



2017第八届中国数据库技术大会

DATABASE TECHNOLOGY CONFERENCE CHINA 2017

基于SparkSQL的海量数据仓库 设计与实践

360大数据基础架构团队 李振炜

2017.5.13

大纲

- Spark在360的实践与改进
- SparkSQL多数据源整合
- SparkSQL海量数据即席查询的实现

Spark在360的实践与改进

- 团队
 - 奇虎360大数据基础架构团队
 - 离线计算
 - 早期spark推广应用到生产环境的团队;
- 集群规模
 - 总物理机结点数超过8k,
 - 单集群最大节点超过3k;
- Spark任务
 - 10w
 - SQL, MLLib, Streaming

Spark在360的实践与改进

- 改进
 - SQL
 - 扩展语法，优化执行，提高效率
 - MLLib
 - 实现多个社群发现算法
 - 改进PageRank算法，比自带算法速度提升5倍
 - 改进LR算法，支持千万+高维特征
 - 为LDA算法扩展gibbs sampling
 - 改进word2vec，精度接近单机版
 - 引入了FM，xgboost等
 - 深度学习
 - TensorFlow，MXnet，Caffe

Spark在360的实践与改进

- SparkSQL替换Hive
 - 封装为spark-hive
 - 现在已经完成了hive作业向Spark的迁移
 - 稳定运行的SQL作业超过5W

Spark在360的实践与改进

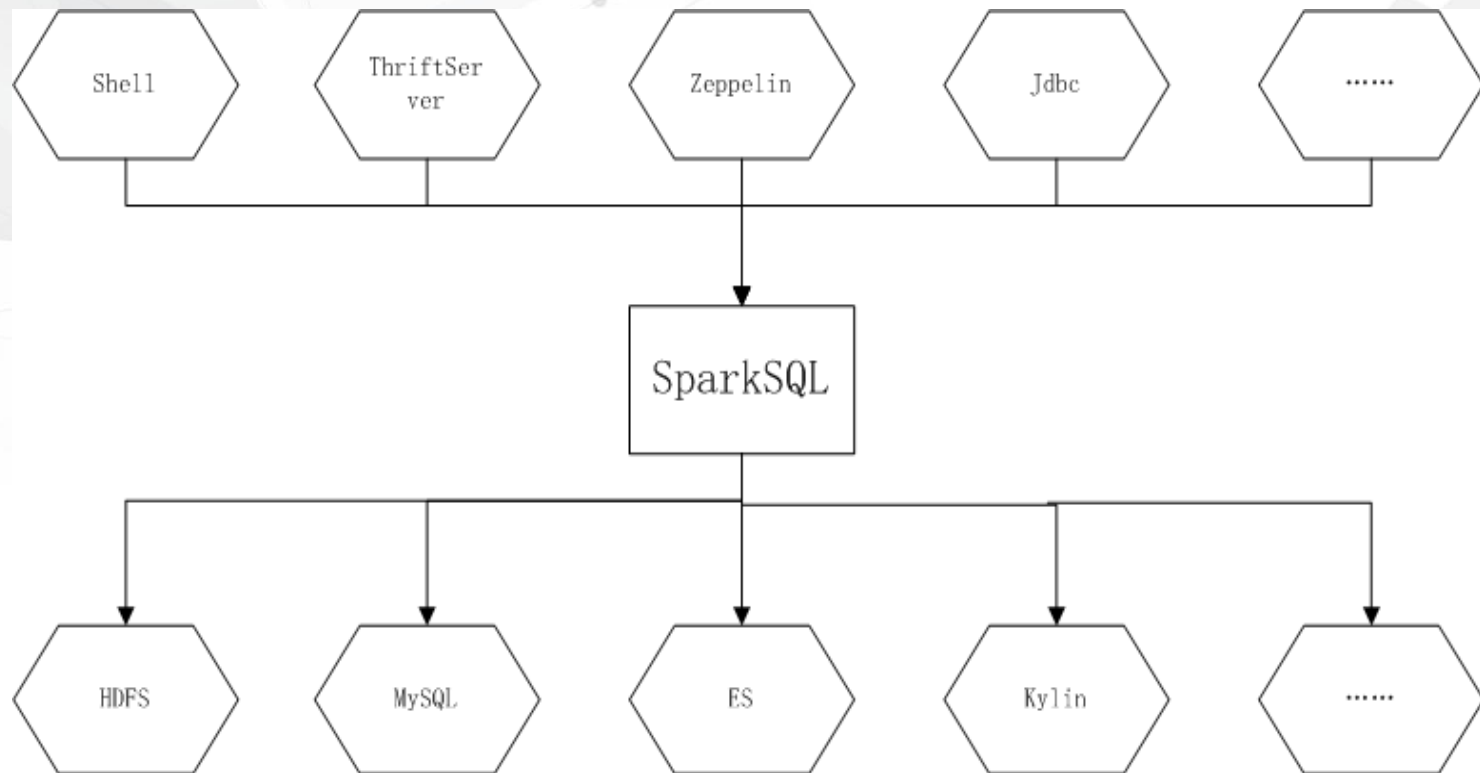
- SparkSQL改进 (50+)
 - Insert overwrite local directory的支持
 - 修复SortMergeJoin 数据倾斜出现OOM
 - 修复Shuffle使用堆外内存，造成executor的内存超限
 - 修复动态资源调整的时候，Driver端出现死锁

