Hive

宋树成、徐展鹏、张子康、崔杰

1. 数据仓库与Hive

1.什么是数据仓库

》数据仓库：

1. 可简写为DW或DWH；
2. 数据仓库软件则是管理，存储数据仓库的软件系统。
3. 数据仓库是出于分析性报告和决策支持目的而创建，供指导业务流程改进，监视时间，成本，质量以及控制。

》Bill lnmon于1990年提出数据仓库

（1）数据仓库是一个面向主题的，集成的，相对稳定的，反应历史变化的数据集合，用于支持管理决策

2.数据仓库的发展

》数据仓库的特点

（1）面向主题的

（2）集成的，数据来源广

（3）不可更新的

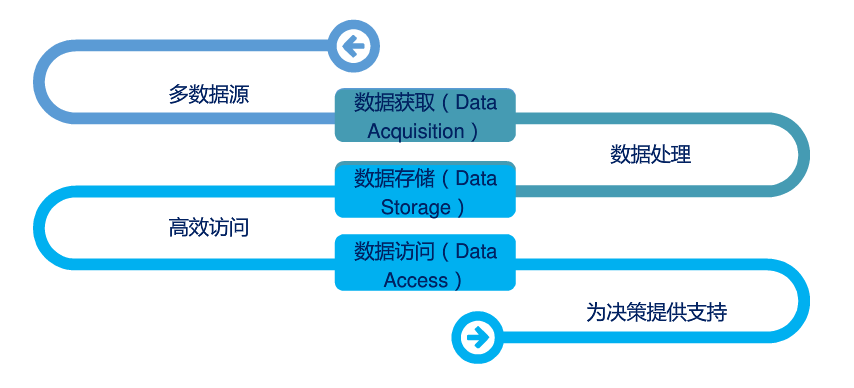
（4）数据仓库是随时间而变化的

（5）汇总的

（6）大容量

（7）非规范化的，经常是冗余的

3．数据仓库软件功能



二．Hive的应用接口

1.Hive的定义

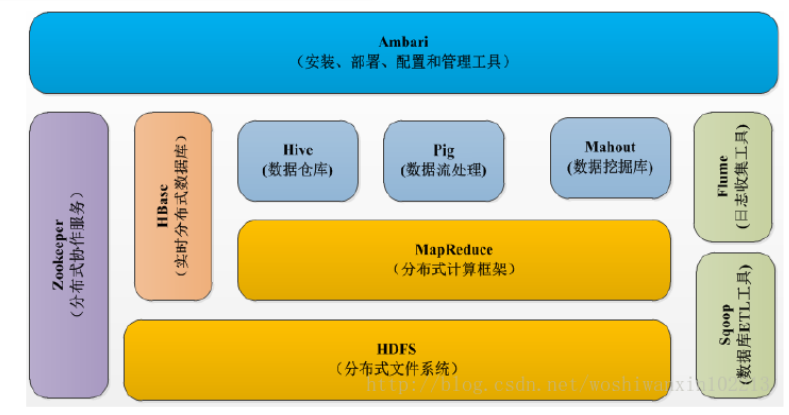
（1）是一个数据仓库软件

（2）目的在于为分布式存储的大数据提供基于SQL的读，写，管理

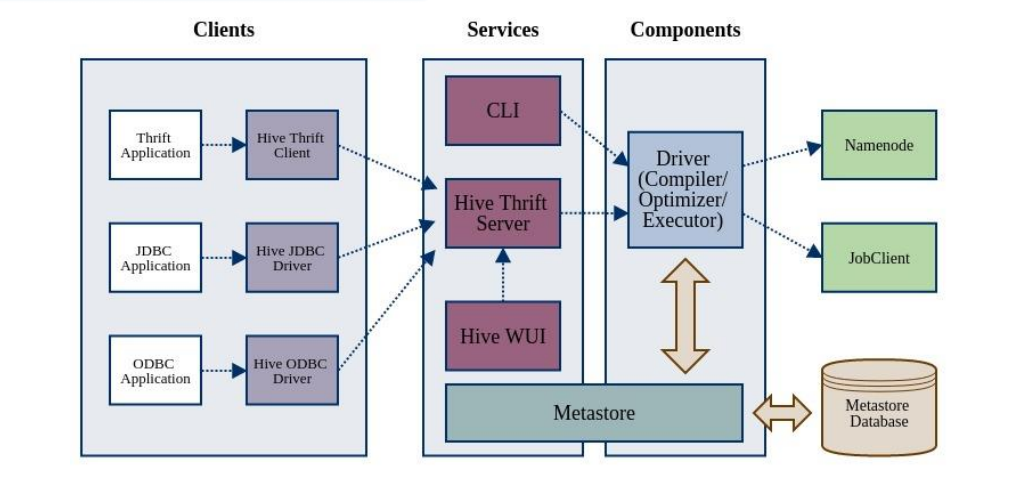
（3）提供分布式存储数据到现有数据的投影

（4）提供命令行工具和JDBC驱动供用户链接Hive

2．Hive在Hadoop2.0中的位置



3．Hive的用户接口



4.Hive的应用接口

（1）CLI（Command Line Interface）是和Hive交互的最简单/最常用的方式，只需在具备完整Hive环境下的Shell终端中键入hive即可启动

