Greenplum优化技巧与实践

Greenplum中文社区
苑泽福（阿福）

AGENDA

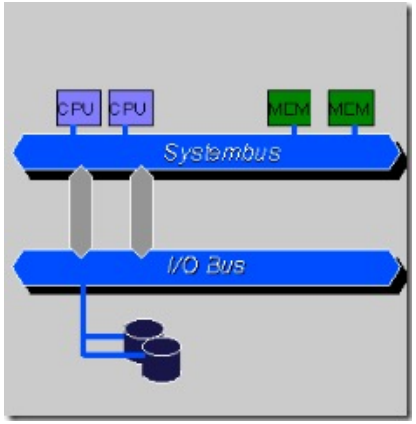
Greenplum架构解析

Greenplum重点功能

Greenplum优化要点

Greenplum架构解析

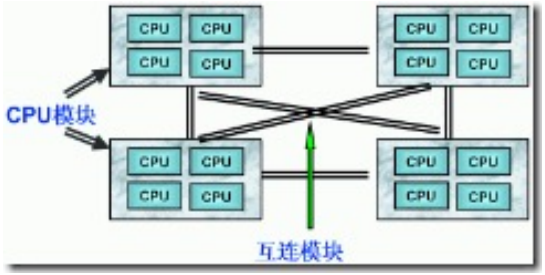
先从系统架构说起



SMP- 对称多处理器结构

特点
共享存储：CPU、内存、IO

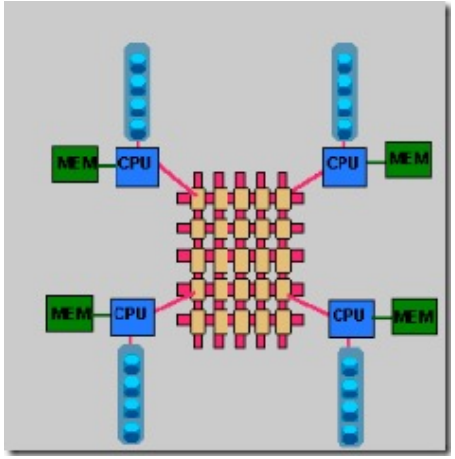
不足
扩展能力有限



NUMA - 非一致存储访问结构

