

Hive总体篇

www.huawei.com





前言

- **FusionInsight HD**中**Hive**组件在社区版本**Hive** 基础上，加入了众多企业级定制化特性，如**Colocation**建表，列加密，语法增强等特性。整个产品在高可靠，高容错，可扩展性以及性能等各方面较社区有巨大提升。



目标

培训与认证部

- 学完本课程后，您将能够：
 - 掌握**Hive**应用场景与基本原理；
 - 掌握**FusionInsight**中**Hive**增强特性；
 - 熟悉常用**Hive SQL**语句；



目录

培训与认证部

1. Hive概述

- 什么是Hive
- Hive在Hadoop中的位置
- Hive的优点以及缺点
- Hive的应用场景
- Hive与传统数据仓库的比较

2. Hive功能与架构

3. Hive 基本操作

什么是Hive

- **Hive**是基于**Hadoop**的数据仓库软件，可以查询和管理PB级别的分布式数据。

- 它提供了如下功能：

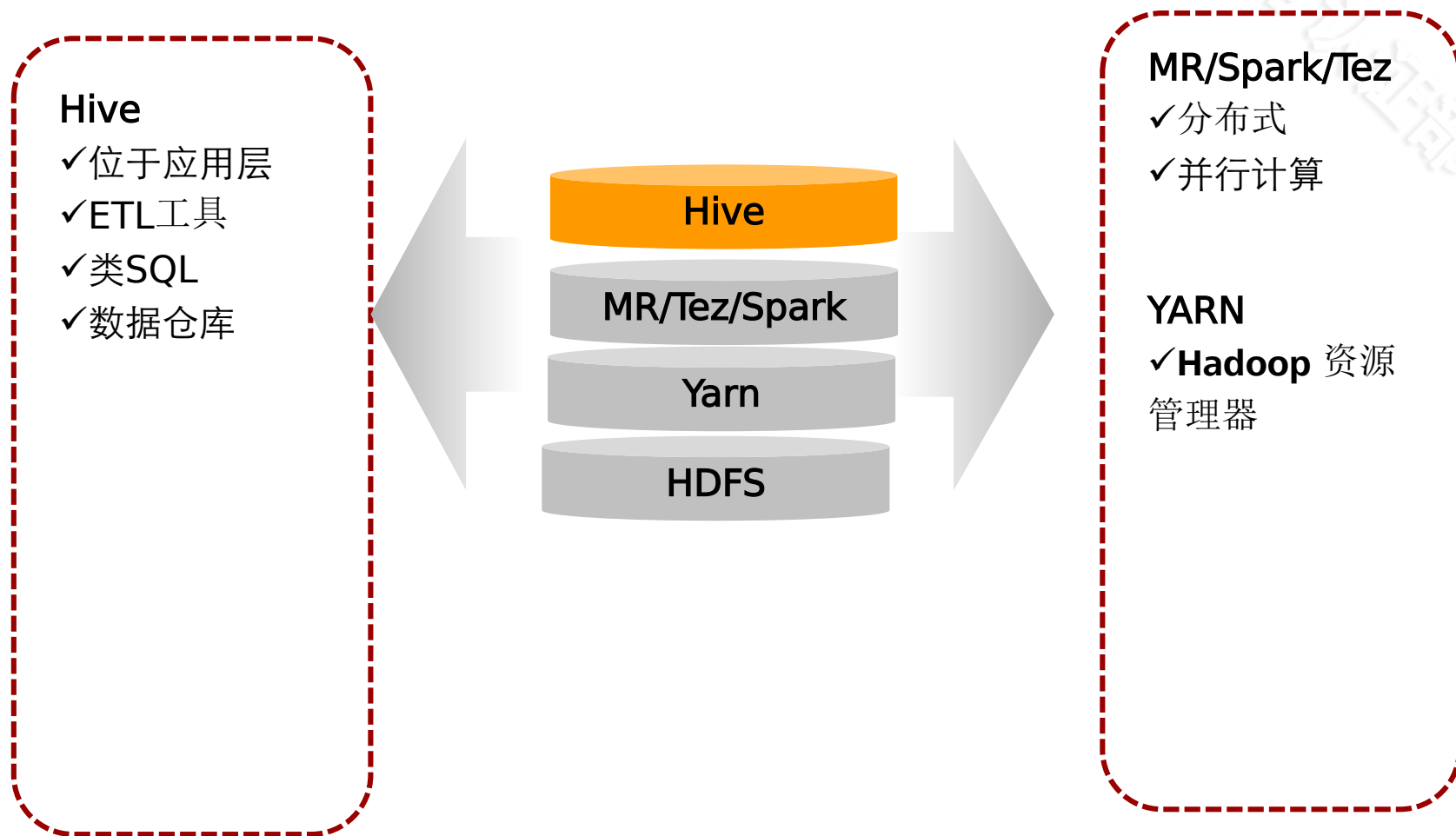
灵活方便的**ETL(extract/transform/load)**

多种文件格式的元数据服务

直接访问**HDFS**文件以及**HBase**

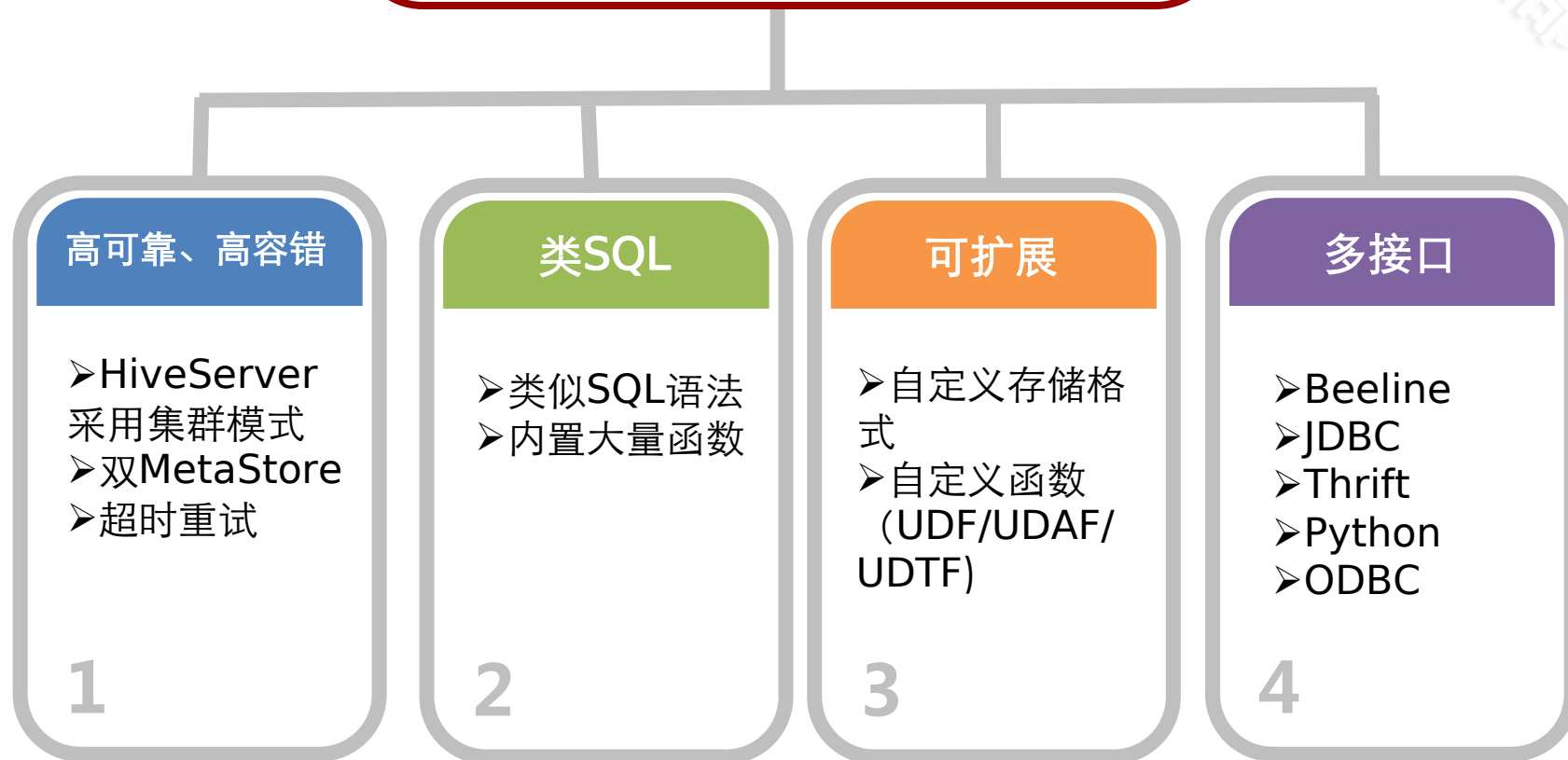
支持**MapReduce**，**Tez**，**Spark**等多种计算引擎

