Datastax Technical Screen

Shuhua (Anna) Liang

May 13, 2013

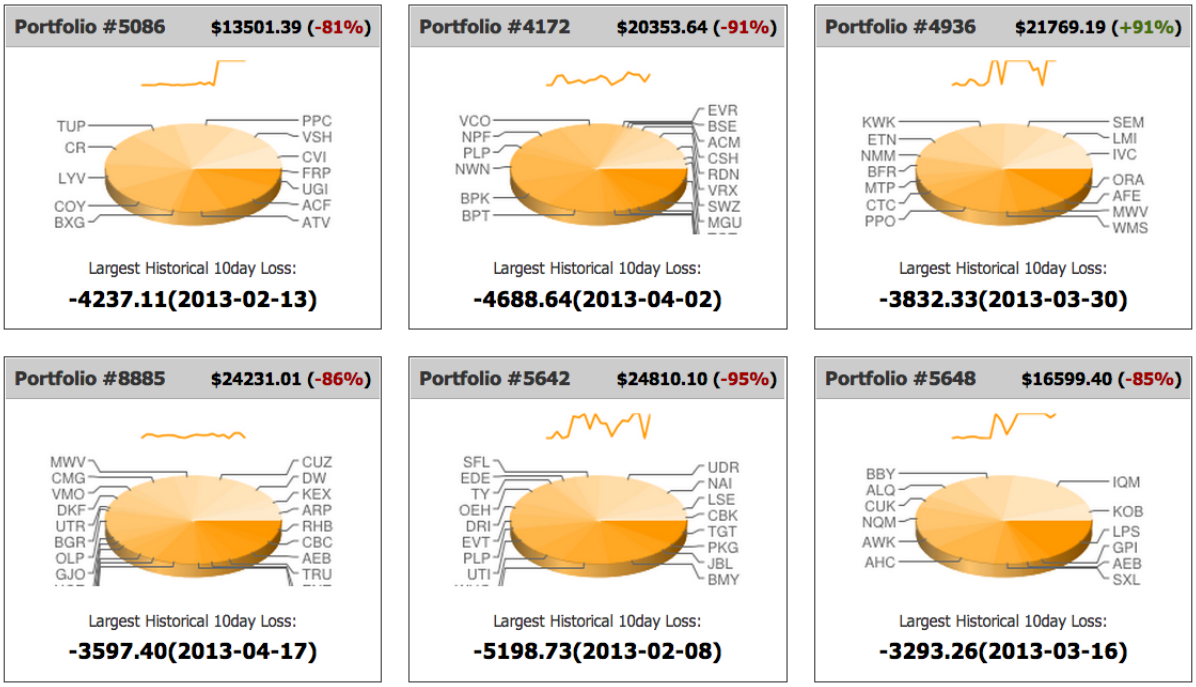
DataStax Enterprise

Gotchas

Having installed and explored the DataStax Enterprise software on my Mac unix system, I learned that it is dealing with the Cassandra and NoSQL-based competitive big data technology. Also, it allows us to easily update real time data and analyze the data in one integrated database solution.

From the demo, I found that there are four column families defined in the keyspace. The four column families include stock portfolio, stock historical end-of-day price, current value, and historical loss. These column families come in a static format of column names and column values. For instance, column family "Portfolio" has portfolio numbers as column names and current price as column values.

This example nicely demonstrated how DataStax Enterprise multitasks on stock analysis and price updates. Following is a screen shot of the results of Largest Historical 10-day Loss for each portfolio from the demo:



While running the MapReduced Job in Hive, I was able to track the job progress:

localhost Hadoop Map/Reduce Administration

State: RUNNING
Started: Thu May 09 19:59:27 PDT 2013
Version: 1.0.4.4, r91c4c2b47eac7ee4e4375864cc6262556858cc4b
Compiled: Tue Apr 16 08:01:52 PDT 2013 by jenkins
Identifier: 201305091959

Cluster Summary (Heap Size is 1004 MB/1004 MB)

Running Map Tasks	Running Reduce Tasks	Total Submissions	Nodes	Occupied Map Slots	Occupied Reduce Slots	Reserved Map Slots	Reserved Reduce Slots	Map Task Capacity	Reduce Task Capacity	Avg. Tasks/Node
0	0	5	1	0	0	0	0	2	2	4.00

Completed Jobs Quick Li

Jobid	Priority	User	Name	Map % Complete	Map Total	Maps Completed	Reduce % Complete	Reduce Total	Reduces Completed
job_201305091959_0001	NORMAL	shuhualiang	INSERT OVERWRITE TABLE 10da...b.column_name) (Stage-1)	100.00%	2	2	100.00%	1	1
job_201305091959_0002	NORMAL	shuhualiang	INSERT OVERWRITE TABLE portfolio_ret...rdate(Stage-1)	100.00%	3	3	100.00%	1	1
job_201305091959_0003	NORMAL	shuhualiang	INSERT OVERWRITE TABLE portfolio_ret...rdate(Stage-2)	100.00%	3	3	100.00%	1	1
job_201305091959_0004	NORMAL	shuhualiang	INSERT OVERWRITE TABLE HistLoss...b.preturn) (Stage-1)	100.00%	1	1	100.00%	1	1