SparkSQL多数据源整合

- 动机
 - 数据存储多样
 - 不同的数据在不同的存储
 - 同一份数据在不同的存储
 - 熟悉不同的平台
 - 数据频繁导入导出
 - 业务逻辑复杂

SparkSQL多数据源整合

多数据源整合架构

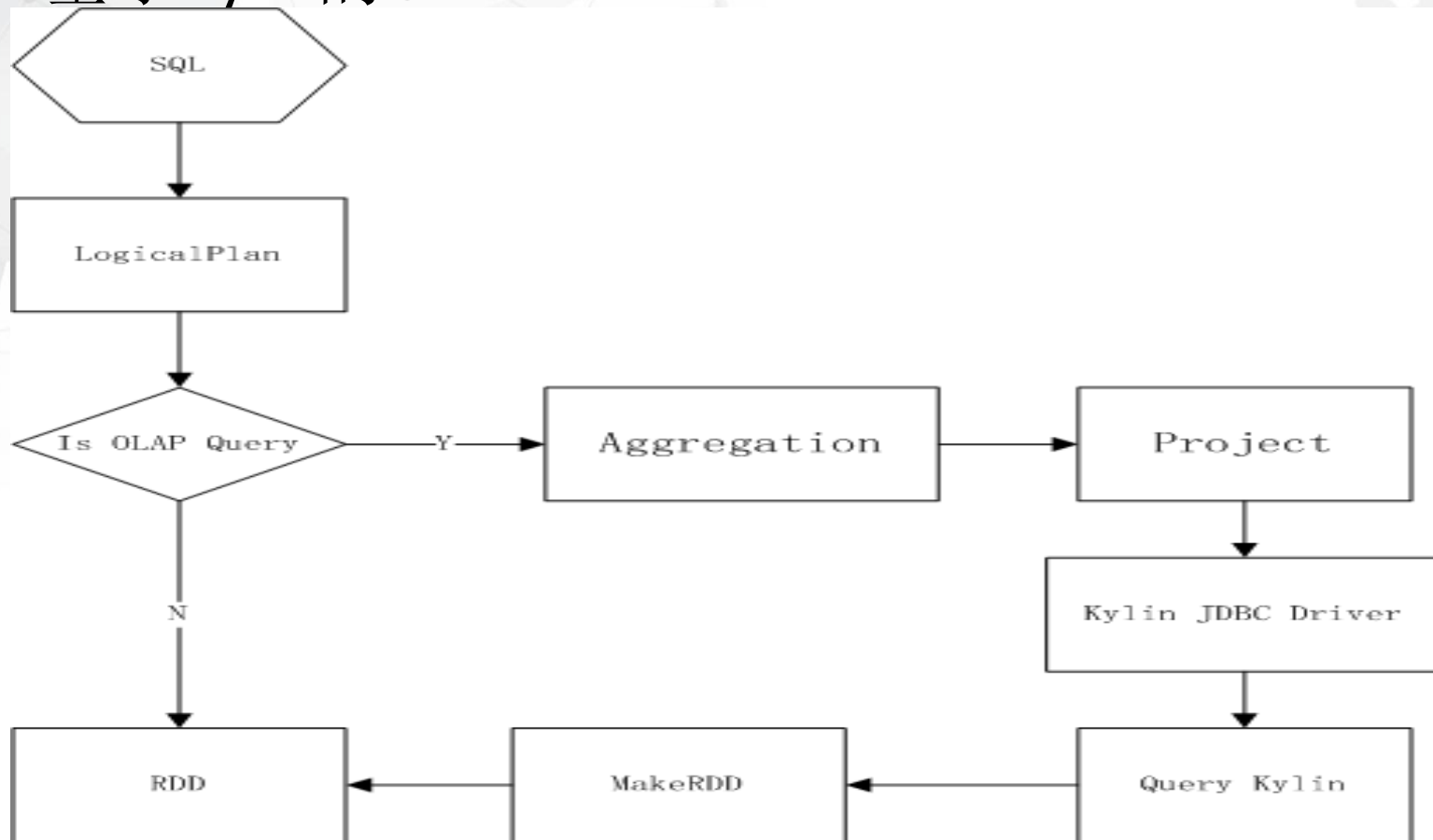


SparkSQL多数据源整合

- 基于ES全文检索
 - 增加TEXT类型，标记分词
 - 扩展建表，删表语法
 - 支持Lucene检索语法
 - SQL查询转换为ES查询

SparkSQL多数据源整合

- 基于Kylin的OLAP



SparkSQL多数据源整合

- 优势