Hive在Hadoop中的位置



Hive的优点和缺点

Hive的优点



Hive的优点和缺点

Hive的缺点

延迟较高

- 默认MR为执行引擎
- MR延迟较高

1

不支持物化视图

- Hive虽然也提供了视图的概念；但还不能支持物化视图
- 不能在视图上更新、插入、删除数据

2

不适用OLTP

- 暂不支持列级别的数据添加、更新、删除操作

3

暂不支持存储过程

- 当前版本还不能支持存储过程，只能通过UDF来实现一些逻辑处理

4

Hive的应用场景

数据挖掘

- 用户行为分析
- 兴趣分区
- 区域展示

非实时分析

- 日志分析
- 文本分析

数据汇总

- 每天/每周用户点击数
- 流量统计

数据仓库

- 数据抽取
- 数据加载
- 数据转换

Hive与传统数据仓库比较

	Hive	传统数据仓库
存储	HDFS ，理论上有无无限拓展的可能。	集群存储，存在容量上限，而且伴随容量的增长，计算速度急剧下降。只能适应于数据量比较小的商业应用，对于超大规模数据无能为力。
执行引擎	有 MR/Tez/Spark 多种引擎可供选择。	可以选择更加高效的算法来执行查询，也可以进行更多的优化措施来提高速度。
使用方式	HQL （类似 SQL ）。	SQL 。
灵活性	元数据存储独立于数据存储之外，从而解耦合元数据和数据。	低，数据用途单一。
分析速度	计算依赖于集群规模，易拓展，在大数据量情况下，远远快于普通数据仓库。	在数据容量较小时非常快速，数据量较大时，急剧下降。

