技术重塑管理

数据驱动的企业

用友 yonyou

yonyou

UAP 大型企业与组织计算平台

Computing Platform for Large-sized Enterprises and Organizations

发版说明

UAP UDH V1.0

版权

目录

1. 概述

UDH(UAP Distribution for Hadoop)是用友企业级大数据处理平台。用于处理大量的非结构化或半结构化类型数据，也适用于超大规模的结构化数据处理分析。

Hadoop 是开源的分布式系统架构，能够让用户在不了解分布式底层运行细节的情况下，对数据进行分布式处理，能够充分利用分布式集群的高效计算和存储能力。UDH 在开源社区软件的基础上，包含 Hadoop 大部分的主流组件，并且对这些组件在安全性，管理，性能，高可用等方面进行了优化。同时整合数据集成工具，集群管理和监控工具，增强了其企业级应用特性。让企业可以更快，更准，更稳地从各类繁杂无序的海量数据中洞察商机。

1.1. UDH 平台定位

UDH 主要用于解决企业的以下需求问题。

- 快速整合，存储，集中管理不同类型的海量数据
- 提供批量和实时数据处理服务
- 与 AE 产品结合为构建企业级数据仓库提供大数据平台支撑
- 结合 BQ 产品和挖掘可视化产品，提供数据分析服务
- 提供平台中各服务组件的管理和系统运行监控

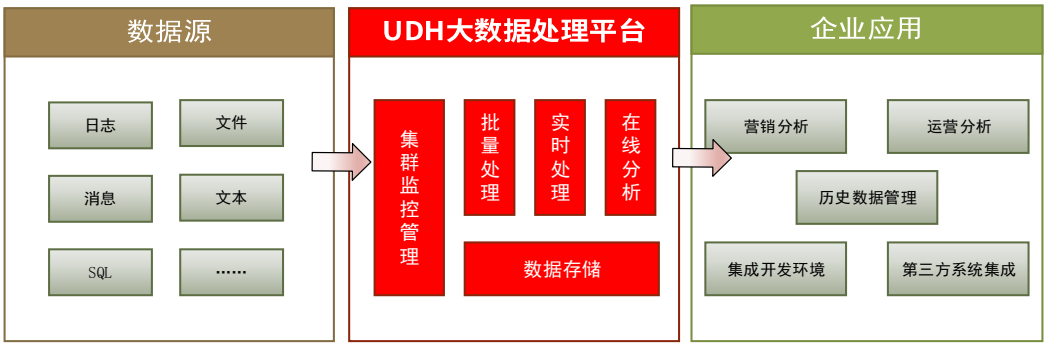

图 1 UDH 的平台定位

1.2. 功能与价值

可靠

UDH 提供全方位的可靠性方案，不仅实现组件的 HA 功能、消除了单点风险，而且提供集群级备份恢复能力。

安全

UDH 支持 Kerberos 安全认证，支持 ACL 文件权限控制列表，并在社区 Hadoop 基础上增强提供操作安全审计、数据加密解密、恶意攻击保护等多种安全机制，消除客户对大数据技术安全问题的担忧。

分析

UDH 与 BQ 结合增强了 BA 分析能力，集成包含 SQL on Hadoop 等各种数据分析组件的同时加强与传统商务智能分析平台的集成，让企业可以更快、更准、更稳的从各类繁杂无序的海量数据中发现价值，洞察企业新商机。

易用

UDH 针对客户设计，全自动化在线运行维护，可自定义 Dashboard、提供自动化的二次开发助手，大幅降低了大数据在传统企业内部的部署难度，帮助传统企业轻松驾驭大数据业务。

UDH 架构

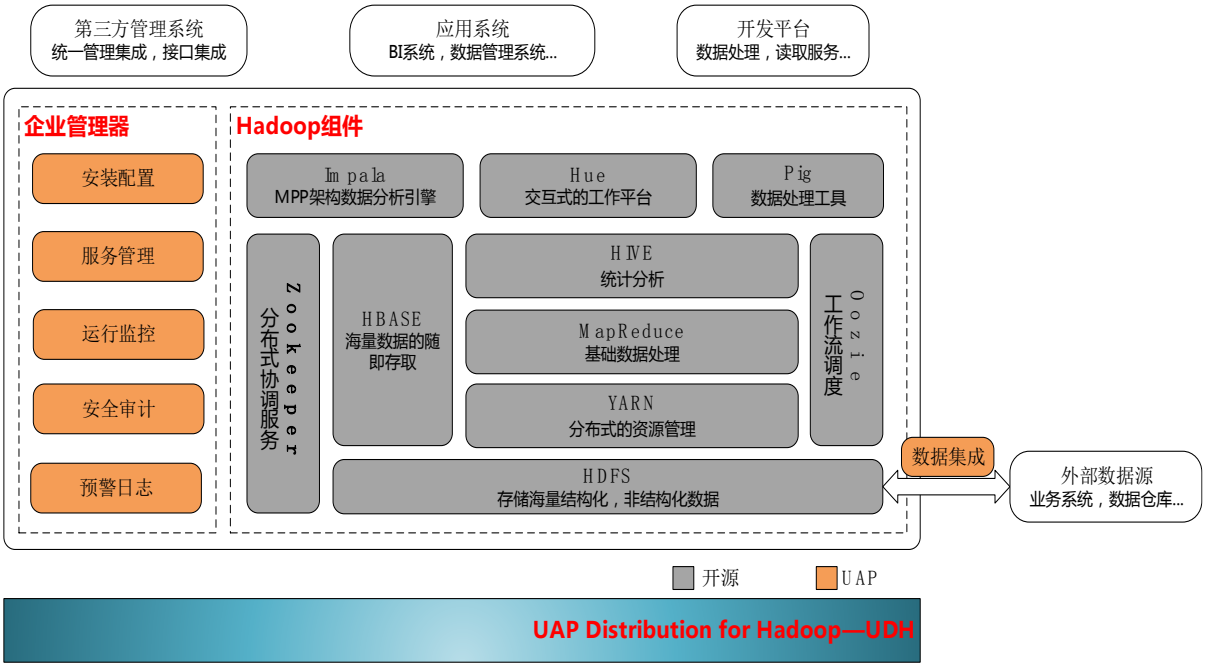

图 2 UDH 产品架构

2. 产品特性

2.1. 硬件规格

硬件	最低配置
CPU	2*8cores Intel Xeon E5-2690 64bit 系列或同规格处理器
内存	> =4GB
磁盘	> =20GB
网络	万兆光纤

