

SQL on Hadoop



作者：许江

走向分布式？







一个系统走向分布式，一定有其不得不为的理由。可扩展性是最常见的理由之一。我先简单的将“可伸缩”的需求分成两种：

- **Data Scalability:** 单台机器的容量不足以 (经济的) 承载所有资料，所以需要分散。如：NoSQL

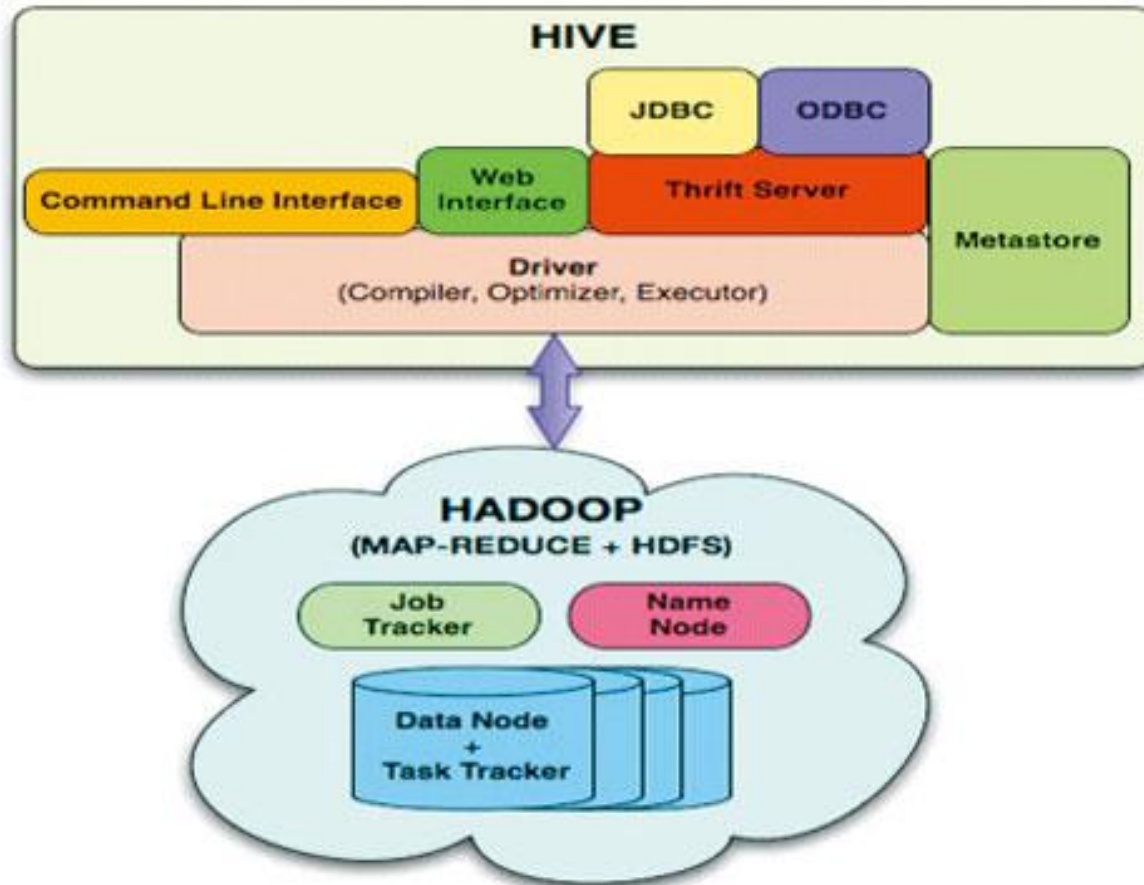
- **Computing Scalability:** 单台机器的运算能力不足以 (经济的) 及时完成运算所以需要分散。如：科学运算。不管是哪一种需求，在决定采用分布式架构时，就几乎注定要接受一些牺牲：

1. 牺牲效率：网路延迟与节点间的协调，都会降低执行效率。
2. 牺牲 AP 弹性：有些在单机上能执行的运算，无法轻易在分布式环境中完成。
3. 牺牲维护维运能力：分散式架构的问题常常很难重现，也很难追踪。另外，跟单机系统一样，也有一些系统设计上的 tradeoffs(权衡)
4. CPU 使用效率优化或是 IO 效率优化
5. 读取优化或是写入优化
6. 吞吐率优化或是网络延迟优化
7. 资料一致性或是资料可得性,选择了不同的 tradeoff，就会有不同的系统架构。