特点
拥有多个CPU模块，每个模块由多个CPU组成，有独立的本地内存；节点之间通过互联模块进行连接和信息交互，较好解决SMP系统的扩展问题

不足
互联模块访问效率和本地内存访问不在一个效率层级，系统性能无法随CPU数线性增加

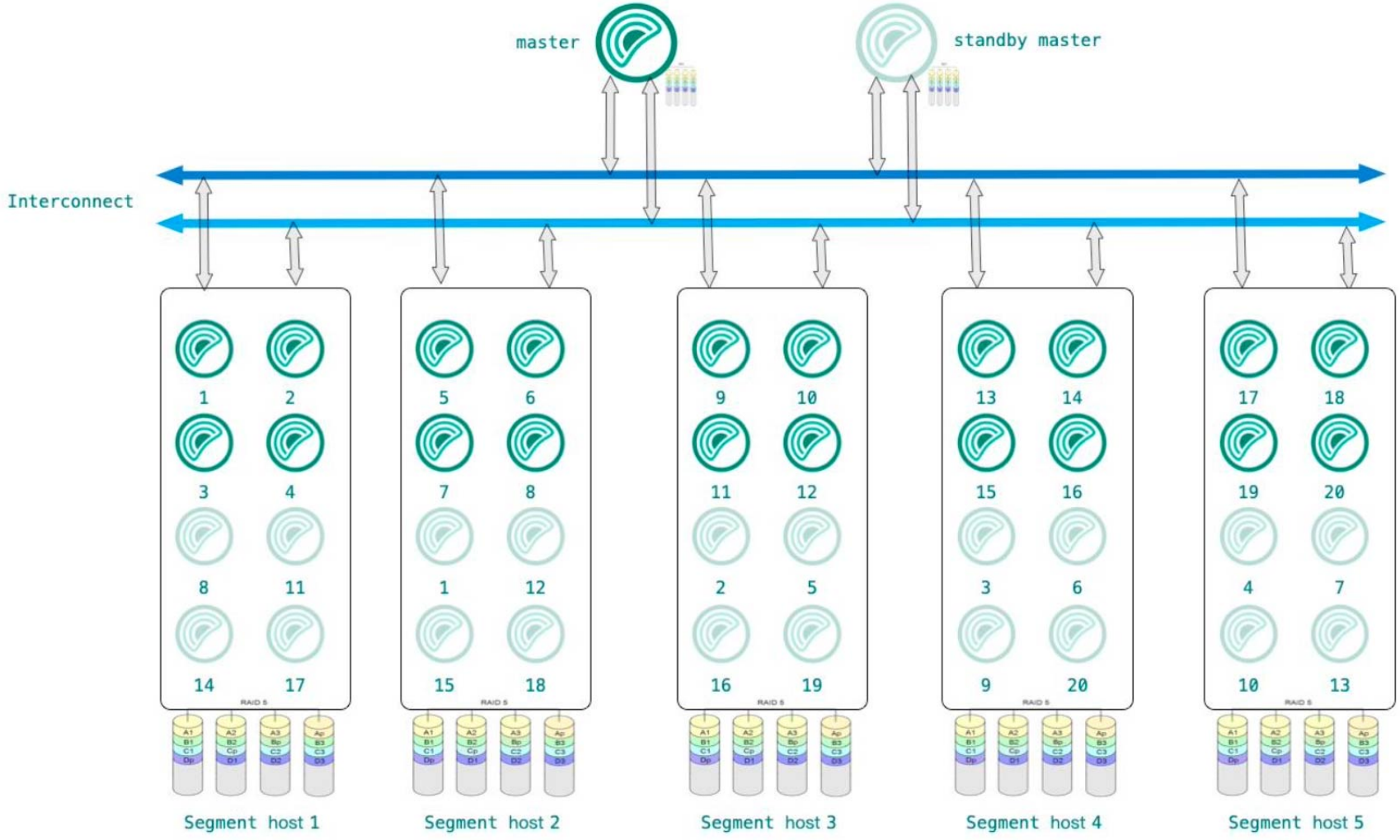


MPP - 海量并行处理架构

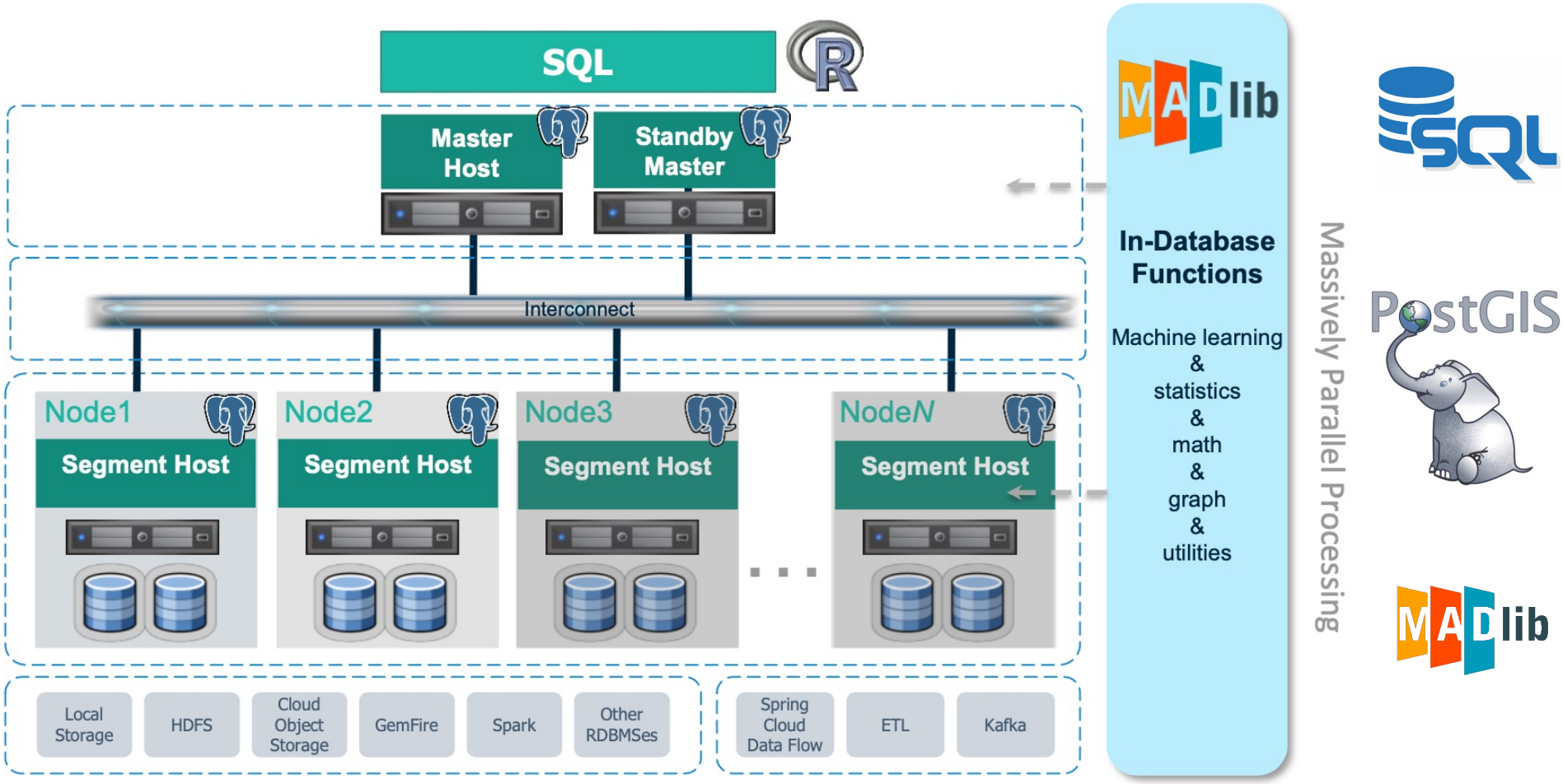
特点
节点互联网络：Share Nothing结构，每个节点只访问本地内存和存储，节点信息交互和节点本身是并行处理的；
对等节点：所有数据节点角色一样。可以提升并行计算能力