服务组件的内存要求

硬件	最低配置
Zookeeper	1GB
HDFS	8GB(包含 NameNode，DataNode)
MapReduce	8GB
Hive	4GB
HBase	6GB(包括 Master，Region Server)
Impala	32GB

2.2. 产品特征

支持的 Hadoop 组件：

1. 支持 HDFS 组件
2. 支持 MapReduce 组件
3. 支持 yarn 组件
4. 支持 HBase 组件
5. 支持 Hive 组件
6. 支持 Impala 组件
7. 支持 Pig 组件
8. 支持 Zookeeper 组件
9. 支持 Oozie 组件
10. 支持 HUE 组件
11. 支持 Hcatalog 组件
12. 支持 Nagios，Ganglia 组件

支持的数据处理服务

1、 支持结构化，非结构化海量数据存储，HDFS，HIVE

2、 支持数据处理服务：支持 MapReduce 的批量处理和 HBase API 处理数据

3、 支持数据查询分析：通过 API,SQL 进行查询分析，通过 HUE 提供执行平台

4、 支持集群节点的自动化安装配置：

a) 支持以本地源或 UAP 源提供安装包

b) 支持以向导方式进行自动化安装部署 UDH 集群

5、 支持 Hadoop 组件的管理监控

a) 提供各组件状态监控

b) 提供组件的基本管理功能，如启动，重启，停止等

c) 支持组件的参数配置

6、 支持对执行作业和任务的状态管理

7、 提供统一的预警服务管理，支持 Heatmap 形式的监控界面

8、 支持服务状态的定时检查，与预警条件结合。

9、 支持 HQL 查询，数据浏览，工作流定义工具：支持 Hive，Impala 的 SQL 查询及其 UI，HDFS 文件浏览和可视化的 Oozie 工作流定义工具

10、 支持对集群节点的监控，如 CPU，磁盘，内存等，提供 Dashboard 监控界面

11、 支持 Kerberos 认证

12、 支持中英文环境

3. 产品范围

产品领域	产品模块
数据处理平台	UDH
	企业管理器

4. 产品主要功能

4.1. 主要数据处理组件

4.1.1. HDFS

Hadoop 分布式文件系统（Hadoop Distributed File System）提供高吞吐量的数据访问，适合大规模数据存储的应用场景。

HDFS 包含主，从 NameNode 和多个 DataNode

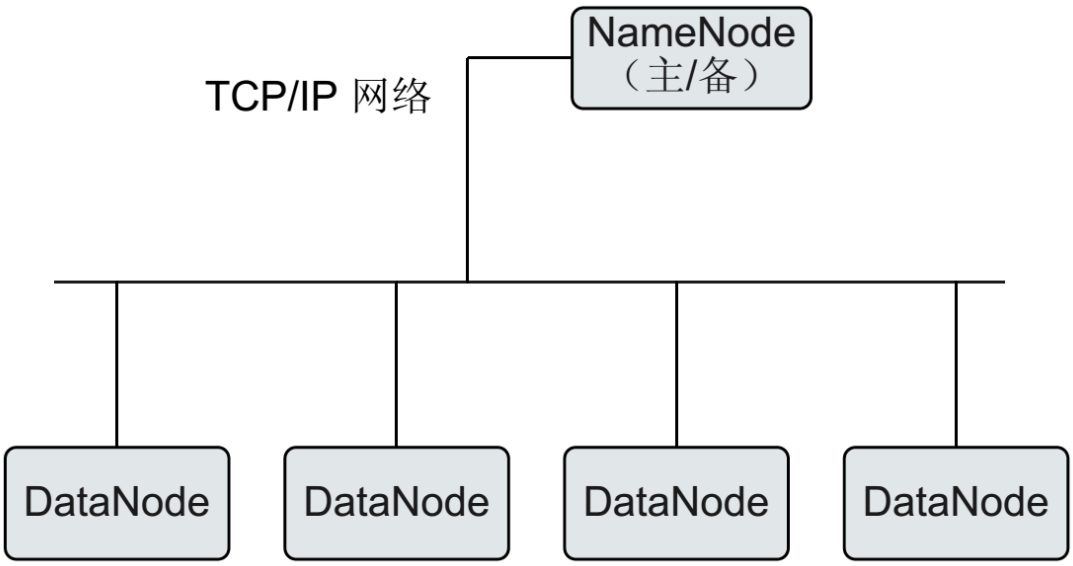

图 3 HDFS 结构

HDFS 是 Master-Slave 结构，在 Master 上运行 NameNode，在每个 Slave 上运行 DataNode。NameNode 和 DataNode 之间通过 TCP/IP 进行通信，NameNode 和 DataNode 需要部署在 Linux 服务器上。N

NameNode：用于管理文件系统的命名空间，目录结构，元数据信息以及提供 HA 机制。

DataNode：用于存储每个文件的“数据块”数据，并且会周期性地想 NameNode 报告存放状态。

在 HDFS 中，一个文件按照定义的块（block）大小被分成多个“数据块”，这些数据

块存储在 DataNode 集合中。客户端链接到 NameNode 中，执行文件系统的“命名空间”操作，例如打开，关闭，重命名等，同时决定“数据块”到具体 DataNode 节点的映射。NameNode 决定数据在 DataNode 上的创建，删除，复制等。

