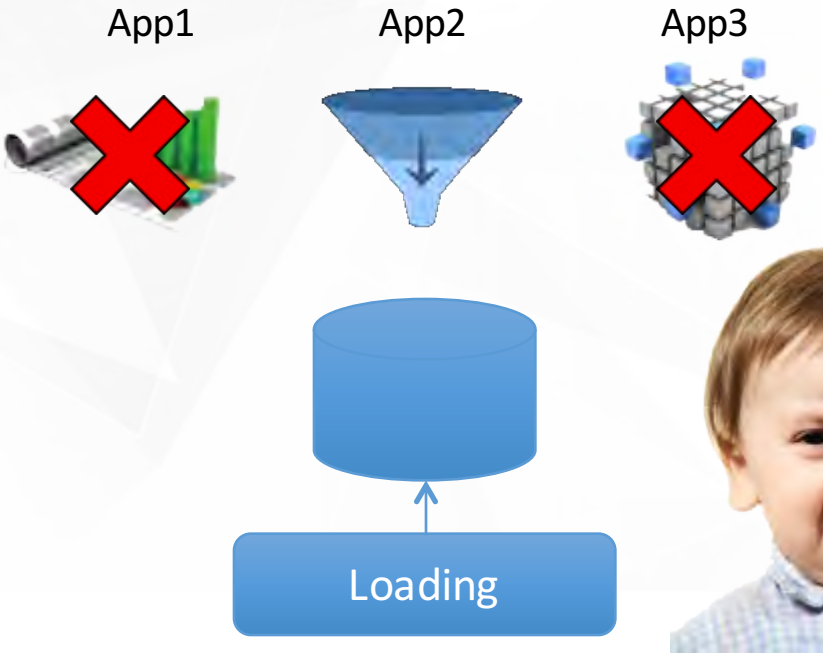
- Designed for search but not OLAP
  - complex computation: TopN, join, multi-level aggregation
- No SQL support

SQL on Hadoop

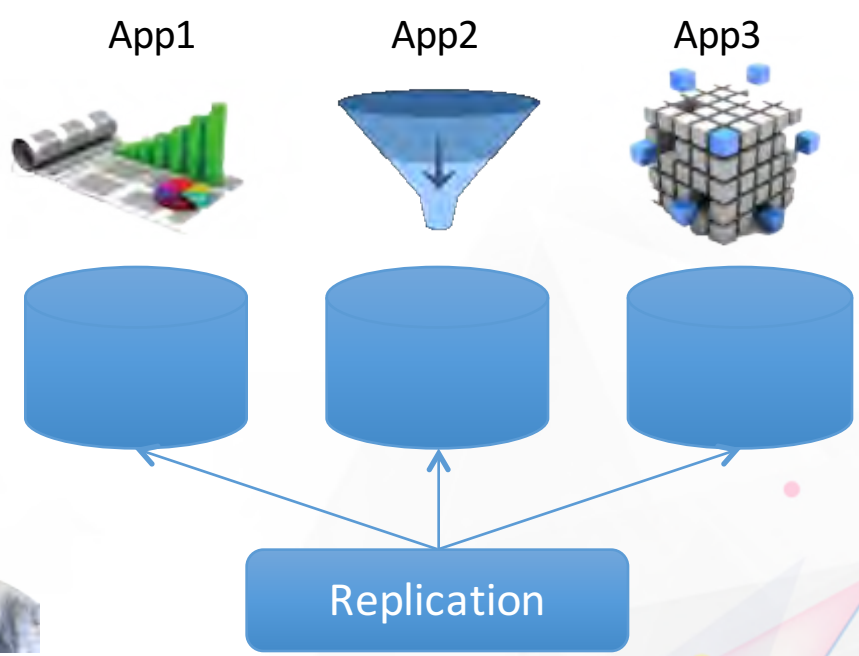
- Modern distributed architecture, scale well in computation.
  - Pipeline based: Impala, Drill, Flink, …
  - BSP based: Hive, SparkSQL
- BUT, still using file format designed for batch job
  - Focus on scan only
  - No index support, not suitable for point or small scan queries

