Project (Optional) -- A Simple Big Data System Using a NoSQL system

CMPUT 391 Database Management Systems (Winter, 2014)

Term Project

Due Time: 18:00 March 31, 2014

1. Introduction

In this project, you are asked to develop a simple big data system for a wireless carrier to store the list all of **call detail record** (**CDR**) in a distributed database management system, and to conduct some interesting SQL queries on the distributed database system.

The collection of the CDRs is so large and complex that it is very difficult, if not impossible, to store and process using traditional object-relational database management systems, such as Oracle. One needs to use a NoSQL system, and a sample system to use is Cassandra, and another one is BigSQL. Cybera, a not-for-profit, technical agency that is helping Alberta advance its IT frontier, has kindly agreed to provide us 8 instances of servers, each with 4 CPU cores, 32G memory, and 4T disk space, for this course project.

2. System Specification

You are going to implement a simple OLAP system, for a wireless carrier, that will be used

- to store the list of all CDRs as an online back up system for their mission-critical system, and
- to evaluate the OLAP queries.

You must first choose a NoSQL system, such as Cassandra, BigSQL, or any other available systems for your project, and install it on the provided Cybera system. Of course, you may develop your own system using the Hadoop.

2.1. Table Creation

You shall first design a table schema or use one of the provided schemes (setup1.sql, or setup2.sql) to create a table named **cdr** to store the list of all CDRs. Note that the table must have at least 100 columns such that each row in the table represents detailed information for each phone call or text message.

Since the table has at lease 100 columns, and will be populated with a huge number of rows, you may have to partition the table to

- a list of node-partitions, i.e., to store the table into a list of distributed nodes; and
- a list of column-partitions, i.e., to store the table as several tables such that (1) all partitioned tables contain the same primary key, and (2) the list of all non-primary key columns of all partitioned tables form a partition of the non-primary-key columns of the given table.

You table must be able to store all the essential fields as listed per **call detail record**.

2.2. Table Population

Populate the table **cdr** based on the pattern as specified in **call detail record** (**CDR**) and the rules are outlined below:

- the list of CDRs for each day should be generated according to a specific pattern. (A sample pattern for setup1.sql is given in partial_data.txt).
- the populated table must contains all the CDRs of the last N days such that the total size of the table must be at least 16TB, in terms of disk space occupied; and
- all the values must be generated using a random number generator, and/or sequences.

Note that a sample data file, data.txt, which contains the list of all CDRs for one specific day, is around 180Mb and contact the instructor for the download details. Further, in the data file, a value 130115040011 of the date type

stands for '2013-01-15:04:00:11'.

2.3. Query Evaluation

According to your understanding of telecommunication transactions, specify five SQL queries to retrieve information from the populated table. You queries must satisfy the followings:

- four of the five queries will retrieve information from all distributed nodes;
- one query must contain at least 10 conditions (atomic formulas) in the where clause;
- two queries must contain both the group by and order by clauses;
- at lease three queries are range queries, i.e., a query retrieving all rows where some column value is between an upper and lower boundary; and
- none of the queries are non-trivial, i.e., a simple key-search.

Justify your choices of these five queries by specifying scenarios in which they are useful.

Evaluate each of the five queries twice and record the average performance of queries in terms of the elapse time (actual running time).

Two lists of queries, (query1.sql and query2.sql) are provided as a reference.

2.4. Partition Analysis

One of the challenges of the project is to arrange proper table partitions (be node-partitions or column-partitions) such that all the queries can be evaluated efficiently. Modifying the table partition in 2.2 and then repeat 2.2 and 2.3 such that at least one of the five queries in 2.3 can be evaluated much more efficiently. Analyze the performance improvement based on your experiments.

3. Requirements

- This is a group project, with two or three students in each group, and of course, all students in a group will receive the same mark for the project. Send an email to Lengdong Wu (lengdong@ualberta.ca) with the list of students in your group, together with your choice of the project (Big Data**)** before February 10. Note that we will take at most four groups for the big data project due to resource limitation, based on the "Early Bird Policy".
- You are required to submit
    - Project Report
      The report must describe
        - the NoSQL system used for the project, and the justification of your choice;
        - the experimental setup, including the configuration of the distributed system, the specification of the data set and queries;
        - result analysis, including the partitions used and the justification for the partitions, and any other measures that help you to improve the performance.
    - Source Code
        - the programs used to populate the data base, and to evaluate the queries.
    - User Documentation
      You must submit an electronic copy of the user documentation. The user documentation must contain
        - the installation guideline that tells the user how to install the NoSQL system,
        - the set up guideline that tells the user how to create and populate the database,
        - the query guideline that tells the user how to conduct the experimental tests.

  A popular tool for writing such online documents is Texi2html.
- Each group will be asked to demo the e*xperimental tests* in our lab, and of course, the demo will use a much smaller data set. Please make sure that the creation and population of the distributed database can be done smoothly and all queries will be evaluated correctly.

4. Important Dates

February 10	Group Registration
March 31, 18:00	Project Report, Source Code and User Documentation
April 1-9	Demo

No later submission will be accepted, except for health reasons.

5. Marking Guidelines

The project will be marked according the following schema.

Project Report	30%
User Documentation	15%
Implementation	55%

The mark for the project report, and user documentation will be based on your understanding of the application system, the description of your implementation strategies, clarity of your document, and of course, English writing.

You must follow proper programming guidelines learned in your previous courses. Poorly documented programs and ill program logic will lead to deduction of marks.

6. Instructions For Submission

- Create a single gzipped tar file with the PDF file of your project report, all your source code, user documentation, and additional files you may need for your demo using using tar -c file1, file2, ..., fileN | gzip -c > project.tgz.
- Submit project.tgz at the bottom of this page.

Submission status

Submission status	Nothing has been submitted for this assignment
Grading status	Not graded
Due date	Monday, 31 March 2014, 6:00 PM
Time remaining	61 days 19 hours

Add submission