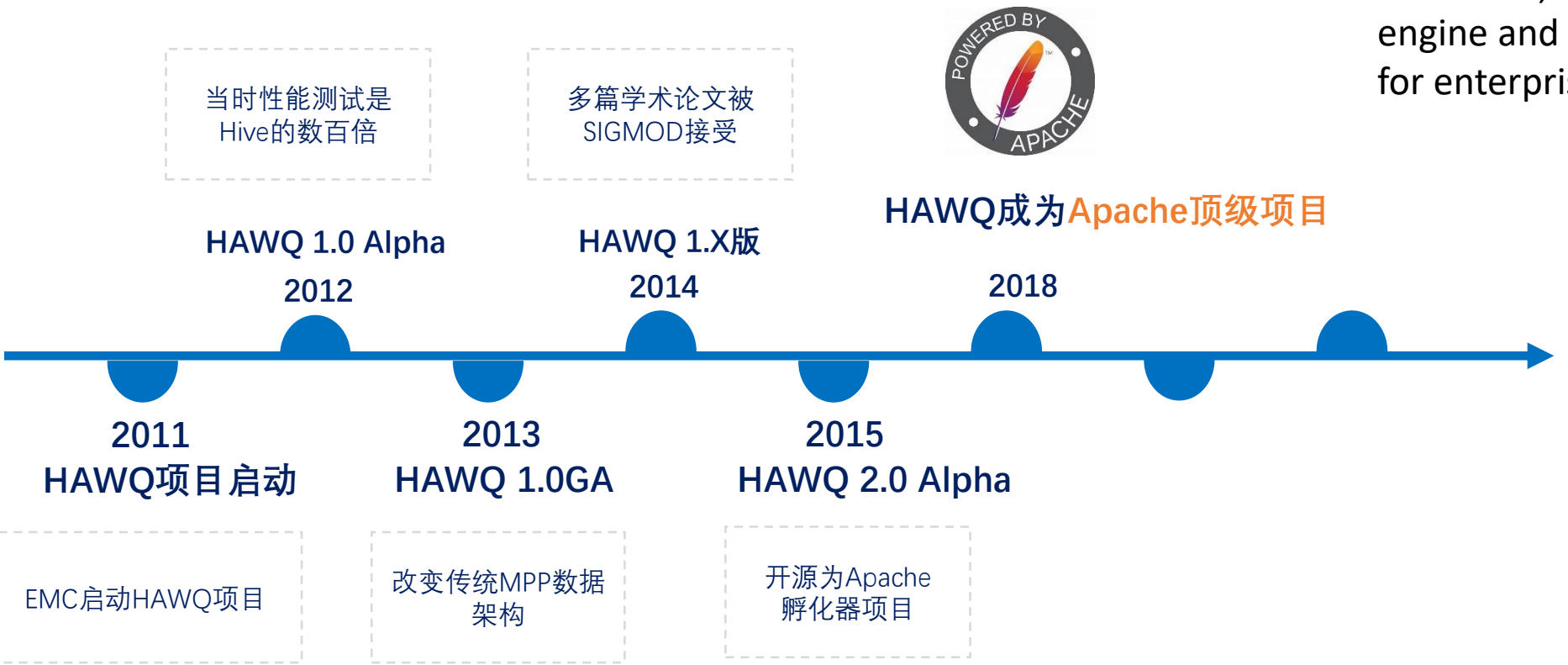
Apache HAWQ课堂授课

让人类只为兴趣而工作

oushu

Apache HAWQ

Apache Hadoop Native SQL. Advanced, MPP, elastic query engine and analytic database for enterprises.



http://hawq.apache.org

Apache社区和数据仓库简介

让人类只为兴趣而工作

目录

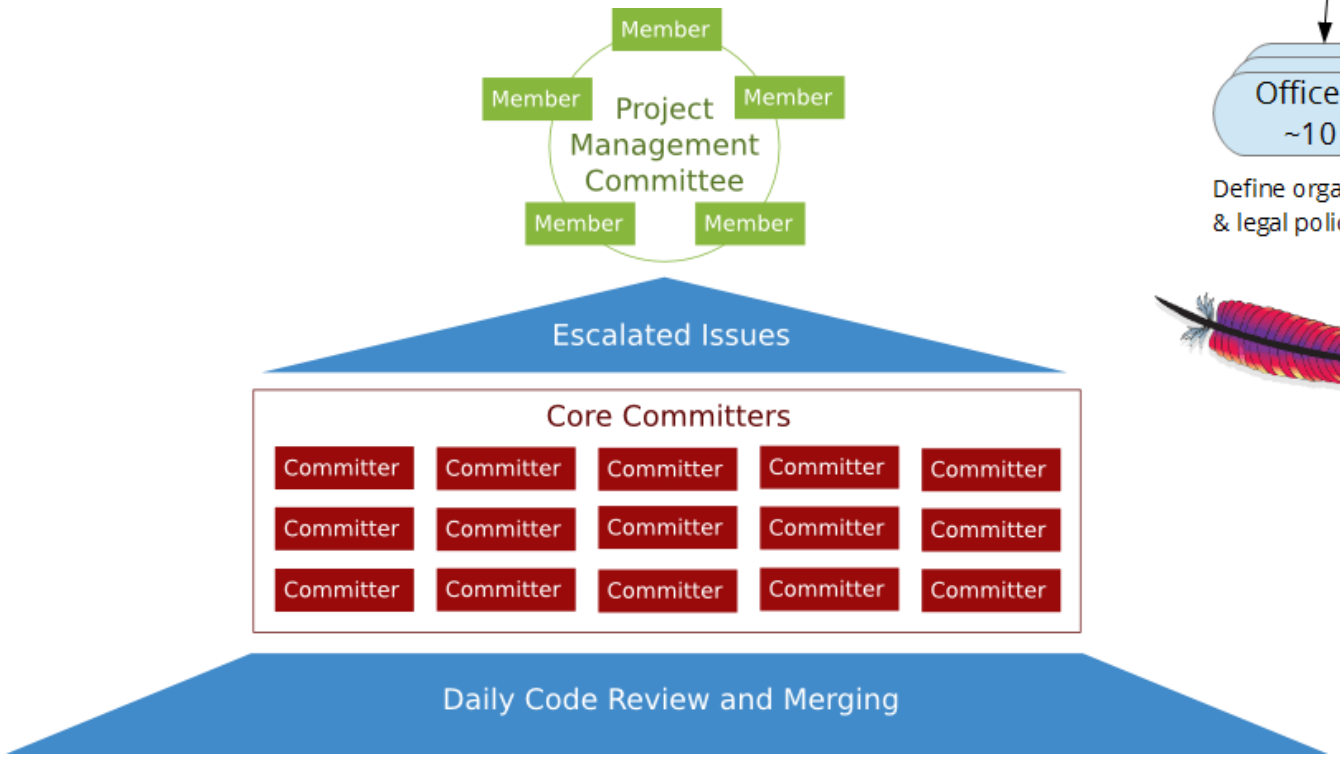
CONTENT

Apache Software Foundation

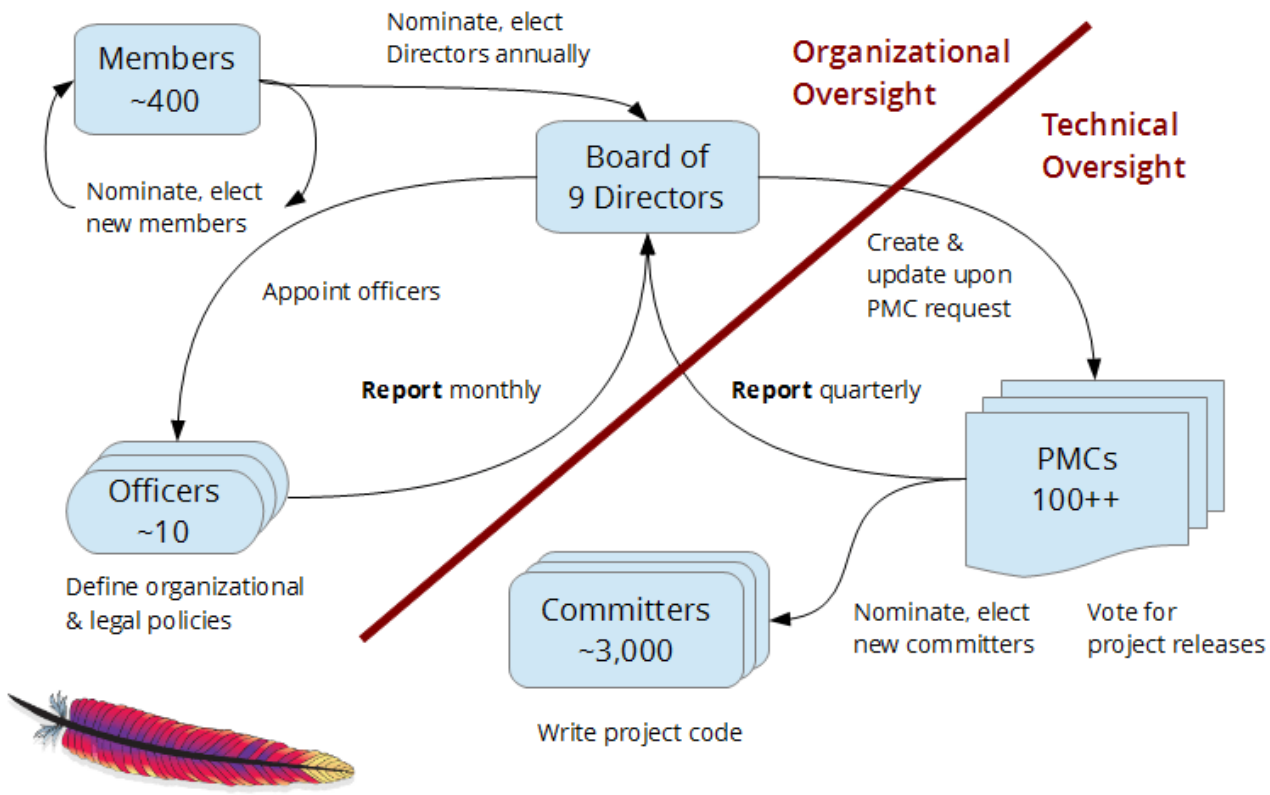
- apache http server
- hadoop

https://apache.org/

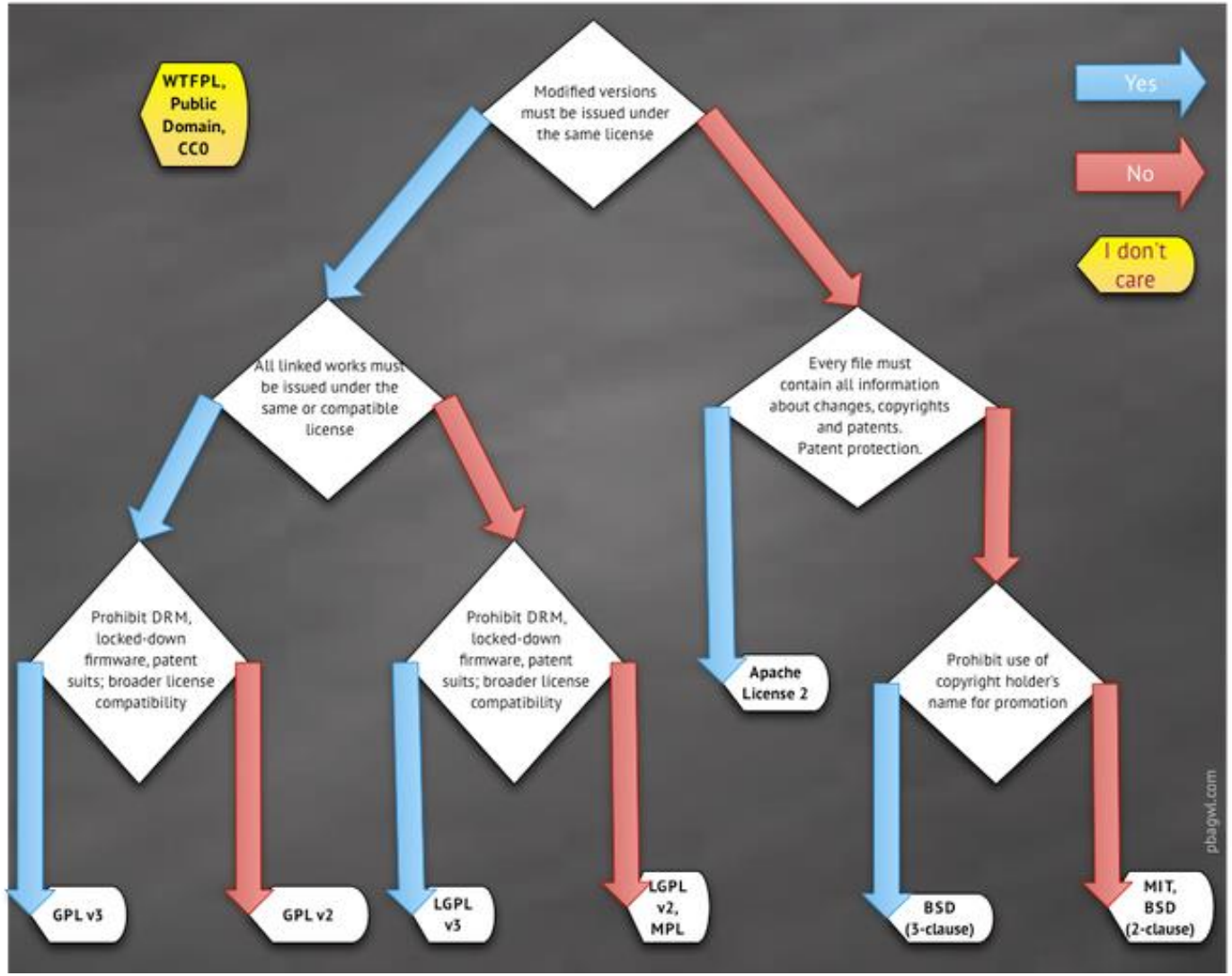
Apache Project



Apache Org Chart



Apache License



- 修改后可以闭源
- 每个源文件必须包含许可声明

目录

CONTENT

- 个体重复系统发育

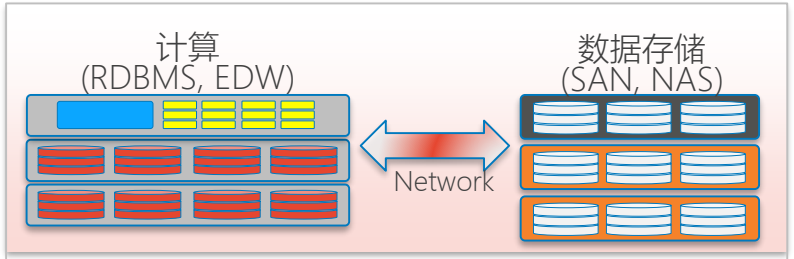
- 计算机科学和许多领域一样，主要是由技术驱动的。古罗马人缺少汽车的原因不是因为他们非常喜欢步行，是因为他们不知道如何造汽车。个人计算机的存在，不是因为数以百万计的人有着迫切的愿望—拥有一台计算机，而是因为现在可以很便宜地制造它们。我们常常忘了技术是如何影响我们对各种系统的观点的，所以有时值得再仔细考虑它们。

数据仓库简介

- 数据库/数据仓库
- 并行数据库/分布式数据库
- 数据划分/并行处理
- 吞吐量/响应时间
- 存储与计算分离/共享存储
- OLAP/DSS
- MPP
- SQL on Hadoop
- Cloud Native