Hive与传统数据仓库比较

	Hive	传统数据仓库
索引	低效，目前还不完善。	高效。
易用性	需要自行开发应用模型，灵活度较高，但是易用性较低。	集成一整套成熟的的报表解决方案，可以较为方便的进行数据的分析。
可靠性	数据存储存储在HDFS，可靠性高，容错性高。	可靠性较低，一次查询失败需要重新开始。 数据容错依赖于硬件Raid。
依赖环境	依赖硬件较低，可适应一般的普通机器。	依赖于高性能的商业服务器。
价格	开源产品。	商用比较昂贵，开源的性能较低。



目录

培训与认证部

1. Hive概述

2. Hive功能与架构

- Hive的架构
- FusionInsight HD中Hive的架构
- FusionInsight HD中Hive增强特性
- Hive数据存储模型

3. Hive 基本操作

Hive的架构

MetaStore：存储表、列和Partition等元数据。

Driver：管理HiveQL执行的生命周期，并贯穿Hive任务整个执行期间。

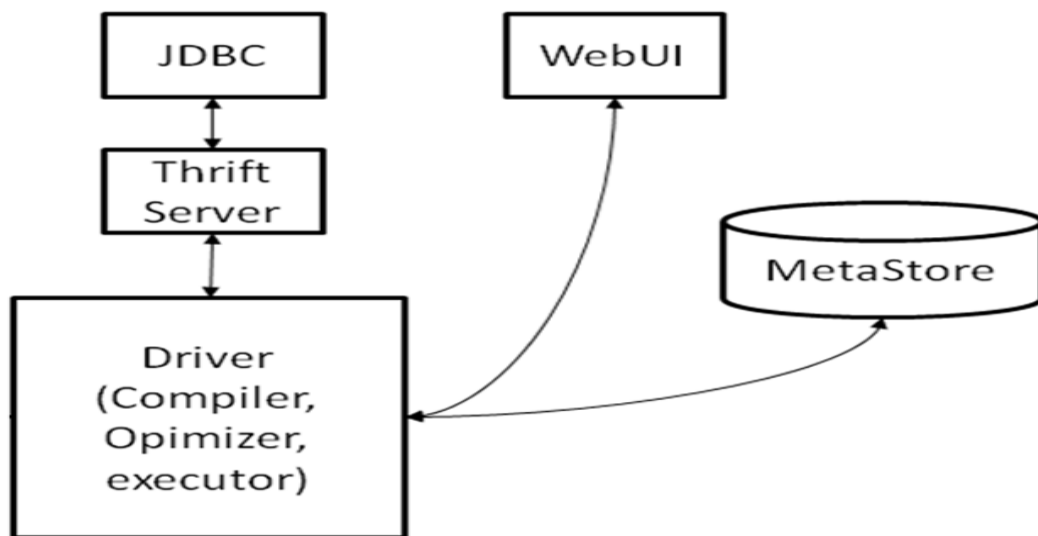
Compiler：编译HiveQL并将其转化为一系列相互依赖的Map/Reduce任务。