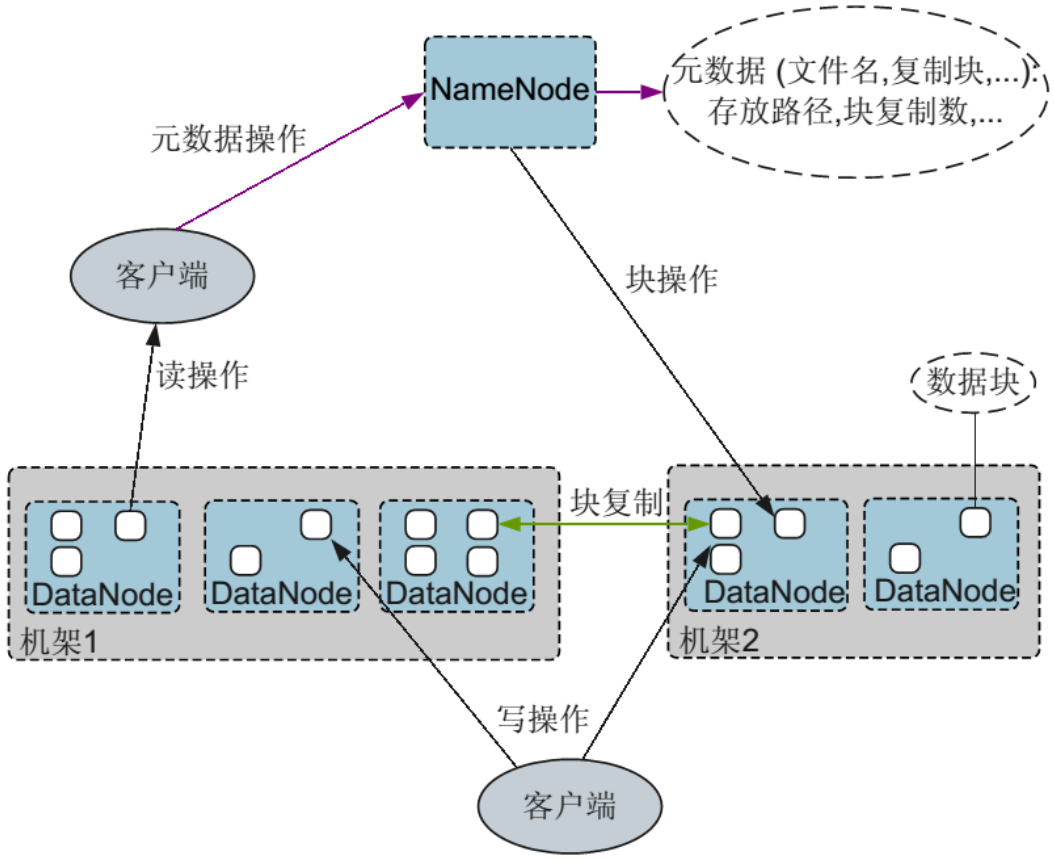

图 4　HDFS 原理

4.1.2.　MapReduce

是一种分布式并行计算模型，名称源于其两个基本操作:Map 和 Reduce。Map 将一个任务分解成几个小任务；Reduce 将分解后的小任务处理结构汇总得到最终的分析结果。

MapReduce 模型主要由 TaskTracker 和 JobTracker 组成。JobTracker 主要执行任务调度，任务监控工作。其中存在一个 Master JobTracker，用于调度和管理其他的 TaskTracker，JobTracker 可以运行于集群上的任意节点上。TaskTracker 负责执行任务，必须运行于 DataNode 上，即 DataNode 既是数据存储节点和是计算节点。

JobTracker 将 Map 任务和 Reduce 任务分发给空闲的 TaskTracker，让这些任务并行运行，并负责监控任务的运行情况，如果某个 TaskTracker 出故障了，JobTracker 会将其负责任务转交给另一个空闲的 TaskTracker 重新运行。

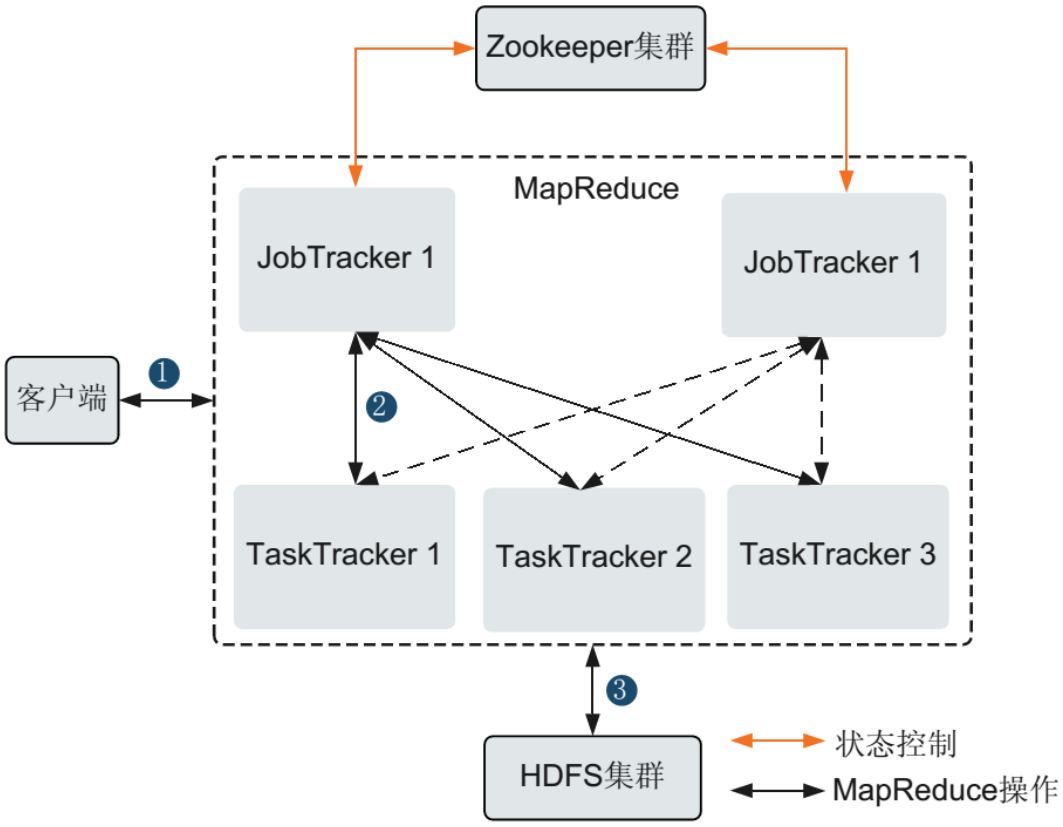

图 4　MapReduce 结构

客户端：向 Active JobTracker（Job Tracker1）发起 MapReduce 操作

Zookeeper 集群：控制系统中 JobTracker 的“Active”和“Standby”状态

HDFS 集群：存储需要运行 MapReduce 的数据和保存 MapReduce 中间结果。

JobTracker：MapReduce 的任务调度进程，分为：

JobTracker1：负责调度作业的所有任务，将作业分布在不同的 TaskTracker 那个，并监视他们的执行以重启失败任务。

JobTracker2:JobTracker1 的备用进程，通过 Zookeeper 对 JobTracker1 进行监控，在 JobTracker1 异常时，JobTracker2 会接替 JobTracker1 的功能，重新执行未完成

任务。

TaskTracker：执行由 JobTracker 指派的任务

4.1.3. HBase

构建在 HDFS 之上的分布式，列式存储系统，具有高可靠，高性能，列式和可伸缩的特性。HBase 适合存储大表数据（可达数十亿和百万列），并对大数据的读，写访问达到实时级别。