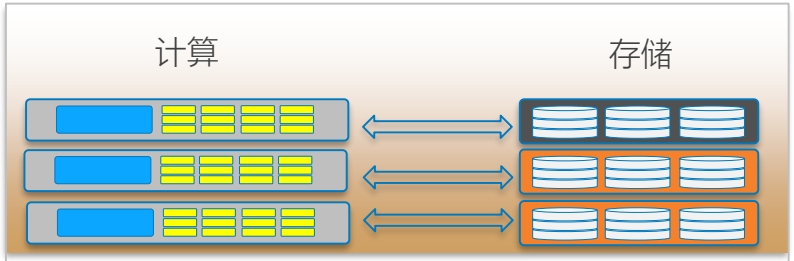
分析型数据库演进路线

传统数据仓库：专有硬件 (1980s)



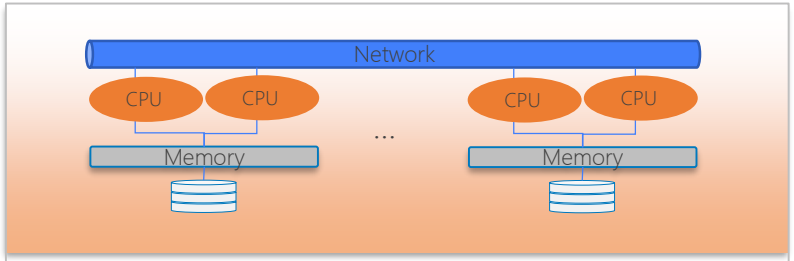
- 应用场景：报表分析
- 节点规模：十几；扩展困难
- SQL兼容性：好
- 性能：中
- 云支持：差
- 系统代表：Oracle，DB2，Teradata

SQL-on-Hadoop：存储与计算分离 (2000s)



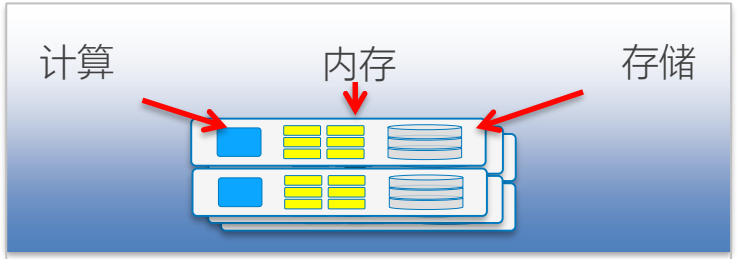
- 应用场景：大数据分析
- 节点规模：上千节点；
- SQL兼容性：差
- 性能：差
- 云支持：中
- 系统代表：Hive，SparkSQL

基于x86的MPP数据库：无共享 (2000s)



- 应用场景：大数据分析
- 节点规模：几百节点；扩展困难
- SQL兼容性：好
- 性能：中等
- 云支持：差
- 系统代表：Greenplum, Vertica

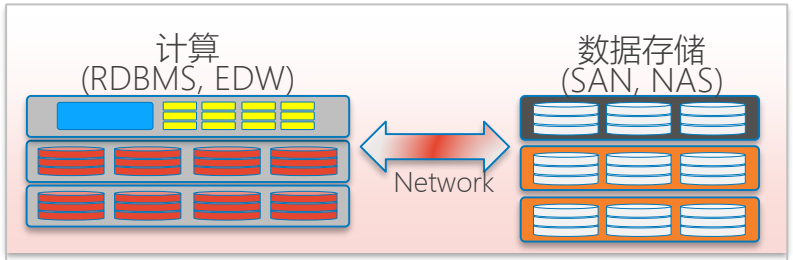
面向AI的云数据库：云原生(2015前后)



- 应用场景：AI、云、物联网和大数据
- 节点规模：数千节点；
- SQL兼容性：好
- 性能：好
- 云支持：原生
- 系统代表：Snowflake

分析型数据库演进路线

传统数据仓库：专有硬件 (1980s)



- 应用场景：报表分析
- 节点规模：十几；扩展困难
- SQL兼容性：好
- 性能：中
- 云支持：差
- 系统代表：Oracle，DB2，Teradata

SQL-on-Hadoop：存储与计算分离 (2000s)

基于x86的MPP数据库：无共享 (2000s)

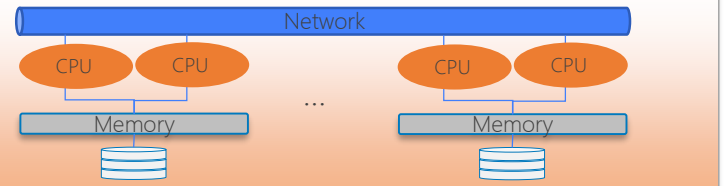
面向AI的云数据库：云原生(2015前后)

分析型数据库演进路线

传统数据仓库：专有硬件 (1980s)

SQL-on-Hadoop：存储与计算分离 (2000s)

基于x86的MPP数据库：无共享 (2000s)



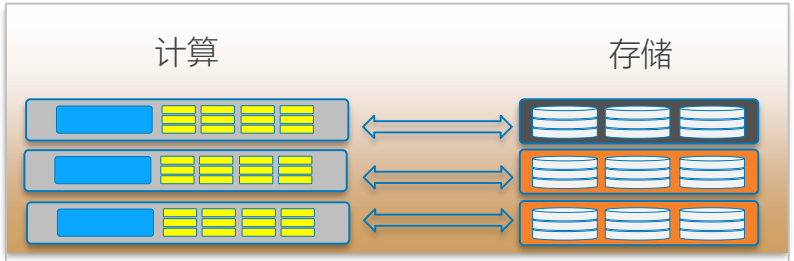
- 应用场景：大数据分析
- 节点规模：几百节点；扩展困难
- SQL兼容性：好
- 性能：中等
- 云支持：差
- 系统代表：Greenplum, Vertica

面向AI的云数据库：云原生(2015前后)

分析型数据库演进路线

传统数据仓库：专有硬件 (1980s)

SQL-on-Hadoop：存储与计算分离 (2000s)



- 应用场景：大数据分析
- 节点规模：上千节点；
- SQL兼容性：差
- 性能：差
- 云支持：中
- 系统代表：Hive，SparkSQL

基于x86的MPP数据库：无共享 (2000s)

面向AI的云数据库：云原生(2015前后)

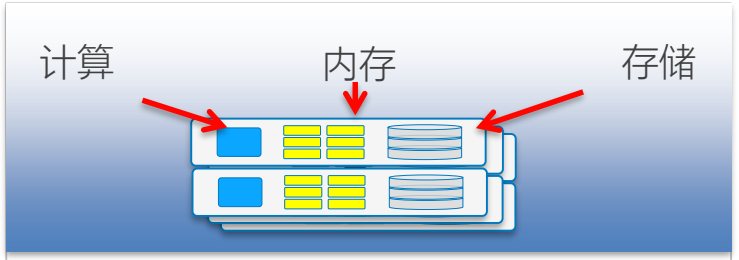
分析型数据库演进路线

传统数据仓库：专有硬件 (1980s)

SQL-on-Hadoop：存储与计算分离 (2000s)

基于x86的MPP数据库：无共享 (2000s)

面向AI的云数据库：云原生(2015前后)