SQL on hadoop

- **Hive** [Tez,hive on spark] 
<http://hive.apache.org/>
- **Spark-sql /shark(spark on hive)** 
<https://spark.apache.org/>
- **Impala** 
<http://www.cloudera.com/>
- **phoenix+hbase** 
<http://phoenix.apache.org/>
- **Drill** 
<http://drill.apache.org/>
- **Presto** 
<http://prestodb.io/docs/current/>

Hive



shark



Development ending;
transitioning to Spark SQL



A new SQL engine designed
from ground-up for Spark

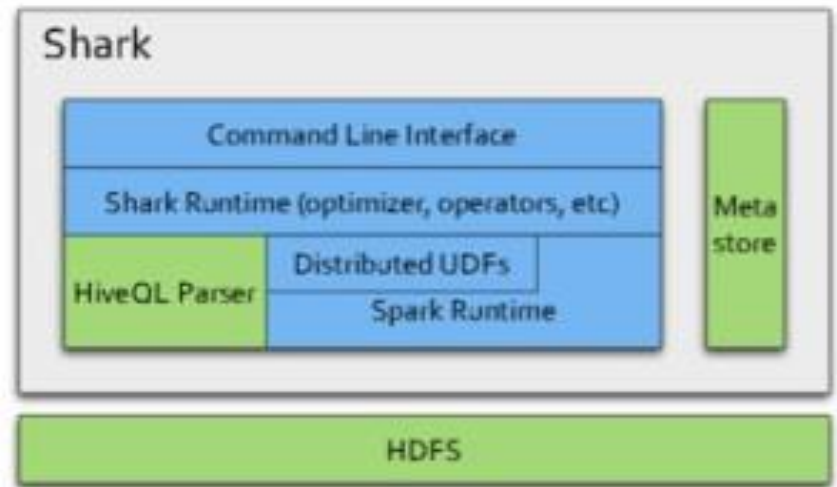
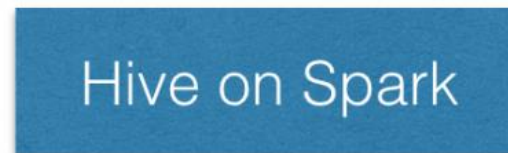
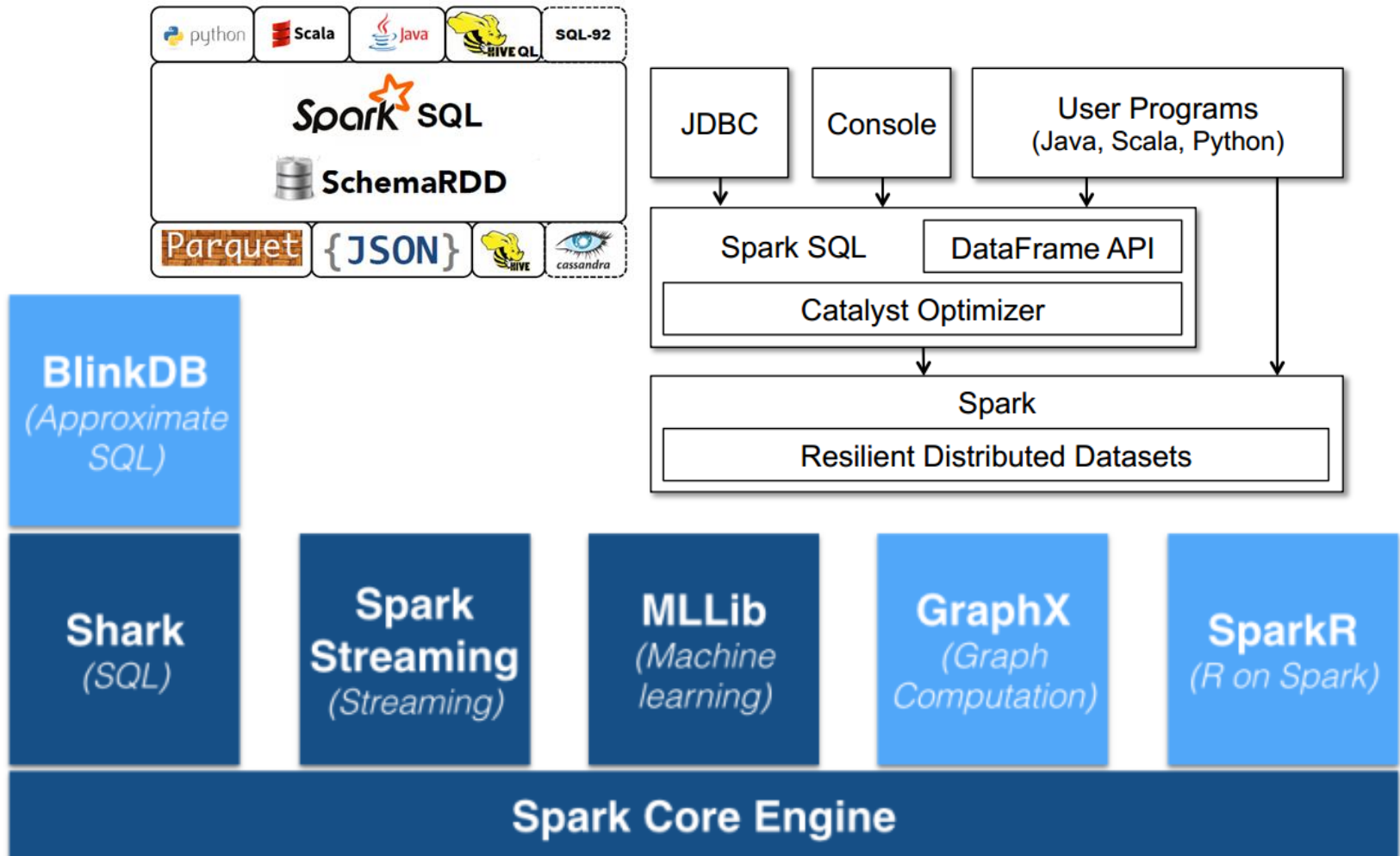


