**HBase Bulkload and Phoenix Index Query**

**Author: Michelle Zhang**

**Company: Cloudera**

**Date: 10/24/2014**

# Install BigDataAnalysis Bulkload in Cluster

## Download Install BigDataAnalysis Bulkload

Copy BigDataAnalysis.tar.gz to one of your cluster nodes(choose CM as simple). More details of BigDataAnalysis Bulkload user guide please refer to

$BIGDATAANALYSIS\_HOME/doc/bulkload/GET\_START\_BL.txt.

Copy phoenix-4.1.0-client-hadoop2.jar into $BIGDATAANALYSIS\_HOME/lib/cdh5 dir.

Take CDH 5.2 cluster as an example.

# Download and Install Phoenix in Cluster

## Download and Install Phoenix

Download phoenix 4.1 tarball(phoenix-4.1.0-bin.tar.gz) from following URL:

<http://mirrors.advancedhosters.com/apache/phoenix/phoenix-4.1.0/bin/>

Unpack it in $PHOENIX\_HOME.

## Configurate HBase and Phoniex

cd into $PHOENIX\_HOME/phoenix-4.1.0-bin/hadoop2/

Copy the phoenix-4.1.0-server-hadoop2.jar into EVERY region server’s /opt/cloudera/parcels/CDH/lib/hbase/lib/

Configure HBase RegionServer hbase-site.xml with following properties:

<property>

  <name>hbase.regionserver.wal.codec</name>

  <value>org.apache.hadoop.hbase.regionserver.wal.IndexedWALEditCodec</value>

</property>

For local indexing also requires special configurations in the master to ensure data table and local index regions co-location.

You will need to add the following parameters to hbase-site.xml on the master:

<property>

<name>hbase.master.loadbalancer.class</name> <value>org.apache.phoenix.hbase.index.balancer.IndexLoadBalancer</value>

</property>

<property>

<name>hbase.coprocessor.master.classes</name> <value>org.apache.phoenix.hbase.index.master.IndexMasterObserver</value>

</property>

Configure hbase-site.xml with following properties in you Phoenix client:

<property>

<name>phoenix.query.timeoutMs</name>

<value>36000000</value>

</property>

## Start Phoenix Service

Restart HBase to enable Phoenix service. Use following shell to start Phoenix Shell:

$ sqlline.py <zookeeper.quorum>

$ sqlline.py localhost ../examples/stock\_symbol.sql

[root@ip-172-31-12-149 bin]# ./sqlline.py 172.31.12.149,172.31.12.150,172.31.12.151

Setting property: [isolation, TRANSACTION\_READ\_COMMITTED]

issuing: !connect jdbc:phoenix:172.31.12.149,172.31.12.150,172.31.12.151 none none org.apache.phoenix.jdbc.PhoenixDriver

Connecting to jdbc:phoenix:172.31.12.149,172.31.12.150,172.31.12.151

14/10/24 09:26:26 WARN impl.MetricsConfig: Cannot locate configuration: tried hadoop-metrics2-phoenix.properties,hadoop-metrics2.properties

Connected to: Phoenix (version 4.1)

Driver: org.apache.phoenix.jdbc.PhoenixDriver (version 4.1)

Autocommit status: true

Transaction isolation: TRANSACTION\_READ\_COMMITTED

Building list of tables and columns for tab-completion (set fastconnect to true to skip)...

74/74 (100%) Done

Done

sqlline version 1.1.2

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1>

Then, your phoenix is installed successfully. You can run SQL queries in Phoenix Shell with “!sql” prefixed.

# Bulkload in HBase

## Prepare Data

Generate 20GB(120000000 records) with TPCH customer.tbl.

## Create Table in Phoenix

Create table in phoenix with setting salt bucket number as following.

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 = 6;

check whether the table is created successfully or not.

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !table

+------------+-------------+------------+------------+------------+------------+

| TABLE\_CAT | TABLE\_SCHEM | TABLE\_NAME | TABLE\_TYPE | REMARKS | TYPE\_NAME |

+------------+-------------+------------+------------+------------+------------+

| null | SYSTEM | CATALOG | SYSTEM TABLE | null | null |

| null | SYSTEM | SEQUENCE | SYSTEM TABLE | null | null |

| null | null | nci20gb | TABLE | null | null |

| null | null | test | TABLE | null | null |

+------------+-------------+------------+------------+------------+------------+

### HBase

hbase(main):006:0> list

TABLE

SYSTEM.CATALOG

SYSTEM.SEQUENCE

nci20gb

test

4 row(s) in 0.0150 seconds

=> ["SYSTEM.CATALOG", "SYSTEM.SEQUENCE", "nci20gb", "test"]

## Bulkload Data into HBase

Use BigDataAnalysis Bulkload to bulk load data into HBase and set “preCreateRegions” and “buildIndex” to “false” in your properties. More details of BigDataAnalysis Bulkload user guide please refer to

$BIGDATAANALYSIS\_HOME/doc/bulkload/GET\_START\_BL.txt.