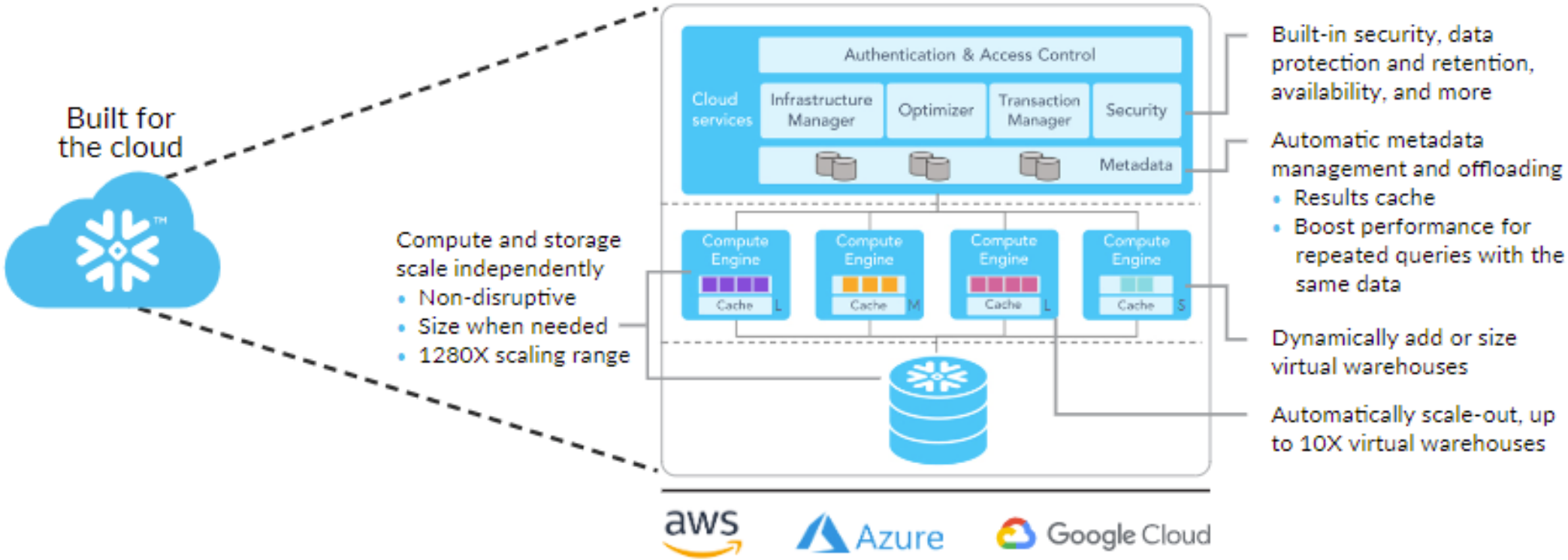


- 应用场景：AI、云、物联网和大数据
- 节点规模：数千节点；
- SQL兼容性：好
- 性能：好
- 云支持：原生
- 系统代表：Snowflake

目录

CONTENT

Snowflake



https://www.snowflake.com/product/architecture/

小结

- Apache社区工作方式
- 数据仓库
- 云原生数据仓库

数据仓库架构和技术介绍

让人类只为兴趣而工作

oushu

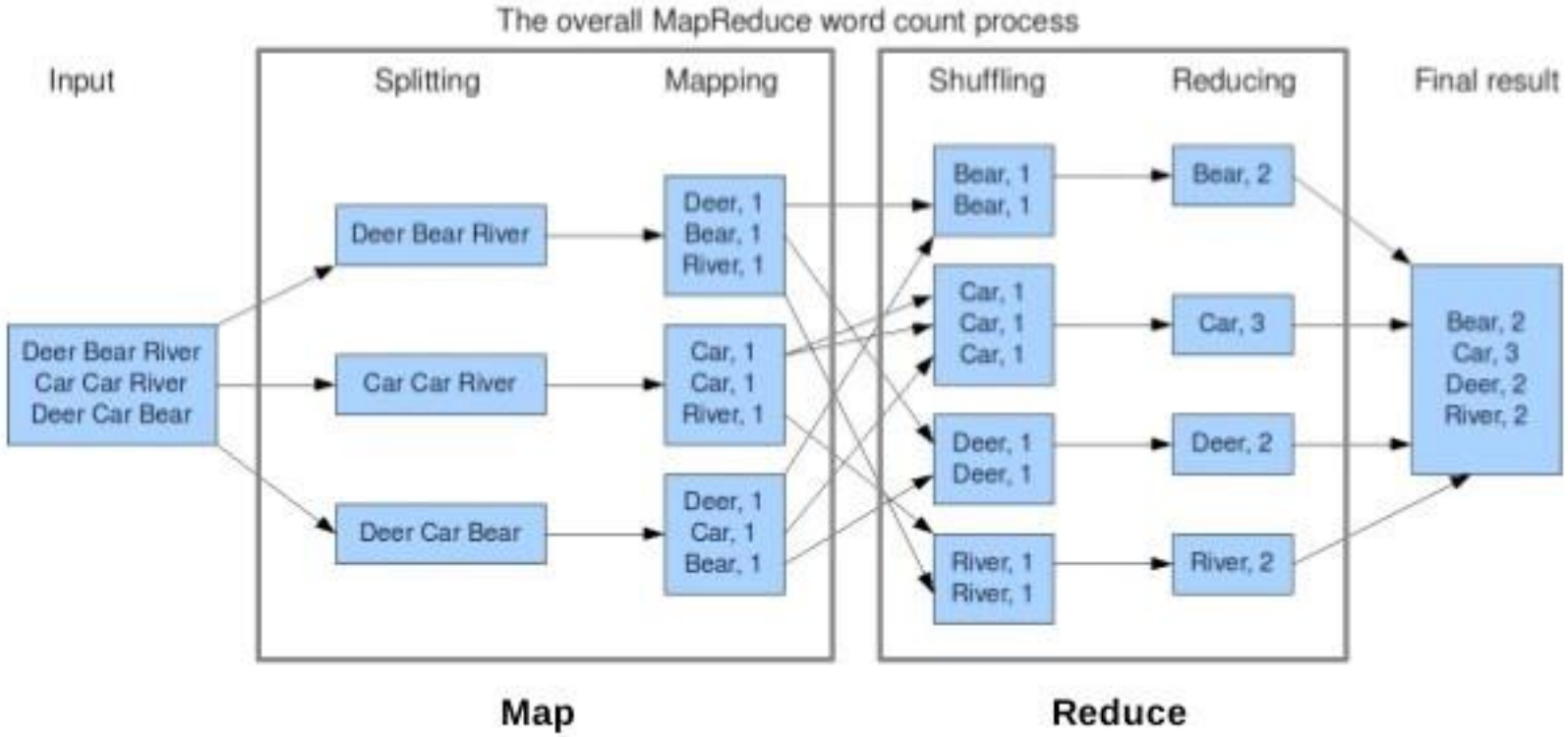
目录

CONTENT

- 主从结构和P2P结构
- 单点失效

- 数据划分与偏斜
- 算子内并行，算子间并行

从MapReduce说起



Google三驾马车：GFS、MapReduce和Bigtable

Google *Re*-Engineering

Google File System (GFS) → HDFS

MapReduce → hadoop

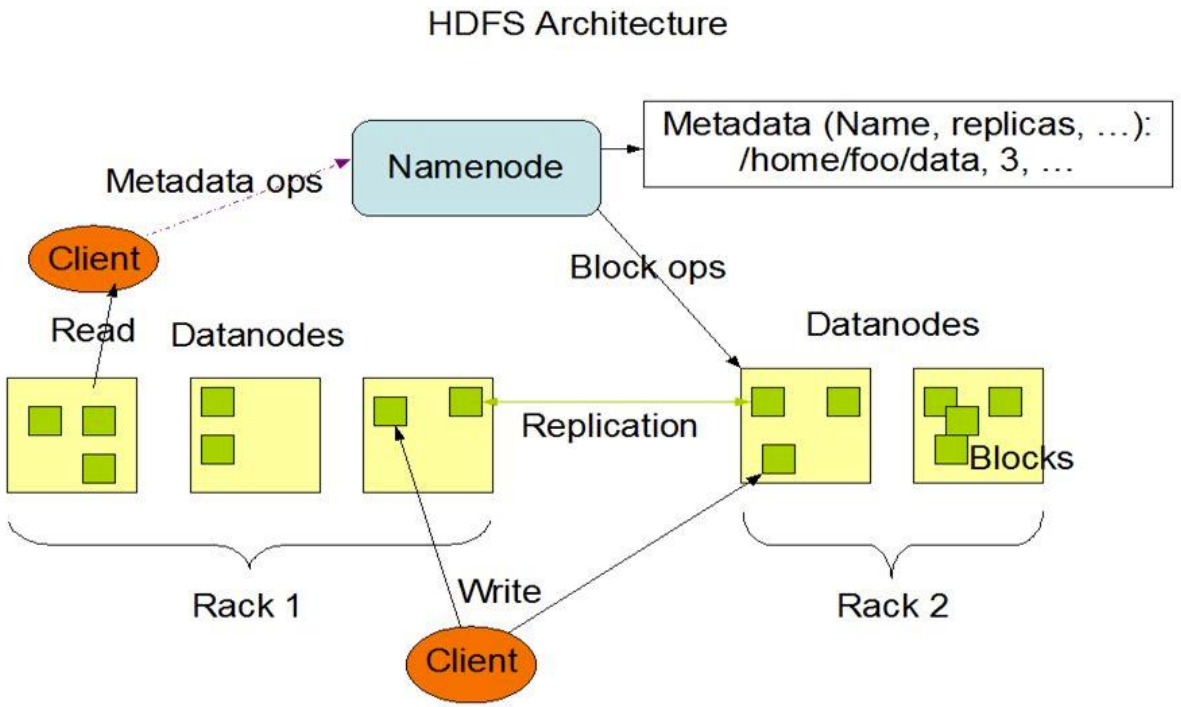
BigTable → APACHE HBASE

目录

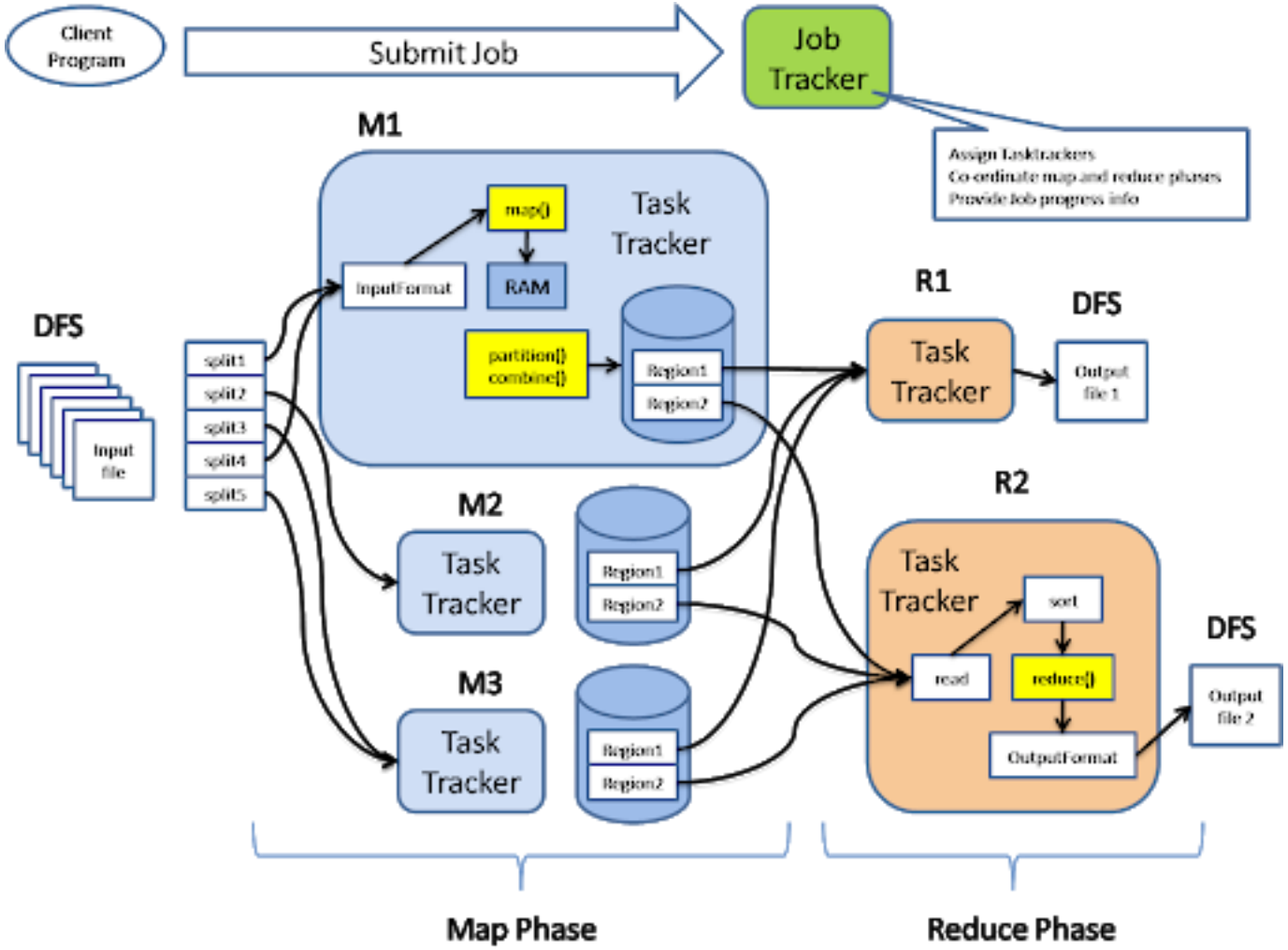
CONTENT

HDFS

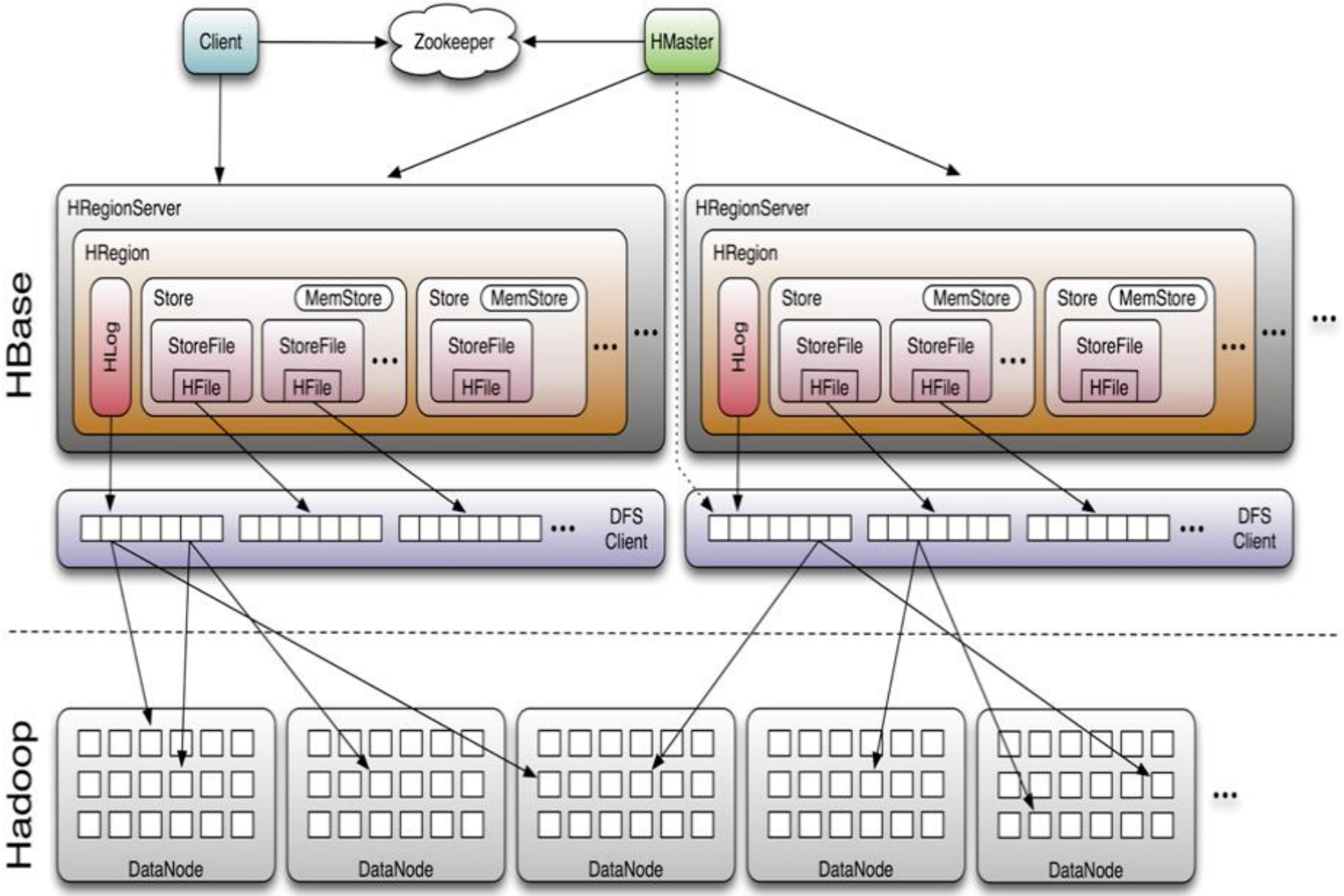
HADOOP ARCHITECTURE



Hadoop MapReduce



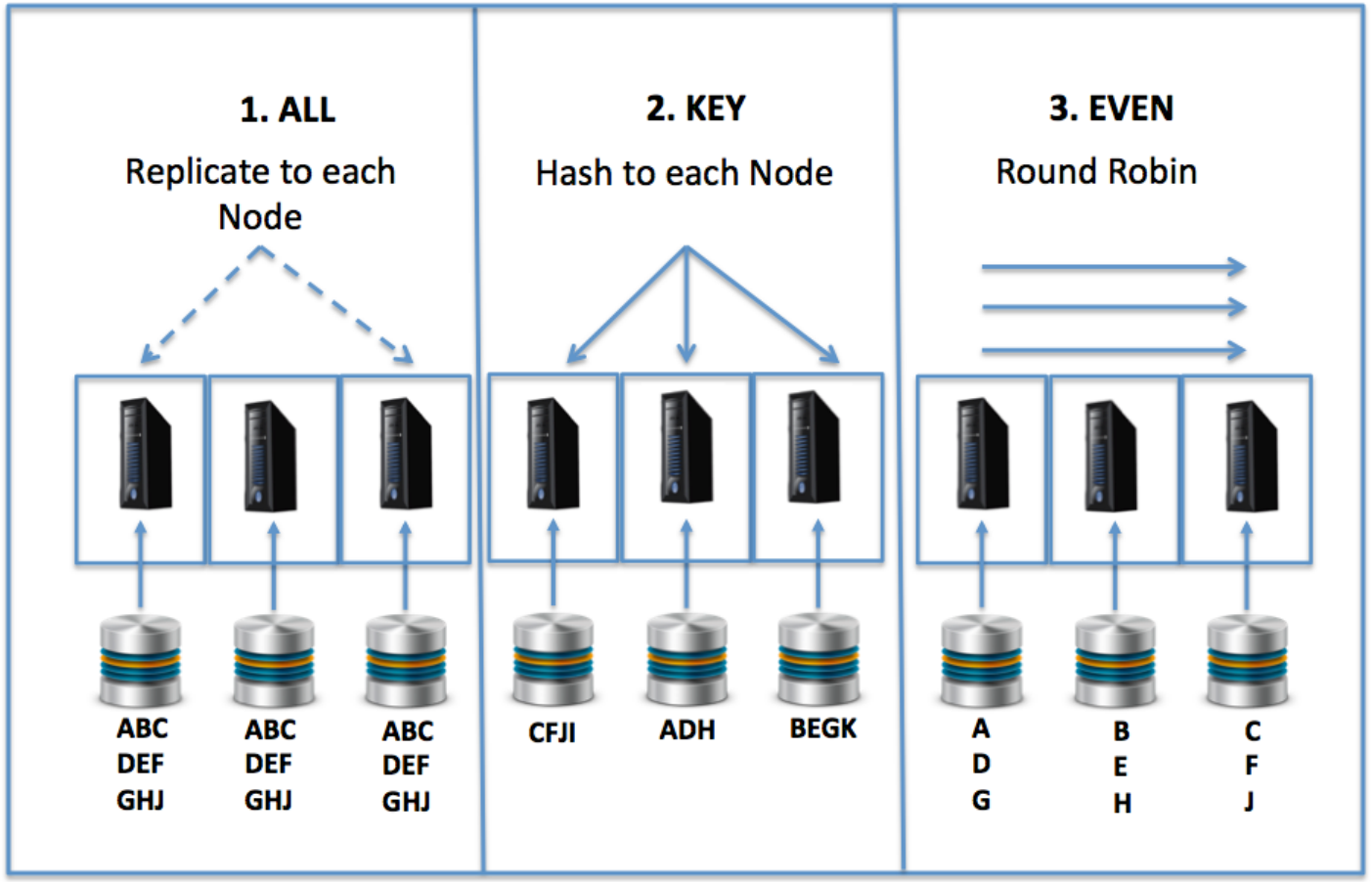
HBASE


Client
Zookeeper
HMaster
HBase
HRegionServer
HRegion
HLog
Store
MemStore
StoreFile
HFile
DFS Client
Hadoop
DataNode

目录

CONTENT

Three MPP Data Distribution Styles



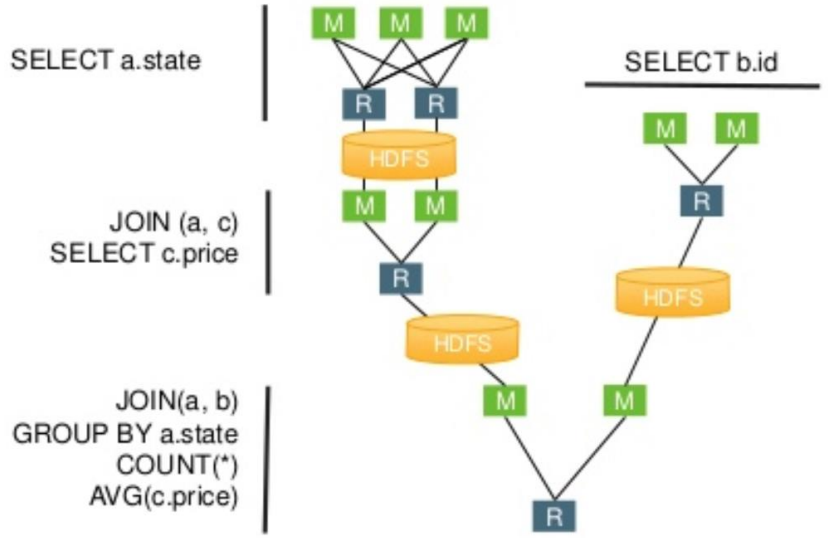
开源数据仓库实例

让人类只为兴趣而工作

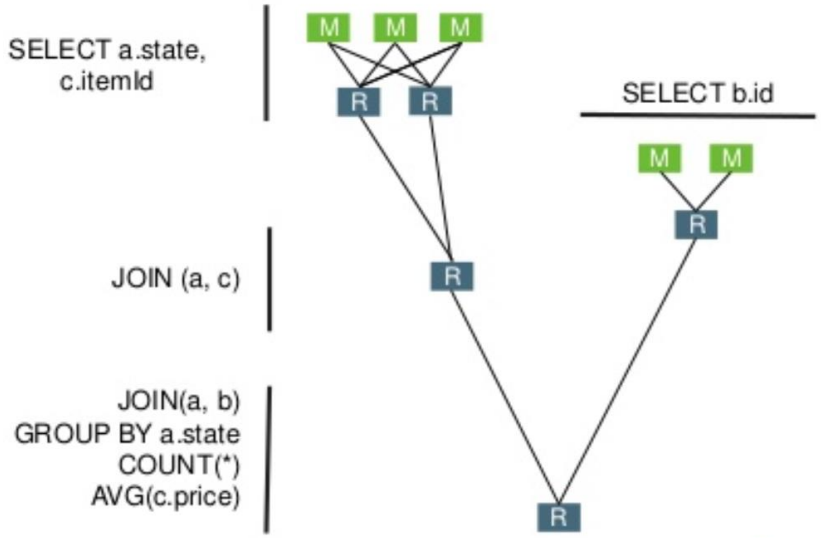
```
SELECT a.state, COUNT(*), AVG(c.price)
FROM a
JOIN b ON (a.id = b.id)
JOIN c ON (a.itemId = c.itemId)
GROUP BY a.state
```