不足
木桶的短板效应。
MPP集群规模不能过大。
并发度不能过高。

Greenplum的MPP实现 - 简洁优雅



一切皆并行


GREENPLUM DATABASE
SQL
Master Host
Standby Master
Interconnect
Node1
Node2
Node3
NodeN
Segment Host
Segment Host
Segment Host
Segment Host
Local Storage
HDFS
Cloud Object Storage
GemFire
Spark
Other RDBMSes
Spring Cloud Data Flow
ETL
Kafka
MADlib
In-Database Functions
Machine learning & statistics & math & graph & utilities
Massively Parallel Processing
SQL
PostGIS
MADlib

使用Greenplum，您能获得什么？

GREENPLUM DATABASE

	之前	之后	性能提升
数据编辑/整理	• 181 行代码 • 75 分钟	• 116 行代码 • 8 分钟	9.35x
特征编辑	• 439 特征 • 4,517 行代码 • 100 分钟	• 934 特征 • 1,438 行代码 • 30 分钟	多 495 个特征，快 3.33x
信息价值	• ~450 个变量，~30分钟计算结果并写入 excel	• 在 GPDB 中花 58 秒计算 ~200 个变量的IV	13.7x/变量
建模	• < 50 个变量，运行一次逻辑回归迭代需要 ~30 分钟	• 376 个变量，运行一次逻辑回归迭代需要 ~1.86 分钟	~16x/迭代