Make sure your “hbase.target.table.name” is the same with the table name you created in Phoenix before in 3.2), it’s case sensitive.

# Phoenix Query Based on Rowkey

## Count Records

Count Bulkload records:

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql SELECT COUNT("f"."q1") FROM "nci20gb";

+-------------+

| COUNT(f.q1) |

+-------------+

| 120000000 |

+-------------+

1 row selected (210.595 seconds)

## Explain Query Based on Rowkey

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql EXPLAIN SELECT "f"."q1" FROM "nci20gb" WHERE MYPK > '1234' LIMIT 5;

+------------+

| PLAN |

+------------+

| CLIENT PARALLEL 10-WAY SKIP SCAN ON 6 RANGES OVER nci20gb [0,'1234'] - [5,\*] |

| SERVER FILTER BY PageFilter 5 |

| SERVER 5 ROW LIMIT |

| CLIENT MERGE SORT |

| CLIENT 5 ROW LIMIT |

+------------+

5 rows selected (0.027 seconds)

## Execute Query Based on Rowkey

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql SELECT "f"."q1" FROM "nci20gb" WHERE MYPK > '1234' LIMIT 5;

+------------+

| q1 |

+------------+

| rXJe7HuU9JV |

| 1XqVHt36TxTROYhHd |

| JErhf4bCPF0s9ruPWkbGZEBBGXJmMBPq,8 |

| J9 NlsTNOhjy1STXslPUA0wIugLR |

| gvPSWiqVT, |

+------------+

5 rows selected (0.091 seconds)

# Phoenix Global Indexer and Queries Based on Non-Rowkey

## Create Global Index Table in Phoenix

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql CREATE INDEX NCI20GB\_Q4 ON "nci20gb" ("f"."q4");

120,000,000 rows affected (2691.383 seconds)

## Explain Global Query

Run explain sql to check whether query is using global index or not.

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql EXPLAIN SELECT "f"."q4" FROM "nci20gb" WHERE "f"."q4" > '1234' LIMIT 5;

+------------+

| PLAN |

+------------+

| CLIENT PARALLEL 30-WAY SKIP SCAN ON 6 RANGES OVER NCI20GB\_Q4 [0,'1234'] - [5,\*] |

| SERVER FILTER BY PageFilter 5 |

| SERVER 5 ROW LIMIT |

| CLIENT MERGE SORT |

| CLIENT 5 ROW LIMIT |

+------------+

5 rows selected (0.036 seconds)

## Query Based on Global Index

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql SELECT "f"."q4" FROM "nci20gb" WHERE "f"."q4" > '1234' LIMIT 5;

+------------+

| q4 |

+------------+

| 1234.00 |

| 1234.00 |

| 1234.00 |

| 1234.00 |

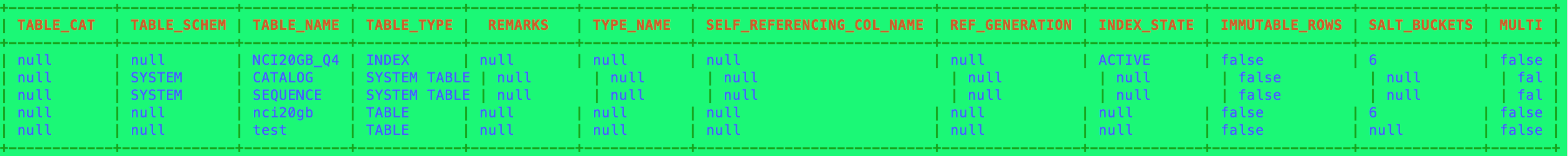
| 1234.00 |

+------------+

5 rows selected (0.163 seconds)