Tez avoids unneeded writes to HDFS

Hive – MapReduce



Hive – Tez



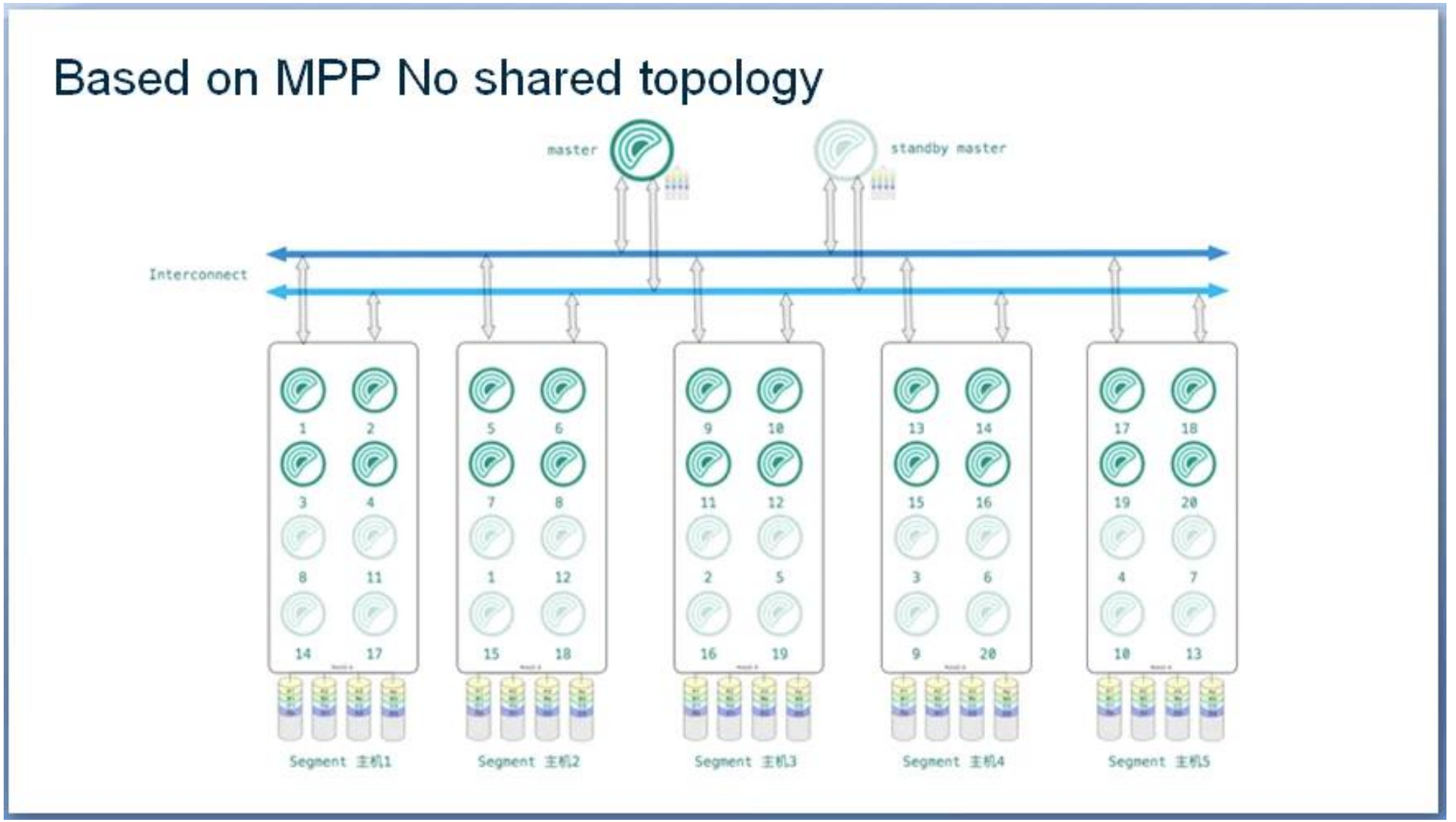
Hortonworks

系统和技术实践

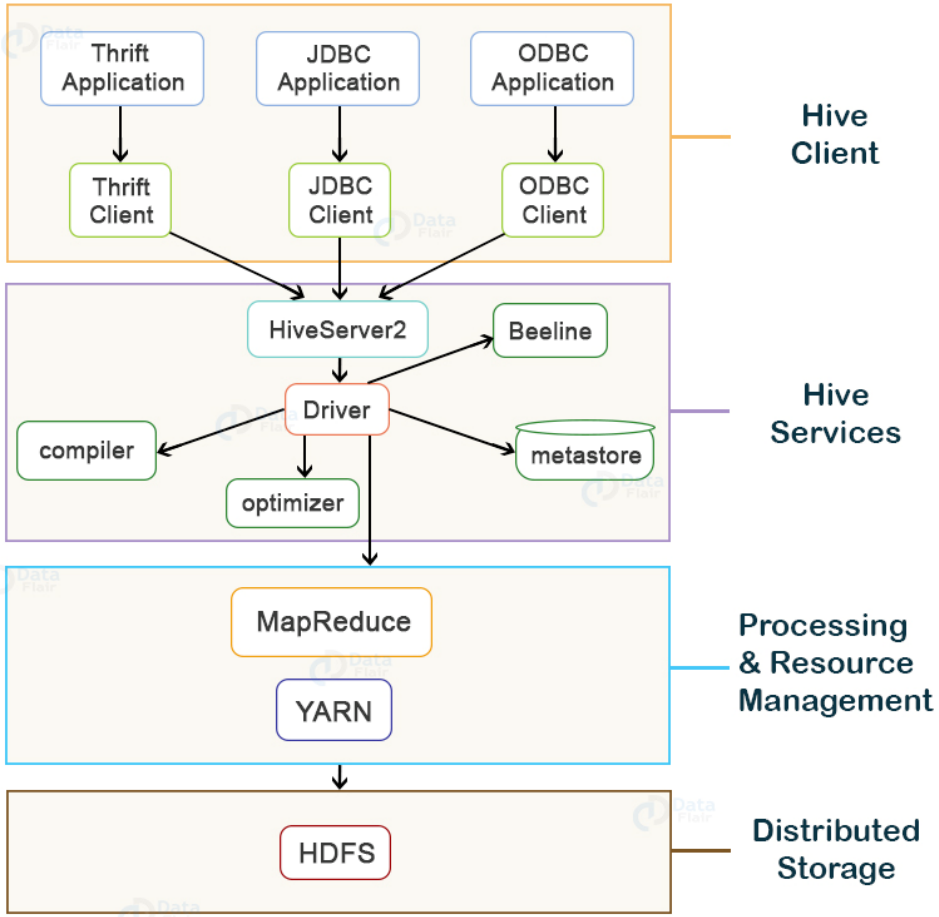
- Greenplum
- Hive
- Impala

- IO划分技术
  - hash
  - random
  - range
- 互联网络
  - RPC
  - udpifc

Greenplum

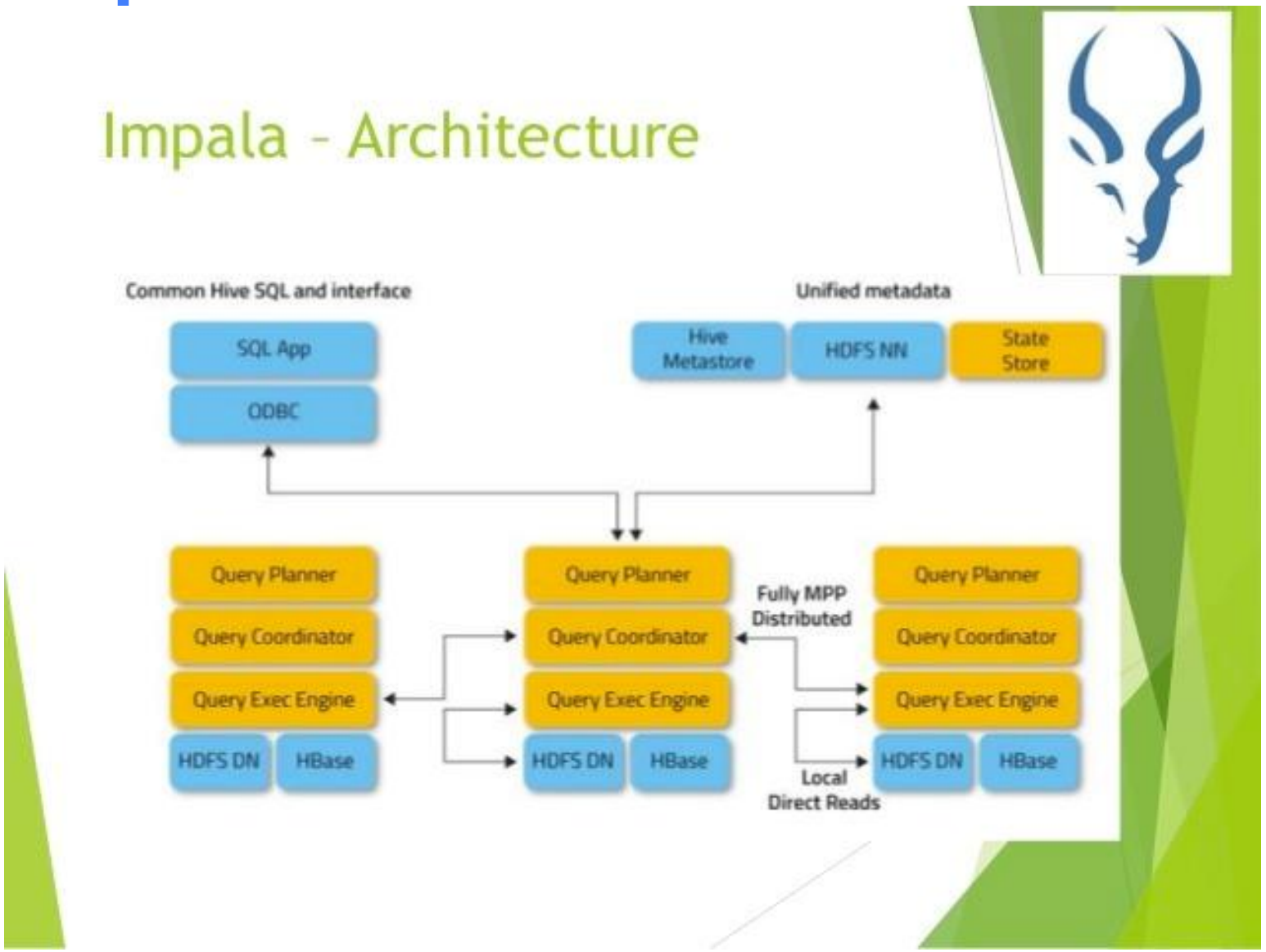

Based on MPP No shared topology
master
standby master
Interconnect
1 2 3 4 8 11 14 17
5 6 7 8 1 12 15 18
9 10 11 12 2 5 16 19
13 14 15 16 3 6 9 20
17 18 19 20 4 7 10 13
Segment 主机1
Segment 主机2
Segment 主机3
Segment 主机4
Segment 主机5

Hive



Hive Architecture & Its Components

Impala


Impala - Architecture
Common Hive SQL and interface
SQL App
ODBC
Unified metadata
Hive Metastore
HDFS NN
State Store
Query Planner
Query Coordinator
Query Exec Engine
HDFS DN
HBase
Query Planner
Query Coordinator
Query Exec Engine
HDFS DN
HBase
Fully MPP Distributed
Query Planner
Query Coordinator
Query Exec Engine
HDFS DN
HBase
Local Direct Reads