Figure 1: Shark Architecture



Help existing Hive users
migrate to Spark

Spark-sql



impala

Impala: Architecture

Common Hive SQL and interface



Unified metadata store

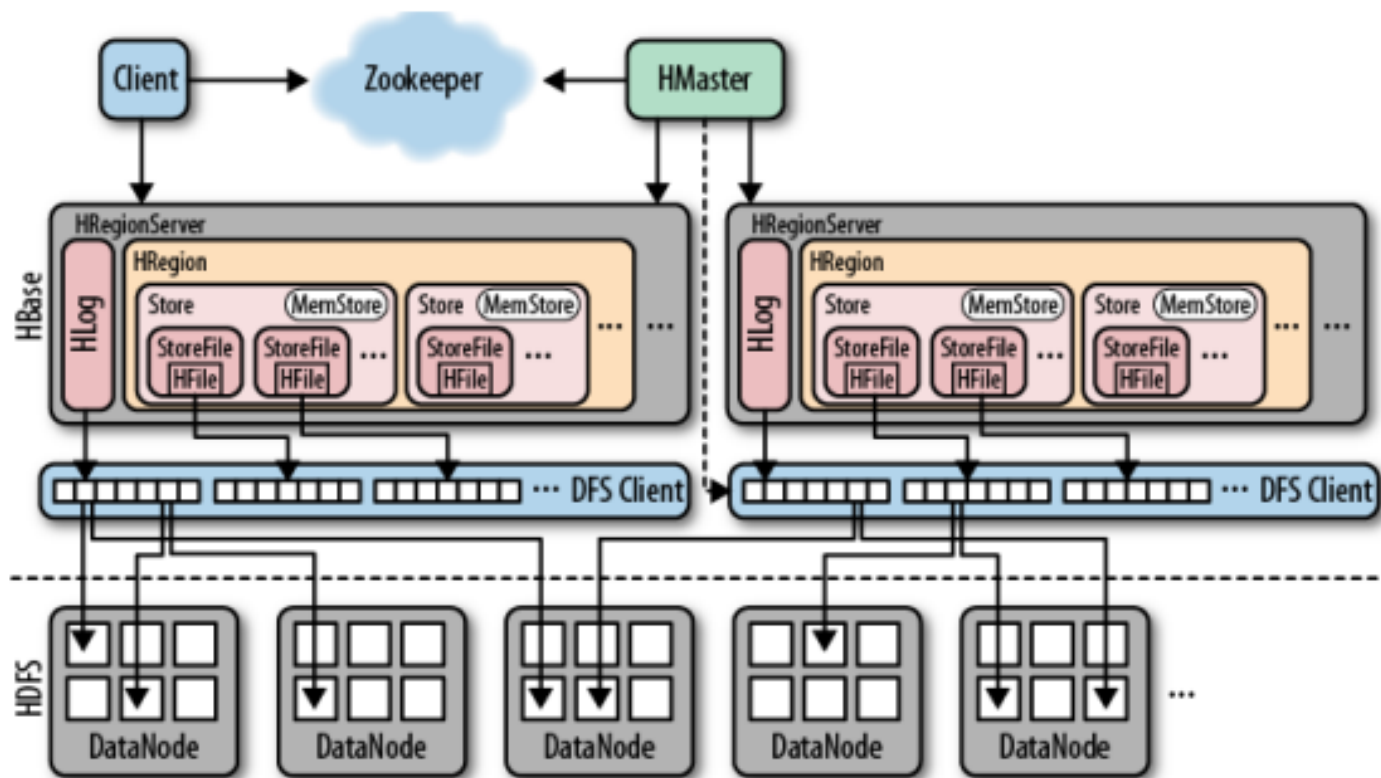


impalad's continually talk to statestore to update their state and to receive metadata to use for query planning