Optimizer：优化器，分为逻辑优化器和物理优化器，分别对HiveQL生成的执行计划和MapReduce任务进行优化。

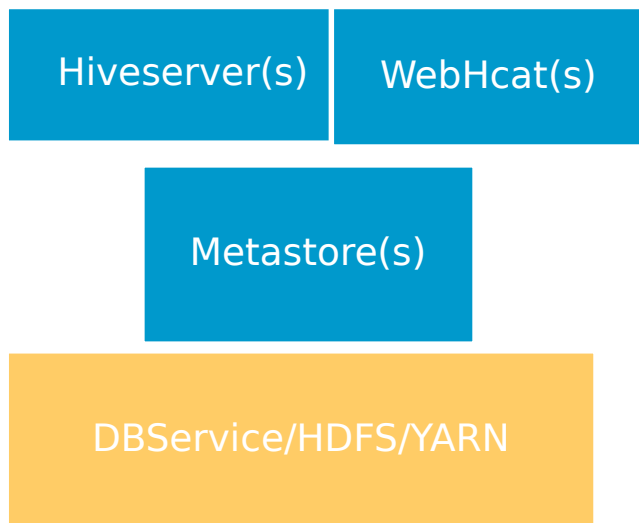
Executor：按照任务的依赖关系分别执行Map/Reduce任务。

ThriftServer：提供thrift接口，作为JDBC和ODBC的服务端，并将Hive和其他应用程序集成起来。

Clients：包含命令行接口(CLI/Beeline)和JDBC/ODBC 接口，为用户访问提供接口。



FusionInsight HD中Hive的架构

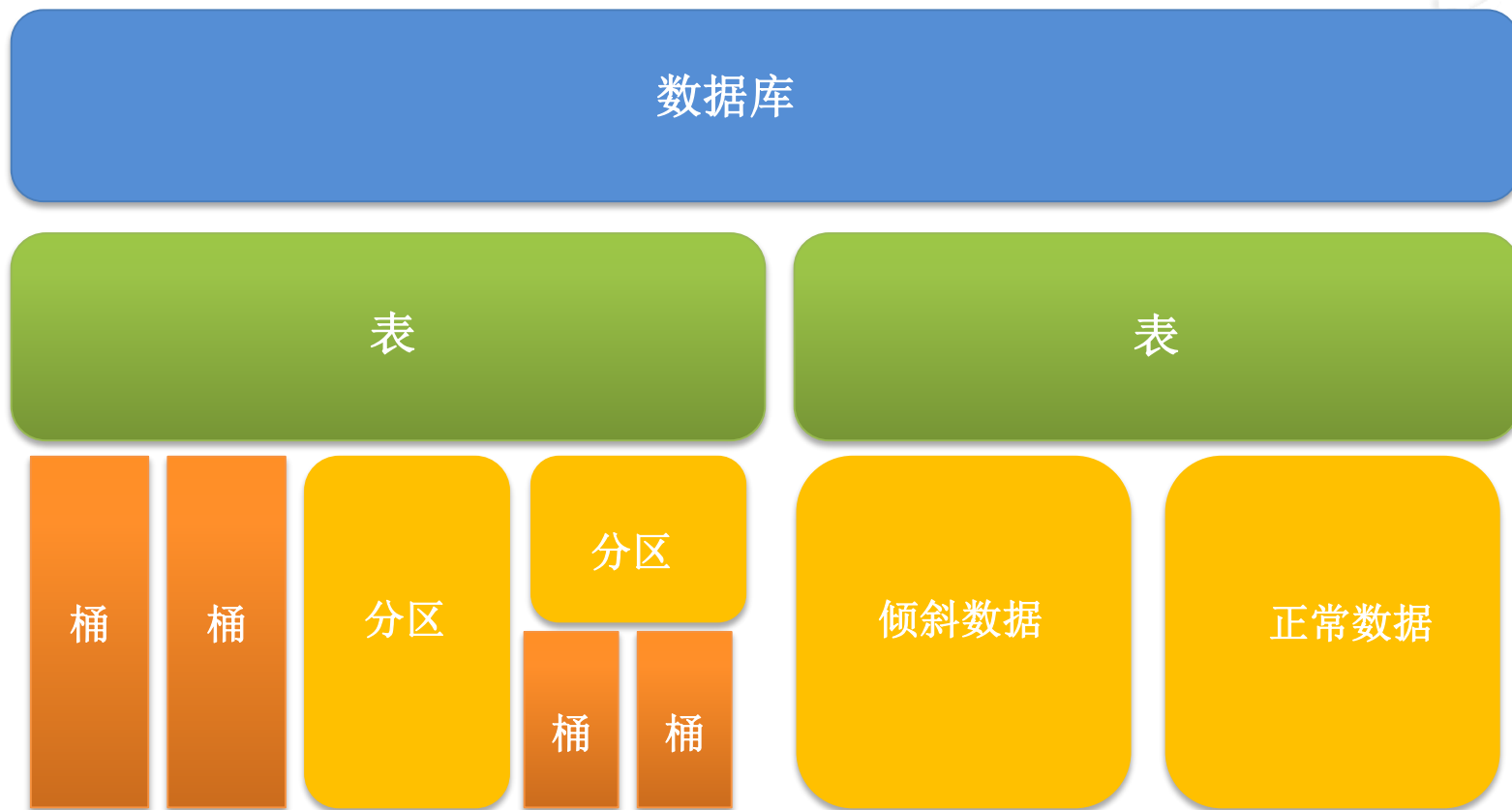


- **HiveServer**使用集群模式，即可同时有两个**HiveServer**提供服务。
- 可以启动一个或两个**MetaStore**进程来提供元数据服务。

Hive分为三个角色**HiveServer**、**MetaStore**、**WebHcat**。

- **HiveServer**负责接受客户端请求、解析、执行**HQL**命令并返回查询结果。
- **MetaStore**提供元数据服务。
- **WebHcat**负责对外提供元数据、**DDL**等**Http**服务。

