Hash Join功能设计文档

修订历史

版本	修订日期	修订描述	作者	备注
0.3	2017-09-29	Hash Join功能设计文档	茅潇潇	无

1 需求分析

Cedar是可扩展的关系数据库，实现了巨大数据量上的跨行跨表事务。在线业务中存在大量的多表连接查询，在处理多表连接时，Cedar支持Merge Join、BloomFilter Join、Semi Join运算，特点是Merge Server（MS）分发请求给相应的Chunk Server（CS），CS把过滤后的数据返回给MS，MS排序后做 Merge操作。但是，如果对两张或两张以上的大表进行连接，MS需要对大量的数据进行排序和比较，并且由于上述算子的连接操作都基于排序归并导致需要缓存每张表的数据和中间结果表的数据，对CPU计算资源和内存资源消耗较大。为解决上述问题，Cedar 0.3版本增加了Hash Join的支持。Hash Join首先对前表构建Hash Table，然后流水线处理后表每行数据做连接，一方面对于选择率较高的查询减少了大量的数据对比次数，明显降低了连接计算的时间，另一方面不再需要对后表进行内存中的缓存和排序，减少了时间和空间资源的消耗。生产环境中检验证明Hash Join的应用显著提高了多表连接查询的性能。

2 原理介绍

Hash Join使用关系S利用公共属性B上在内存中建立哈希表，然后扫描R表并探测哈希表，找出相匹配的行。假设，节点1上的关系R和节点2上的关系S，在属性R.A=S.B上做连接操作，它的工作流程如下：首先，节点2对关系S的每个元组在属性B上的值使用特定的哈希函数进行计算，并在内存里产生一张哈希表；然后，节点1把关系R中的记录传送给节点2，在节点2上对关系R的每个元组在属性A上的值使用相同的哈希函数进行计算，将计算的结果在哈希表中进行探测；如果探测成功，则一条新的纪录将被创建。

3 功能简述

Hash Join相当于把数据进行了分片，对两表中相同哈希值的数据进行对比，减少了大量的数据对比次数，并且充分利用了流水线技术的优势，通过对右表数据的流水线处理，降低了数据操作带来的时间和空间资源的消耗，显著提高了多表连接操作的处理性能。

4 设计思路

4.1 子功能模块划分

Hash Join分为词法语法、逻辑计划、物理计划和物理操作符四个子功能模块。

1. 词法语法解析层
   增加Select中的Hint的解析，不涉及其他部分的词法语法解析，影响范围是Select Hint的语法解析部分。
2. 逻辑计划层
   修改Select中的Hint的逻辑计划生成部分的代码，未动其他部分的逻辑计划，所以影响范围限于Select Hint的逻辑计划生成部分。
3. 物理计划层
   修改Hash Join的物理计划生成部分的代码，没有动其他部分的物理计划生成，影响范围限于Hash Join的物理计划生成部分。
4. 物理操作符
   增加ObHashJoinSingle操作符的代码，并未动其他物理操作符，影响范围限于ObHashJoinSingle操作符。

4.2 总体设计思路

Hash Join实际分为四个阶段：

1. 将左表的所有数据发送到一台MergerServer上，其中既包括ChunkServer存储的基线数据又包括UpdateServer存储的增量数据；
2. 在MergerServer上根据左表的数据在连接属性上的值使用特定的哈希函数进行计算，在内存中生成哈希表；
3. 右表所在的ChunkServer通过合并UpdateServer上的增量数据获得右表的全部数据后，将右表的所有数据发送到MergerServer；
4. MergerServer获取右表的一行数据，使用相同的哈希函数在哈希表中进行探测，如果探测成功，则一条新的纪录将被创建并返回给客户端。