Architect’s choice

Choice 1: Compromising



Choice 2: Replicating of data



目录:

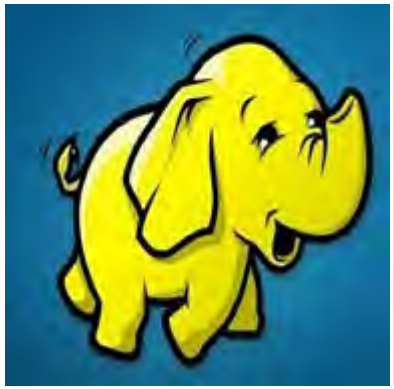
- **CarbonData项目背景和适合的场景**
- 关键技术介绍
- 性能和DEMO演示
- Apache CarbonData社区和路标

客户需求：

多维组合即席分析

详单过滤查询

按列扫描查询



开源生态集成

当前大数据生态系统，没有一种存储方式同时满足上面所有的需求！

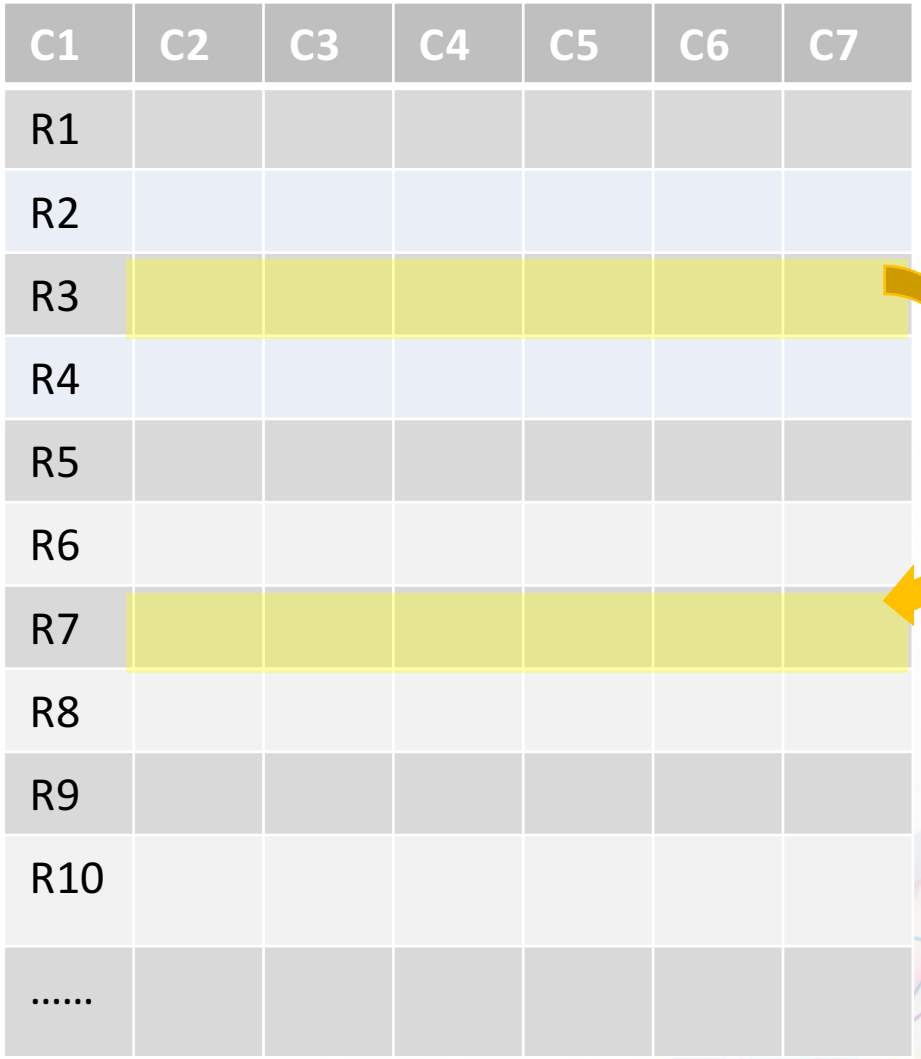
典型场景1：按列扫描查询

- 按列扫描查询（Full Scan）：
  - 没有过滤条件，仅仅做汇总计算等
  - 只查询几列信息
- 典型的场景如：
  - 数据清洗处理
  - 日志分析

C1	C2	C3	C4	C5	C6	C7
R1						
R2						
R3						
R4						
R5						
R6						
R7						
R8						
R9						
R10						
.....						