## Show Tables Results

### Phoenix



### HBase

hbase(main):006:0> list

TABLE

NCI20GB\_Q4

SYSTEM.CATALOG

SYSTEM.SEQUENCE

nci20gb

test

5 row(s) in 0.0150 seconds

=> ["NCI20GB\_Q4", "SYSTEM.CATALOG", "SYSTEM.SEQUENCE", "nci20gb", "test"]

## Rebuild Global Index

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql ALTER INDEX IF EXISTS NCI20GB\_Q4 ON "nci20gb" REBUILD;

# Phoenix Local Indexer and Queries Based on Non-Rowkey

## Create Local Index Table in Phoenix

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql CREATE LOCAL INDEX NCI20GB\_LOCAL\_Q5\_N ON "nci20gb" ("f"."q5");

120,000,000 rows affected (1036.288 seconds)

## Explain Local Query

Run explain sql to check whether query is using global index or not.

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql EXPLAIN SELECT "f"."q5" FROM "nci20gb" WHERE "f"."q5" > '1234' LIMIT 5;

+------------+

| PLAN |

+------------+

| CLIENT PARALLEL 6-WAY RANGE SCAN OVER \_LOCAL\_IDX\_nci20gb [-32762,'1234'] - [-32762,\*] |

| SERVER FILTER BY PageFilter 5 |

| SERVER 5 ROW LIMIT |

| CLIENT MERGE SORT |

| CLIENT 5 ROW LIMIT |

+------------+

5 rows selected (0.032 seconds)

## Query Based on Local Index

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql SELECT "f"."q5" FROM "nci20gb" WHERE "f"."q5" > '1234' LIMIT 5;

+------------+

| q5 |

+------------+

| AUTOMOBILE|! Tiresias cajole sometimes. carefully close dolphins ought to ar| |

| AUTOMOBILE|! Tiresias haggle blithely. bold asymptotes ought to sleep bravely finally ironic platelets. blithely ironic pint| |

| AUTOMOBILE|! Tiresias nag even, final requests. evenly regular theodolites boost slyly after the fluffy ideas. final| |

| AUTOMOBILE|! accounts about the blithely ironic packages haggle carefully special packages. regular, ir| |

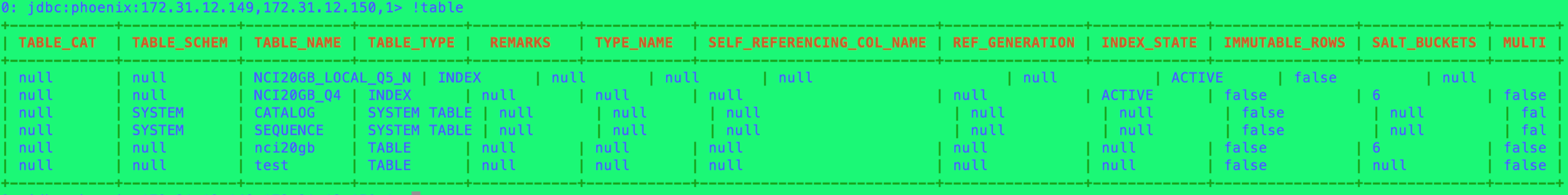
| AUTOMOBILE|! accounts about the boldly regular foxes boost fluffily pending instructions. even warhorses integrate.| |

+------------+

5 rows selected (0.11 seconds)

## Show Tables Results

### Phoenix



### HBase

hbase(main):001:0> list

TABLE

NCI20GB\_Q4

SYSTEM.CATALOG

SYSTEM.SEQUENCE

\_LOCAL\_IDX\_nci20gb

nci20gb

test

6 row(s) in 1.7580 seconds

=> ["NCI20GB\_Q4", "SYSTEM.CATALOG", "SYSTEM.SEQUENCE", "\_LOCAL\_IDX\_nci20gb", "nci20gb", "test"]

## Rebuild Local Index

0: jdbc:phoenix:172.31.12.149,172.31.12.150,1> !sql ALTER INDEX IF EXISTS NCI20GB\_LOCAL\_Q5\_N ON "nci20gb" REBUILD;

# Other Commands in Phoenix Shell

## Disable Index Table

!sql ALTER INDEX IF EXISTS NCI20GB\_LOCAL\_Q5\_N ON "nci20gb" DISABLE;

## Drop Index Table

!sql DROP INDEX NCI20GB\_LOCAL\_Q5\_N on "nci20gb";

## Drop Original Table

!sql DROP table "nci20gb";

## Show Table Description

!describe NCI20GB\_LOCAL\_Q5\_N;