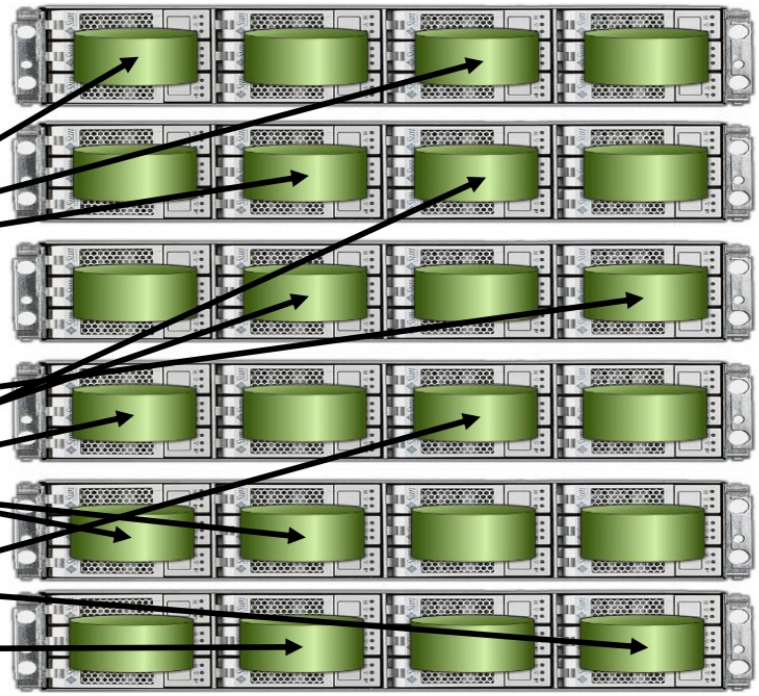
Greenplum重点功能

数据分布:每个节点 1/n 数据

GREENPLUM DATABASE

最重要的策略和目标是均匀分布。

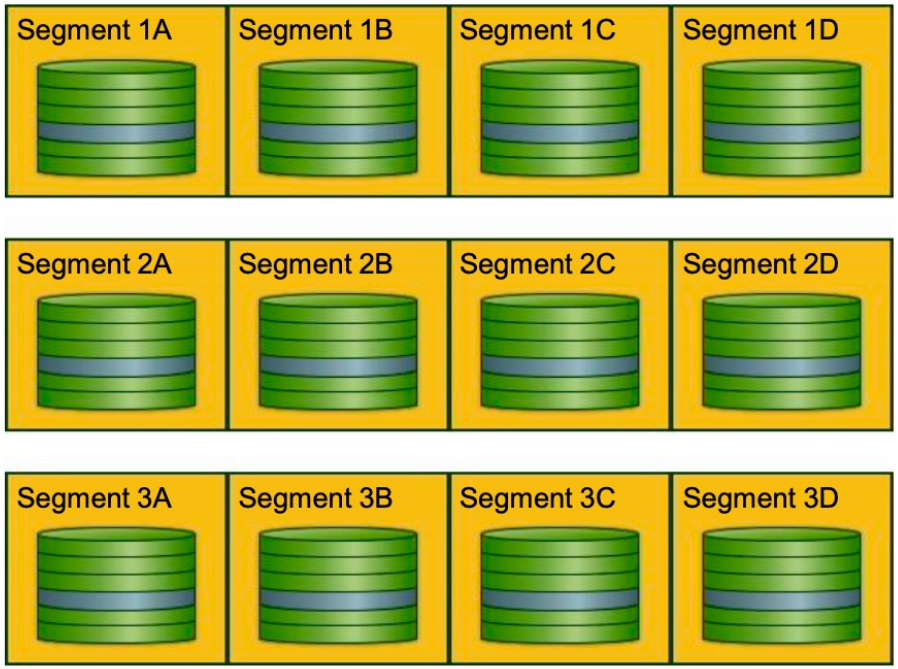
Order		
Order #	Order Date	Customer ID
43	Oct 20 2005	12
64	Oct 20 2005	111
45	Oct 20 2005	42
46	Oct 20 2005	64
77	Oct 20 2005	32
48	Oct 20 2005	12
50	Oct 20 2005	34
56	Oct 20 2005	213
63	Oct 20 2005	15
44	Oct 20 2005	102