HBase 集群由主备 HMaster 进程和多个 RegionServer 进程组成。

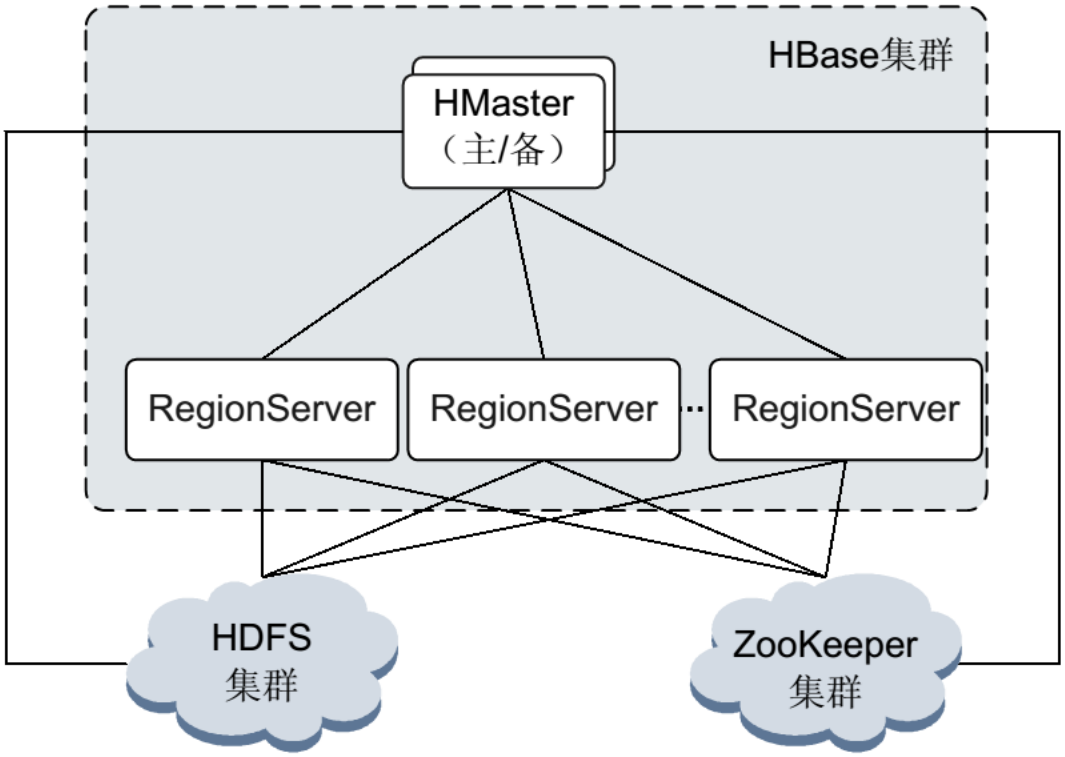

图 5 HBase 结构

主 HMaster：负责管理 RegionServer 节点，维护集群网络拓扑以及集群负载均衡

备用 HMaster：当主 HMaster 故障时，替代对外提供服务，当主恢复后，原主 HMaster 为备用。

RegionServer：负责提供表的读写服务，是数据处理和计算单元，一般与 HDFS 的

DataNode 一起部署，实现数据的存储功能。

Zookeeper 集群:为 HBase 集群中各进程提供分布式协作服务。各 RegionServer 将自己的信息注册到 Zookeeper 中，HMaster 获取各个 RegionServer 的状态。

HDFS 集群：为 HBase 提供文件存储服务。

HBase 的数据模型

HBase 以表的形式存储数据，表中的数据划分为多个 Region，并由 HMaster 分配给对应的 RegionServer 进行管理。每个 Region 包含了表中一段 RowKey 区间范围内的数据。HBase 的数据表会根据 Region 的大小，达到上限后，自动分为一个新的 Region。

RowKey	TimeStamp	Column Family 1		Column Family 2		Column Family N		
		URI	Content	Catalog	Article	Column 1	Column 2	
1	t1	www.yyuap.co	<html>...	…	…	…	…	Region
	t2	www.yyuap.co	<html>…	…	…	…	…	
2	t1	…	…	…	…	…	…	
…	…	…	…	…	…	…	…	
M	…	…	…	…	…	…	…	
M+1	t1	…	…	…	…	…	…	Region
M+2	t1	…	…	…	…	…	…	
	t2	…	…	…	…	…	…	
	t3	…	…	…	…	…	…	
…	…	…	…	…	…	…	…	
N	t1	…	…	…	…	…	…	Region
…	…	…	…	…	…	…	…	

图 6　HBase 数据模型

RowKey：表的主键，存储时表中记录按照 RowKey 的字符顺序进行排序

TimeStamp：插入数据时的时间戳，HBase 支持相同 RowKey 的多版本数据存储

Column Family：列族。一张表可以有多个由 Column 组成的 Column Family 组成。

Column：列。与数据库中表中的列类似。

Region 数据存储

RegionServer 主要负责管理由 HMaster 分配的 Region。

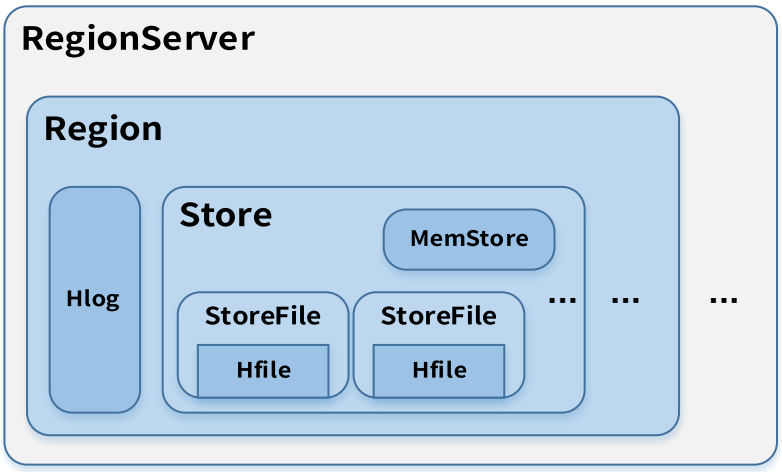

图 7　RegionServer 数据存储结构

Store：一个 Region 由一个或多个 Store 组成，每个 Store 对应一个 Column Family。

MemStore：一个 Store 中包含一个 MemStore。缓存客户端向 Region 插入的数据，当 MemStore 大小达到配置容量的上限时，RegionServer 会将 MemStore 中的数据 Flush 到 HDFS 中。

StoreFile：Flush 到 HDFS 后成为 StoreFile，随着数据的增多会产生多个 StoreFile，当数量达到上限时，RegionServer 会将多个 StoreFile 合并成一个大的 StoreFile。