数据仓库实例讲解——
Apache HAWQ
系统架构和基本原理

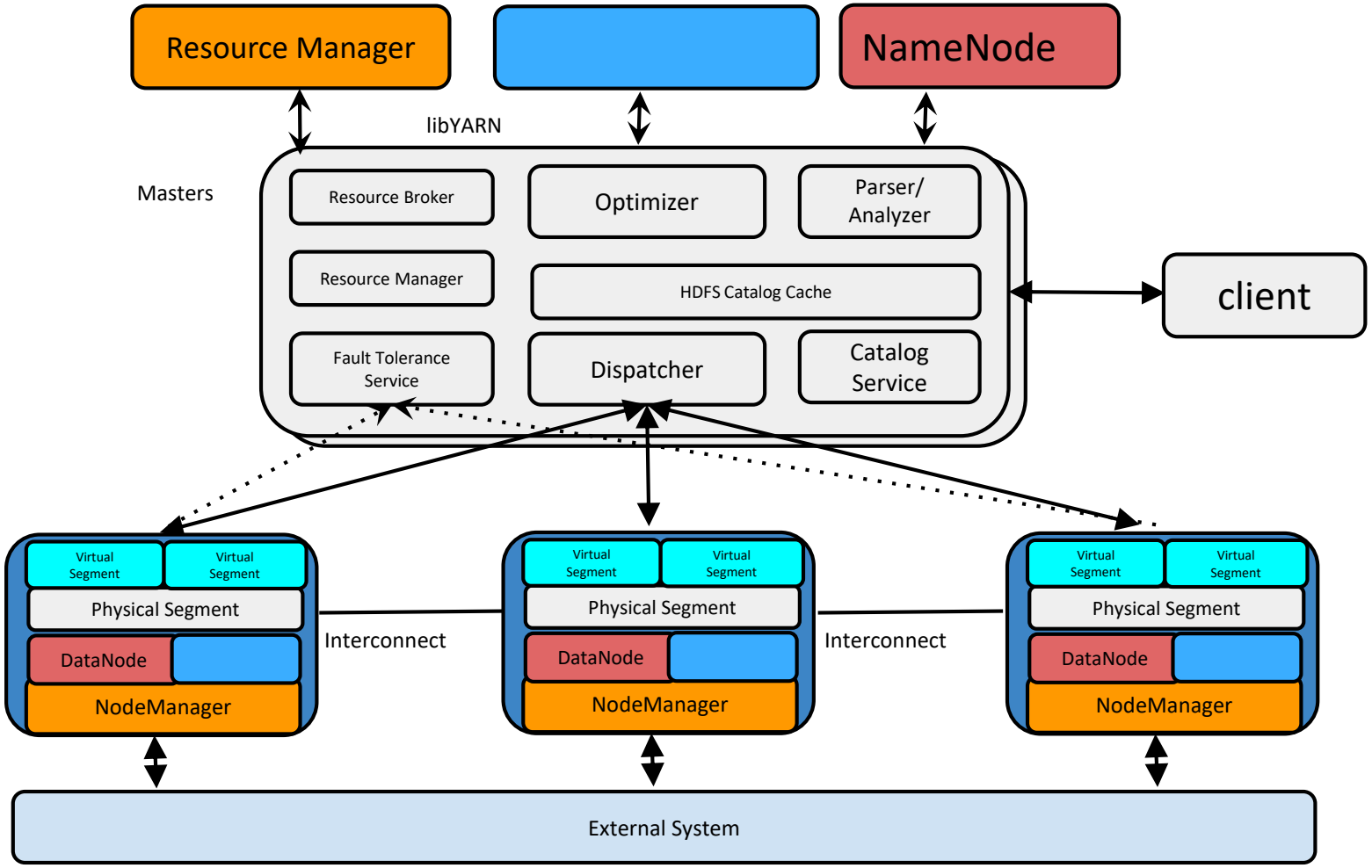
oushu

目录

CONTENT

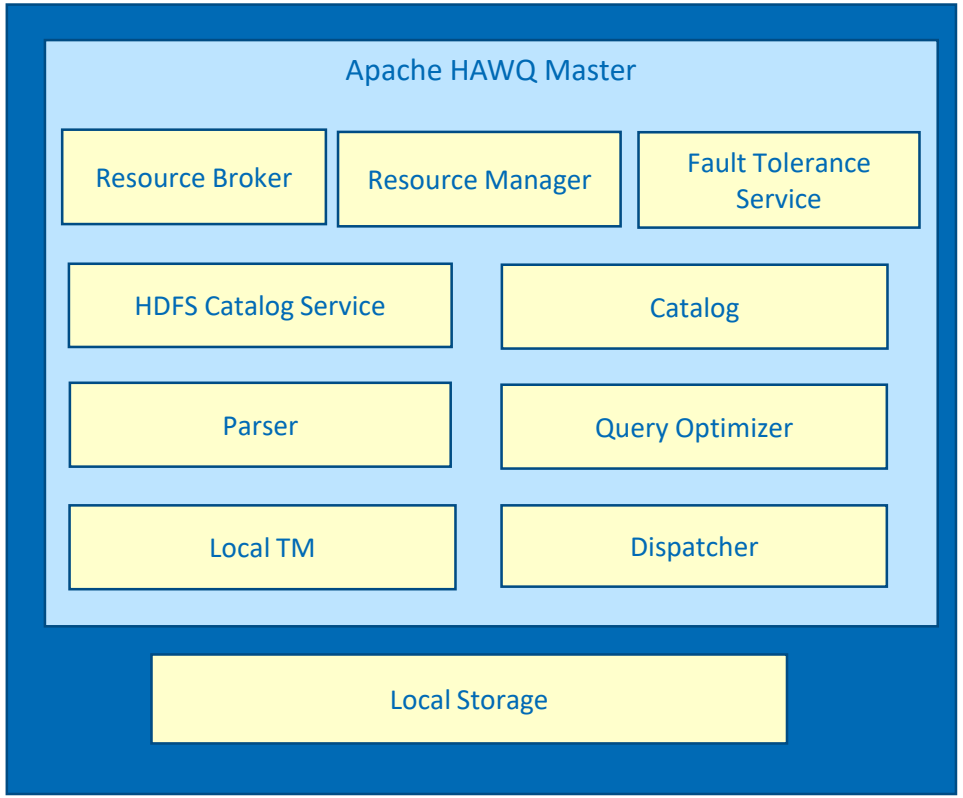
01 Apache HAWQ 基本组件

Apache HAWQ基本组件



Apache HAWQ Master

- 元数据信息仅存储在master的本地存储中
- 认证客户和接受客户的要求
- 将SQL转换为解析树数据结构
- SQL优化器决定最有效的并行查询执行
- 将查询计划分发segment
- 将结果返回给客户端
- 执行事务管理



Apache HAWQ Physical Segment & Virtual Segment

- 一个 Apache HAWQ segment host 是一个计算节点
- 每个segment host可以有多个虚拟segment,一般为（2，4，6，8）， 缺省为8
- 一个虚拟segment是一个并行查询的最基本计算单元（资源的container， 并行度的弹性）
- 多个部分协同工作，形成一个单一的并行查询处理系统。
- Apache HAWQ segments向HDFS传递请求所需的数据节点完成数据查询。
- Segment和Master的区别：
  - 无状态
  - 不存储数据库表元数据
  - 不存储本地文件系统中的数据

Apache HAWQ基本组件

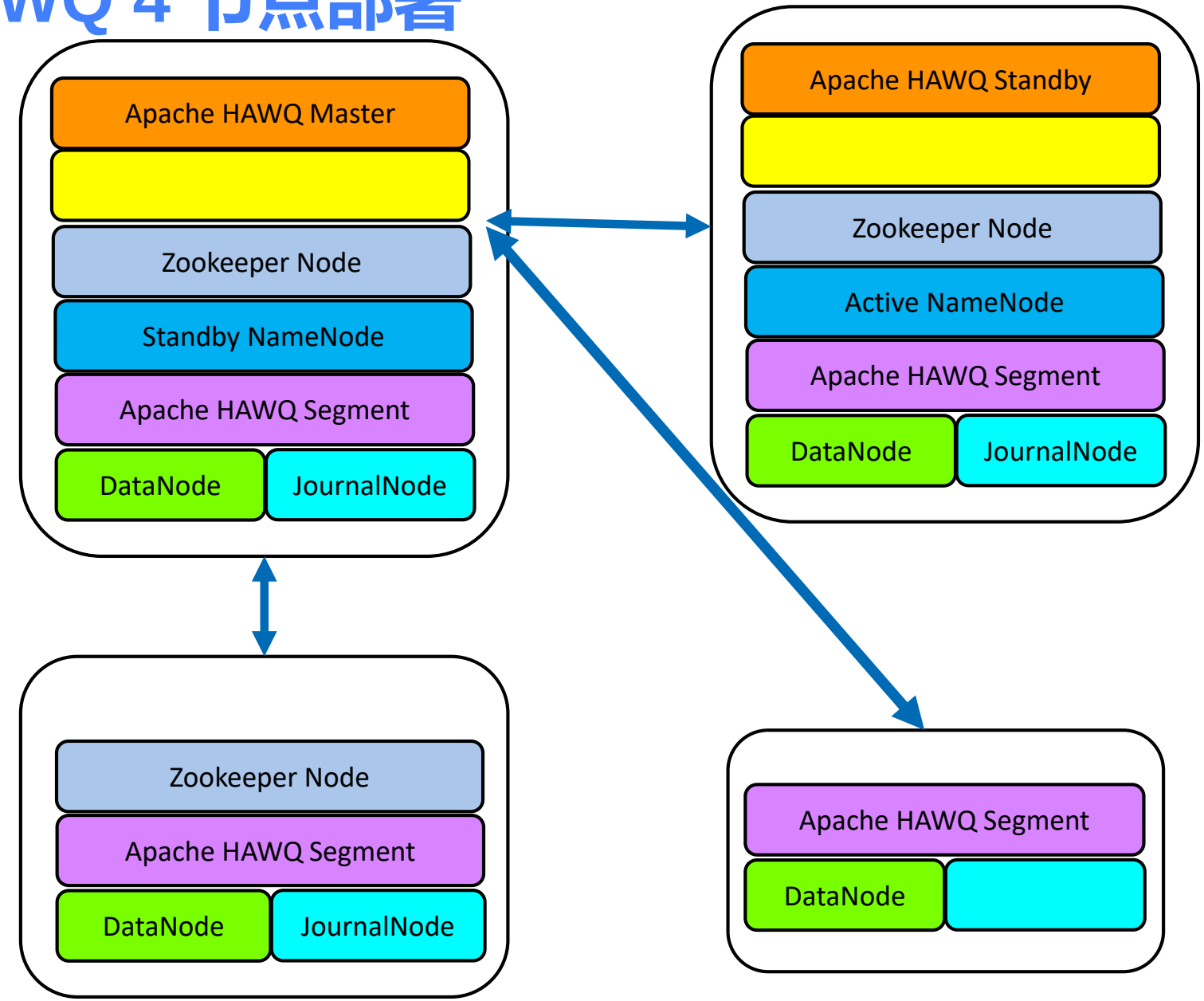
- HDFS（可选）
  - Namenode：管理文件系统的元数据，在一个集群中配置两个不同的机器来提供HA的服务
  - Datanode： 管理该节点上的数据存储
- Zookeeper （可选）
  - 负责为分布式应用提供一致性服务。
  - Zookeeper 会实时获取HDFS 两个Namenode 的心跳信息，从而在必要时触发namenode 的切换。

Apache HAWQ基本组件

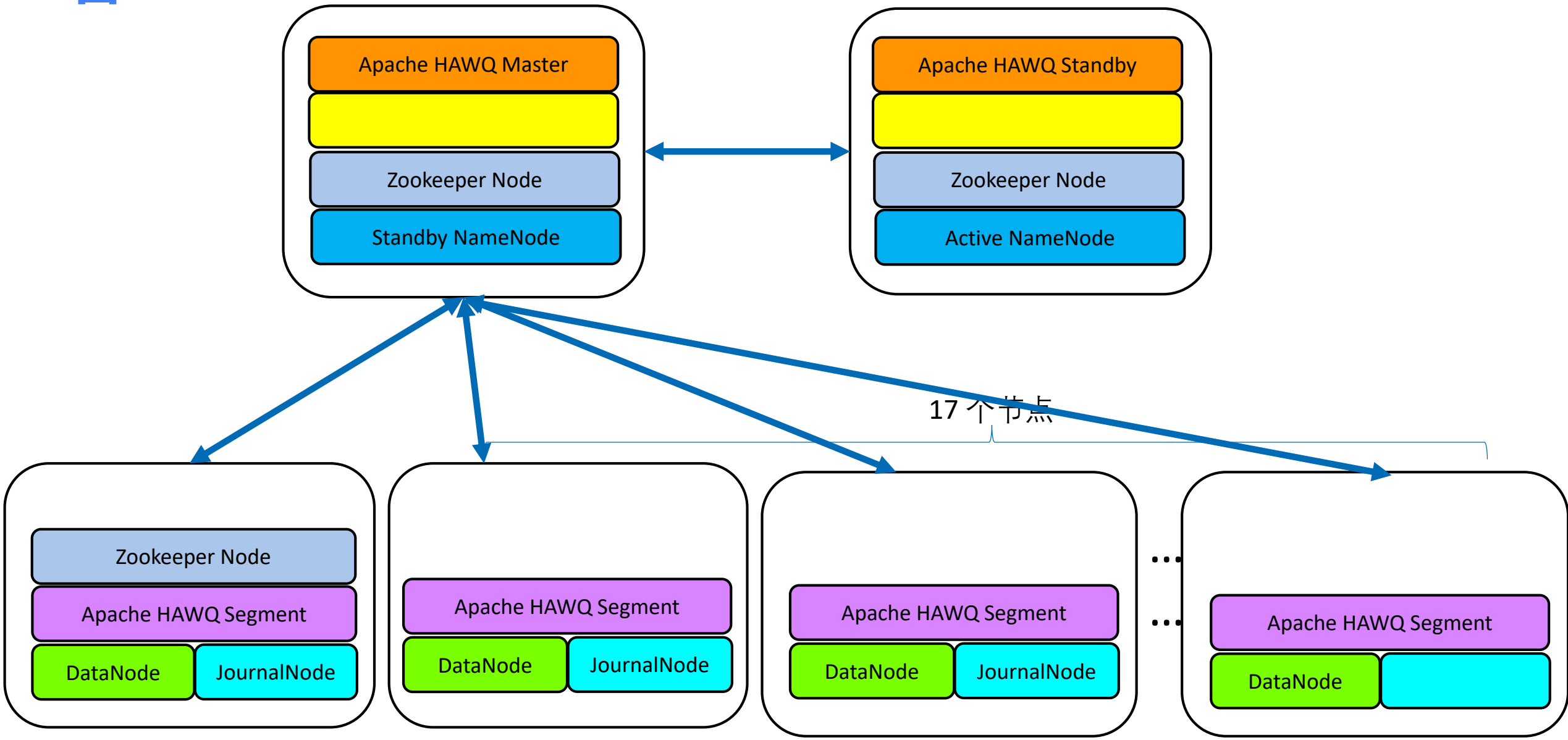
- Yarn（可选）
- Apache HAWQ 提供了Standalone 和Yarn 两种模式来进行资源管理
- Standalone： 缺省模式， 独占系统资源。
- Yarn： Apache HAWQ 作为Yarn 的一个应用程序，通过libyarn向Yarn 来申请资源。
- Resource Manager
  - 全局的资源管理，负责整个系统的资源管理和分配。
- Node Manager
  - 负责本地节点资源管理

02 Apache HAWQ 部署架构

Apache HAWQ 4 节点部署



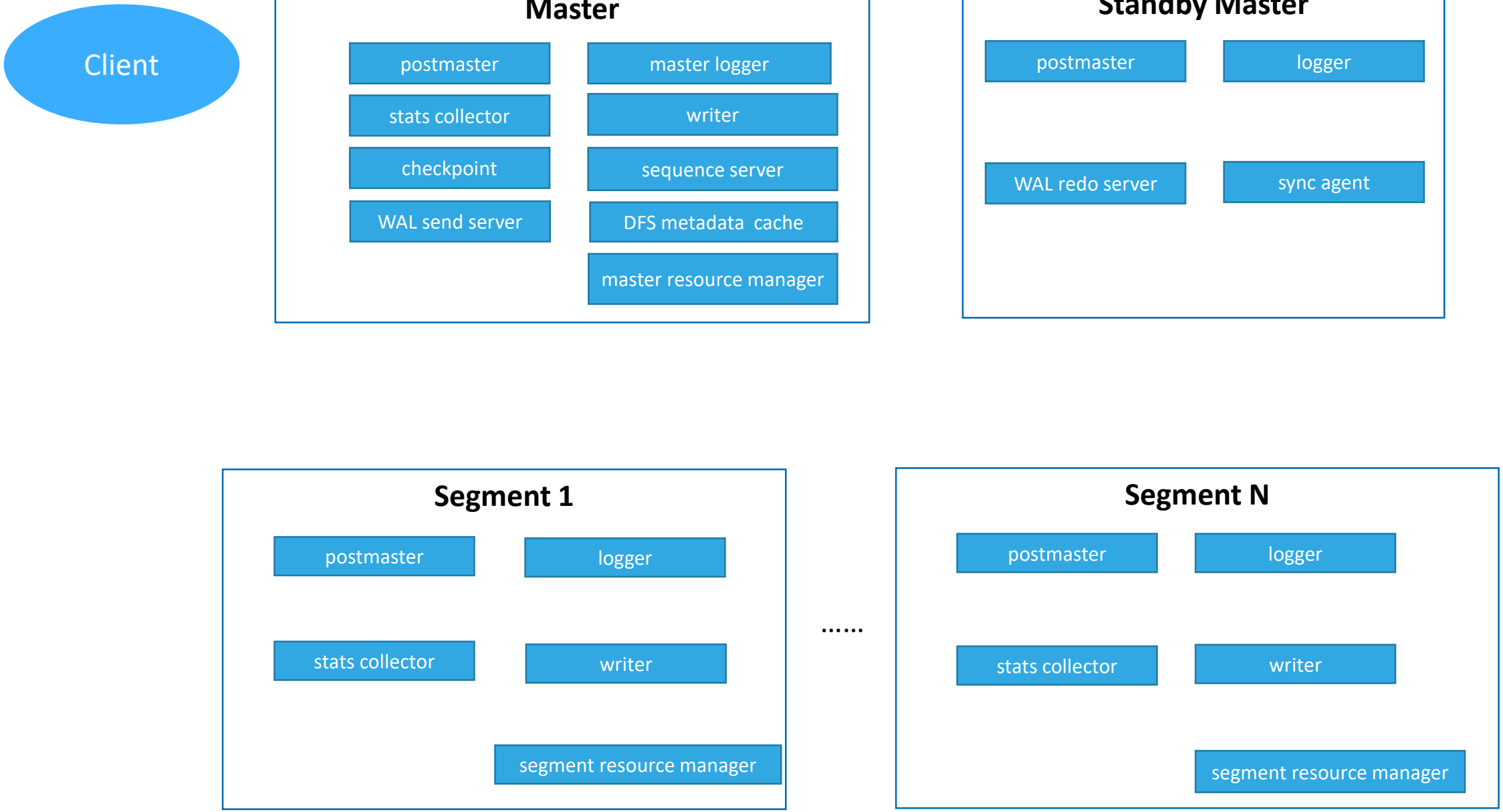
Apache HAWQ 20 节点部署



03 Apache HAWQ 进程架构

oushu

进程架构

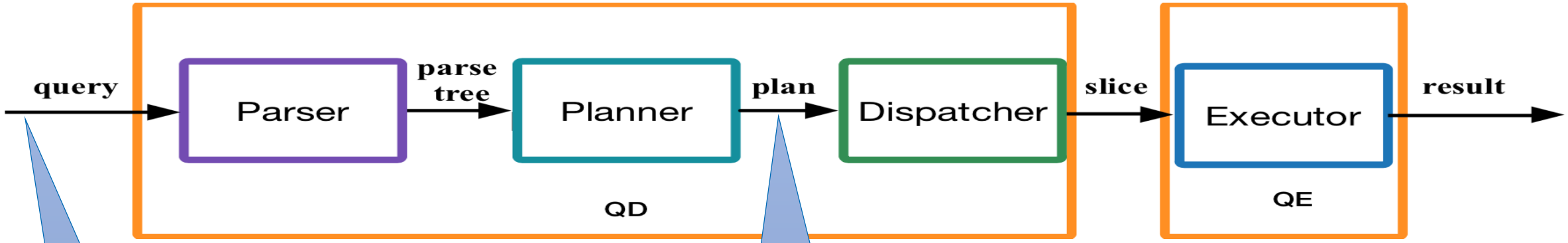

Client
Master
postmaster
master logger
stats collector
writer
checkpoint
sequence server
WAL send server
DFS metadata cache
master resource manager
Standby Master
postmaster
logger
WAL redo server
sync agent
Segment 1
postmaster
logger
stats collector
writer
segment resource manager
.....
Segment N
postmaster
logger
stats collector
writer
segment resource manager

