1、impala有什么特点?

1、基于内存进行计算，能够对PB级数据进行交互式实时查询、分析

2、无需转换为MR，直接读取HDFS数据

3、C++编写，LLVM统一编译运行

4、兼容HiveSQL

5、具有数据仓库的特性，

6、可对hive数据直接做数据分析

7、支持列式存储

8、支持Data Local

9、支持JDBC/ODBC远程访问

2、impala有哪些核心组件?

Statestore Daemon

•实例*1-statestored

–负责收集分布在集群中各个impalad进程的资源信息、各节点健康状况，同步节点信息.

–负责query的调度

Catalog Daemon

•实例*1-catalogd

–分发表的元数据信息到各个impalad中

–接收来自statestore的所有请求

Impala Daemon（具有数据本地化的特性所以放在DataNode上）

•实例*N–impalad

–接收client、hue、jdbc或者odbc请求、Query执行并返回给中心协调节点

–子节点上的守护进程，负责向statestore保持通信，汇报工作

3、impala和hive有什么区别?

1、Impala是Cloudera公司主导开发的新型查询系统，它提供SQL语义，能查询存储在Hadoop的HDFS和HBase中的PB级大数据。已有的Hive系统虽然也提供了SQL语义，但由于Hive底层执行使用的是MapReduce引擎，仍然是一个批处理过程，难以满足查询的交互性。相比之下，Impala的最大特点也是最大卖点就是它的快速。

2、相同点：

- 数据存储：使用相同的存储数据池都支持把数据存储于HDFS, HBase。
- 元数据：两者使用相同的元数据。
- SQL解释处理：比较相似都是通过词法分析生成执行计划。

3、不同点：

1.执行计划：

Hive: 依赖于MapReduce执行框架，执行计划分成map->shuffle->reduce->map->shuffle->reduce...的模型。如果一个Query会被编译成多轮MapReduce，则会有更多的写中间结果。由于MapReduce执行框架本身的特点，过多的中间过程会增加整个Query的执行时间。

Impala: 把执行计划表现为一棵完整的执行计划树，可以更自然地分发执行计划到各个Impalad执行查询，而不用像Hive那样把它组合成管道型的map->reduce模式，以此保证Impala有更好的并发性和避免不必要的中间sort与shuffle。

2.数据流：

Hive: 采用推的方式，每一个计算节点计算完成后将数据主动推给后续节点。

Impala: 采用拉的方式，后续节点通过getNext主动向前面节点要数据，以此方式数据可以流式的返回给客户端，且只要有1条数据被处理完，就可以立即展现出来，而不用等到全部处理完成，更符合SQL交互式查询使用。

3.内存使用：

Hive: 在执行过程中如果内存放不下所有数据，则会使用外存，以保证Query能顺序执行完。每一轮MapReduce结束，中间结果也会写入HDFS中，同样由于MapReduce执行架构的特性，shuffle过程也会有写本地磁盘的操作。

Impala: 在遇到内存放不下数据时，当前版本0.1是直接返回错误，而不会利用外存，以后版本应该会进行改进。这使用得Impala目前处理Query会受到一定的限制，最好还是与Hive配合使用。Impala在多个阶段之间利用网络传输数据，在执行过程不会有写磁盘的操作（insert除外）。

4.调度：

Hive: 任务调度依赖于Hadoop的调度策略。

Impala: 调度由自己完成，目前只有一种调度器simple-schedule，它会尽量满足数据的局部性，扫描数据的进程尽量靠近数据本身所在的物理机器。调度器目前还比较简单，在SimpleScheduler::GetBackend中可以看到，现在还没有考虑负载，网络IO状况等因素进行调度。但目前Impala已经有对执行过程的性能统计分析，应该以后版本会利用这些统计信息进行调度吧。

5.容错：

Hive: 依赖于Hadoop的容错能力。

Impala: 在查询过程中，没有容错逻辑，如果在执行过程中发生故障，则直接返回错误（这与Impala的设计有关，因为Impala定位于实时查询，一次查询失败，再查一次就好了，再查一次的成本很低）。但从整体来看，Impala是能很好的容错，所有的Impalad是对等的结构，用户可以向任何一个Impalad提交查询，如果一个Impalad失效，其上正在运行的所有Query都将失败，但用户可以重新提交查询由其它Impalad代替执行，不会影响服务。对于State Store目前只有一个，但当State Store失效，也不会影响服务，每个Impalad都缓存了State Store的信息，只是不能再更新集群状态，有可能会把执行任务分配给已经失效的Impalad执行，导致本次Query失败。

6.适用面：

Hive: 复杂的批处理查询任务，数据转换任务。

Impala：实时数据分析，因为不支持UDF，能处理的问题域有一定的限制，与Hive配合使用,对Hive的结果数据集进行实时分析。

4、Impala相对于Hive用到了什么优化技术?