（2）WUI（Web User Interface）是通过浏览器访问Hive

（3）Hive Thrift Server是基于Thrift软件框架开发的，它提供Hive的RPC的通信接口

（4）Thrift Client:Thrift客户端采用Hive Thrift Server提供的接口来访问Hive

（5）Hive也提供了封装好Thrift RPC的Python Client 和 Ruby Client

（6）Thrift Client的优点在于程序员不再依赖Hive环境来访问Hive数据仓库

（7）JDBC Client:Hive官方已经实现了JDBC Driver

（8）ODBC Client:可以在微软的官网下载ODBC Driver

三．Hive存储的几个概念

1.Hive的元数据

Hive中的元数据包括表的名字，表的列和分区及属性，表的属性（是否为外部表等），表的数据所在目录等。由于Hive的元数据需要不断的更新丶修改，而HDFS系统中的文件是多读少改的，这显然不能将Hive的元数据存储在HDFS中。目前Hive将元数据存储在数据库中，如Mysql,Derby中。我们可以通过修改Hive的配置文件以修改元数据的存储方式。

1. Hive存储模型
2. Hive的数据存储

HIve是基于Hadoop分布式文件系统的，它的数据存储在Hadoop分布式文件系统中。

HIve本身是没有专门的数据存储格式，也没有为数据建立索引，只需要在创建表的时候告诉Hive数据

中的列分隔符和行分隔符，Hive就可以解析数据。

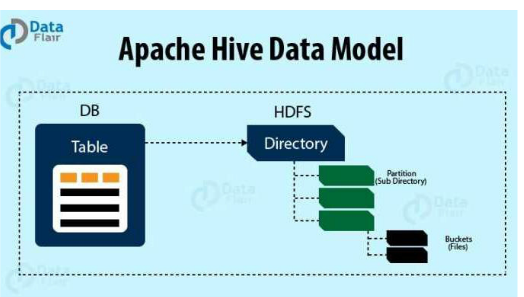
往Hive表里面导入数据只是简单的将数据移动到表所在的目录

1. 几个相关概念

Table(表)，External Table(外部表),Partition(分区),Bucket(桶)

①表：Hive中每个表在HDFS中都有相应的目录用来存储表的数据。

实例：若有表 tbl\_stu，那么在HDFS中会创建一个目录： ${HIVE\_HOME}/ warehouse/ tbl\_stu ，该表所有的数据都存放在这个目录中



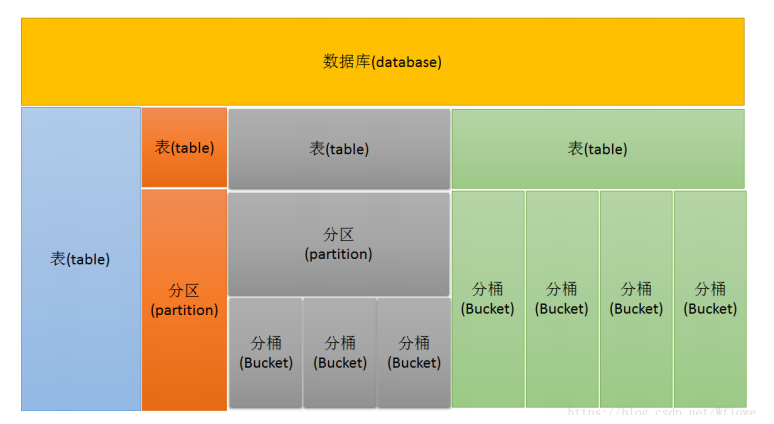
②分区：在Hive中，表的每一个分区对应表下的相应目录，所有分区的数据都是存储在对应的目录中。 

实例：比如tbl\_stu表有dept和grade两个分区，则对应dept=SE,grade=2017对 应表的目录为${HIVE\_HOME}/warehouse/dept=SE/grade=2017，所有属于这个分区的数据都存放在这个目录中

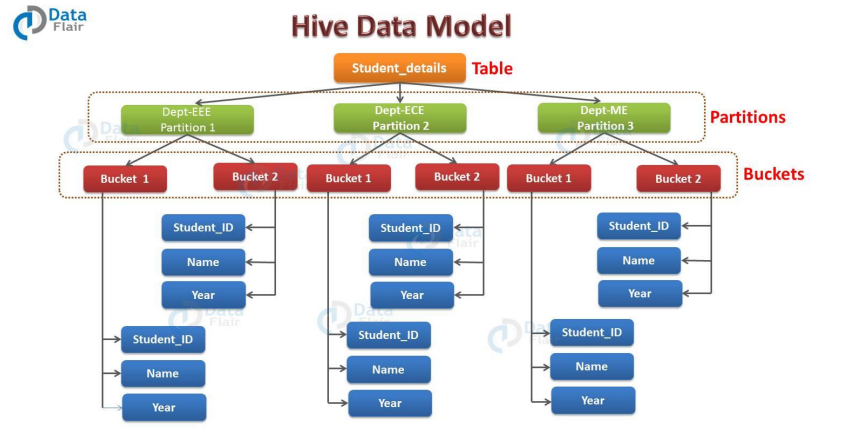
③桶：对指定的列计算其hash， 根据hash值切分数据，目的是 为了提高并行，每一个桶对应一个文件。