 - 同时分析不同的数据源中的数据
 - 根据不同的SQL自动选择合适的数据源
 - 分析结果写入合适的数据源

SparkSQL海量数据即席查询的实现

- 面临的痛点
 - 数据量与查询效率的矛盾
 - 顺序读与随机读的矛盾
- 现有方案
 - 不同的需求对应不同的存储
- 影响
 - 数据多份存储，增加存储成本
 - 引入多个平台，运维成本高
 - 有些平台的分布式横向扩展遇到瓶颈

SparkSQL海量数据即席查询的实现

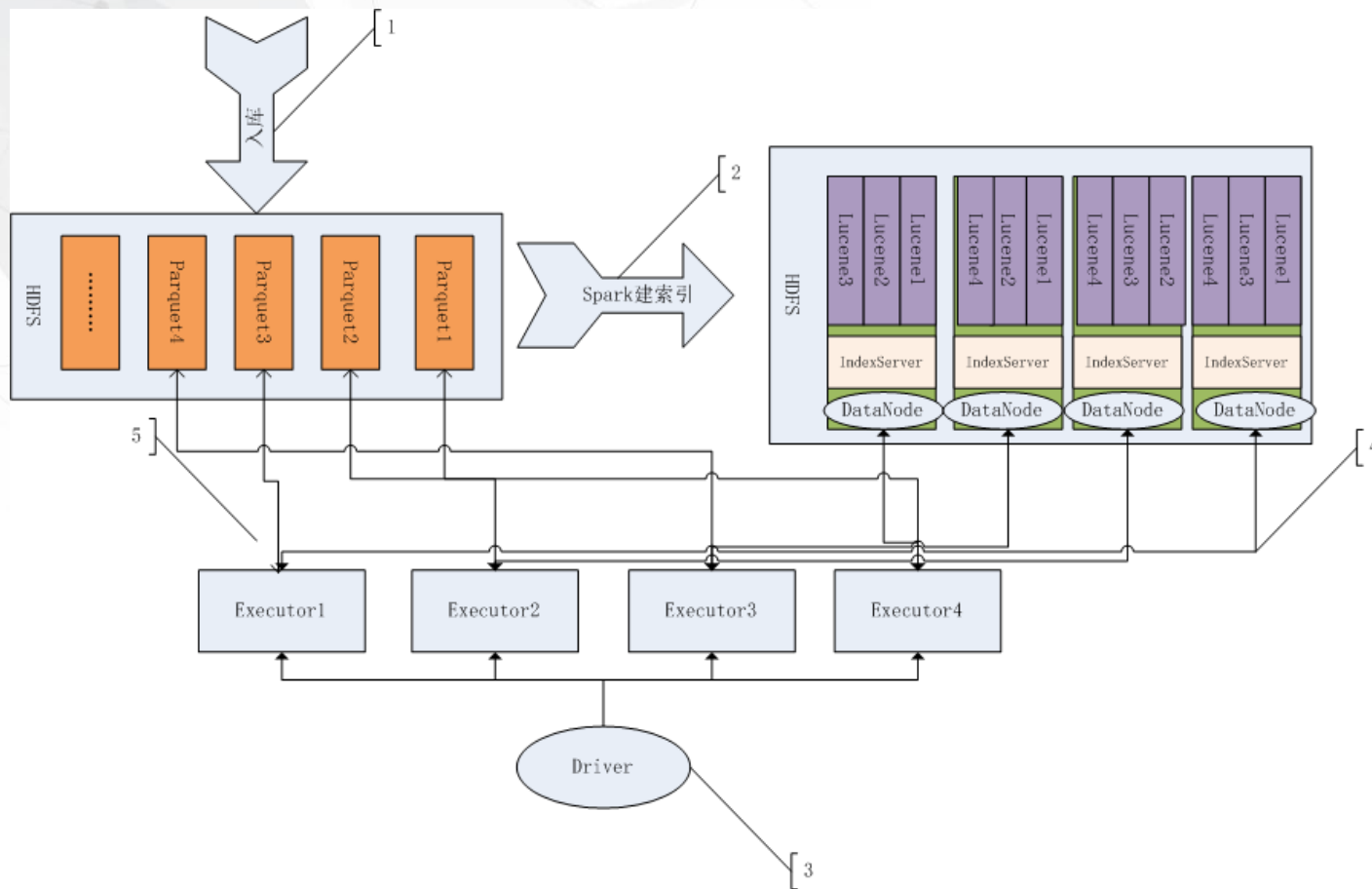
- 前提
 - 在Spark+hadoop的框架内解决
 - 最好不引入新的存储格式
 - 兼容现有的业务逻辑