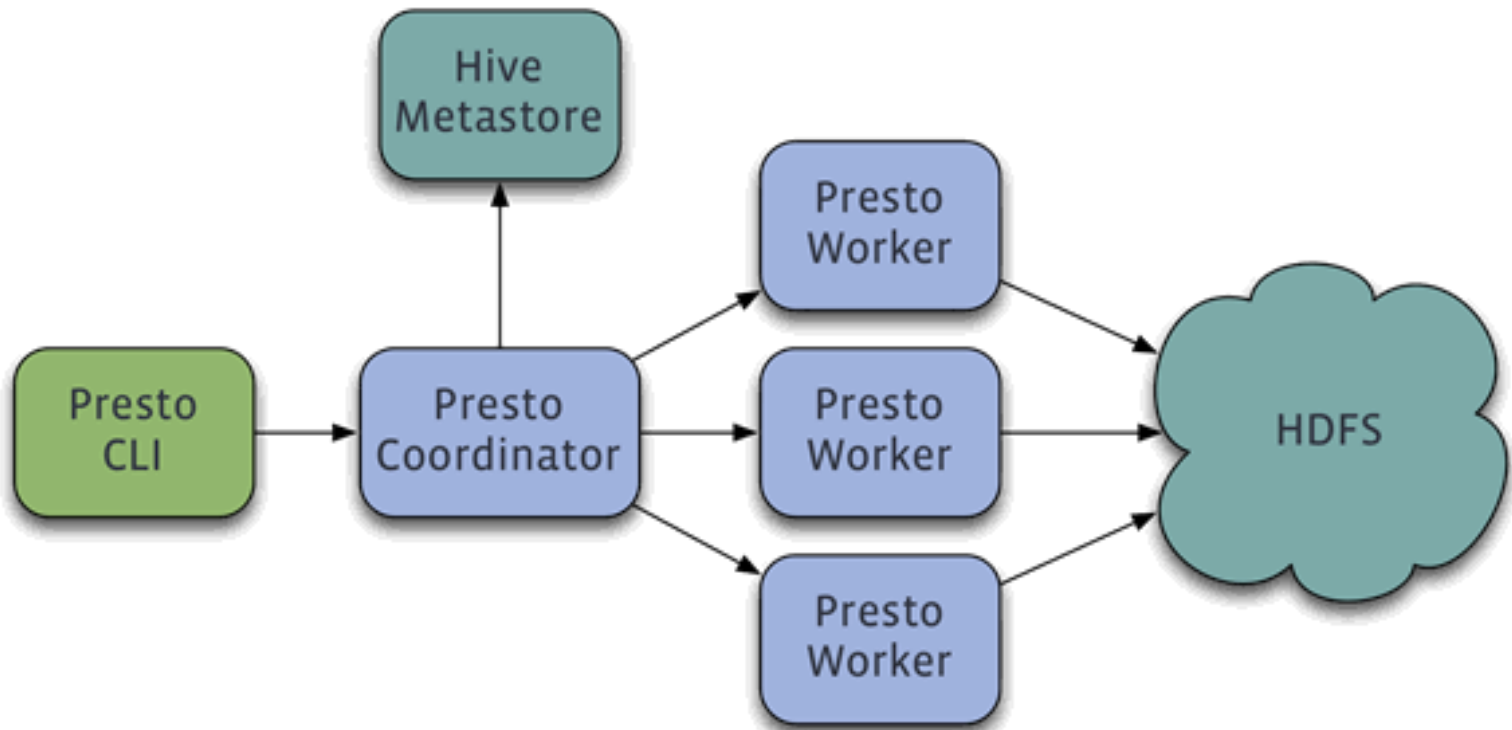


Phoenix(sql on hbase)

- Phoenix实现类sql查询Nosql数据库
- Phoenix将Query Plan直接使用HBaseAPI实现
- SQL语句解析为hbase查询语法



Presto



性能1?

Parquet表(亿级,5个维度group by)

组件	耗时	资源消耗	其它问题记录
Hive	176.491s	3G	第一次执行
Impala	4.899s	5.4 GiB	第一次执行
Impala	4.584s	5 GiB	第二次执行
Spark-sql	9.92s	3.5 MB	第一次执行
Spark-sql	5.009s	3.5 MB	第二次执行

Text表(亿级, ,5个维度group by)

组件	耗时	资源消耗	其它问题记录
Hive	808.005s	2G	第一次执行
Impala	240.829s	2.7 G	第一次执行
Impala	238.893s	2.4 G	第二次执行
Spark-sql	318.5s	9.1 MB	第一次执行