53	Oct 20 2005	82
55	Oct 20 2005	55



多级分区

GREENPLUM DATABASE

```
SELECT COUNT(*)
  FROM orders
 WHERE order_date >= 'Oct 20 2007'
   AND order_date <  'Oct 27 2007'
```



多模存储/多态存储

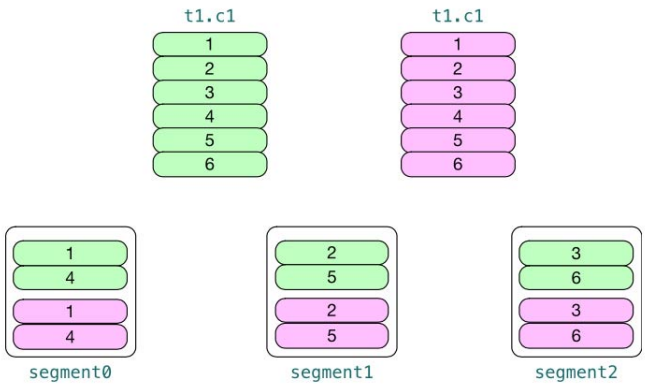
- 适合 OLTP 业务
- 适合频繁更新或者访问大部分字段的场景

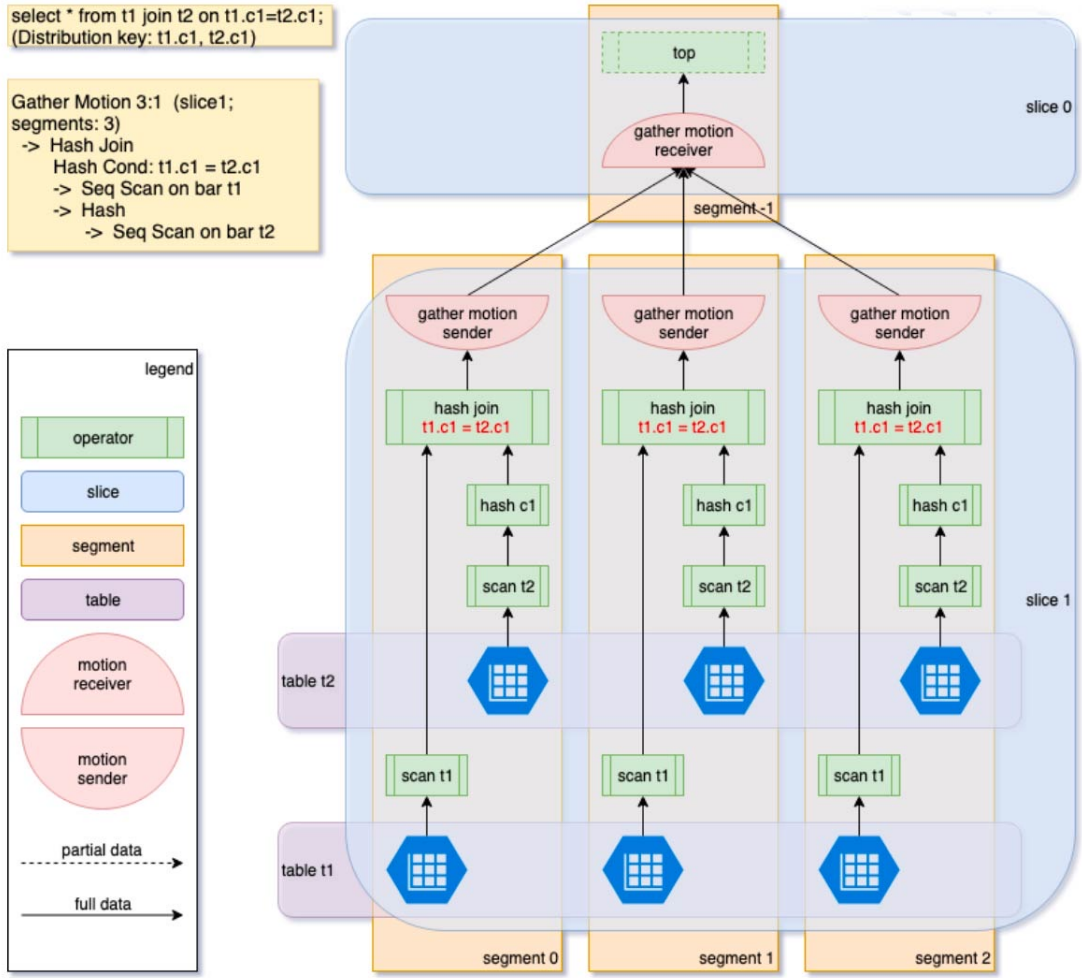
- 列存储更适合压缩
- 查询列子集时速度快
- 不同列可以使用不同压缩方式: gzip (1-9), quicklz, delta, RLE，zstd