进程架构

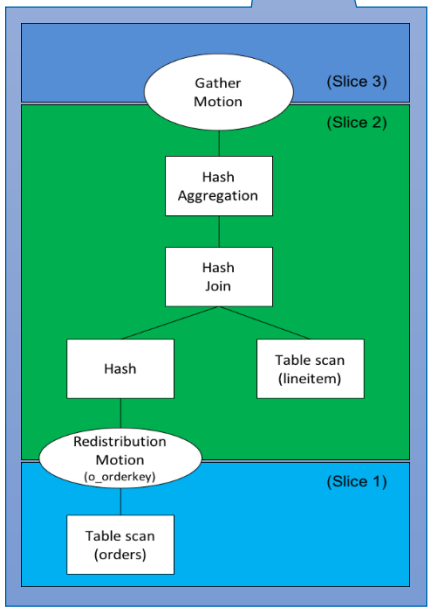
- Postmaster: 监听用户请求的守护进程，为每个客户端连接fork单独的postgres服务，当postgres 进程出错时尝试修复。
- Logger: 负责收集各个子进程的输出，并将其写到pg_log。
- Stats collector: 统计数据收集进程, 会生成描述数据库，表等统计信息的.stat文件。
- Writer：就是定期将共享内存的数据写到磁盘上的进程
- Master resource manager：管理 / 分配 / 回收资源，定期查询 / 接收 / 处理segment心跳信息，从而获取集群可用的节点。
- Segment resource manager：查询本节点状态，发送 segment心跳信息
- DFS metadata cache：读取并缓存block location，方便计算data locality信息，从而提高生成查询计划时决定哪个节点上读取哪些数据的速度。
- Checkpoint: 负责周期性做checkpoint或响应常规的checkpoint请求
- Sequenceserver：负责产生序列的进程
- Wal send server： 负责把write ahead log发给standby master
- Sync agent：负责和master上wal send server通信的进程，处理Master 和standby 节点状态

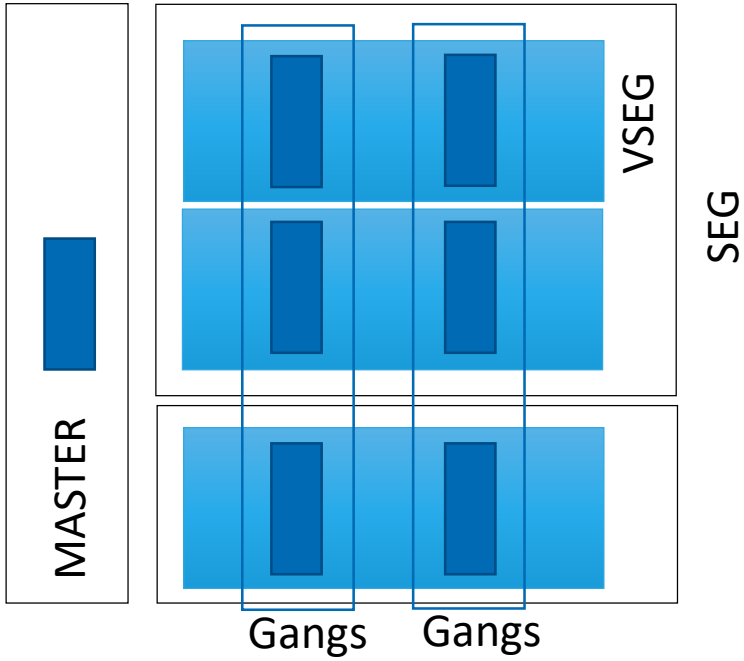
04 Apache HAWQ 查询执行过程

查询执行流程

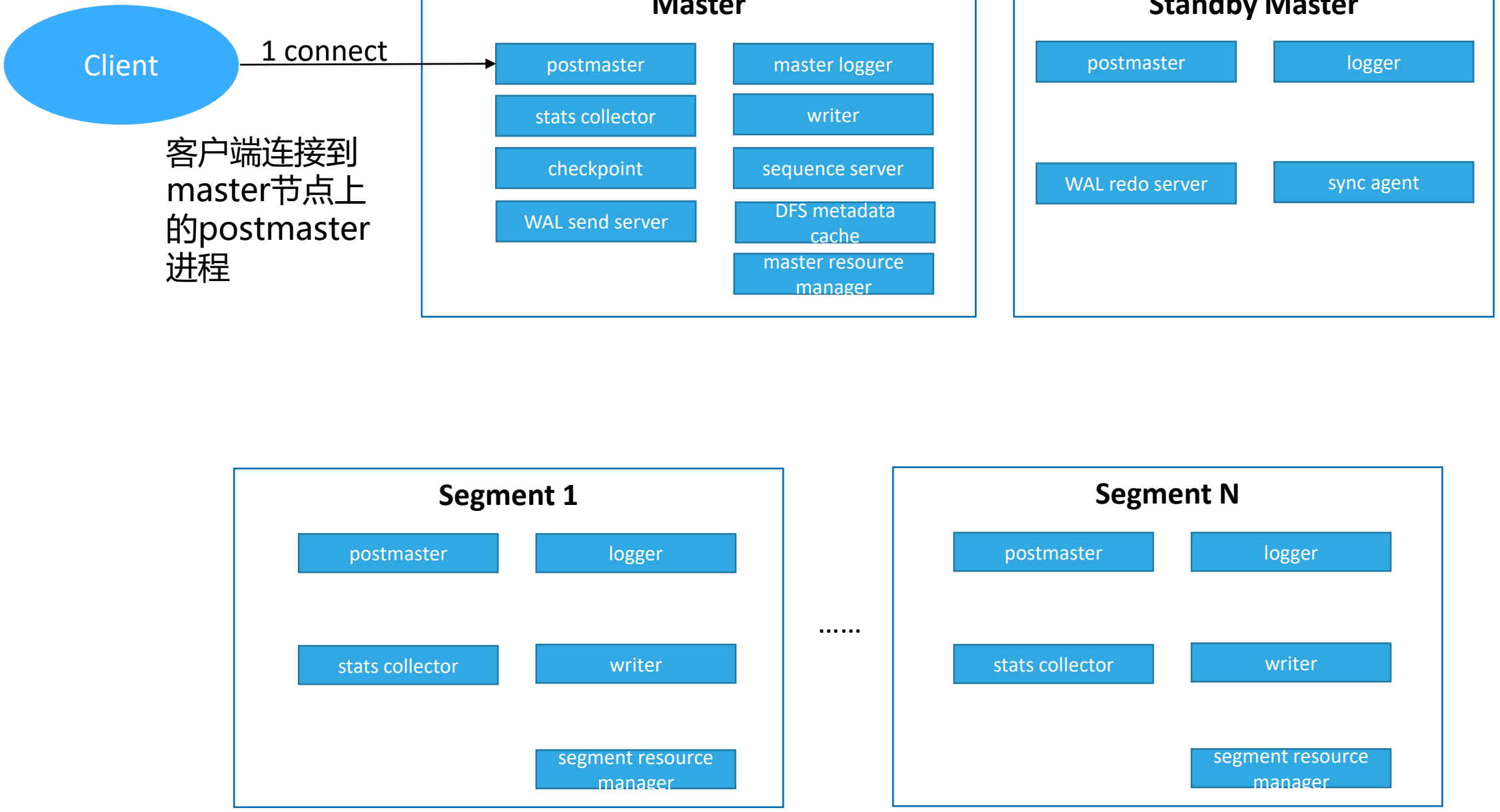



```
SELECT l_orderkey,
count(l_quantity)
FROM lineitem, orders
WHERE l_orderkey=o_orderkey
AND l_tax>0.01
GROUP BY l_orderkey;
```

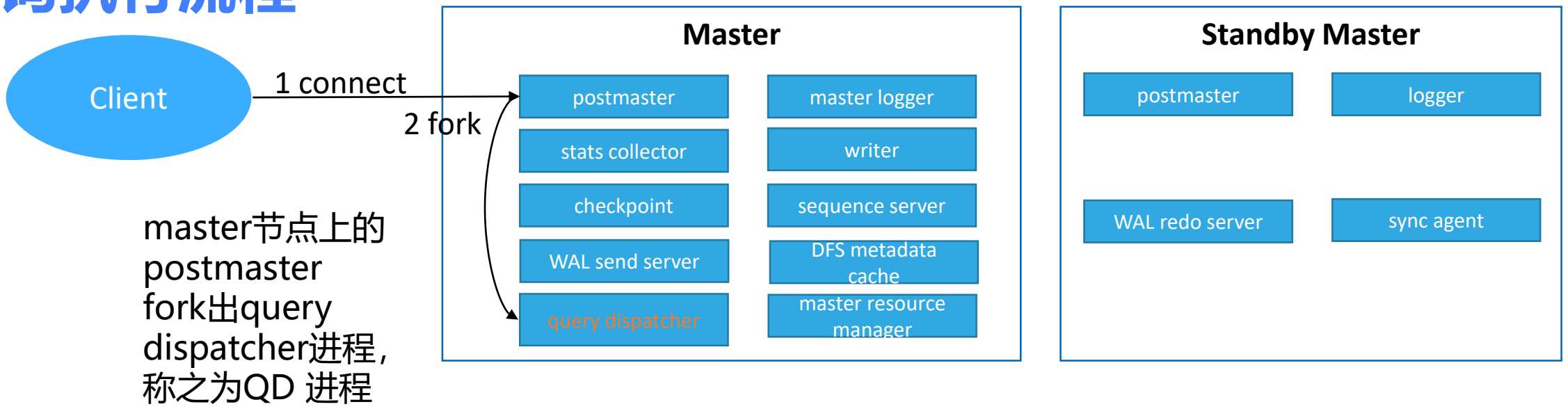




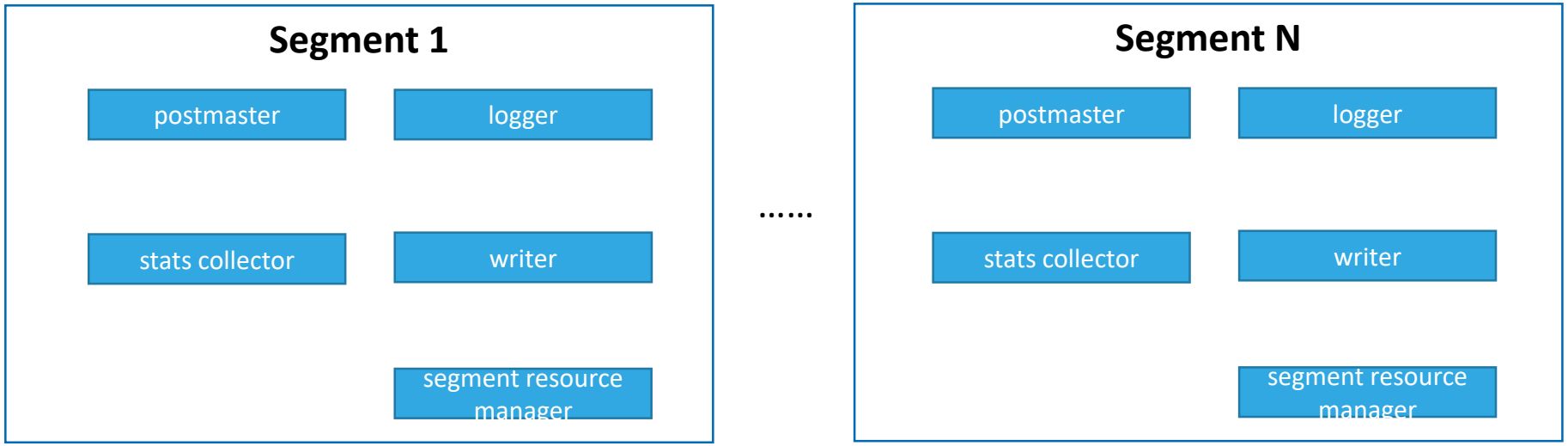
查询执行流程


Client
1 connect
客户端连接到master节点上的postmaster进程
Master
postmaster
master logger
stats collector
writer
checkpoint
sequence server
WAL send server
DFS metadata cache
master resource manager
Standby Master
postmaster
logger
WAL redo server
sync agent
Segment 1
postmaster
logger
stats collector
writer
segment resource manager
.....
Segment N
postmaster
logger
stats collector
writer
segment resource manager

