**Tuning HBase Bulkload and Phoenix Index Query**

**Author: Michelle Zhang**

**Company: Cloudera**

**Date: 10/25/2014**

# Environment

Cluster: 3 nodes

DataNode, Regionserver, HMaster: 3 nodes

Memory: 32GB

CPU: 8\*core Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz

# Tuning HBase Bullkload

## Check Network Speed

Use SCP or other command to check each node’s network transmission speed is reasonable and match the network devices.

## Set Appropriate SALT\_BUCKETS Number

Set Appropriate SALT\_BUCKETS number according to Resgionserver when create table in phoenix, better similar to core number.

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql CREATE TABLE "nci20gb" (myPK VARCHAR PRIMARY KEY, "f"."q1" VARCHAR, "f"."q2" VARCHAR, "f"."q3" VARCHAR, "f"."q4" VARCHAR, "f"."q5" VARCHAR) SALT\_BUCKETS = 24;

# Tuning HBase for Phoenix Index Builder

## Enlarge HBase RegionServer Java Heap Size

Stop other not related service,s enlarge HBase RegionServer java heap size, for example, set to 80% Memory to reduce GC operations.

## Optimize HDFS configuration

Separate datanode dir and namenode dir on different disks, you can use CM OS system disk as namenode dir disk, and not use CM OS system disk as hdfs data dir disk.

## Enlarge Memstores to Reduce New Update Blocking

hbase.regionserver.global.menstore.upperLimit=0.6

hbase.regionserver.global.menstore.lowerLimit=0.6

## Reduce Hfile Block Cache

hfile.block.cache.size=0.1

## Stop HBase Split

hbase.regionserver.regionSplitLimit=1

## Enlarge HStoreFile Size

hbase.hregion.max.filesize=1024GB

## Enlarge Multiple Threads Concurrent Flush

hbase.hstore.blockingStoreFiles=2000

hbase.hstore.blockingWaitTime=150min

Add fowlloing properties in “COnfiguration” -->”RegionServer Default Group” -->”Advance”--> Regionserver hbase-site.xml window.

<property>

<name>hbase.regionserver.flusher.count</name>

<value>20</value>

</property>

<property>

<name>hbase.hstore.flush.thread</name>

<value>20</value>

</property>

## Enlarge Block Updates Memstores Flush Size

hbase.hregion.memstore.flush.size =512MB

hbase.hregion.memstore.block.multiplier=2000

## Enlarge RPC Instances

hbase.regionserver.handler.count =3000

## Disable HBase Major Compaction

hbase.hregion.majorcompaction=0

hbase.hstore.compaction.max=2000

Then, restart HBase service to enable the new configuration, which is optimized for HBase heavy WRITE.

# Tuning HBase for Phoenix Index Query

After write, run major compaction, read at once, ensure locality, no hbase restart, no balance.

## Manually Run Major Compaction and Flush

Manually run flush memstore and major compaction in hbase shell.

hbase> flush ‘nci20gb’

hbase> major\_compact ‘nci20gb’

## Reduce Memstores

hbase.regionserver.global.menstore.upperLimit=0.1

hbase.regionserver.global.menstore.lowerLimit=0.1

## Enlarge Hfile Block Cache For Heavy Read

hfile.block.cache.size=0.7

(hfile.block.cache.size + hbase.regionserver.global.menstore.upperLimit <= 0.8)

## Set Opposite Values of Above Heavy Write Properties

## Set opposite values of above HBase heavy write properties

# Tuning Phoenix for Phoenix Index Query

## Enlarge Max Global Memory Percentage

phoenix.query.maxGlobalMemoryPercentage=95

## Enlarge Max Global Memory Size

Specify “phoenix.query.maxGlobalMemorySize” larger

## Enlarge Target Concurrency Threads Number

phoenix.query.targetConcurrency=128

## Enlarge Max Concurrency Threads Number

phoenix.query.maxConcurrency=256

## Enlarge Groupby Max Cahce Size

phoenix.groupby.maxCacheSize=24GB

## Enlarge Client Scanner Cache

Specify “hbase.client.scanner.caching” larger.

hbase.client.scanner.caching=100000

# Tuning on 120 million 20GB data

Use TPCH to generate 20GB customer.tbl table.

Data size: 20GB

Record: 120000000(120 million)

Record fields’ number: 9

## Bulkload

|  |  |  |
| --- | --- | --- |
| **Region Number** | **Before Tuning** | **After Tuning** |
| 6 | 31min 26s | 16min 12s |
| 24 |  | 15min 58s |
| 120 |  | 12min 9s |

## Count Query Based on Rowkey

|  |  |
| --- | --- |
| **Region Number** | **Response Time** |
| 6 | 210.5s |
| 24 | 178.4s |

## Query Based on Rowkey

|  |  |  |
| --- | --- | --- |
| **Region Number** | **Before Tuning** | **After Tuning** |
| 6 | 0.091s |  |

## Phoenix Indexer

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Region Number** | **Before Tuning** | **After Tuning** |
| Create Global Index | 6 | 2570.6s |  |
| Rebuild Global Index | 6 |  |  |
| Create Local Index | 6 | 2397.6s | 1032.3s |
| Rebuild Local Index | 6 |  |  |

## Phoenix Query

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Region Number** | **Before Tuning** | **After Tuning** |
| Query Global Index | 6 | 0.098s |  |
| Query Local Index | 6 | 0.081s |  |

## HFile Size

Hfile size of tables unber /hbase/data/default/ in HDFS:

|  |  |
| --- | --- |
|  | **Hfile Size** |
| Original Table | 22.4GB |
| Create Global Index Table | 6.2GB |
| Create Local Index Table | 5.3GB |

## Deferent Column Qualifier Data Type

Modify Bulkload code can support different data type for phoenix load, index and query (2GB data):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data Type** | **Bulk Load Time** | **Count Time** | **Index Time** | **Query Time** |
| VARCHAR | 2min 37s | 1.283s | 139.876s | 0.055s |
| UNSIGNED\_LONG | 2min 30s | 1.511s | 139.153s | 0.046s |
| UNSIGNED\_INT |  | 1.388s | 136.428 |  |