**Spark SQL 2.0 Experiences Using TPC-DS**

**（Berni Schiefer）**

# Abstract

This talk summarizes the results of using the TPC-DS workload to characterize the SQL capability, performance and scalability of Apache Spark SQL 2.0 at the multi-Terabyte scale in both single user dedicated and multi-user concurrent execution modes. We track the evolution of Spark SQL across versions 1.5, 1.6 and 2.0 to underscore the pace of improvement in Spark SQL capability and performance. We also provide best practices and configuration tuning parameters to support the concurrent execution of the 99 TPC-DS queries at scale. The key takeaways include

1) See the substantial progress made by Spark SQL 2.0

2) Understand what TPC-DS is and why it has become the preferred workload of SQL on Hadoop systems.

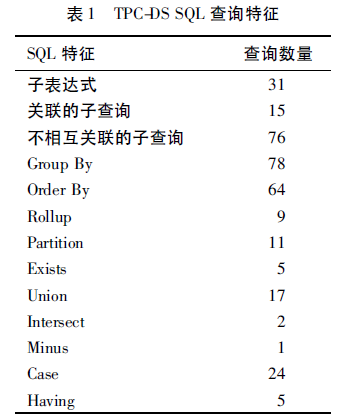
3) Experimental results supporting the optimized execution of multi-user, multi-terabyte TPC-DS-based workloads

4) Tuning and configuration changes used to attain excellent performance of Spark SQL.

# TPC-DS简介

TPC-DS测试基准是TPC组织推出的用于替代TPC-H的下一代决策支持系统测试基准

当前为2.3版，TPC-DS采用星型、雪花型等多维数据模式。它包含7张事实表，17张维度表平均每张表含有18列。其工作负载包含99个SQL查询，覆盖SQL99和2003的核心部分以及OLAP（联机分析处理）。