- 历史数据和不常访问的数据存储在 HDFS 或者其他外部系统中
- 无缝查询所有数据
- Text，CSV，Binary，Avro，Parquet，ORC 格式

N 个节点并行执行

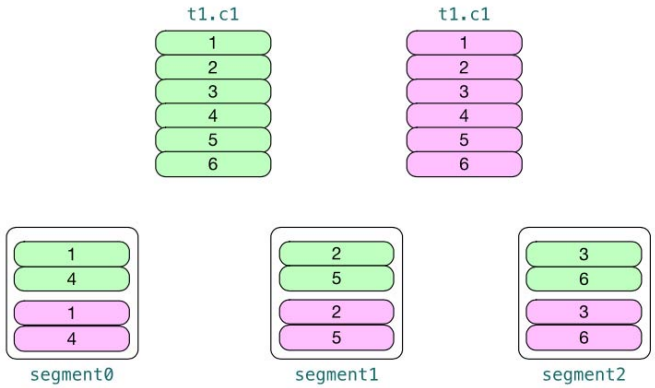
GREENPLUM DATABASE

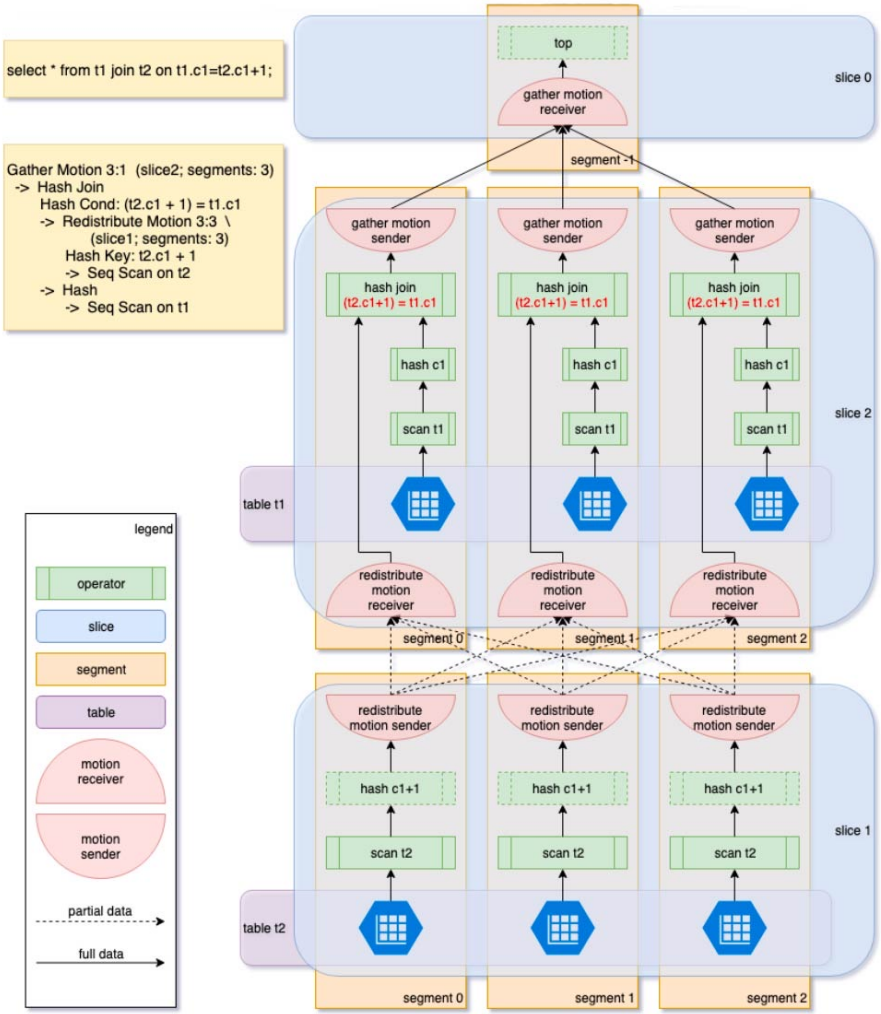




数据shuffle

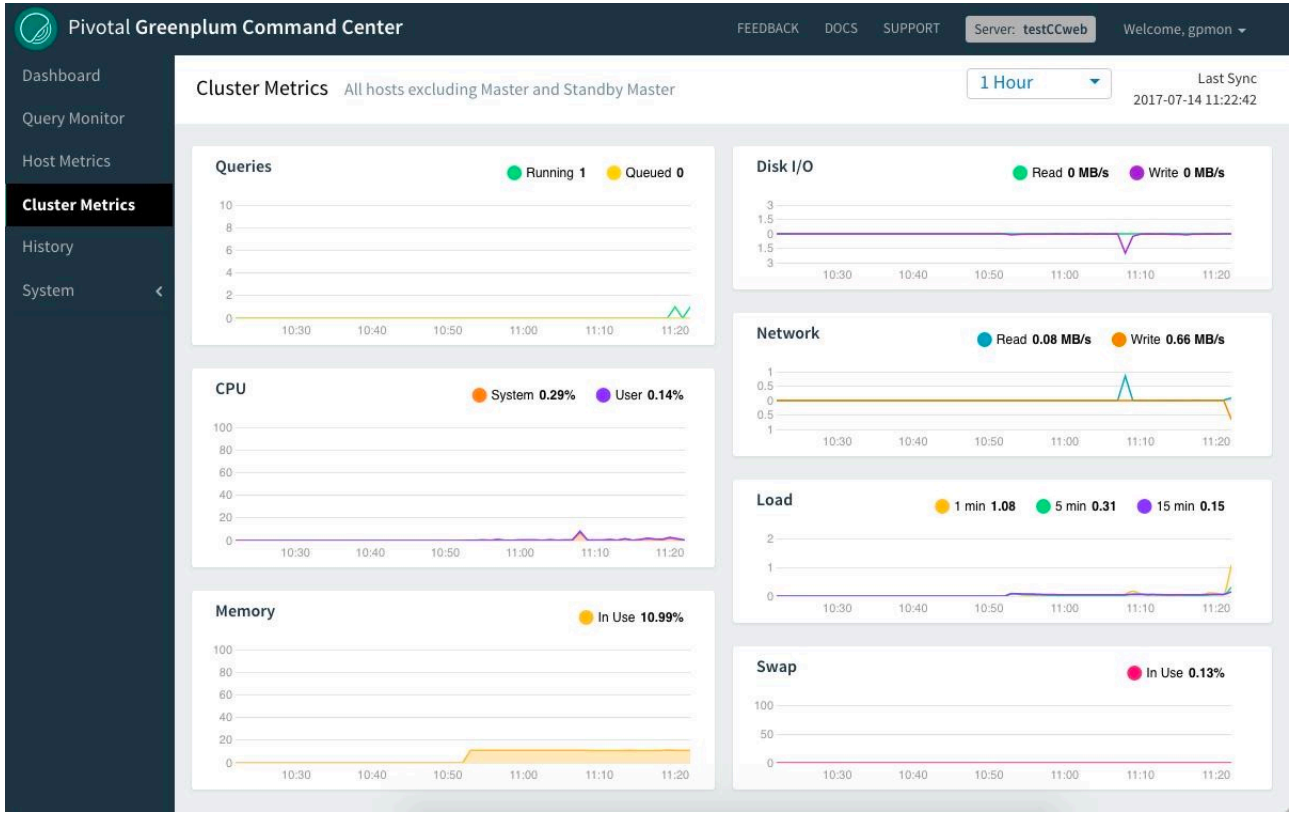
GREENPLUM DATABASE





其他功能

- 全局死锁检测(GDD)
- 复制表
- 灵活索引
- 资源负载管理
- 企业级的监控



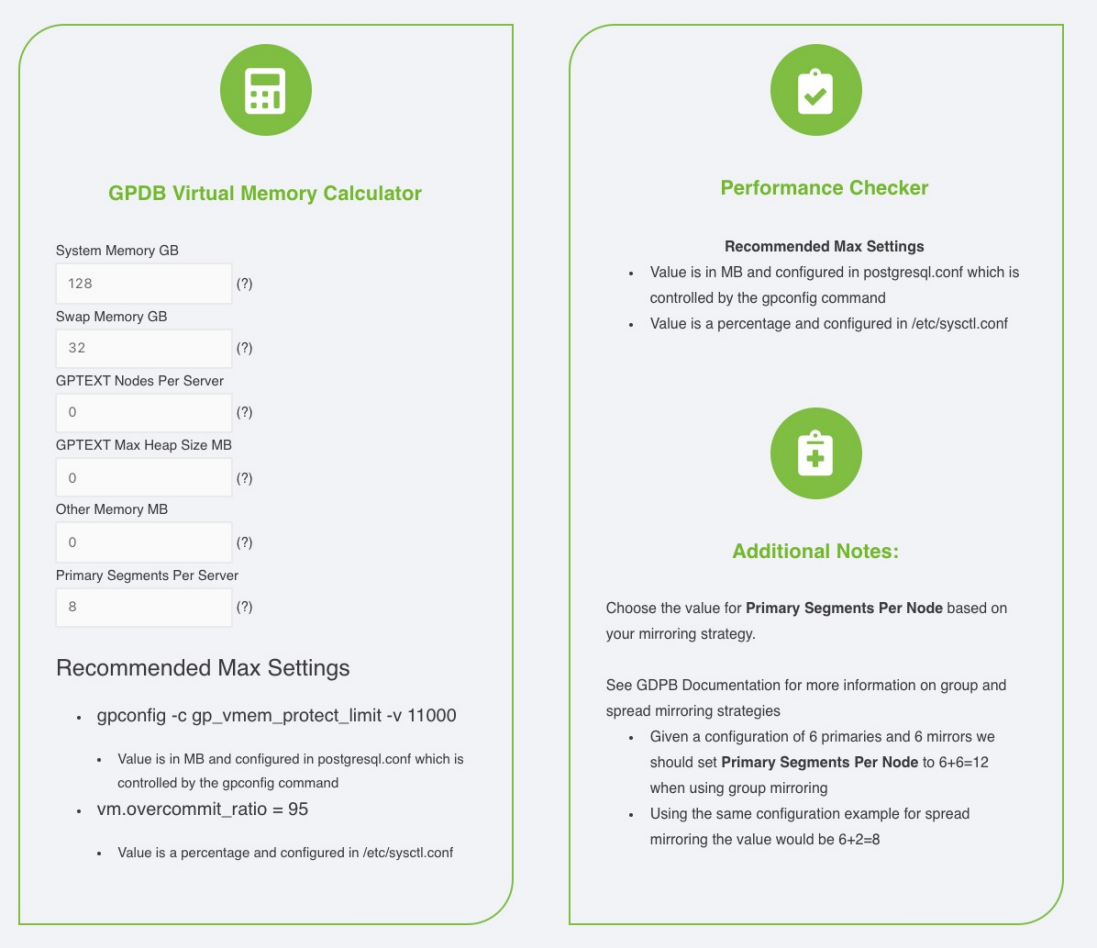
Greenplum优化要点

社区提供的工具及文档

数据库初始化参数预估
参考工具：https://greenplum.org/calc/

数据库初始化安装前，根据硬件规划情况，使用右侧计算器粗略计算各个vmem protect值；