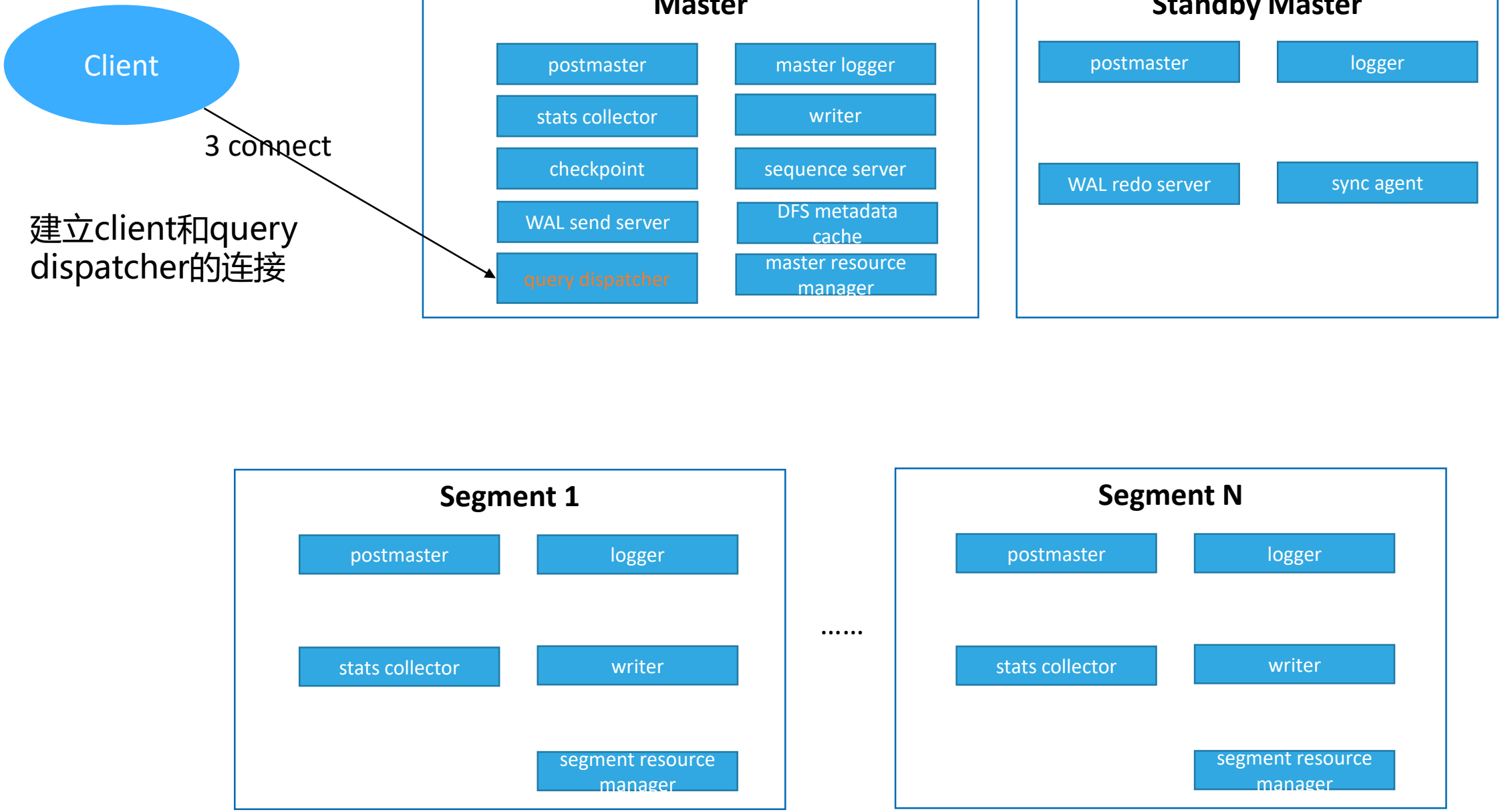
查询执行流程



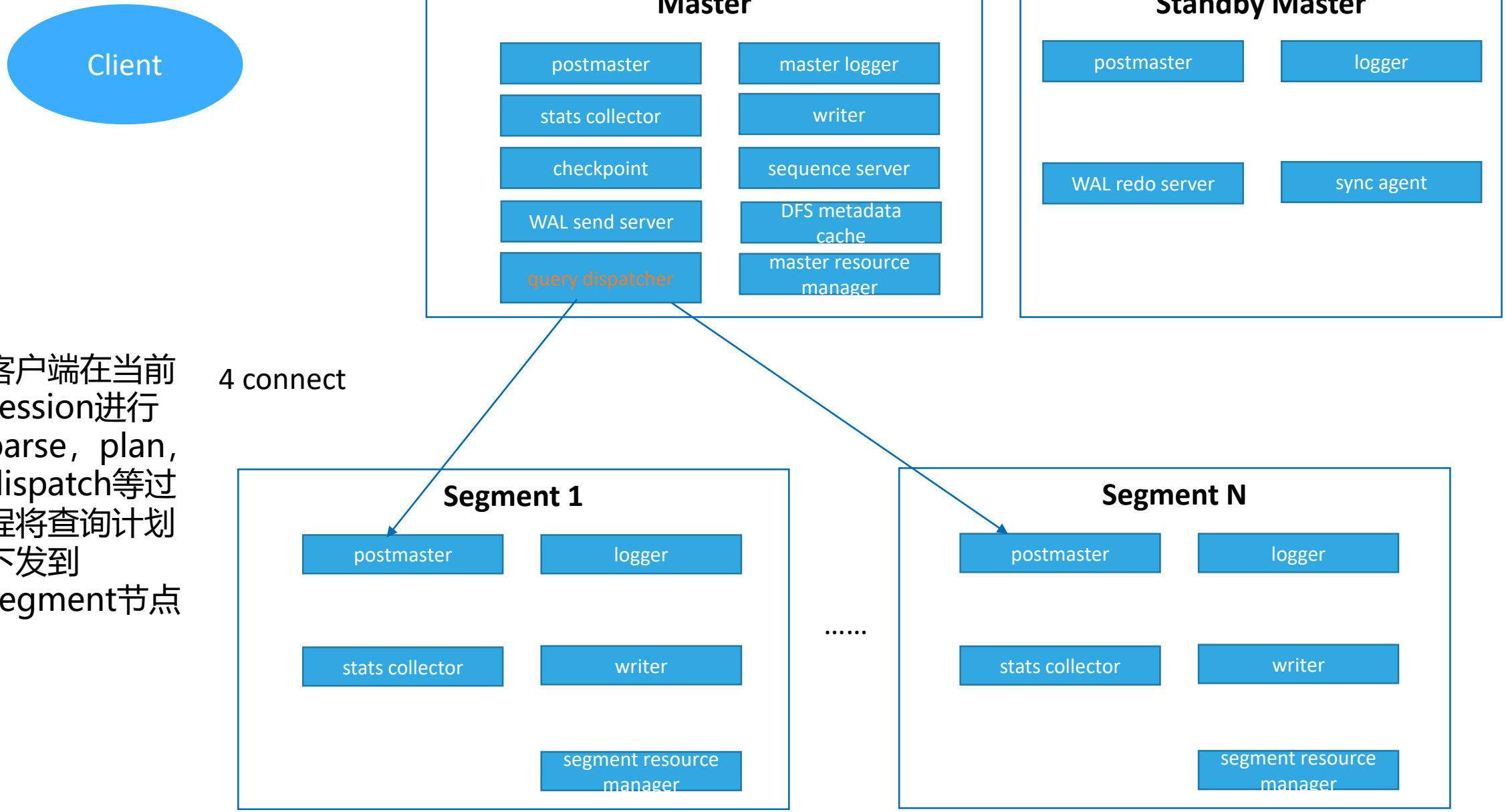
master节点上的postmaster fork出query dispatcher进程，称之为QD 进程



查询执行流程

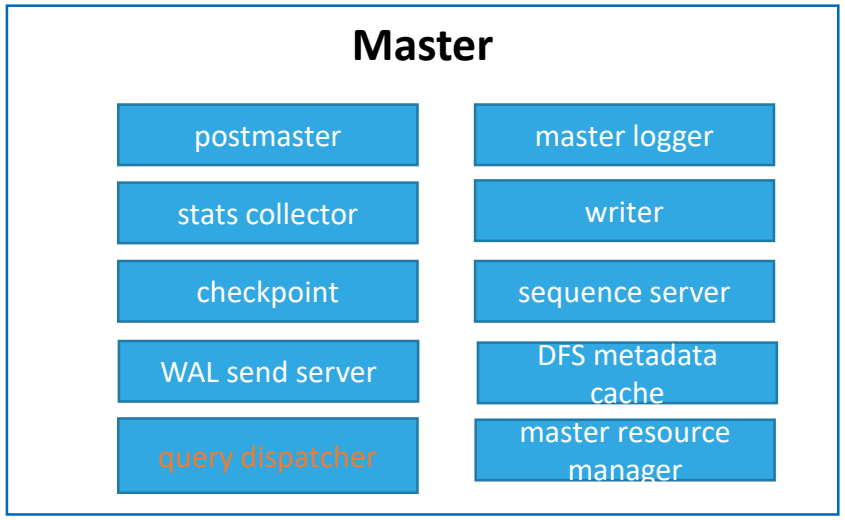

Client
3 connect
建立client和query dispatcher的连接
Master
postmaster
master logger
stats collector
writer
checkpoint
sequence server
WAL send server
DFS metadata cache
query dispatcher
master resource manager
Standby Master
postmaster
logger
WAL redo server
sync agent
Segment 1
postmaster
logger
stats collector
writer
segment resource manager
.....
Segment N
postmaster
logger
stats collector
writer
segment resource manager

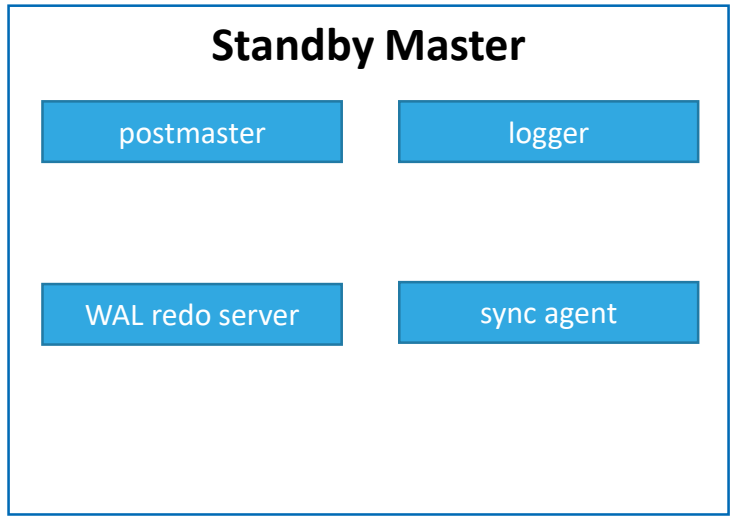
查询执行流程



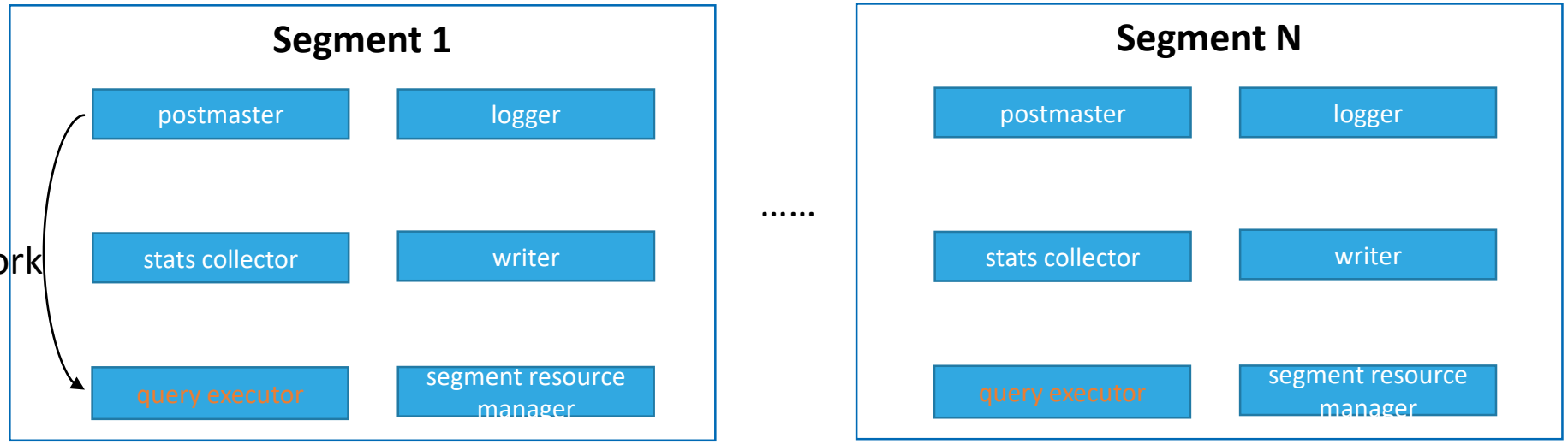
客户端在当前session进行parse，plan，dispatch等过程将查询计划下发到segment节点