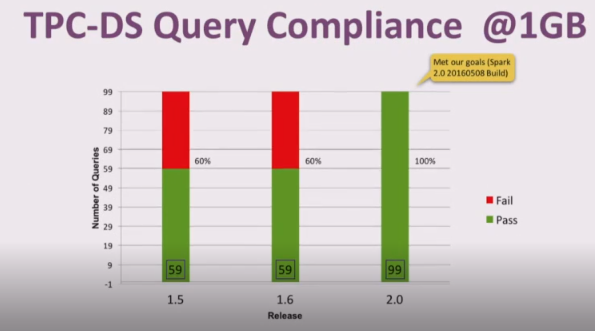
没有开发商对自己产品发布过正式的TPC-DS测试

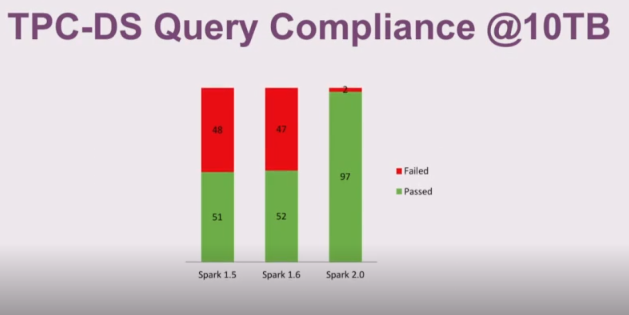
多个scale（100G,300G,1T,3T,10T,30T,100T）

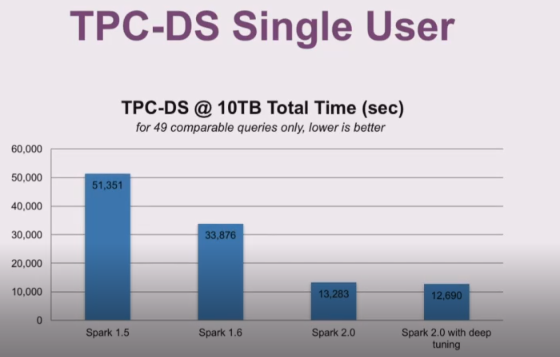
可以支持多用户并行执行

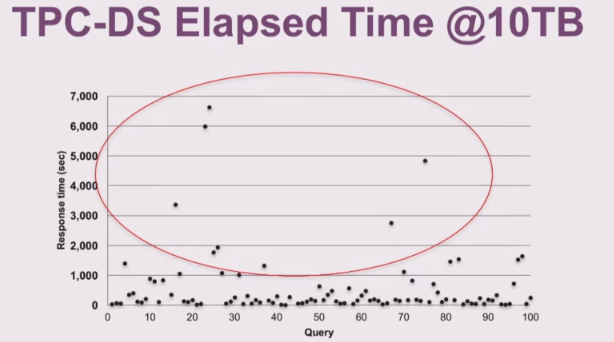
默认为了spark sql benchmark

# 简要测试结果









大多数的query能快速计算，但是有一些query花费了很长的时间，总结原因有两点：

1. 这个query确实是一个非常tough的query，进行了多个fat table的连接操作
2. 这些query采用了bad执行计划，并没有采用right执行计划

# 下一步计划

More data

10TB ->100TB

more user

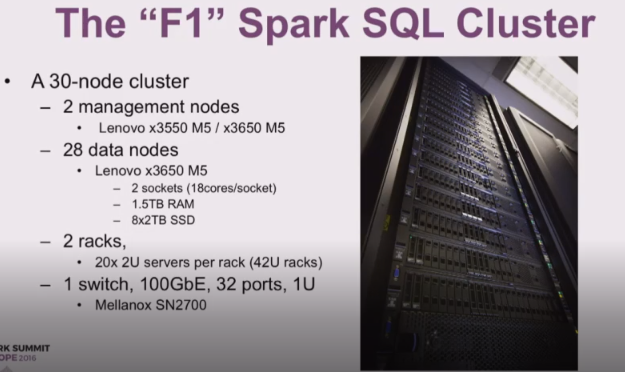
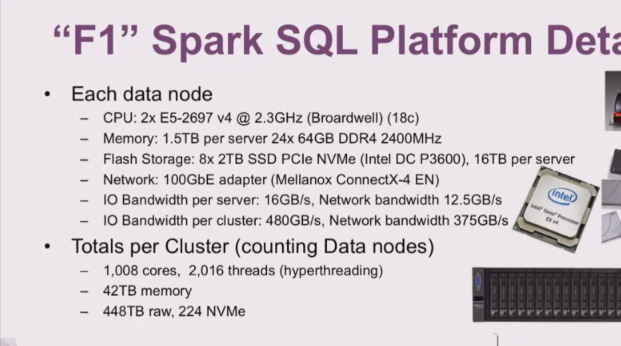
single user -> 4 concurrent users