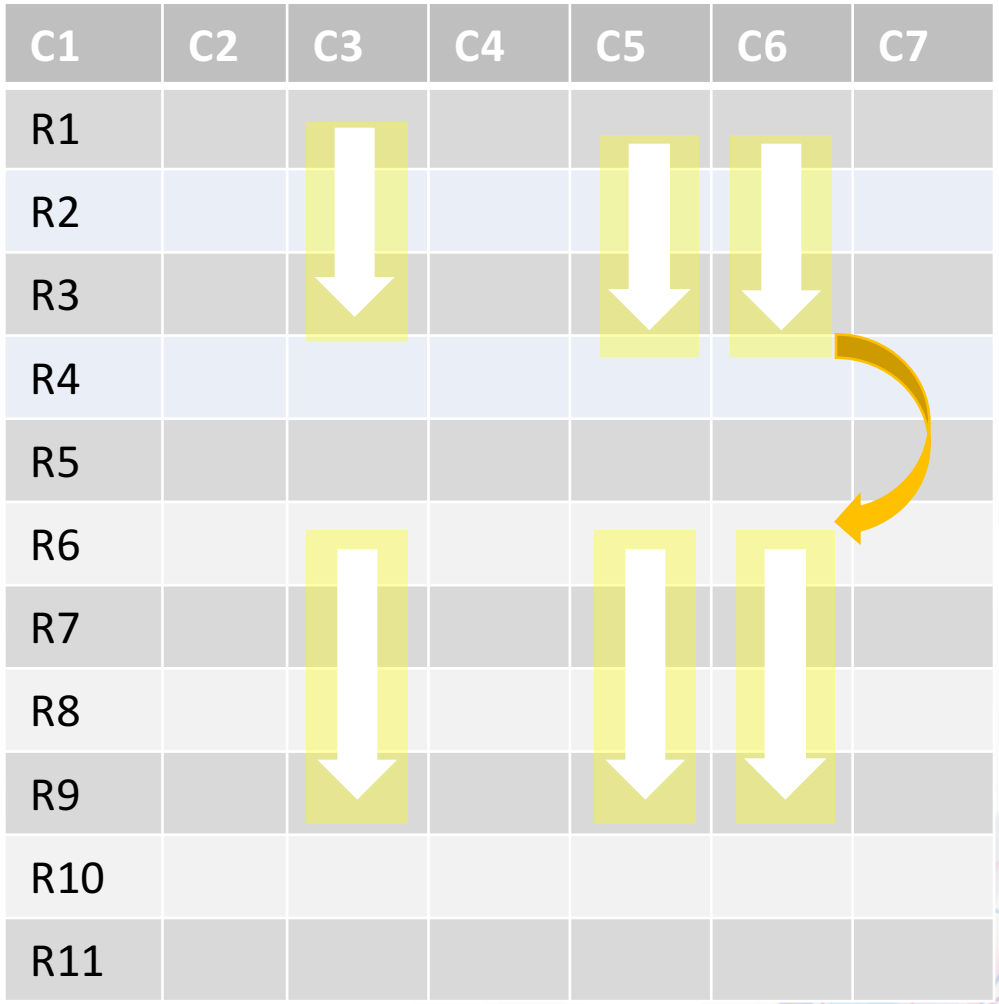
典型场景2：详单过滤查询

- 详单过滤查询（Small Scan）：
  - 按关键字快速过滤查询 (类似HBase)
  - 多组过滤条件组合，查询所有列
  - 要求查询性能秒级响应
- 典型的场景如：
  - 运维查询
  - 用户行为分析