HFile：定了 StoreFile 在文件系统中存储的格式。

HLog：保证了在 RegionServer 故障情况下用户写入的数据不丢失，RegionServer 的多个 Region 共享一个相同的 Hlog。

元数据表

元数据表是 HBase 中一种特殊的表，用来帮助 Client 定位到具体的 Region。包括“.META.”和“-ROOT-”表。

.META.表：记录用户表的 Region 信息。例如 Region 位置，起始 RowKey，结束 RowKey 等。

-ROOT-表：记录.META.表的 Region 信息。

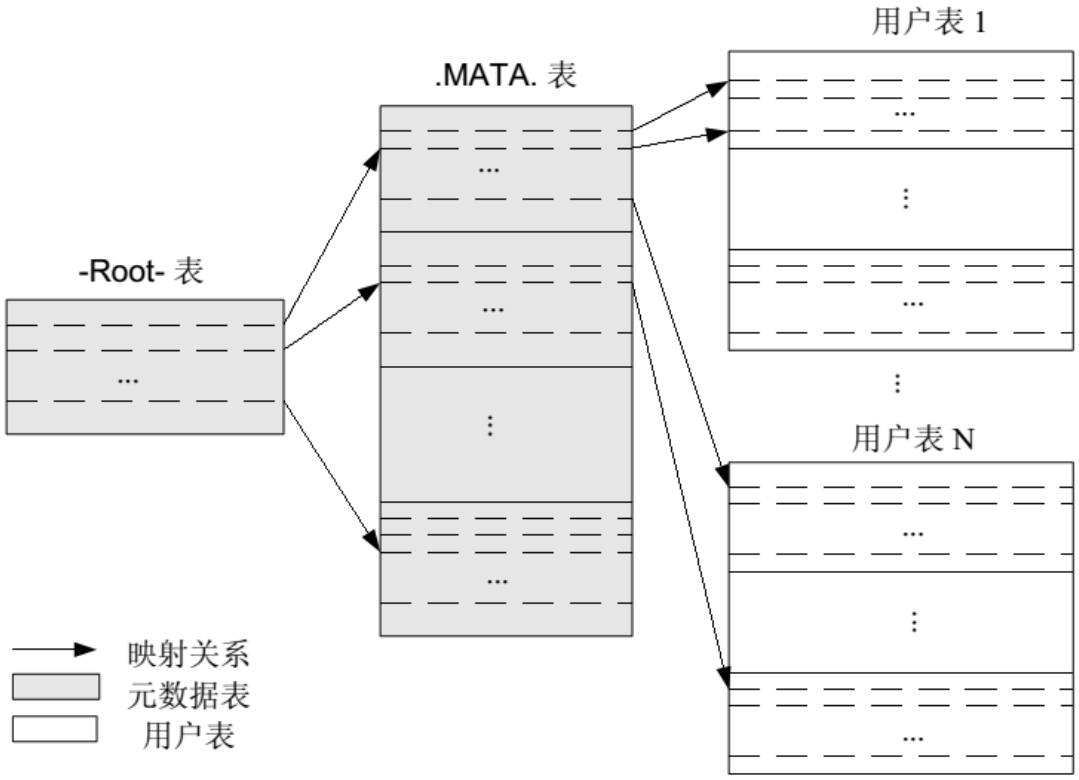

图 8　元数据表和用户表映射关系

HBase 数据操作流程：

1. 当 UDH 对 HBase 进行增，删，改，查操作时，Client 首先连接 Zookeeper 获取“-ROOT-”表所在的 RegionServer 的信息。
2. Client 连接到“-ROOT-”表 Region 所在 RegionServer 获取“.META.”表 Region 信息。
3. Client 连接到对应“.META.”表的 Region 所在的 RegionServer，获取到相应的用户表的 Region 所在的 RegionServer 信息。
4. Client 连接到对应用户表 Region 所在的 RegionServer，并将操作命令发送给 RegionServer，RegionServer 接受并执行，完整数据操作。

为了提升数据操作效率，Client 会缓存“-ROOT-”和“.META.”和用户表信息，当应用操作下次请求时，会先从缓存中获取这些信息；当内存中的缓存与系统实际信息不符时，Client

会重复上述步骤。

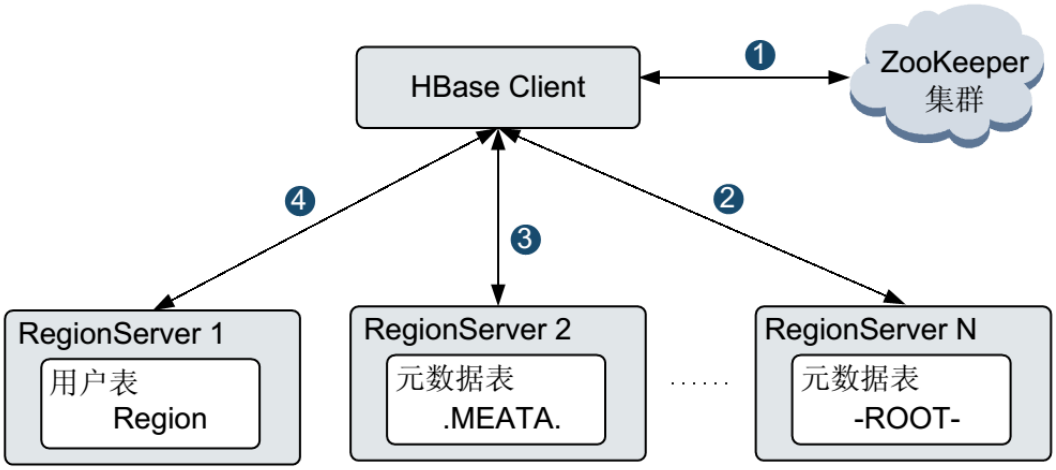

图 9　HBase 数据操作流程

4.1.4.　Hive

是建立在 Hadoop 基础上的开源数据仓库，提供类似 SQL 的 Hive SQL 操作结构化存储数据服务和基本的数据分析服务。

Hive 主要特点：

1. 海量结构化数据分析汇总
2. 将复杂的 MapReduce 任务简化为 SQL
3. 灵活的数据存储格式：支持 JSON，CSV，TextFile，RCFile，SequenceFile 等格式
4. 支持 HA 和安全特性，保证高可用，数据安全和访问控制。