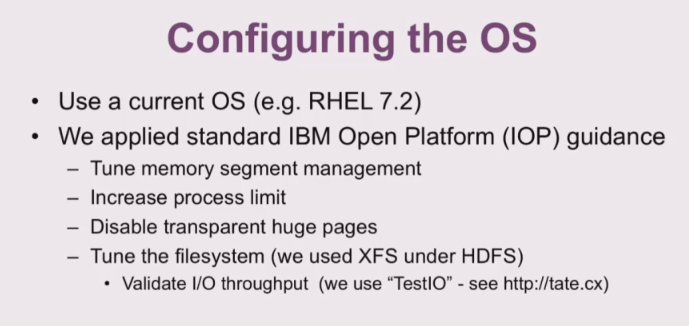
more performance

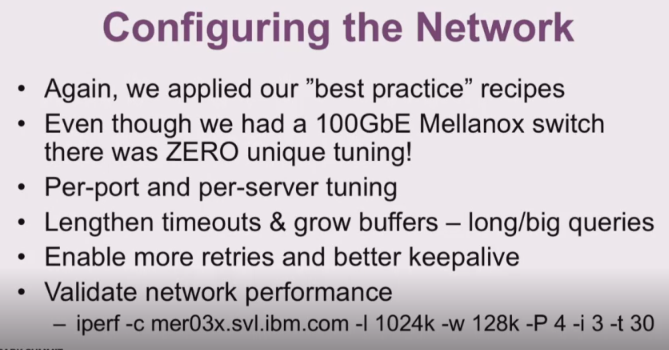
focus on long-running queries

develop and submit changes for2.x

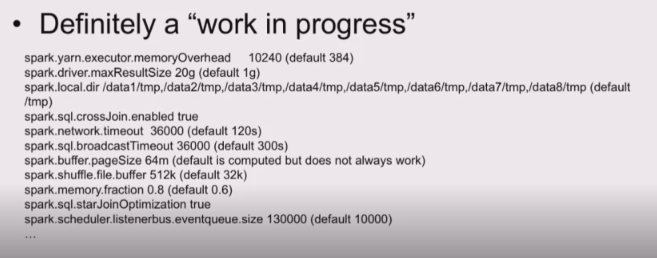
# 硬件配置与软件环境



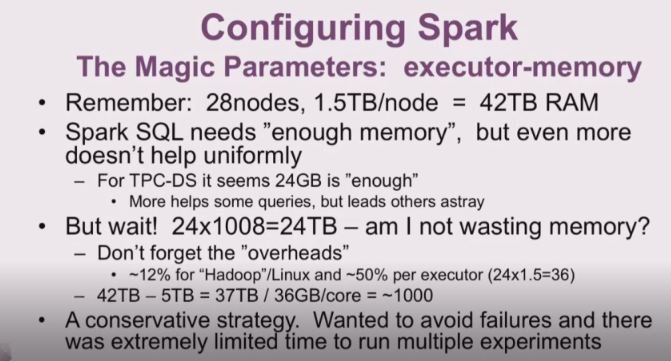


# Spark参数配置



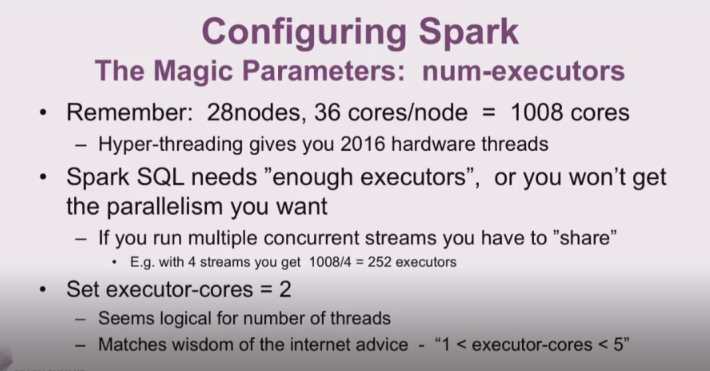
## 两个参数

Executor-memory



策略：

先放开内存使queries测试通过，然后减少内存



## 具体参数设置:

|  |  |
| --- | --- |
| driver-memory | 20g |
| master | Yarn |
| executor-memory | 24g |
| num-executor | 1008 |
| executor-cores | 2 |
|  |  |

# 一个优化issue

Spark-17791 Join Reordering using star schema detection

* Finds the star join with the largest fact table and places it on the driving arm of the left-deep join. This plan avoids large tables on the inner, and thus favors hash joins.

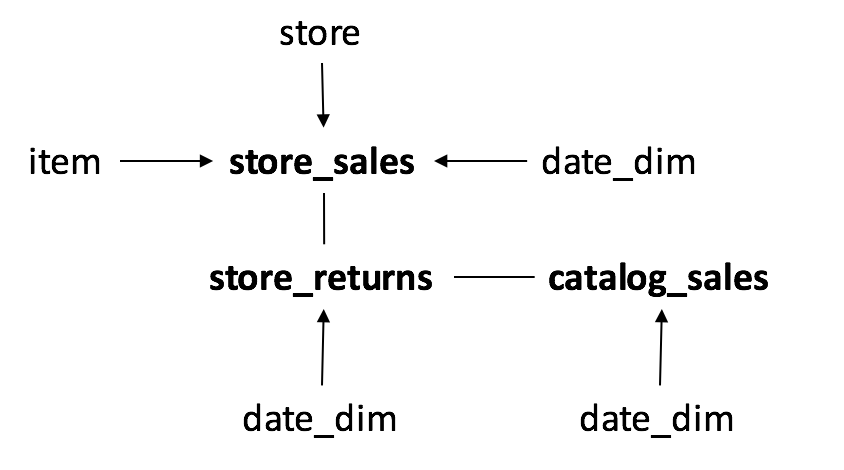
找到最大的星形事实表，将其放到左深树的驱动臂首先进行join，这样可以避免大表放在内层，因此有可能进行hash join