典型场景3：多维组合即席分析

- 即席分析/Adhoc查询：
  - 汇总计算
  - 多维度组合OLAP分析
  - 低时延即席查询
- 典型的场景如：
  - Dash-Board报表
  - Ad-hoc分析



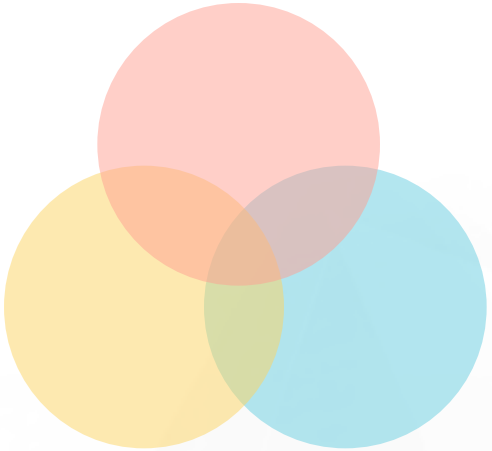
为什么开始CarbonData项目?

多维组合即席分析
(OLAP analysis)

按列扫描查询
(Full scan)

详单过滤查询
(Small scan)

CarbonData
(一份数据满足所有cases)



Apache CarbonData实现一份数据同时满足多种业务需求， 与Spark引擎对接后形成一套分布式多维分析解决方案。

目录:

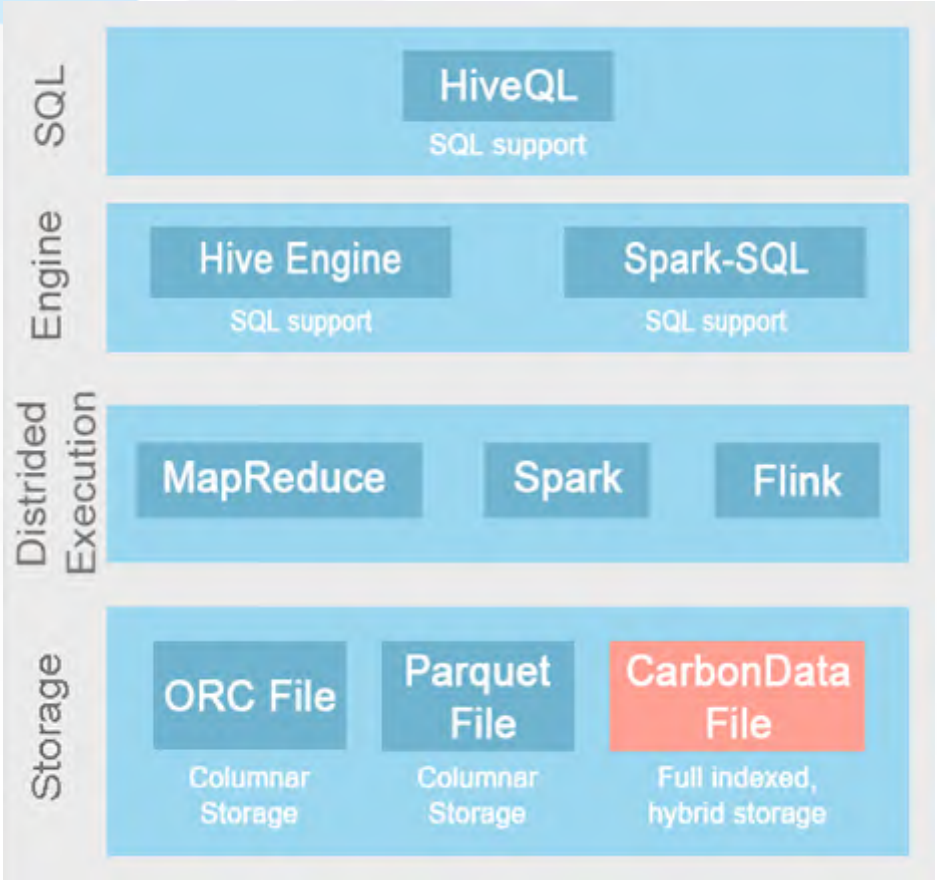
- CarbonData项目背景和适合的场景
- **关键技术介绍**
- 性能和DEMO演示
- Apache CarbonData社区和路标