Hive数据存储模型



Hive数据存储模型 – 分区和分桶

分区：数据表可以按照某个字段的值划分分区

- 每个分区是一个目录
- 分区数量不固定
- 分区下可再有分区或者桶
- 分区可以很明显的提高查询效率

桶：数据可以根据桶的方式将不同数据方式不同的桶中

- 每个桶是一个文件
- 建表时指定桶个数，桶内可排序
- 数据按照某个字段的值**Hash**后放入某个桶中
- 对于数据抽样、特定**join**的优化很有意义

Hive数据存储模型-托管表和外部表

Hive 默认创建托管表，由**Hive**来管理数据，意味着**Hive**会将数据移动到数据仓库目录。

另外一种选择是创建外部表，这时**Hive**会到仓库目录以外的位置访问数据。

如何选择？

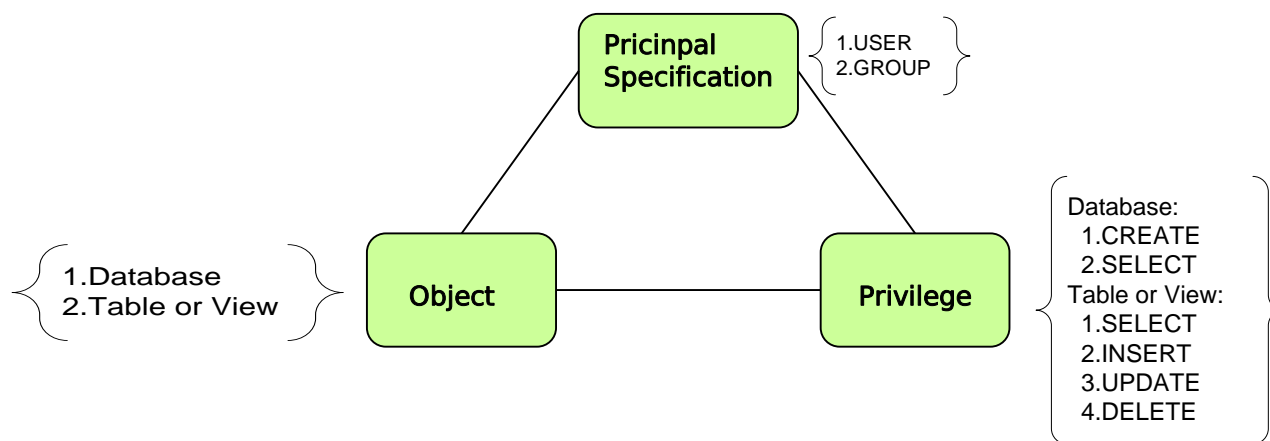
- 如果所有处理都由**Hive**完成，建议使用托管表
- 如果要用**Hive**和其它工具来处理同一个数据集，应该使用外部表。

	托管表	外部表
CREATE/LOAD	把数据移到仓库目录	创建表时指明外部数据的位置
DROP	元数据和数据会被一起删除	只删除元数据

Hive权限管理

- 权限模型

- Principal Specification--用户对象:Hive中的用户对象包括:USER|GROUP
- Object--数据库对象:Hive中可操作的数据库对象包括数据库、表和视图
- Privilege--权限类型:Hive中可以授权的权限主要有
CREATE|SELECT|INSERT|DELETE|UPDATE



Hive权限管理 -权限分布

培训与认证部

Action	Select	Insert	Update	Delete	Ownership	Admin
CREATE TABLE					Y (of database)	
DROP TABLE					Y	
DESCRIBE TABLE	Y					
SHOW PARTITIONS	Y					
ALTER TABLE LOCATION					Y	
ALTER PARTITION LOCATION					Y	
ALTER TABLE ADD PARTITION		Y				

Hive权限管理 - 权限分布

Action	Select	Insert	Update	Delete	Ownership	Admin
ALTER TABLE DROP PARTITION				Y		
ALTER TABLE (all of them except the ones above)					Y	
TRUNCATE TABLE					Y	
CREATE VIEW	Y + G					
ALTER VIEW PROPERTIES					Y	
ALTER VIEW RENAME					Y	
DROP VIEW PROPERTIES					Y	
DROP VIEW					Y	
ANALYZE TABLE	Y	Y				
SHOW COLUMNS	Y					
SHOW TABLE STATUS	Y					

Hive权限管理 -权限分布

Action	Select	Insert	Update	Delete	Ownership	Admin
SHOW TABLE PROPERTIES	Y					
CREATE TABLE AS SELECT	Y				Y (of database)	
CREATE INDEX					Y (of table)	
DROP INDEX					Y	
ALTER INDEX REBUILD					Y	
ALTER INDEX PROPERTIES					Y	
SELECT	Y					
INSERT		Y		Y		
UPDATE			Y			
DELETE				Y		
LOAD		Y		Y		
SHOW CREATE TABLE	Y+G					
CREATE FUNCTION						Y