SparkSQL海量数据即席查询的实现

- 方案
 - 为数据建立一套外部索引
- 设计
 - 存储: Parquet + Hdfs
 - 索引: Lucene + Hdfs + IndexServer
 - 计算: Spark

SparkSQL海量数据即席查询的实现

即席查询架构



1.清洗后的数据以Parquet格式写入HDFS;

2.用Spark对每个Parquet 文件批量建Lucene索引。

3.SQL查询转化为Spark任务;

4.每个Executor由所处理Parquet 文件,得到对应索引文件所在DataNode的地址,向其IndexServer发起查询请求并得到命中的索引;

5.根据索引读取对应的Page, 返回查询结果。

SparkSQL海量数据即席查询的实现

入库优化

- 按行数划分Page。
- 每个RowGroup中都记录每列值最大值和最小值，可以作为一个粗粒度的索引。在此基础上，我们对数值列做了进一步改进：计算出均值和标准差，对应列数值做如下处理

$$x' = \frac{x - \mu}{\sigma} \quad x_{new} = \begin{cases} 3 & x' \geq 3 \\ x' & other \\ -3 & x' \leq -3 \end{cases} \quad y_n = \begin{cases} 1 & n == Round\left(\frac{x_{new} + 3}{6} N\right) \\ 0 & else \end{cases}$$