CarbonData设计思路

- 分布式能力
- 快速查询秒级响应
- 高效数据存储方式
- 无缝与大数据生态集成

开源是为了构建生态，CarbonData是数据存储层技术，要发挥价值，需要与计算层、查询层有效集成在一起，形成E2E生态发挥最大价值。

CarbonData独特的价值特性



- ❖多种索引(MDK,MinMax,倒排)，快速找到目标数据
- ❖字典编码，减少计算开销
- ❖支持数据更新IUD(开发中ing)
- ❖与大数据生态无缝集成，具有HDFS分布式、可靠性等所有优点

多维Key索引介绍

数据即索引 (multi-dimensional keys)

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

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

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]

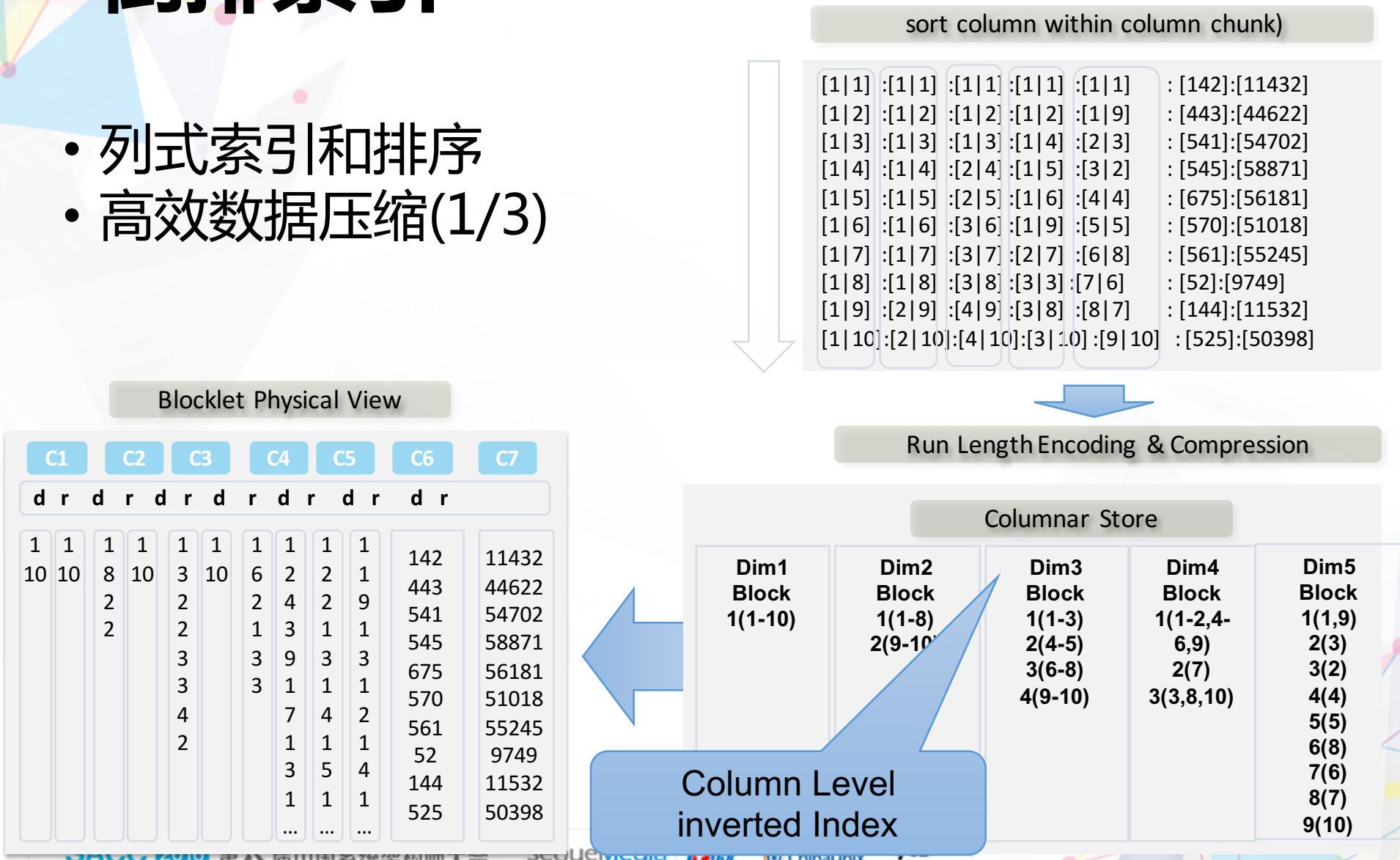
Sort
(MDK Index)