Hive权限管理 - 权限分布

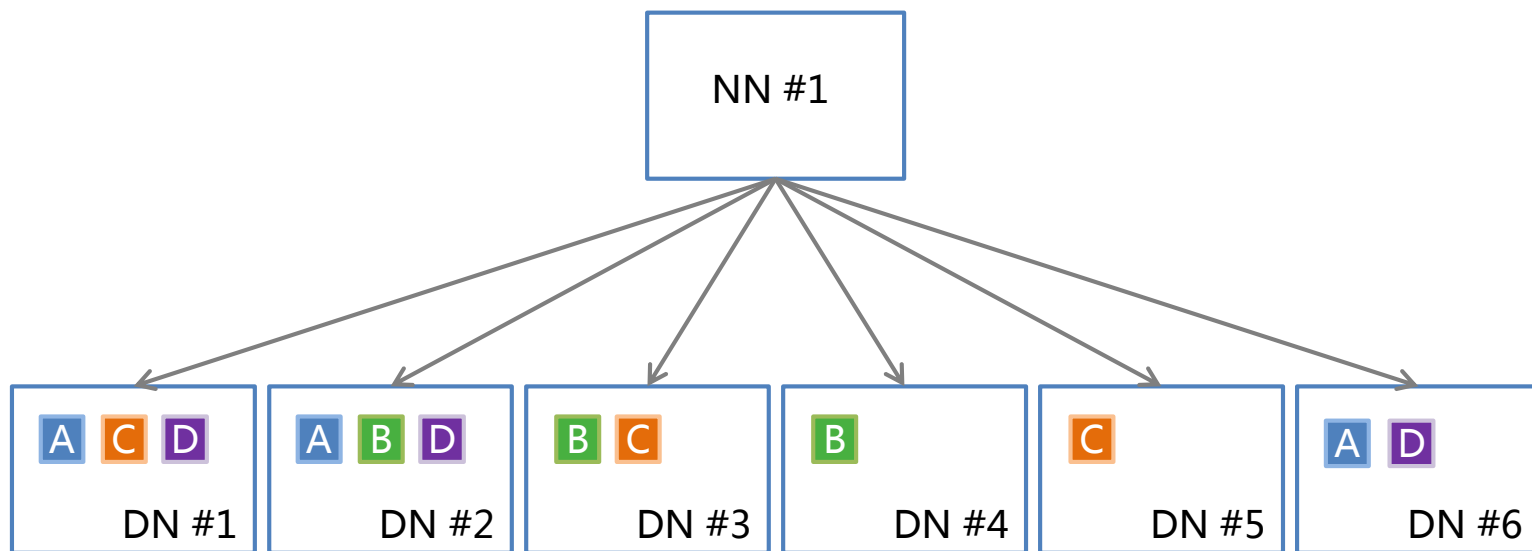
Action	Select	Insert	Update	Delete	Ownership	Admin
DROP FUNCTION						Y
MSCK (metastore check)						Y
ALTER DATABASE						Y
CREATE DATABASE						
EXPLAIN	Y					
DROP DATABASE					Y	
CREATE ROLE						Y
DROP ROLE						Y
Y: Privilege required.						
Y + G: Privilege "WITH GRANT OPTION" required.						

Hive增强特性

- 基于**HDFS Colocation**特性建表
- 列加密
- 语法增强
- **HBase**表批量记录删除功能
- 流控特性
- 指定行分隔符

Hive增强特性 – Colocation简介

Colocation (同分布): 将存在关联关系的数据或可能要进行关联操作的数据存储在相同的存储节点上。



文件级同分布实现文件的快速访问，避免了因数据搬迁带来的大量网络开销。

Hive增强特性 – Colocation使用

- 步骤1：通过HDFS接口创建groupid

```
hdfs colocationadmin -createGroup -groupId groupid -  
locatorIds locatorid1,locatorid2,locatorid3
```

- 步骤2：在Hive中使用Colocation

```
CREATE TABLE tbl_1 (id INT, name STRING) stored as  
RCFILE  
TBLPROPERTIES ("groupId"="group1", "locatorId"="locator1"  
);
```

tbl_1和tbl_2有
关联关系

```
CREATE TABLE tbl_2 (id INT, name STRING) row format  
delimited fields terminated by '\t' stored as  
TEXTFILE  
TBLPROPERTIES ("groupId"="group1", "locatorId"="locator1"  
);
```

Hive增强特性 – 列加密

- 步骤1：在创建表时指定相应的加密列和加密算法

```
create table encode_test (id INT, name STRING, phone  
STRING, address STRING) row format serde  
'org.apache.hadoop.hive.serde2.lazy.LazySimpleSerDe'  
WITH SERDEPROPERTIES (  
    'column.encode.columns'='phone,address', 'column.encode.  
    classname'='org.apache.hadoop.hive.serde2.AESRewriter'  
)
```

- 步骤2：使用insert语法向设置列加密的表中导入数据

```
insert into table encode_test select id, name, phone,  
address from test;
```

Hive增强特性—语法增强

语法Intersect: $A \cap B$ 交集, 要求字段数量相同, 顺序匹配

```
select name,id from tb_1 intersect select name,id from tb_2
```

转换成:

```
select name,id from tb_1 a left semi join tb_2 b on a.name = b.name and  
a.id = b.id;
```