实例：将tbl\_stu表sid列分散 至16个桶中，对sid列的值计 算hash，对应hash值为0的数据存储的HDFS目录为： ${HIVE\_HOME}/warehouse/ tbl\_stu /part-00000;而hash值 为2的数据存储的HDFS 目录为： ${HIVE\_HOME}/ warehouse/tbl\_stu/part-00002。

（3）



1. Hive存储模型实例



1. Hive的外部表

（1）外部表：Hive中的外部表和表很类似，但是其数据不是放在自己表所属的目录中，而是存放到别处。

（2）优点：如果你要删除这个外部表，该外部表所指向的数据是不会被删除的，它只会删除外部表对应的元数据；而如果你要删除表，该表对应的所有数据包括元数据都会被删除。

四．Hive的简单CLI操作

1.Hive的命令行接口

（1）CLI

 ①DML（Data Manipulation Language）Operations

LOAD, INSERT, UPDATE, DELETE, IMPORT/EXPORT,…

 ②DDL （Data Definition Language）Operations

CREATE, DROP, SHOW, TRUNCATE, DESCRIBE, ALTER,…

 ③ Hive-QL （Hive Query Language）Operations

1. Hive的数据定义语言DDL

（1）一些DDL列表

①CREATE DATABASE/SCHEMA, TABLE, VIEW, FUNCTION, INDEX

②DROP DATABASE/SCHEMA, TABLE, VIEW, INDEX

③TRUNCATE TABLE

④ALTER DATABASE/SCHEMA, TABLE, VIEW

⑤MSCK REPAIR TABLE (or ALTER TABLE RECOVER PARTITIONS)

⑥SHOW DATABASES/SCHEMAS, TABLES, TBLPROPERTIES, VIEWS,

PARTITIONS, FUNCTIONS, INDEX[ES], COLUMNS, CREATE TABLE

⑦DESCRIBE DATABASE/SCHEMA, table\_name, view\_name

 （2） 参考网址：

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DDL>

（3） 创建 Hive Tables

hive> CREATE TABLE pokes (foo INT, bar STRING);

hive> CREATE TABLE invites (foo INT, bar STRING) PARTITIONED BY (ds STRING);

 （4）列出所有Hive Tables

hive> SHOW TABLES;

 （5）列出所有以’s’结尾的表

hive> SHOW TABLES '.\*s';

 （6） 显示表pokes结构

hive> DESCRIBE pokes;

 （7） 删除表:

hive> DROP TABLE pokes;

1. Hive的数据操作语言DML

 （1）一些DML列表

 ① LOAD

 ② INSERT

 ③ into Hive tables from queries

 ④ into directories from queries

 ⑤ into Hive tables from SQL

 ⑥ UPDATE

 ⑦DELETE

 ⑧ MERGE

参考网址：

<https://cwiki.apache.org/confluence/display/Hive/LanguageManual+DML>

（2）从一般文本文件中加载数据到Hive表

hive> LOAD DATA LOCAL INPATH './examples/files/kv1.txt'

OVERWRITE INTO TABLE pokes;

要求：文本文件包含两列，有ctrl-a分割；LOCAL说明文件来自本地

文件系统，缺省情况下，说明文件来自HDFS

（3）从HDFS文件加载数据

hive> LOAD DATA INPATH '/user/myname/kv2.txt'

OVERWRITE INTO TABLE invites PARTITION (ds='2008-08-15');

1. Hive的HQL支持--SELECT

（1） Select查询

SELECT [ALL | DISTINCT] select\_expr, select\_expr, ...

FROM table\_reference

[WHERE where\_condition]

[GROUP BY col\_list]

[ORDER BY col\_list]

[CLUSTER BY col\_list

| [DISTRIBUTE BY col\_list] [SORT BY col\_list]

]

[LIMIT [offset,] rows]

（2） SELECTS and FILTERS

hive> SELECT a.foo FROM invites a WHERE a.ds='2008-08-15';

hive> SELECT a.\* FROM invites a WHERE a.ds='2008-08-15';

hive> SELECT a.\* FROM pokes a; 

（3） GROUP BY

hive> SELECT a.bar, count(\*) WHERE a.foo > 0 GROUP BY a.bar;

（4） JOIN

hive> SELECT t1.bar, t1.foo, t2.foo

FROM pokes t1 JOIN invites t2 ON (t1.bar = t2.bar)