查询执行流程

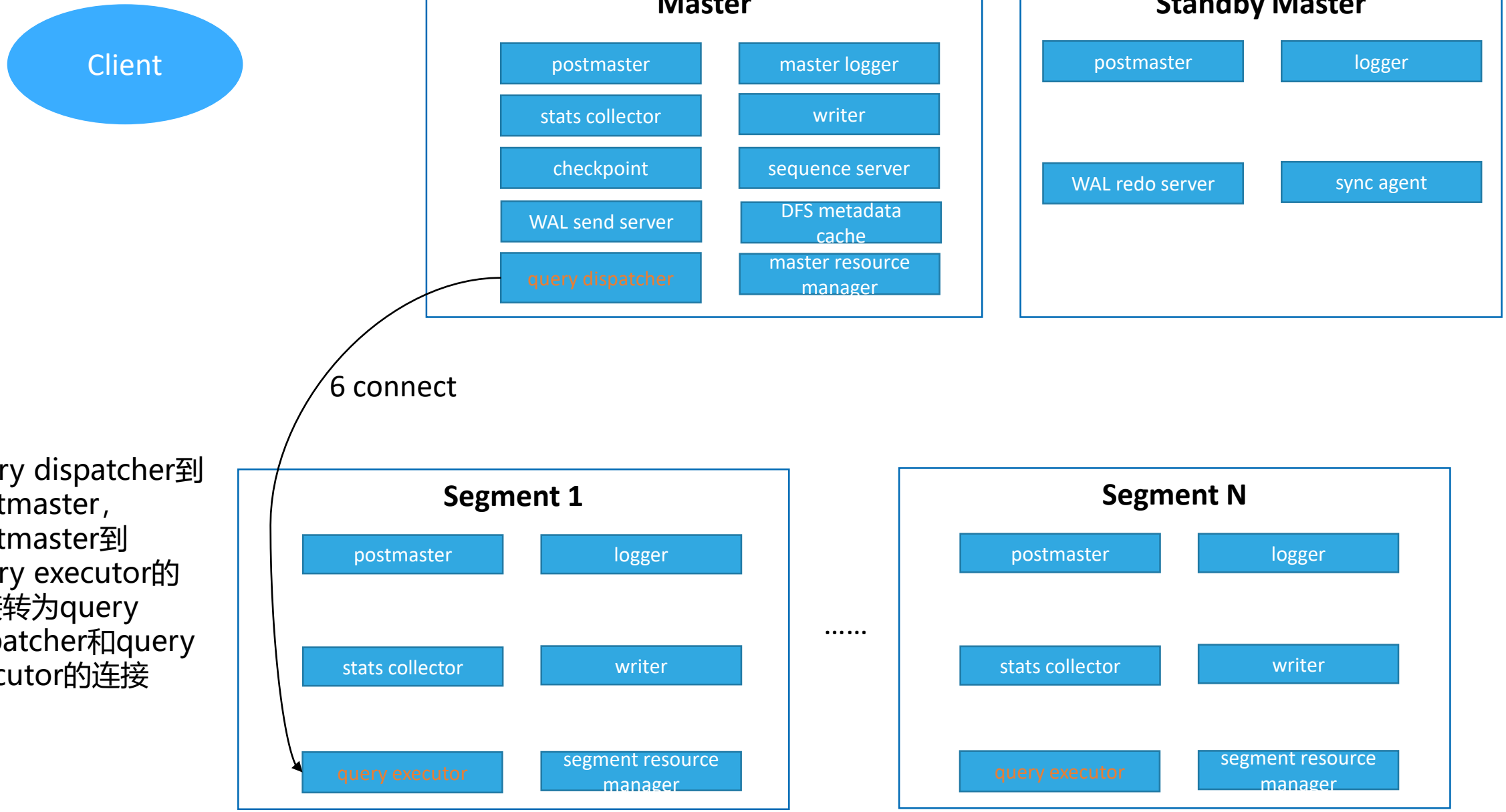




segment节点上的postmaster fork出query executor进程

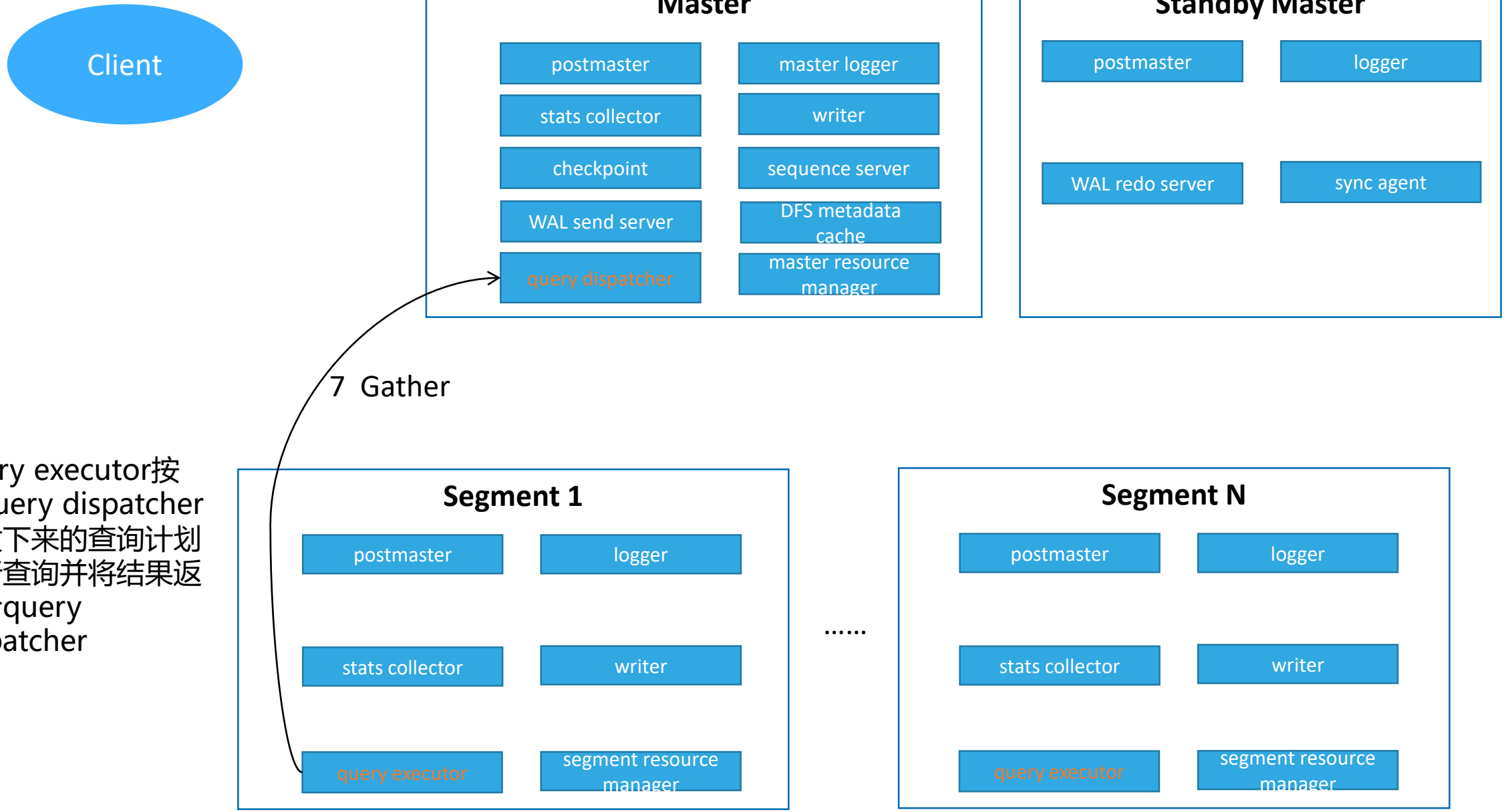


查询执行流程



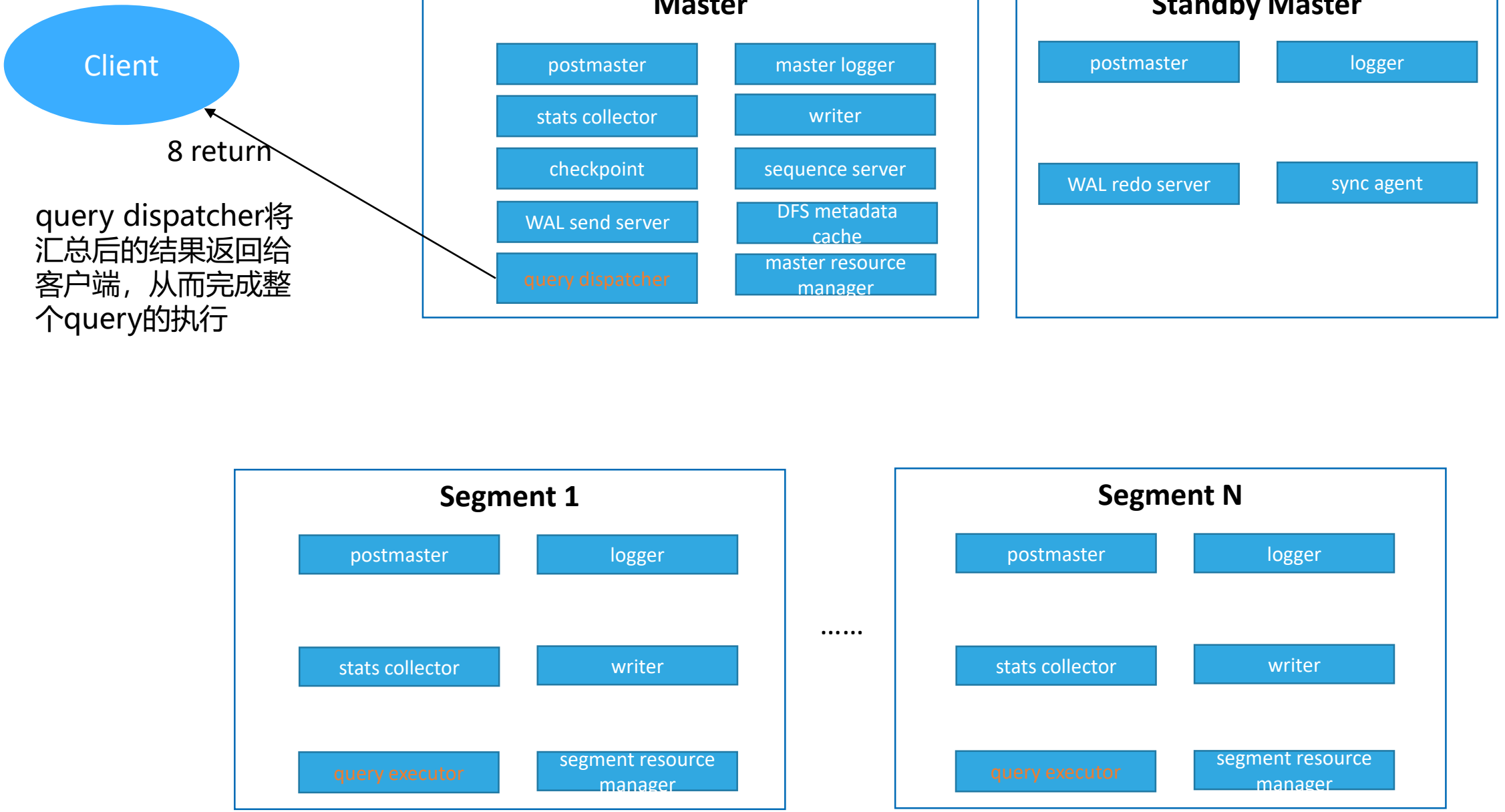
query dispatcher到postmaster，postmaster到query executor的连接转为query dispatcher和query executor的连接

查询执行流程



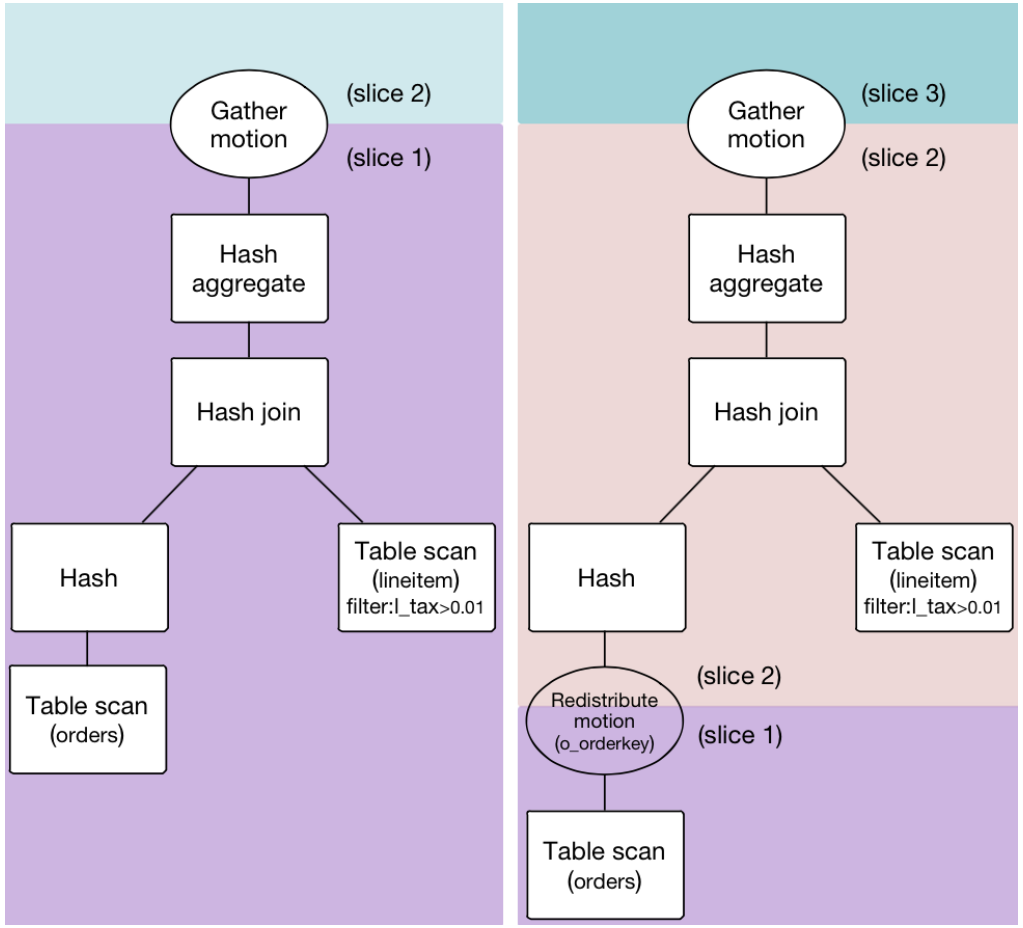
query executor按照query dispatcher下发下来的查询计划执行查询并将结果返回给query dispatcher

查询执行流程


Client
8 return
query dispatcher将汇总后的结果返回给客户端，从而完成整个query的执行
Master
postmaster
master logger
stats collector
writer
checkpoint
sequence server
WAL send server
DFS metadata cache
query dispatcher
master resource manager
Standby Master
postmaster
logger
WAL redo server
sync agent
Segment 1
postmaster
logger
stats collector
writer
query executor
segment resource manager
.....
Segment N
postmaster
logger
stats collector
writer
query executor
segment resource manager

05 Apache HAWQ 查询例子

查询例子

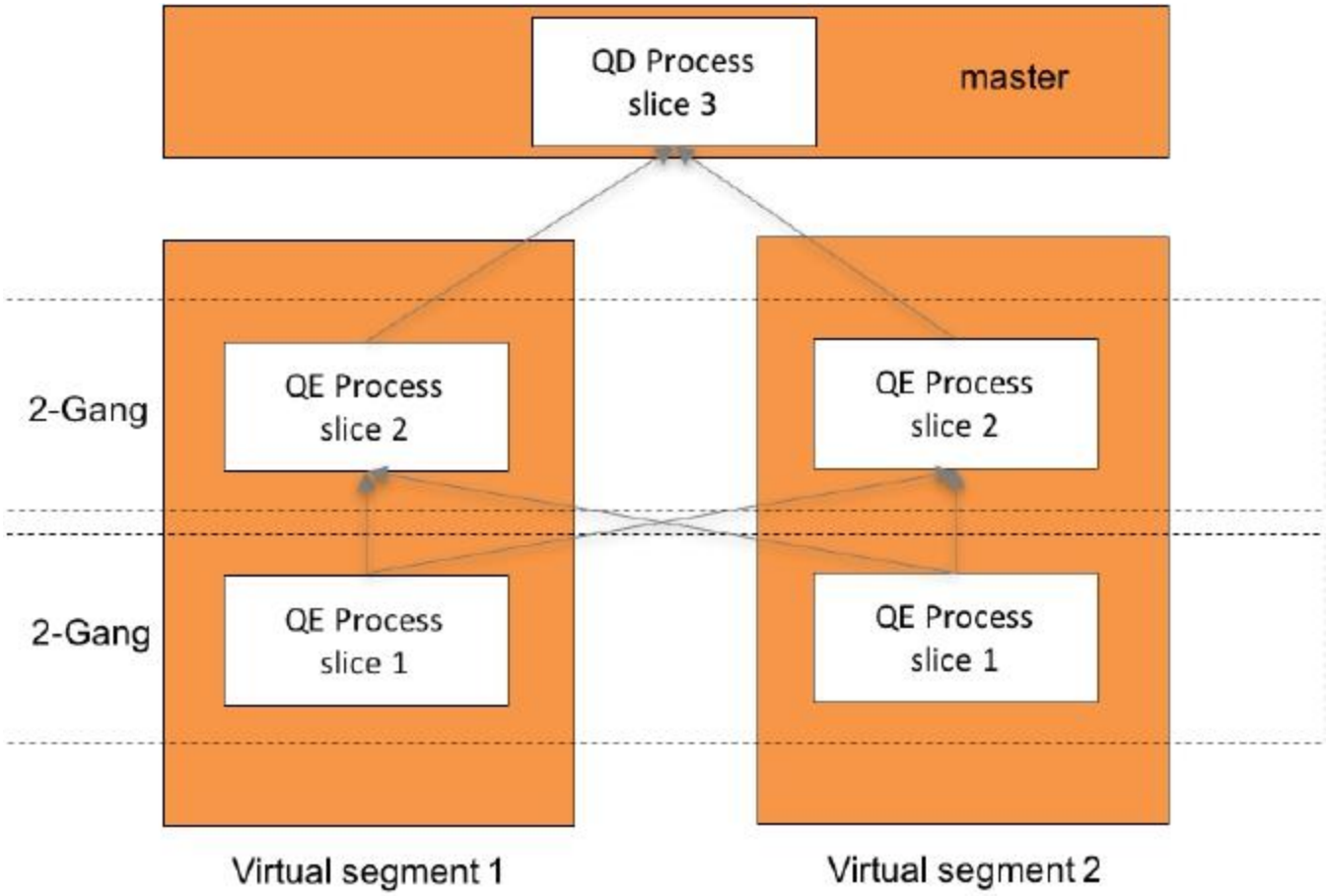


Motion:

- Redistribution
- Broadcast
- Gather
- 每个Motion 划分一个slice

```
SELECT l_orderkey, count(l_quantity)
FROM lineitem, orders
WHERE l_orderkey=o_orderkey AND l_tax>0.01
GROUP BY l_orderkey;
```

查询执行流程


QD Process
slice 3
master
2-Gang
QE Process
slice 2
QE Process
slice 2
2-Gang
QE Process
slice 1
QE Process
slice 1
Virtual segment 1
Virtual segment 2

06 外部接口

客户端工具

- PSQL
  - PSQL是 PostgreSQL 中的一个命令行交互式客户端工具。 安装的Apache HAWQ binary 里面会自动安装了PSQL。
- JDBC
  - 可以兼容PG官方JDBC
- ODBC
  - 目前没有提供单独的ODBC，可以兼容PG官方ODBC
- PGAdmin
  - Postgres 官方UI 客户端

07 调试

日志

- 初始化 / 启动 / 停止日志
  - /home/gpadmin/hawqAdminLogs/
  - hawq_init_日期.log / hawq_start_日期.log / hawq_stop_日期.log / hawq_config_日期.log
- Master 日志
  - 在配置的hawq_master_directory 下的pg_log，每天一个新的文件，每次重启一个新的文件。
- Segment 日志
  - 在配置的hawq_segment_directory 下的pg_log，当query 出错时，会出现在某个节点的segment 出现问题。这时候需要去对应的segment 上查看相应的log。

报使用问题所需要的信息

- 问题描述
  - 重现步骤与预期行为
  - 配置：
  - 节点数（Master / Segment 个数）
  - 影响的版本号 select version() 的输出
  - 如果有coredump， 打出错误栈 上传coredump 和对应的binary
  - 出错时间段前后的log。根据出错信息里来决定拿出错节点的对应的log（hawq master log/hawq segment log)

感谢观看

让人类只为兴趣而工作