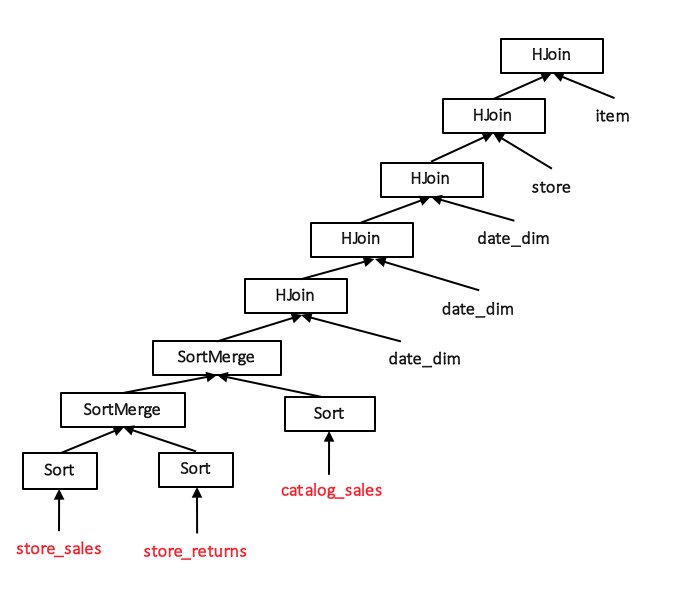
* Applies the most selective dimensions early in the plan to reduce the amount of data flow.