其他各个指标，例如sysctl.conf/limits.conf/statement_mem等参数的配置，请参考安装文档具体计算公式：Installation Guide -> Setting the Greenplum Recommended OS Parameters



常用压测工具及目的

GREENPLUM DATABASE

- gpcheckperf - 定期对内存、网络、磁盘性能进行压测

  定期执行该测试，用于掌握集群硬件环境的变化情况，及时发现存在的潜在性能影响因素，并针对性的安排硬件调整计划；

- TPCH/TPC-DS - 选型场景下的标准测试集压测

  在选型场景下，与其他数据库选用同样的分析型测试模型，更有利于选择更有的方案，并能在同一测试场景下，不断查找并学习针对性优化方法；

  https://github.com/RunningJon/TPC-DS

- 业务场景定期压测

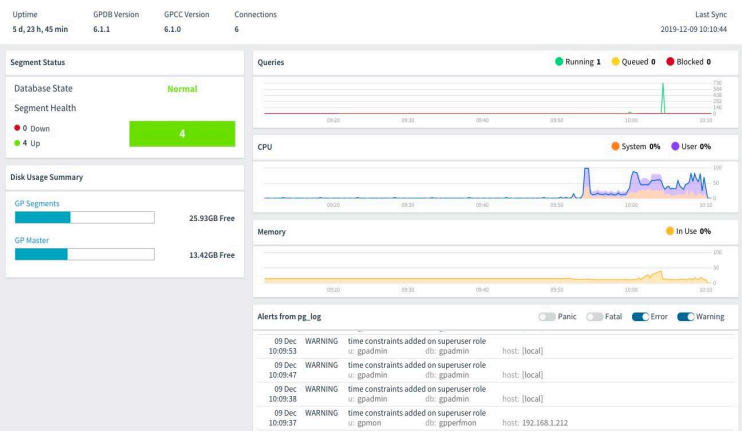
  定期进行业务压测，方便提前察觉长期运行状态下产生的性能下降问题，并针对性的进行性能提升操作，制定日常vacuum analyze操作计划，调整数据模型。