把均值和标准差以及由 $y_0 y_1 \dots y_{N-1} y_N$ 组成的Byte数组写入到RowGroup元信息中。

- 为整个Parquet文件也建立改进后的索引。

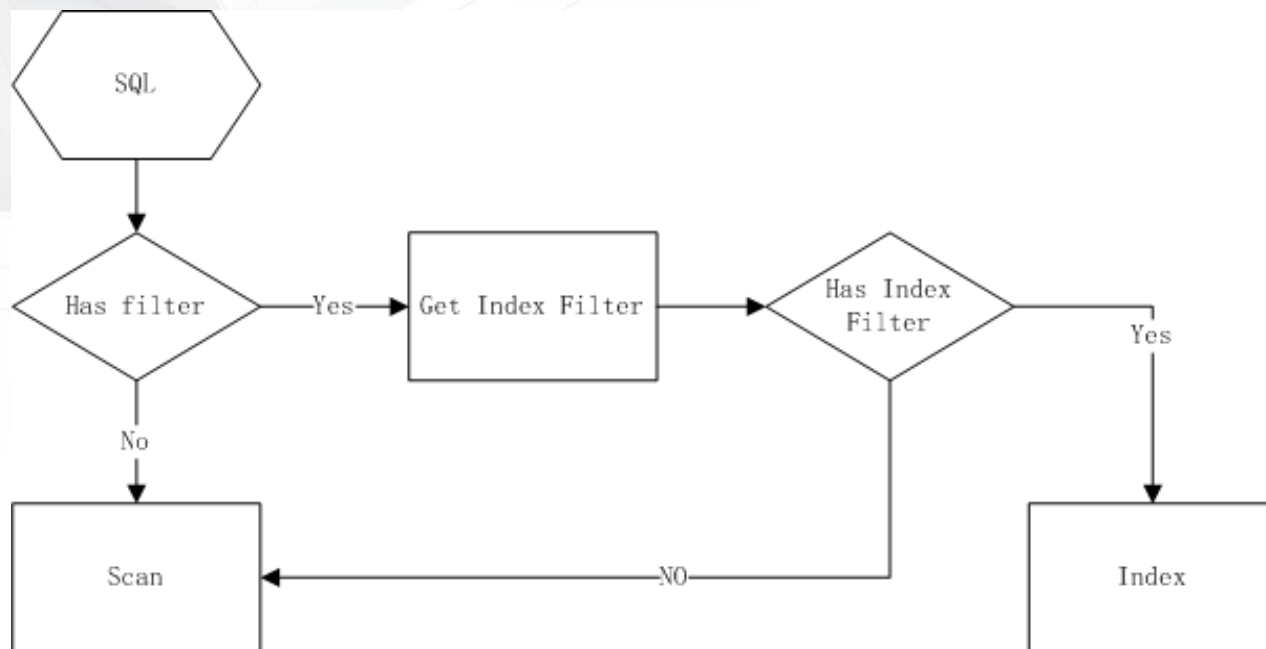
SparkSQL海量数据即席查询的实现

建立索引

- 读取元信息
- 写Lucene文件
 - 每行作为一个Doc
 - 索引列作为Field
 - 为每个Parquet文件单独建索引
- 优化
 - 只索引而不存储
 - 增大数值类型切分精度
 - 删减Lucene功能

SparkSQL海量数据即席查询的实现

SparkSQL转化



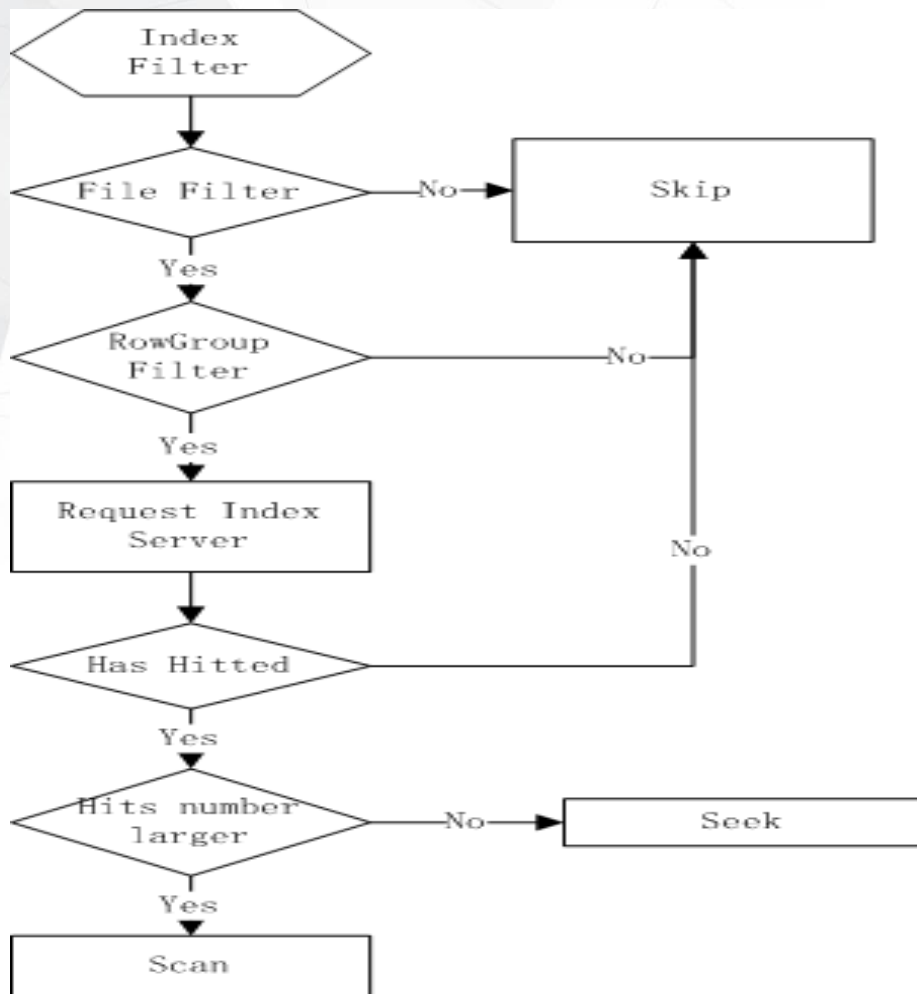
SparkSQL海量数据即席查询的实现

SparkSQL转化

- 优化
 - 每个Parquet为一个Task
 - 缓存元信息
 - 合并Spark Task
 - 复杂Filter下推
 - 字符串查找匹配

SparkSQL海量数据即席查询的实现

SparkSQL执行查询



每个Task根据索引查询结果
自动选择是执行Scan还是
Seek.