详细特征见：https://cwiki.apache.org/confluence/display/Hive/DesignDocs

用户认证采用 Kerberos 认证。Kerberos 是一种适用于在公共网络上进行分布计算的工业标准的安全认证系统。用户通过 Hive 客户端和 Hive Server 建立连接时，需要双向认证。当用户是 Kerberos KDC 的合法用户时，才能通过认证访问 Hive。认证方法为使用用户名（Principle）和 Keytab 文件登录 KDC，登录成功则表示认证通过。

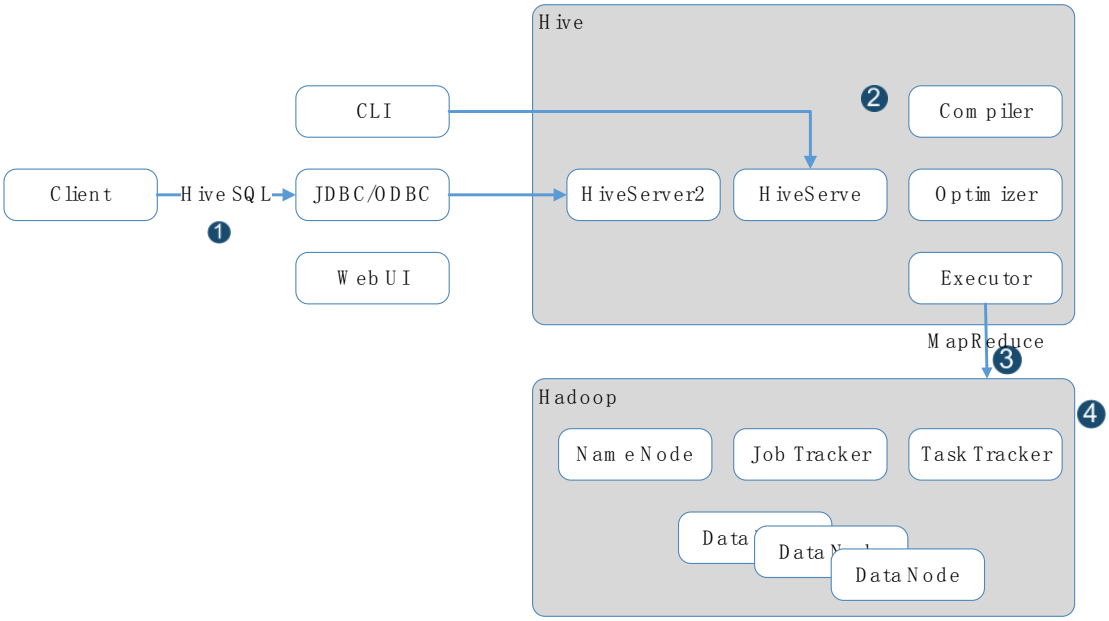

图 10 HIVE 架构

由于 HiveServer 只能支持单一客户端的请求不支持多客户端并发，在 Hive-0.11 版本中引入 HiveServer2 支持多客户端并发访问和认证，为客户端 API 对 JDBC,ODBC 提供了更好的支持。

HIVE 执行流程：

1. 通过命令行（CLI）,API(JDBC/ODBC)和 Web UI 方式向 HIVE 提交 SQL 查询请求。
2. Hive SQL 通过编译器，进行语法解析生成语法结构树，通过分析器进行优化，如列剪枝（RBO 优化），最后通过语义分析转换成 MapReduce 任务。
3. 执行器根据生成的任务信息启动 MapReduce 任务。
4. MapReduce 执行查询任务并返回结果。

4.1.5. Impala

是支持 SQL 交互式查询的实时大数据分析引擎，可以让 HDFS 文件系统和 HBase 数据库中的数据支持实时查询。相比 Hive，Impala 最大的特点就是速度快，性能较 Hive 提升 3-90 倍。

Impala 的主要特点

1. Impala 可以作为可靠的数据仓库，可用于数据分析，商业智能，监控等应用。
2. Impala 可以将交互式查询缩短为几秒，可以作为即席查询的工具。
3. 可以在部署 MapReduce 相同集群上运行 Impala，也可以单独部署 Impala 集群作为专有的分析集群。
4. 与 Hive 一样，Impala 也支持 SQL，可以方便地将查询由 Hive 转换为 Impala，为 Hive 中部分查询提升性能。
5. 同时 Impala 支持以 ODBC，JDBC 的方式与集群连接，使得 Impala 可以作为可视化工具和应用程序的数据引擎。

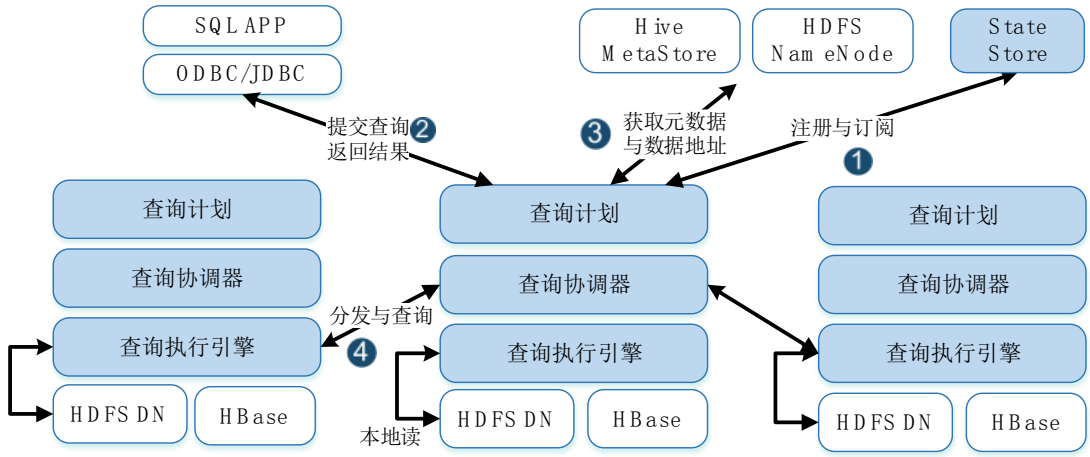

图 11 Impala 架构

Impala 可以分为两大部分 StateStore 和 Impalad。StateStore 为集群状态服务进程，处理 Impalad 的注册订阅和与各 Impalad 保持心跳连接。在整个集群中只存在一个实例，用于查找数据，便于分布式资源的查询分配。Impalad 是服务端，接受数据查询请求（此 Impalad 协调器，Coordinator）和解析查询生成执行计划，并行执行数据读写请求。Impalad 会定期向 StateStore 的更新状态和地址。