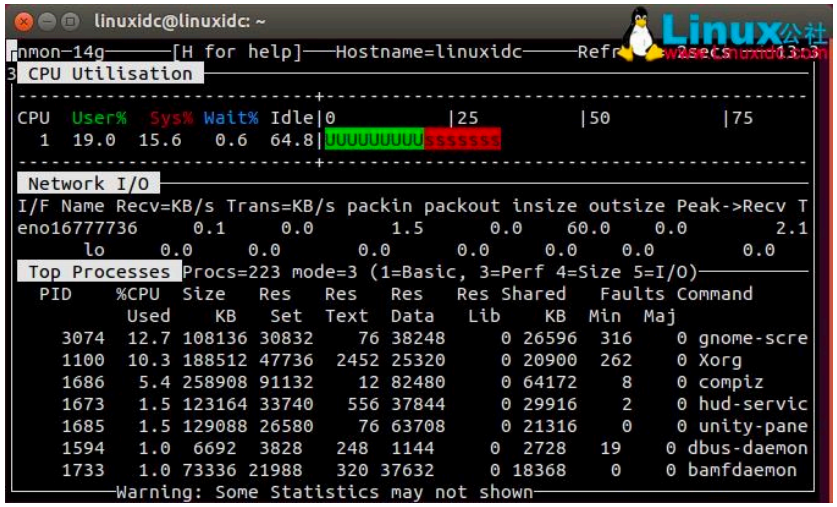
如何进行性能瓶颈分析

GREENPLUM DATABASE

```
postgres=# explain analyze select t4.name from t3 join t4 on t3.id=t4.id where t3.id < 5;
                                                    QUERY PLAN
-----------------------------------------------------------------------------------------------------------------------
 Gather Motion 2:1  (slice1; segments: 2)  (cost=0.00..862.01 rows=5 width=1) (actual time=2.064..2.765 rows=4 loops=1)
   ->  Hash Join  (cost=0.00..862.01 rows=3 width=1) (actual time=1.943..2.544 rows=3 loops=1)
         Hash Cond: (t3.id = t4.id)
         Extra Text: (seg0)   Hash chain length 1.0 avg, 1 max, using 3 of 524288 buckets.
         ->  Seq Scan on t3  (cost=0.00..431.00 rows=3 width=4) (actual time=0.002..0.006 rows=3 loops=1)
               Filter: (id < 5)
         ->  Hash  (cost=431.00..431.00 rows=3 width=4) (actual time=0.015..0.015 rows=3 loops=1)
               ->  Seq Scan on t4  (cost=0.00..431.00 rows=3 width=4) (actual time=0.008..0.012 rows=3 loops=1)
                     Filter: (id < 5)
 Planning time: 6.284 ms
   (slice0)    Executor memory: 59K bytes.
   (slice1)    Executor memory: 4170K bytes avg x 2 workers, 4170K bytes max (seg0).  Work_mem: 1K bytes max.
 Memory used:  128000kB
 Optimizer: Pivotal Optimizer (GPORCA)
 Execution time: 3.099 ms
(15 rows)

postgres=#
```





问题定位调优一：

GREENPLUM DATABASE

问题现象：

现场人员反馈，前端某一简单查询报表前两天查询还挺快的，这两天查询突然变慢了，而且越来越慢。

问题调查：

反馈的该业务表为分区事实表，每天凌晨数据入库；入库有支撑两种业务，一种为统计分析出报表到其他表；一种为前端网页程序提供简单查询。

请大家思考原因

解决办法：

首先查看该表上是否为前端查询提供了针对的索引，如果未提供，尝试使用索引；

如果已经提供索引，请尝试从执行计划下手查看是否正常走索引；

查看凌晨数据入库逻辑中是否有索引重建及统计信息收集操作，如果没有，请增加后处理。

问题定位调优二：

GREENPLUM DATABASE

问题现象：

现场人员反馈，基于某张表的查询全部超时，客户投诉严重。

问题调查：

反馈的业务所涉及的表是分区表，没有索引，均为批量入库操作，日常分析型查询响应时间均在秒级，今天的查询卡住属于突发现象。

请大家思考原因

解决办法：

登陆数据库，查看活动查询视图是否正常：select * from pg_stat_activity;

1）有锁 - 查看锁存在原因，八成因为入库晚正在执行truncate分区表等操作；

2）无锁 - 如果活动视图显示一切正常，那极有可能是segment节点由于某些非常规操作，存在锁表的孤立进程。

问题定位调优三：

GREENPLUM DATABASE

问题现象：

新开发的业务，在一张分区表上运行缓慢；该表上的其他查询业务均正常。

问题调查：

该表数据量比较大，也做了有效的分区；开发提供的查询业务属于统计分析语句，不需要索引的参与。

请大家思考原因

解决办法：

首先第一反应就应该是：没走正常的分区裁剪；

通过EXPLAIN ANALYZE查看开发人员提供的语句是否正常；如果不正常，需要针对性的做查询语句调优，通常情况下，这种问题出现在业务开发与数据库设计不匹配上。

GREENPLUM DATABASE®

微信技术讨论群
微信搜索添加“gp_assistant”
加入技术讨论

微信公众号
搜索添加“Greenplum中文社区”
技术干货、行业热点、活动预告

GREENPLUM
DATABASE
Q&A

感谢观看！