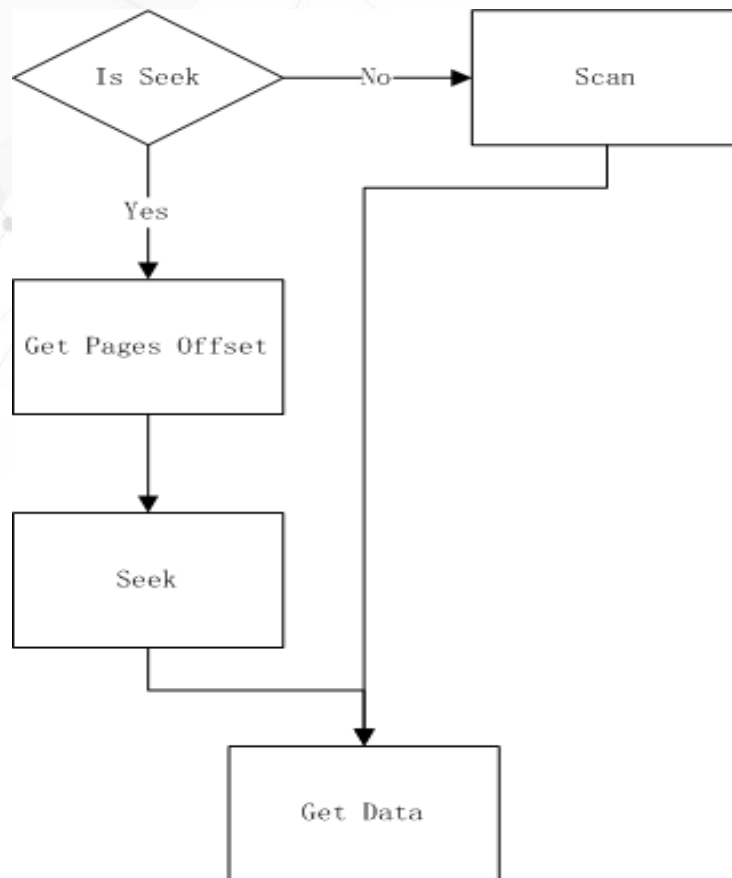
SparkSQL海量数据即席查询的实现

SparkSQL执行查询

- 索引查询策略：固定数据，移动查询。
- 索引文件多副本，发起多个查询，最先返回的作为结果，避免查询长尾的出现，保证查询的稳定性。
- 索引文件的管理全部依赖于Hdfs，具有极强扩展能力，同时减少运维压力。
- IndexServer 是无中心，无状态独立服务，依托于多副本，自动实现容错。

SparkSQL海量数据即席查询的实现

SparkSQL读取数据



SparkSQL海量数据即席查询的实现

性能分析

- 分析背景
 - 数据
 - 选取万亿规模的真实业务数据
 - 包括20W个Parquet
 - Parquet文件总大小约70T
 - 每个Parquet包含500W条记录
 - Spark集群
 - 125个Executors,
 - 每个Executor: 4cores and 5G
 - 测试SQL
 - `Select count(*) from table where
a="748e8837b09e6b253de7389de6c7e2c4"
and b="748e8837b09e6b253de7389de6c7e2c4"
and c like "31100002037%"`
 - 命中结果为27710, 大约分布在5%的文件中

SparkSQL海量数据即席查询的实现

性能分析

- 过程分析

- 对于全表遍历的数据的情况：
 - 读取单个Parquet文件，逐行遍历，并返回命中的结果，需要的时间5s
 - 实测时间为2376s
- 利用索引，随机读数据：
 - 查索引且未命中的代价平均为50ms
 - 查索引命中且从Hdfs读取对应的Page，遍历Page，返回命中结果，需要时间平均200ms
 - 实测时间为27s

SparkSQL海量数据即席查询的实现

总结

- 数据和索引分离，完全兼容现有的业务；
- 数据和索引存储于Hdfs，可以支持海量的数据；
- IndexServer无中心，无状态，容错及负载均衡天然与Hdfs绑定，运维成本低；
- 查询找数据，避免了数据的网络传输；
- 顺序读和随机读可以根据具体的执行逻辑，自动切换，并且代价很低；
- 对于不再使用的索引，可以单独删除，节约空间。



THANKS

SequeMedia
盛拓传媒

IT168.com

ITPUB

ChinaUnix.net