查询处理流程：

1. Impalad 节点向 StateStore 注册信息，并定期更新自身的状态。同时各 Impalad 都会缓存一份 State Store 中的信息，如果 StateStore 离线后，仍然可以工作，但也会由

于缓存无法更新，导致把执行计划分配给了失效的节点，导致查询失败。当 StateStore 重新加入集群后，自动恢复正常。

2. 客户端向某个 Impalad（协调器）发送查询请求，同时此协调器会汇总并行计算的结果返回给客户端。

3. 协调器对查询进行语法分析，生成执行计划。执行计划分为多个片段 PlanFragment，每个片段可以由多个 Impalad 并行执行。协调器根据执行计划访问 Hive MetaStore 和 NameNode 获取数据文件所在的节点信息。

4. 通过调度算法，把执行计划分配给后端的 Impalad 执行。查询时采用 LLVM 进行代码生成，编译和执行，提升执行效率。每个节点直接读取本地数据来完成查询任务，减少了网络 IO。各节点的执行结果在协调器节点进行汇总并返回给客户端完成查询执行过程。

与 Hive 的区别

1. Impala 使用大规模并行处理（MPP）引擎执行 SQL 查询，而 Hive 使用 MapReduce 执行，Hive 的查询需要执行 MapReduce 任务，而 Impala 没有这个开销，所以更快。

2. Impala 执行会使用大量的内存资源，故集群内存的大小会限制查询的数据量和性能，而 Hive 没有这方面限制，使用相同的硬件，Hive 可以处理更大的数据集。Impala 用于快速、交互式查询，而 Hive 更适用于大数据集的 ETL 工作任务。通常，Impala 适用于对速度要求高，需要对非常大的数据集做分析并且内存资源能够满足的场景下；相反，对处理的数据量大，但不要求速度的场景，更适合 Hive。

4.2. 集群管理器

UDH 集群管理器（UDH Manager）是 UDH 的集群管理工具，其集节点管理、服务部署、集群监控、数据分析及管理等功能于一身，支持 UDH 所有 Hadoop 组件的管理，包括 HDFS, Yarn, Mapreduce, Hive, Hbase, Impala, Zookeeper, WebHcat, Oozie, Hue, Nagios 和 Ganglia 等，利用它可以轻松搭建和管理大数据存储、分析平台。

