# 实用 | 如何使用 Hadoop 提升 Hive 查询性能

2017-03-10 Cloudera中国



本文来源：开源中国
译者：leoxu
原文地址：https://dzone.com/articles/how-to-improve-hive-query-performance-with-hadoop
链接：https://www.oschina.net/translate/how-to-improve-hive-query-performance-with-hadoop?utm_source=tuicool&utm_medium=referral

Apache Hive 是一个 Hadoop 之上构建起来的数据仓库，用于数据的分析、汇总以及查询。Hive 提供了一种类 SQL 的接口来查询被存储在各种数据源和文件系统中的数据。

## 使用 Tez Engine

Apache Tez Engine 是一种用来构建高性能批处理与交互式数据处理的可扩展框架。在 Hadoop 中它借助 YARN 实现协作。Tez 通过提高处理速度来对 MapReduce 样例进行提升，并且保持着 MapReduce 向 PB 量级数据的扩展能了。

你可以通过在环境中将 hive.execution.engine 设置为 tez 来启用 Tez 引擎:

```
set hive.execution.engine=tez;
```

## 利用矢量化（ Vectorization ）

矢量化（Vectorization） 通过在一次操作中提取t 1024 行数据提升性能，而不是一次只取一条。它提升了像过滤, 联合, 聚合等等操作的性能。

Vectorization 可以通过在环境中执行如下命令而得到启用。

```
set hive.vectorized.execution.enabled=true;
set hive.vectorized.execution.reduce.enabled=true;
```

## 使用 ORCFile

优化的行列格式（Optimized Row Columnar）提供了通过借助比较原来节省 75% 数据存储的格式来存储hive数据的高效方法。当要对数据进行读、写以及处理操作时，ORCFile 格式较 Hive 文件格式更优。它使用了像谓词下推、压缩以及更多其它的技术来提升查询的性能。

考虑有这样两个表: 雇员（employee）以及雇员详情（employee_details），这两个表被存储在一个文件文件中。假如说我们要使用联合来从两个表取出详情数据。

```
Select a.EmployeeID, a.EmployeeName, b.Address,b.Designation from Employee a
Join Employee_Details b
On a.EmployeeID=b.EmployeeID;
```

上面的查询操作会花掉较长的时间, 因为数据是以文本形式存储的。将该表转换成 ORCFile 格式将会显著减少查询的执行时间。

```
Create Table Employee_ORC (EmployeeID int, EmployeeName varchar(100),Age int)
STORED AS ORC tblproperties("compress.mode"="SNAPPY");

Select * from Employee Insert into Employee_ORC;

Create Table Employee_Details_ORC (EmployeeID int, Address varchar(100)
                                  ,Designation Varchar(100),Salary int)
STORED AS ORC tblproperties("compress.mode"="SNAPPY");

Select * from Employee_Details Insert into Employee_Details_ORC;
Select a.EmployeeID, a.EmployeeName, b.Address,b.Designation from Employee_ORC a
Join Employee_Details_ORC b
On a.EmployeeID=b.EmployeeID;
```

ORC 支持压缩 (ZLIB 和 Snappy), 还有解压缩的存储。

## 利用分区

有了分区, 数据就可以被存储在 HDFS 上的多个文件夹下。查询时不回去查询整个数据集，而是查询分区的数据集。

## 创建临时表并将数据导入临时表

```
Create Table Employee_Temp(EmployeeID int, EmployeeName Varchar(100),
                           Address Varchar(100),State Varchar(100),
                           City Varchar(100),Zipcode Varchar(100))
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;

LOAD DATA INPATH '/home/hadoop/hive' INTO TABLE Employee_Temp;
```

## 创建分区表

```
Create Table Employee_Part(EmployeeID int, EmployeeName Varchar(100),
                           Address Varchar(100),State Varchar(100),
                           Zipcode Varchar(100))
PARTITIONED BY (City Varchar(100))
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

## 创建动态的Hive分区

```
SET hive.exec.dynamic.partition = true;
SET hive.exec.dynamic.partition.mode = nonstrict;
```

## 从临时表向分区表导入数据

```
Insert Overwrite table Employee_Part Partition(City) Select EmployeeID,
EmployeeName,Address,State,City,Zipcode from Emloyee_Temp;
```

## 利用桶装数据

Hive 表被分割成许多分区而被叫做 Hive 分区。Hive 分区可以被进一步细分成卷或者桶，而被称作是数据卷或者桶装数据。

```
Create Table Employee_Part(EmloyeeID int, EmployeeName Varchar(100),
                        Address Varchar(100),State Varchar(100),
                        Zipcode Varchar(100))
PARTITIONED BY (City Varchar(100))
Clustered By (EmployeeID) into 20 Buckets
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE;
```

## 基于成本的查询优化

Hive 在向最终的执行提交之前会对每一个查询逻辑和物理执行计划进行优化。不过，这样的优化并非基于初始版本 Hive 中的查询操作的成本来进行的。
在 Hive 的后续版本中，查询已经根据查询操作的成本（显示会执行哪种类型的联合，如何对联合操作进行排序，并行的程度等等）进行了优化。
为了利用到基于成本的优化，在查询的开始部分要设置好如下一些参数。

```
set hive.cbo.enable=true;
set hive.compute.query.using.stats=true;
set hive.stats.fetch.column.stats=true;
set hive.stats.fetch.partition.stats=true;
```

## 总结

Apache Hive 是一种非常强大的数据分析工具，而它也支持批量和可交互式的数据处理操作。它是数据分析师和数据科学家最常被用到的工具。当你在处理 PB 量级的数据时，知道如何提升查询操作的性能是非常重要的。现在你就知道了如何去提升 Hive 查询操作的性能！

◆◆◆

请点击 **“阅读全文”** 进入微站