Spark-sql	291.191s	8.9 MB	第二次执行

集群配置: mem:32g, cpu:32core, 软件版本: saprk1.2,impala2.2,hive0.13.1,hadoop2.4

注意: 这里的资源消耗情况并不准确, 是根据个人想法填写, 仅作参考!

性能2？

parksql cache table(亿级, ,5个维度group by)

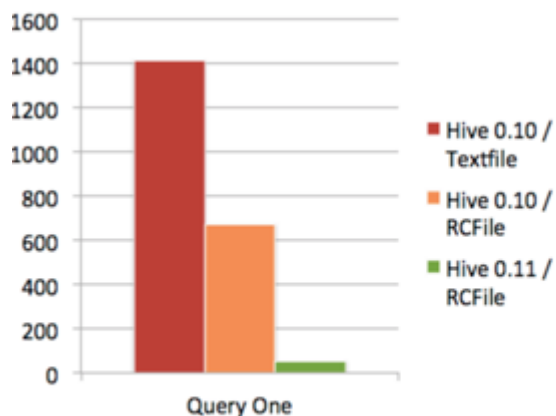
组件	耗时	资源消耗	其它问题记录
Hive	154.805s	5GB	第一次执行
Impala	128.204s	9.6 GiB	第一次执行
Impala	365.958s	9.3 GiB	第二次执行
Spark-sql[cache]	15.602s	9.5 MB	第一次执行
Spark-sql[cache]	7.707s	9.5 MB	第二次执行
Spark-sql[nocache]	32.382s	9.5 MB	第一次执行
Spark-sql[nocache]	32.941s	9.5MB	第二次执行