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]

倒排索引

- 列式索引和排序
- 高效数据压缩(1/3)



目录:

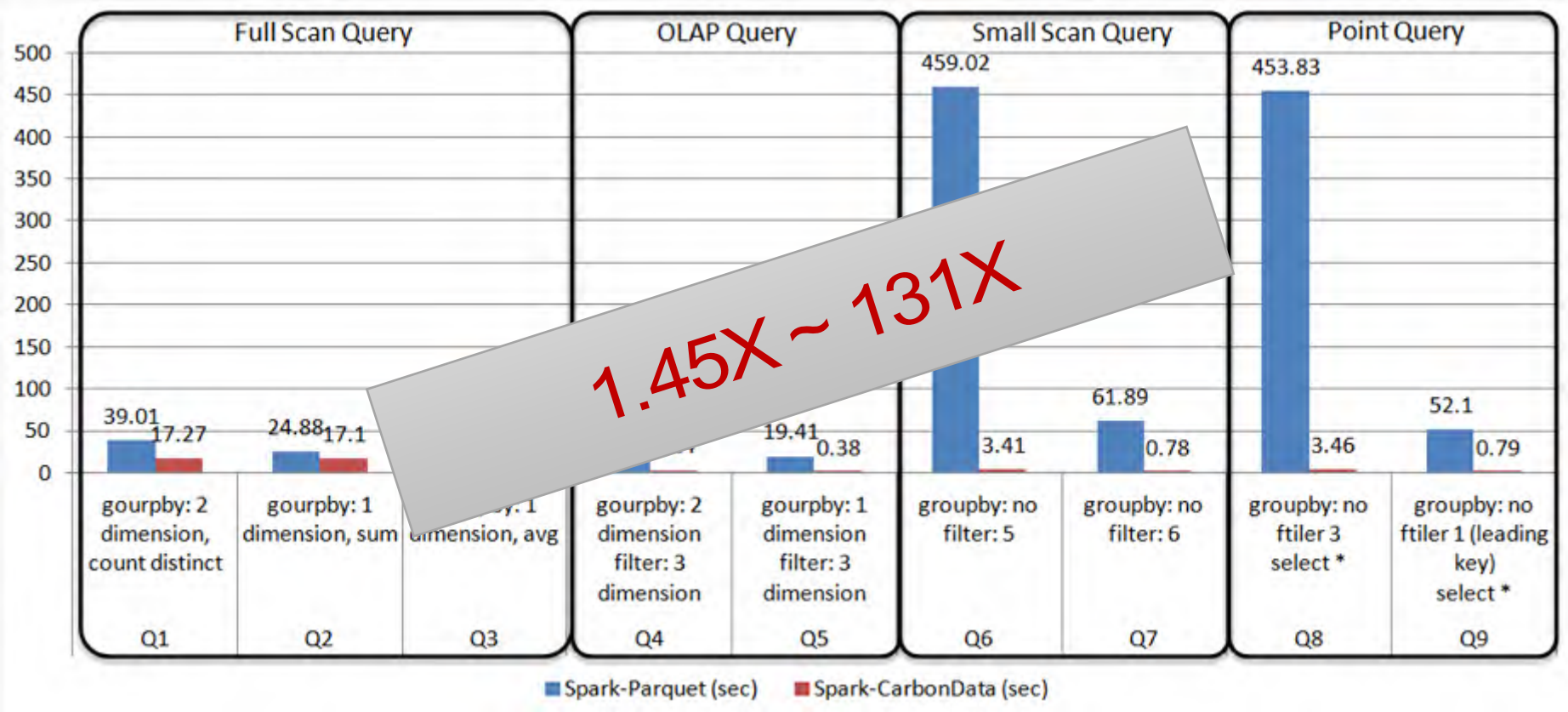
- CarbonData项目背景和适合的场景
- 关键技术介绍
- **性能和DEMO演示**
- Apache CarbonData社区和路标