优先join选择度大的维度表来减少data flow

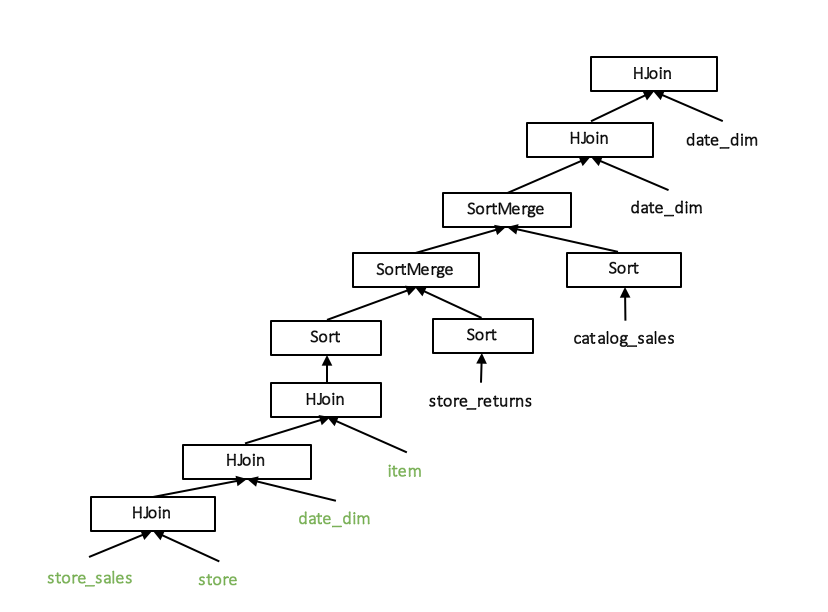
Query25

**Star schema:**





**Default, positional join reordering:** suboptimal join of large fact tables.

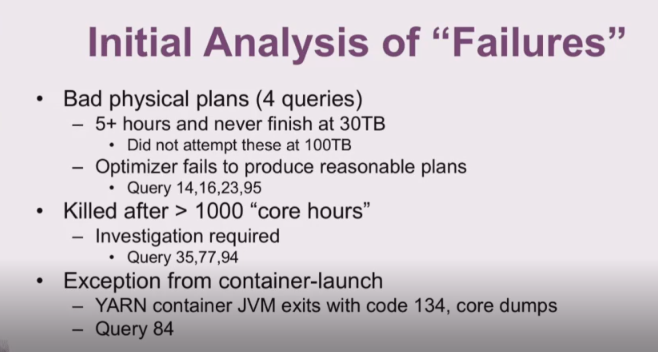


**Star join reordering:** selective star join early in the plan.

Result

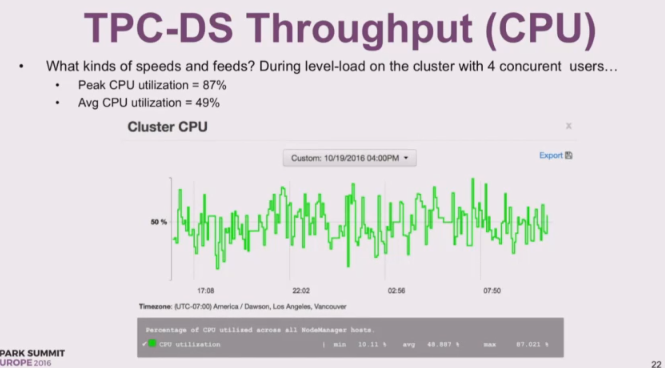
提升效率在60%-70%

# 错误分析

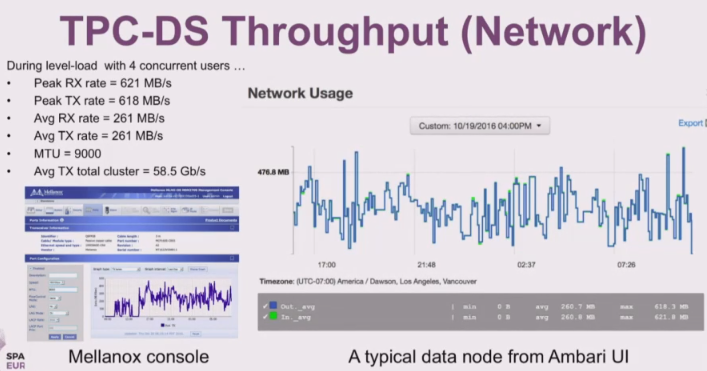


# Throughput Test

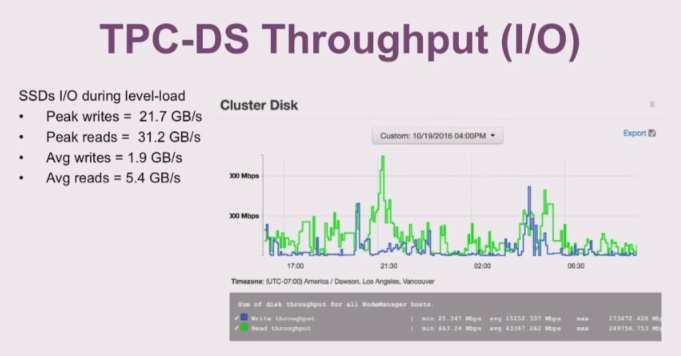
## CPU使用率



## Network



## IO速率



# Summary

1. Spark2.0相对于spark1.x在SQL方面有巨大的进步，spark2.1应该会更加稳定
2. IBM，intel，Lenovo和Mellanox尝试部署出适合新spark的集群拓扑结构
3. Tpc-ds是个很有价值和挑战性的workload
4. 测试在single user 上运行不错（91/99）
5. 在4个并行users的情况下，TPC-DS的throughput test体现了很好的可扩展性
6. Spark2.0仍然有可以改进的地方

When spark sql runs, it’s great, but it doesn’t always run