Extrapolate the Column Status for partitions

Pengcheng Xiong

This document discusses the implementation of the extrapolation for the column status.

**1. Assumption:**

We discuss aggregation value of the column status for PARTITIONED tables only. Non-partitioned tables are not discussed.

We extrapolate for the aggregation value of the column status for PARTITIONED tables only. We do not extrapolate the column status for every single partition.

**2. Problem specification**

In a PARTITIONED table, there are many partitions. For example,

*create table if not exists loc\_orc (*

*state string,*

*locid int,*

*zip bigint*

*) partitioned by(year string) stored as orc;*

We assume there are 4 partitions, partition(year='2000'), partition(year='2001'), partition(year='2002') and partition(year='2003').

We can use the following command to compute statistics for columns state,locid of partition(year='2001')

*analyze table loc\_orc partition(year='2001') compute statistics for columns state,locid;*

We need to know the “aggregated” column status for the whole table loc\_orc. However, we may not have the column status for some partitions, e.g., partition(year='2002') and also we may not have the column status for some columns, e.g., *zip bigint* for *partition(year='2001')*

We propose a method to extrapolate the missing column status for the partitions.

**3. Current solution**

*"select \"COLUMN\_NAME\", \"COLUMN\_TYPE\", "*

*+ "min(\"LONG\_LOW\_VALUE\"), max(\"LONG\_HIGH\_VALUE\"), min(\"DOUBLE\_LOW\_VALUE\"), max(\"DOUBLE\_HIGH\_VALUE\"), "*

*+ "min(\"BIG\_DECIMAL\_LOW\_VALUE\"), max(\"BIG\_DECIMAL\_HIGH\_VALUE\"), sum(\"NUM\_NULLS\"), max(\"NUM\_DISTINCTS\"), "*

*+ "max(\"AVG\_COL\_LEN\"), max(\"MAX\_COL\_LEN\"), sum(\"NUM\_TRUES\"), sum(\"NUM\_FALSES\") from \"PART\_COL\_STATS\""*

*+ " where \"DB\_NAME\" = ? and \"TABLE\_NAME\" = ? and \"COLUMN\_NAME\" in ("*

*+ makeParams(colNames.size()) + ") AND \"PARTITION\_NAME\" in ("*

*+ makeParams(partNames.size()) + ") group by \"COLUMN\_NAME\", \"COLUMN\_TYPE\"";*

As we can see, current solution just runs over the columns and partitions that have column status.

**4. Proposed solution**

The solution works like this

**If (the status all the columns and all the partitions exists)**

**Run the existing solution**

**Else**

**For Column column: columns,**

**If(all the partitions have status data for this column)**

**Run the existing solution**

**Else**

**Run extrapolation**

Extrapolation works like this

Case 1: the aggregation function is min/max

Assume that, partition 2000,2001,2002 has NUM\_DISTINCTS as 10,11,20, we do not know the value for 2003.

In the current solution, we will use max(10,11,20)=20 as the aggregated value.

In our proposed solution,

We take the min(10,11,20) = 10

We take the max(10,11,20) = 20

So, we draw a line (2000,10) and (2002,20)

Thus, the point at 2003 (this is the right border) should be (2003,25). And the point at 2000 is already (2000,10) (this is the left border)

Thus, the value should be max(left border,rightborder)=max(10,25)=25 now.

Case 2: the aggregation is sum

In the current solution, we will use sum(10,5,3)=18 as the aggregated value.

In our proposed solution,

We take the sum(10,5,3)=18

We take the sum(1,1,1) = 3 as the number of partitions that have data

We then scale up by 4(all partitions)/3(the partitions that have status)

Thus, the value should be 18\*4/3=24 now.

Our assumptions

(1) we ignore \_\_HIVE\_DEFAULT\_PARTITION\_\_

(2) we assume that the partitions are uniformly generated.