语法Except: $A - B$ 差集, 要求字段数量相同, 顺序匹配

```
select name,id from tb_1 except select name,id from tb_2
```

转换成:

```
select name,id from tb_1 a left outer join tb_2 b on a.name = b.name and  
a.id = b.id where b.name is null and b.id is null;
```

Hive增强特性 – HBase记录批量删除

概要说明：

- 在**Hive on HBase**功能中，**FusionInsight HD Hive**提供了对**HBase**表的单条数据的删除功能，通过特定的语法，**Hive**可以将**HBase**表中符合条件的一条或者多条数据清除。

使用：

- 如果要删除某张**HBase**表中的某些数据，可以执行**HQL**语句：
remove table hbase_table where *expression*;
- 其中**expression**规定要删除数据的筛选条件。

Hive增强特性 – 流控特性

通过流控特性，可以实现：

- 当前已经建立的总连接数阈值控制；
- 每个用户已经建立的连接数阈值控制；
- 单位时间内所建立的连接数阈值控制；

Parameter	Value
hive.default.fileformat	<input type="checkbox"/> TextFile <input type="checkbox"/> SequenceFile <input checked="" type="checkbox"/> RCfile <input type="checkbox"/> ORC
* hive.exec.compress.output	<input type="radio"/> true <input checked="" type="radio"/> false
hive.exec.reducers.max	999
hive.mapred.mode	<input checked="" type="checkbox"/> nonstrict <input type="checkbox"/> strict
hive.server.session.control.maxconnection.peruser	500
hive.server.session.control.maxconnections	500
hive.server.timewindow.delaytime	60
hive.server.timewindow.maxsessions.in.delaytime	500
hive.server2.idle.session.timeout	4320m
hive.server2.session.check.interval	3000ms

Hive增强特性 – 指定行分割符

- 步骤1：创建表时指定InputFormat和OutputFormat

```
CREATE [TEMPORARY] [EXTERNAL] TABLE [IF NOT EXISTS]
[db_name.]table_name
[(col_name data_type [COMMENT col_comment], ...)]
[ROW FORMAT row_format]
STORED AS
inputformat
'org.apache.hadoop.hive.contrib.fileformat.SpecifiedDelimitedTextInputFormat'
outputformat
'org.apache.hadoop.hive.ql.io.HiveIgnoreKeyTextOutputFormat';
```

- 步骤2：查询之前指定配置项

```
set hive.textinput.record.delimiter='!@!\';
```



目录

培训与认证部

1. Hive应用场景介绍
2. Hive功能与架构
3. Hive基本操作
 - ▣ Hive SQL介绍
 - ▣ Hive基本操作 – DDL
 - ▣ Hive基本操作 – DML
 - ▣ Hive基本操作 – DQL
 - ▣ Hive中的函数

Hive SQL介绍

DDL-数据定义语言

- ◆ 建表，修改表，删表、分区、数据类型

DML-数据管理语言

- ◆ 数据导入
- ◆ 数据导出

DQL-数据查询语言

- ◆ 一般查询
- ◆ Group by, Order by, Cluster by, Join, Union
- ◆ 子查询

函数（内置函数/UDF/UDAF/UDTF）

Hive基本操作 – DDL之创建表

```
CREATE TABLE IF NOT EXISTS example.employee
(Id INT COMMENT 'employeeid',
DateInCompany STRING COMMENT 'date come in company',
Money FLOAT COMMENT 'work money',
Mapdata Map<STRING,ARRAY<STRING>>,
Arraydata ARRAY<INT>,
StructorData STRUCT<col1:STRING,col2:STRING>)
PARTITIONED BY (century STRING COMMENT 'centruey come in
company',
year STRING COMMENT 'come in company year')
CLUSTERED BY (DateInCompany) into 32 buckets
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','
COLLECTION ITEMS TERMINATED BY '@'
MAP KEYS TERMINATED BY '$'
STORED AS TEXTFILE;
```

数据格式示例: 1,huawei,1000.0,m1\$a,1@2@3@4,c1@c2

Hive基本操作 – DDL之创建外部表

```
CREATE EXTERNAL TABLE IF NOT EXISTS company.person  
(Id int,Name string,Age int,Birthday string)  
PARTITIONED BY (century string ,year string)  
CLUSTERED BY (Age) INTO 10 buckets  
ROW FORMAT DELIMITED FIELDS TERMINATED BY ','  
STORED AS sequencefile  
LOCATION '/localtest';
```

注：创建外部表使用**External**关键字，同时需要指定**HDFS**上的一个路径（非必须）为外部表的数据源。

Hive基本操作 – DDL之修改表

--修改列

```
ALTER TABLE employee1 CHANGE money string COMMENT 'changed by alter' AFTER dateincompany;
```

--添加列

```
ALTER TABLE employee1 ADD columns(column1 string);
```

--修改文件格式

```
ALTER TABLE employee3 SET fileformat TEXTFILE;
```

--添加分区

```
ALTER TABLE employee ADD IF NOT EXISTS PARTITION  
(century= '21',year='2012');
```