注意：这里的资源消耗情况并不准确，是根据个人想法填写，仅作参考！

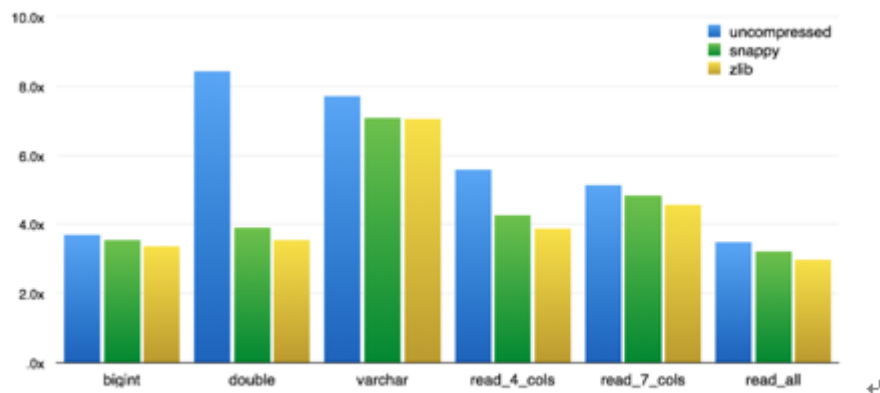
文件格式

组件↴	格式↴	数据量↴	数据大小↴	不支持↴	支持↴
Imapla↴	Parquet↴	1118681226↴	30.1 G↴	Orcfile↴	Rcfile↴
Sparksql↴	Parquet↴	1118681226↴	↴	无↴	全↴
Hive↴	Rcfile↴	1118681226↴	93.4 G↴	无↴	全↴
Presto↴	Orcfile↴	1118681226↴	16.2 G↴	无↴	全↴
Sparksql↴	Textfile↴	1118681226↴	↴	无↴	全↴

seconds

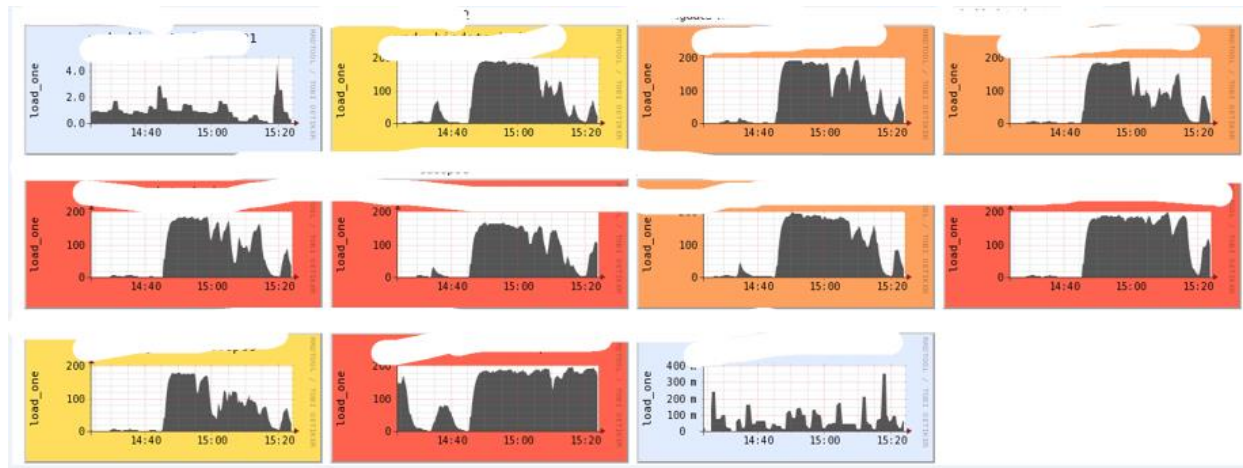


ORC Raw Decoding Speedup



性能3？

Hive生成各种格式表	time	Filesize	问题记录
Rcfile	689.045s,	93.4 G	
Orcfile	1566.637s	30.1 G	
Parquetfile	1251.149s	68.1G	
Textfile	990.122s	98.6 G	



生成各种文件格式严重消耗集群资源，如果为parquet格式生成情况！

问题？

- 生成各种不同文件格式时间换空间
 - orc,rcfile,parquet,textfile
 - 文件格式转换效率确实不高
- 各种oom, overcommit
 - yarn资源调优
 - Container xxx is running beyond physical memory limits
 - Java heap space(YARN)
- 都是吃内存大户,内存是个大问题!
 - 各种列式存储格式超级耗内存
 - impala入门级mem:128g+
- 性能调优,调错参数,整个系统奔溃!
 - 一步错,满盘皆输
 - 运维需要有敬畏之心,充分验证理解各种调养参数,不如不如不调,选择绕过!

SQL on hadoop选型？

演示安装过程

大数据



云计算



Openstack,docker,spark等.

Thank you

提问时间?

qq群: 288396468