1. impala没有使用MapReduce进行并行计算，虽然MapReduce是非常好的并行计算框架，但它更多的面向批处理模式，而不是面向交互式的SQL执行。与MapReduce相比：Impala把整个查询分成一执行计划树，而不是一连串的MapReduce任务，在分发执行计划后，Impala使用拉式获取数据的方式获取结果，把结果数据组成按执行树流式传递汇集，减少了把中间结果写入磁盘的步骤，再从磁盘读取数据的开销。Impala使用服务的方式避免每次执行查询都需要启动的开销，即相比Hive没了MapReduce启动时间。

2. 使用LLVM产生运行代码，针对特定查询生成特定代码，同时使用Inline的方式减少函数调用的开销，加快执行效率。

3. 充分利用可用的硬件指令。

4. 更好的IO调度，Impala知道数据块所在的磁盘位置能够更好的利用多磁盘的优势，同时Impala支持直接数据块读取和本地代码计算checksum。
5. 通过选择合适的数据存储格式可以得到最好的性能（Impala支持多种存储格式）。
6. 最大使用内存，中间结果不写磁盘，及时通过网络以stream的方式传递。

5、impala优缺点?

1. 优点：
    a. 支持SQL查询，快速查询大数据。
    b. 可以对已有数据进行查询，减少数据的加载，转换。
    c. 多种存储格式可以选择（Parquet, Text, Avro, RCFile, SequeenceFile）。
    d. 可以与Hive配合使用。
2. 缺点：
    a. 不支持用户定义函数UDF。
    b. 不支持text域的全文搜索。
    c. 不支持Transforms。
    d. 不支持查询期的容错。
    e. 对内存要求高。

6、Impala如何通过Hive外部表方式和HBase进行整合?

•步骤1：创建hbase 表，向表中添加数据

```
create 'test_info', 'info'
put 'test_info','1','info:name','zhangsan'
put 'test_info','2','info:name','lisi'
```

•步骤2：创建hive表

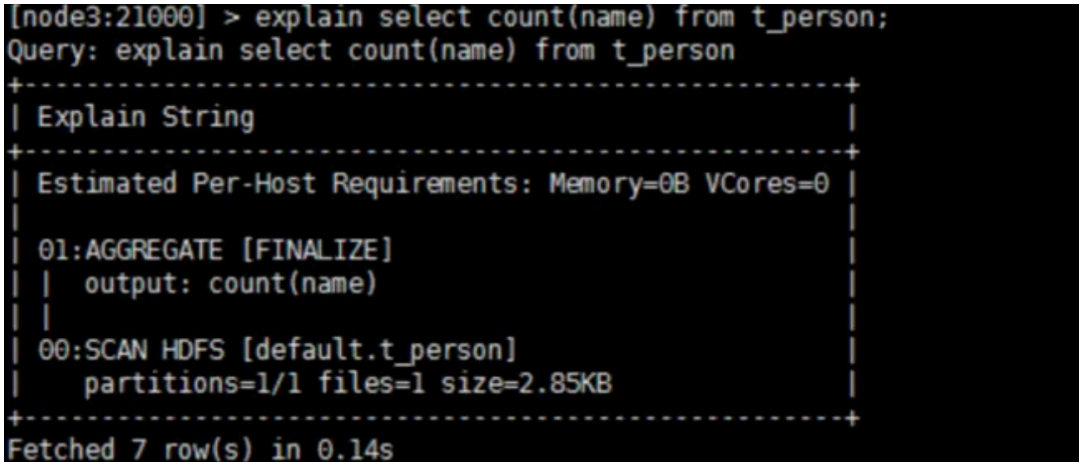
```
CREATE EXTERNAL TABLE test_info(key string,name string )
ROW FORMAT SERDE 'org.apache.hadoop.hive.hbase.HBaseSerDe'
STORED by 'org.apache.hadoop.hive.hbase.HBaseStorageHandler'
WITH SERDEPROPERTIES
("hbase.columns.mapping"=":key,info:name")
TBLPROPERTIES ("hbase.table.name" = "test_info");
```

•步骤3：刷新Impala表

```
invalidate metadata
```

7、impala性能优化?

执行计划：查询sql执行之前，先对该sql做一个分析，列出需要完成这一项查询的详细方案。命令：explain sql、profile



1、SQL优化，使用之前调用**执行计划**

2、选择合适的文件格式进行存储

3、避免产生很多小文件（如果有其他程序产生的小文件，可以使用中间表，将小文件存放到中间表然后写入到要执行的表里A—>V—>B）

4、使用合适的分区技术，根据分区粒度测算

5、使用compute stats进行表信息搜集

6、网络io的优化：

–a.避免把整个数据发送到客户端

–b.尽可能的做条件过滤

–c.使用limit字句

–d.输出文件时，避免使用美化输出

7、使用profile输出底层信息计划，在做相应环境优化