--删除分区

```
ALTER TABLE employee DROP PARTITION (century=  
'23',year='2010');
```

Hive基本操作-DML

--加载数据

--主要有从本地加载、从HDFS加载和从另一个表加载三种。

对应的命令：

LOAD --本地加载、从HDFS加载

INSERT INTO --从另一个表加载

--导入和导出数据

EXPORT -- 从Hive表中导出数据到HDFS

IMPROT -- 从HDFS导入数据到Hive表

Hive基本操作-DML

--从本地文件家中数据到Hive表

```
LOAD DATA LOCAL INPATH 'employee.txt' OVERWRITE INTO TABLE  
example.employee partition (century='21',year='2012');
```

--从另一个表加载数据到Hive表

```
INSERT INTO TABLE company.person PARTITION(century=  
'21',year='2010')  
SELECT id, name, age, birthday FROM company.person_tmp  
WHERE century= '23' AND year='2010';
```

--导出数据到HDFS

```
EXPORT TABLE company.person TO '/department';
```

--从HDFS导入数据

```
IMPROT TABLE company.person FROM '/department';
```

注：导入数据到Hive表时，不会检查数据合法性，只会在读取数据时候检查。

Hive基本操作-查询

--Group by having

```
SELECT dateincompany, sum(money) AS mm FROM employee11 GROUP BY  
DateInCompany HAVING mm>3;
```

--Union ALL & sub-Query

```
SELECT u.id, actions.date FROM ( SELECT av.uid AS uid,av.date  
    FROM action_video av WHERE av.date = '2008-06-03' UNION ALL  
    SELECT ac.uid AS uid,ac.date  
    FROM action_comment ac  
    ) actions  
JOIN users u ON (u.id = actions.uid)  
Limit 100;
```

Hive基本操作-表连接

- 表连接（Join）语句将数据库中的两个或多个表组合起来作为查询数据集。常用于两个或者多个表跨表查询的场景。

- Inner join

Hive只支持等值连接；

JOIN 子句中表的顺序很重要，一般最好将最大的表在最后。

- Outer join

外连接可以让你找到连接表中不能匹配的数据行；


- Semi join

可以使用LEFT SEMI JOIN替代in关键字以达到相同效果。

- Map join

该查询Job没有reducer；

使用时充分利用Bucketed Table，需要设置hive.optimize.bucketmapjoin为true；



慎重使用！
看懂再用

Hive基本操作 – 函数

Hive内置函数:

- 数学函数，如`round()`,`floor()`,`abs()`,`rand()`等。
- 日期函数，如`to_date()`,`month()`,`day()`等。
- 字符串函数，如`trim()`,`length()`,`substr()`等。

UDF (User- Defined Funcation)

UDAF (User- Defined Aggregation Funcation)

UDTF (User- Defined Table-Generating Funcation)

注：如果内置函数不能满足用户需求时，Hive可支持自定义函数。

Hive基本操作 – 函数使用举例

--UDF: 普通的字段处理函数, 如trim

```
SELECT trim(a.val), b.val, c.val FROM a JOIN b ON (a.key = b.key1) JOIN c  
ON (c.key = b.key1);
```

--UDAF: 聚合函数, 如sum, count, avg

```
SELECT count(1) FROM a;
```

--UDTF: 分组数据的侧视图, 即将数组展开, 放在不同的列中。

```
SELECT explode(a.arrdata) FROM a;
```



本章总结

- 通过本章的学习，介绍了**Hive**应用场景与基本原理，然后介绍了**Hive**在**FusionInsight** 中的增强特性；接下来介绍常用的**Hive SQL**语句。

思考题

以下哪些是Hive适用的场景？（ ）

- A. 实时的在线数据分析
- B. 数据挖掘（用户行为分析，兴趣分区，区域展示）
- C. 数据汇总（每天/每周用户点击数，点击排行）
- D. 非实时分析（日志分析，统计分析）



思考题

以下哪几项不属于FusionInsight HD中Hive的增强特性？（ ）

- A. 建表时可针对列指定加密算法
- B. 建表示可指定数据存储类型如TextFile,ORC,RCFile等，同时也可自定义存储类型
- C. 创建表时可以指定行分隔符
- D. 创建表时可以指定列分隔符



思考题

培训与认证部

以下关于Hive SQL基本操作描述正确的是？（ ）

- A. 创建外部表使用external关键字，创建普通表需要指定internal关键字
- B. 创建外部表必须要指定location信息
- C. 加载数据到Hive时源数据必须是HDFS的一个路径
- D. 分区可以在创建表时指定也可在创建表后通过Alter命令添加分区



学习推荐

- Hive社区
 - <https://cwiki.apache.org/confluence/display/Hive/GettingStarted>
- 华为产品资料
 - <http://e.huawei.com/cn/products/cloud-computing-dc/cloud-computing/bigdata/fusioninsight>

谢谢

www.huawei.com