在此只对主要的功能做概要性介绍，具体的操作请参考《UDH 集群管理器操作手册》。

4.2.1. 控制台

图 12 管理器控制台

主要分为五部分：仪表板，热点图，服务，主机和管理。其中根据管理用户角色不同会有所区别，非管理角色的用户，无法看到管理部分的功能节点以及其他部分与管理相关的功能。

仪表板

“仪表板”页面包含两部分内容，其中，左边的列表罗列出当前已安装的所有服务；右边代表当前集群的健康状态和性能等的仪表板。

服务组件列表通过颜色可以标识出当前组件的运行状态，同时根据组件的不同，提供了启动，停止和添加服务等快捷辅助服务。

图 13 服务组件状态

在指标仪表盘显示区域，对整个集群的健康状态和各个服务的性能表示为不同可视化图表，并给出主要的性能指标，便于管理。

热点图

通过选择预置指标集中具体的指标，根据定义的阈值条件，可视化显示出各节点的当前状态。

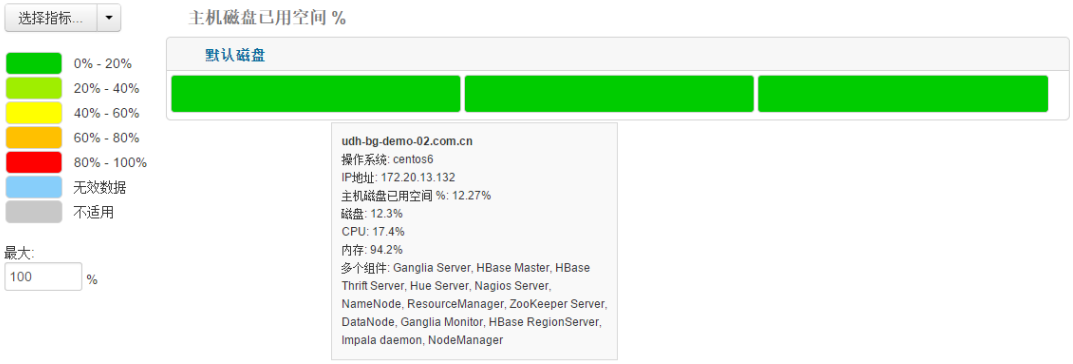

图 14 磁盘使用率

服务

记录了每个服务的概要、配置,警告和健康检查等信息。方便用户查看、修改服务设置。X 详细的组件服务信息，参见《UDH 集群管理器操作手册》相关章节。

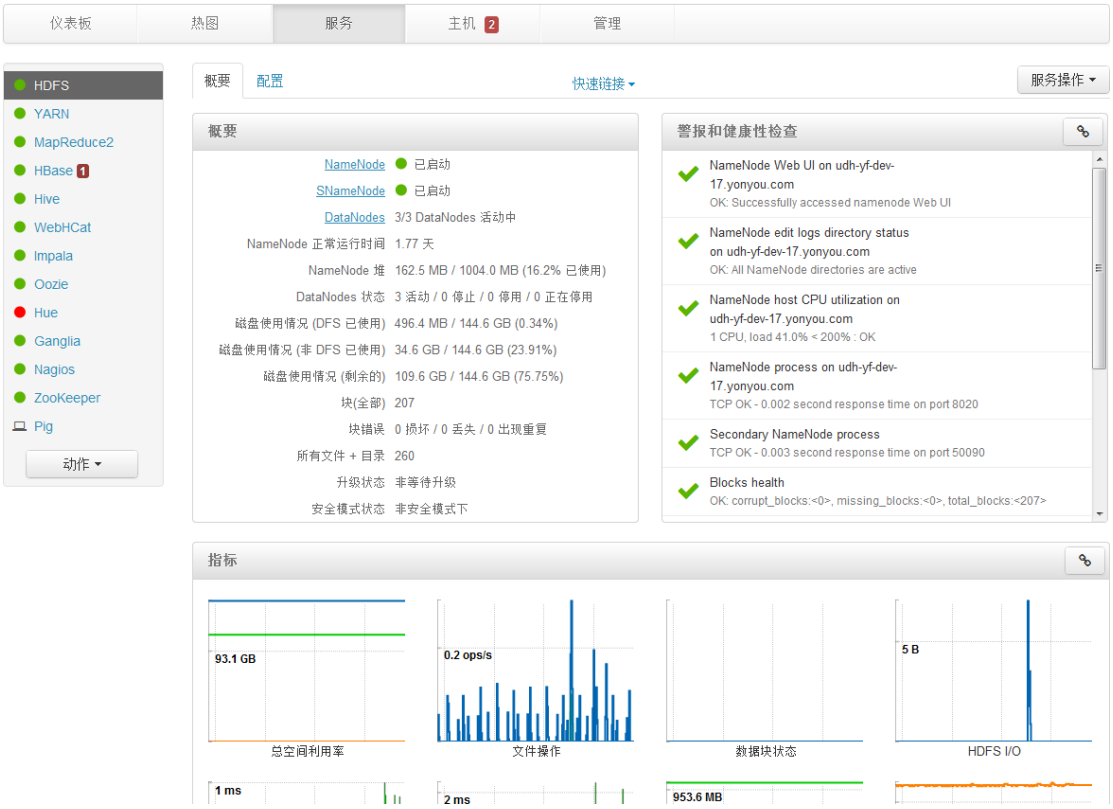

图 15 HDFS 组件详细信息

主机

记录了各个主机的名称、IP 地址、核心 CPU、RAM、磁盘使用情况、平均负载、组件等信息。通过醒目的标识显示各主机的当前状态，可以对某个主机上的运行的服务组件进行批量操作，如全部停止，启动，重启等，也可以针对具体的主机和个别的组件进行管理操作。

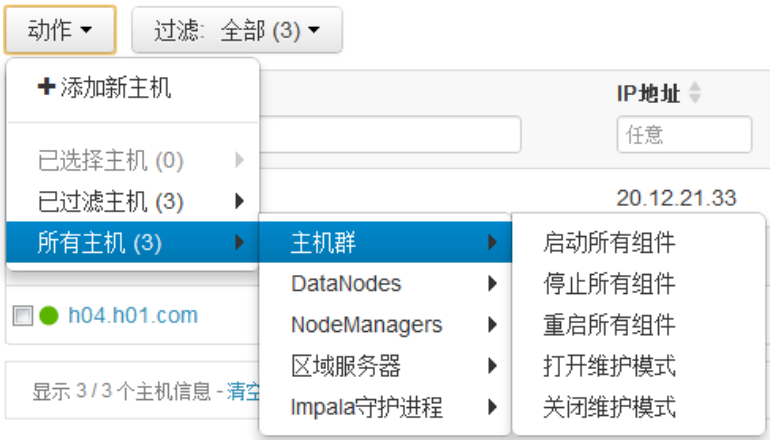

图 16 主机的批量管理操作

主机与服务相区别，提供了以运行主机的角度，按照主机粒度进行划分，对系统资源，服务组件状态的监控和管理。

管理

具有管理角色的用户可以看到“管理”页签，主要功能有：

1. 在“用户群”中，可对用户信息进行管理；
2. 在“High Availability”中，可以启动 HA；
3. 在“集群”中，可以查看当前集群管理器、各个服务的版本，以及各个服务的说明；
4. 在“杂项”中，可以查看集群中各个服务对应的用户信息，此用户信息指的是操作系统的用户。而非登录集群管理器的用户账号。

4.2.2. 服务组件管理

UDH 集群管理器支持多达十几种 Hadoop 服务组件，方便用户对集群中各组件进行统一监控管理，使用户从繁杂的系统维护中解放出来。

目前支持的服务组件有：HDFS，YARN,MapReduce2，HBase，Hive， WebHCat，Impala，Oozie，Hue，Ganglia，Nagios，Zookeeper。

详细说明，参见《UDH 集群管理器操作手册》。

4.2.3. 数据处理服务

UDH 管理器面向 UDH（hadoop）提供基于 Web 的数据分析处理工具。能够实现：

1. HDFS 的文件浏览，编辑，下载等
2. 面向 Hive 的查询编辑，运行，历史保存等
3. 面向 Impala 的查询编辑，运行，历史保存等。
4. 支持 Pig 脚本的编辑，保存，提交，运行日志等
5. Oozie 的可视化设计，提交以及运行监控等
6. Hive MetaStore 数据的浏览，可视化图表分析以及导出等。
7. HBase 数据的浏览，可视化图表分析以及导入等。
8. 查看 Zookeeper 的运行状态信息
9. 自定义 MapReduce，Streaming，Java 作业等以及调度执行和监控。

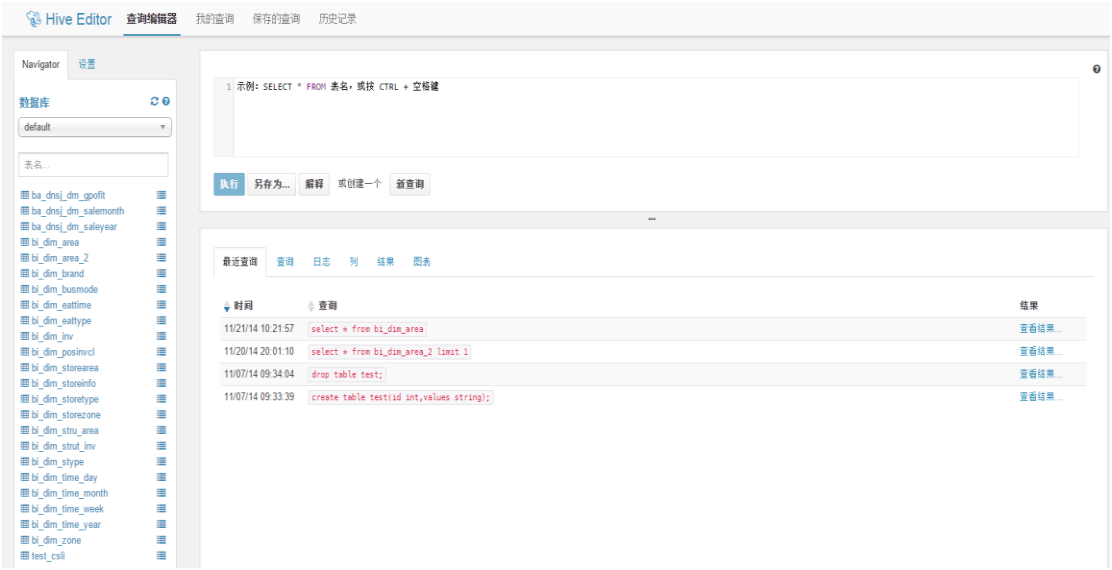

图 17 Hive 查询编辑器

演示地址：http://172.20.13.178:8000/

欢迎关注微信用友数据平台和 QQ 讨论群加入，反馈您的意见。