测试环境

DEMO Environment	
Number of Nodes	1 master + 3workers
vCPU	40cores
Memory	384G
Data Size	1.9TB
#Records	1 billion rows * 300 columns

Data Model	
#Columns	300 (150 String, 150 Double)
# High Cardinality Columns	2 Columns (10 Million) 3 Columns (0.5 Million)
# Medium Cardinality Columns	4 Columns (0.4 Million) 2 Columns (0.2 Million) 11 Columns (0.1 Million)
#Row Size	2KB

性能


Full Scan Query
OLAP Query
Small Scan Query
Point Query
500
450
400
350
300
250
200
150
100
50
0
39.01
17.27
24.88
17.1
19.41
0.38
459.02
3.41
61.89
0.78
453.83
3.46
52.1
0.79
1.45X ~ 131X
gourpby: 2 dimension, count distinct
gourpby: 1 dimension, sum
gourpby: 2 dimension filter: 3 dimension
gourpby: 1 dimension filter: 3 dimension
groupby: no filter: 5
groupby: no filter: 6
groupby: no ftiler 3 select *
groupby: no ftiler 1 (leading key) select *
Q1
Q2
Q3
Q4
Q5
Q6
Q7
Q8
Q9
Spark-Parquet (sec)
Spark-CarbonData (sec)

DEMO演示: CSV，Parquet，CarbonData

- 构造300万行数据
- 用同样的SQL语句分别查询CSV，Parquet，CarbonData数据：

benchmark { csvdf.filter($"name" === "Allen" and $"gender" === "Male" and $"province" === "NB" and $"singler" === "false").count }

```
scala> benchmark { csvdf.filter($"name" === "Allen" and $"gender" === "Male" and $"province" === "NB" and $"singler" === "false").count }
9811.510703ms
res17: Long = 110

scala> benchmark { parquetdf.filter($"name" === "Allen" and $"gender" === "Male" and $"province" === "NB" and $"singler" === "false").count }
868.01382ms
res18: Long = 110

scala> benchmark { carbondf.filter($"name" === "Allen" and $"gender" === "Male" and $"province" === "NB" and $"singler" === "false").count }
145.831882ms
res19: Long = 110
```

目录:

- CarbonData项目背景和适合的场景
- 关键技术介绍
- 性能和DEMO演示
- **Apache CarbonData社区和路标**

Apache CarbonData社区

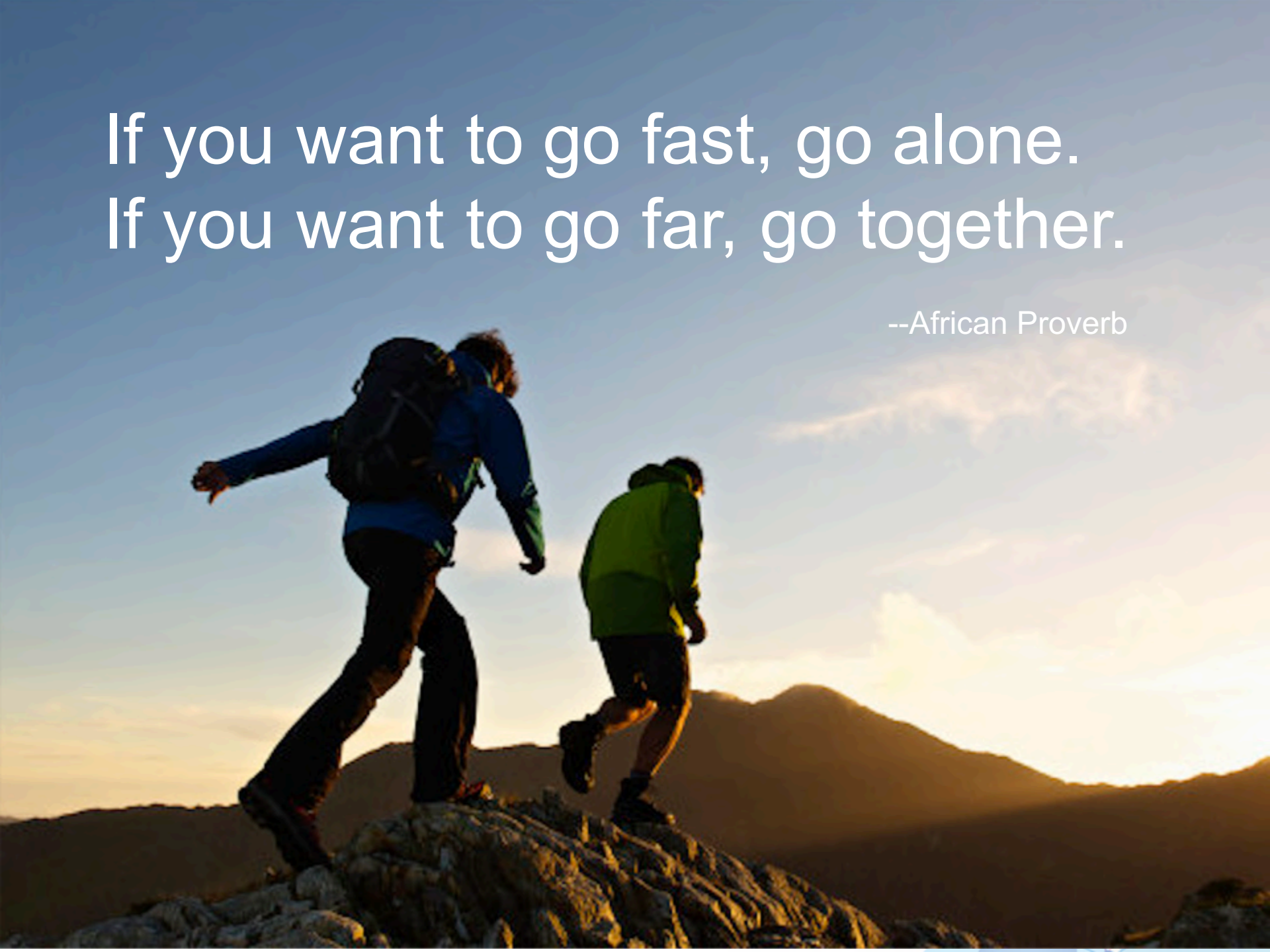
已发布了社区稳定版本 Apache CarbonData 0.1.0,0.1.1

- 深度解读Apache CarbonData：http://www.infoq.com/cn/news/2016/07/huwei-CarbonData-data-second-res
- Apache CarbonData源代码地址: https://github.com/apache/incubator-carbondata
- 订阅Dev Mailing，参与社区讨论：dev@carbondata.incubator.apache.org
- 如果有任何需求、建议、defects反馈，请提交到Apache JIRA：https://issues.apache.org/jira/browse/CARBONDATA
- Contributors from: Huawei, Talend, eBay, Intel, Inmobi, 美团

关注CarbonData微信公众号：ApacheCarbonData，及时获得最新进展信息。

路标计划

- 与Spark 2.x集成
- 与各种主流BI tools集成
- 支持流式数据导入，实时查询
- 支持预聚合
- 支持与主流大数据生态系统集成


If you want to go fast, go alone.
If you want to go far, go together.
--African Proverb

THANKS

SequeMedia 盛拓传媒 | IT168 专注导购16年 | ChinaUnix | ITPUB www.itpub.net