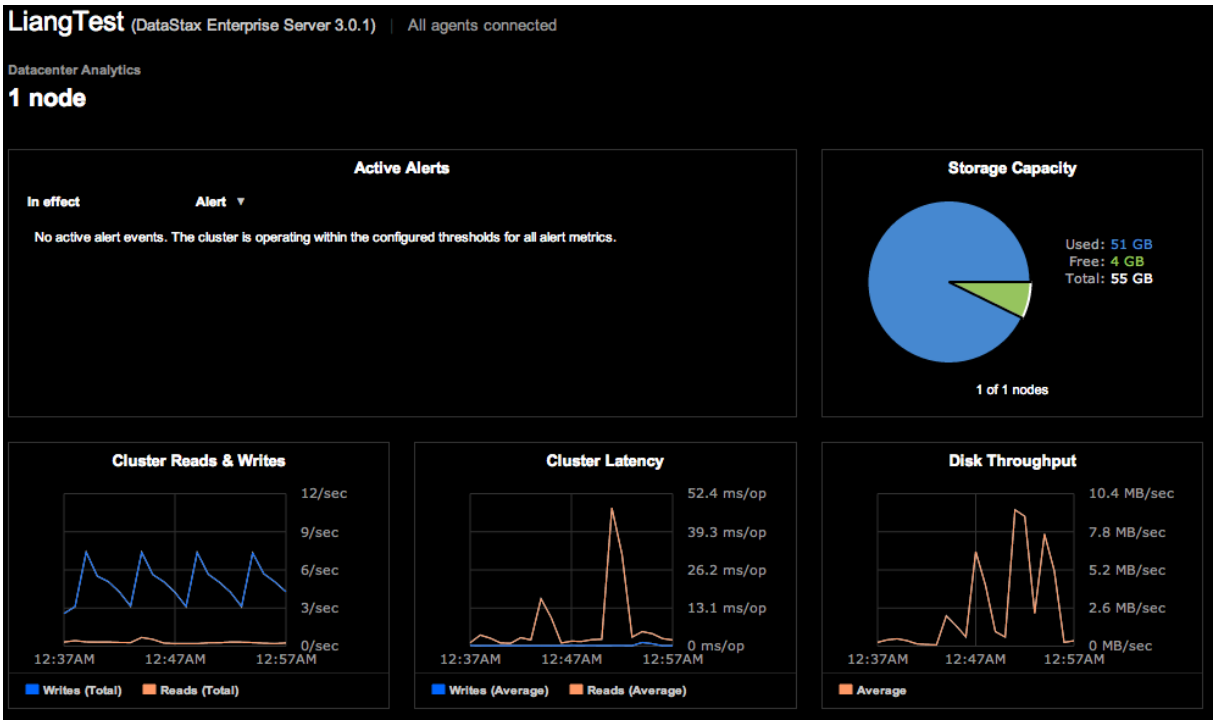
Suggestion

A little suggestion for the Portfolio Manager Demo would be the row names for column family Portfolio. Instead of providing users with the portfolio numbers, if might be helpful to show the stock ticker symbols. For example, it can display “APPL” for Apple, Inc.

Opscenter

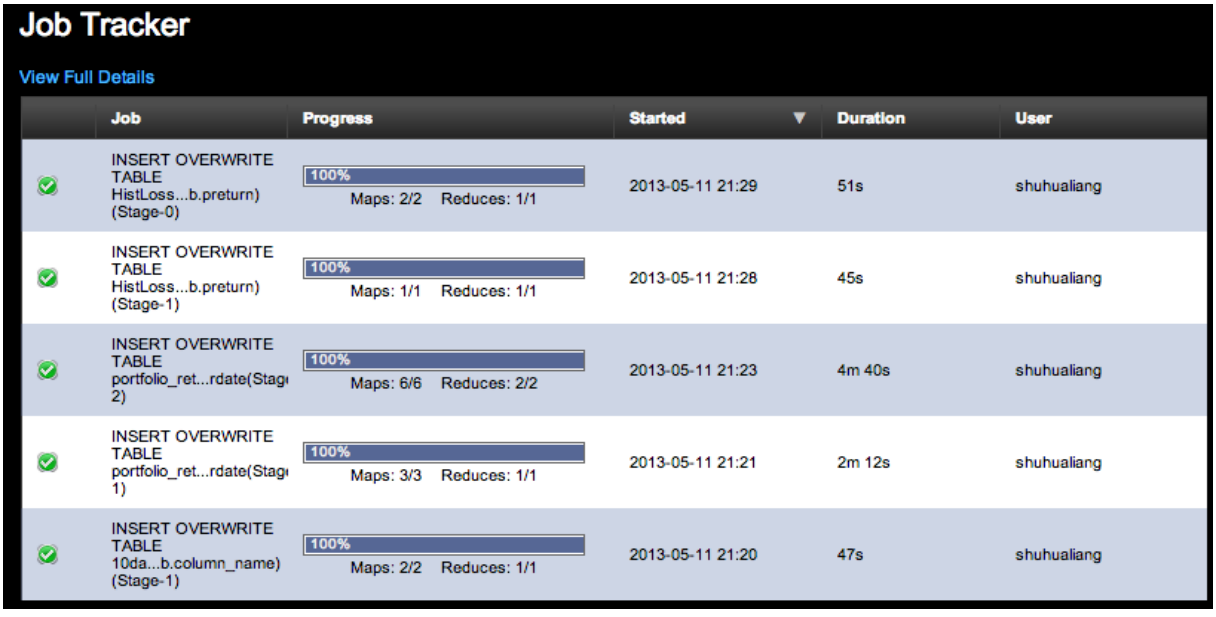
Gotchas

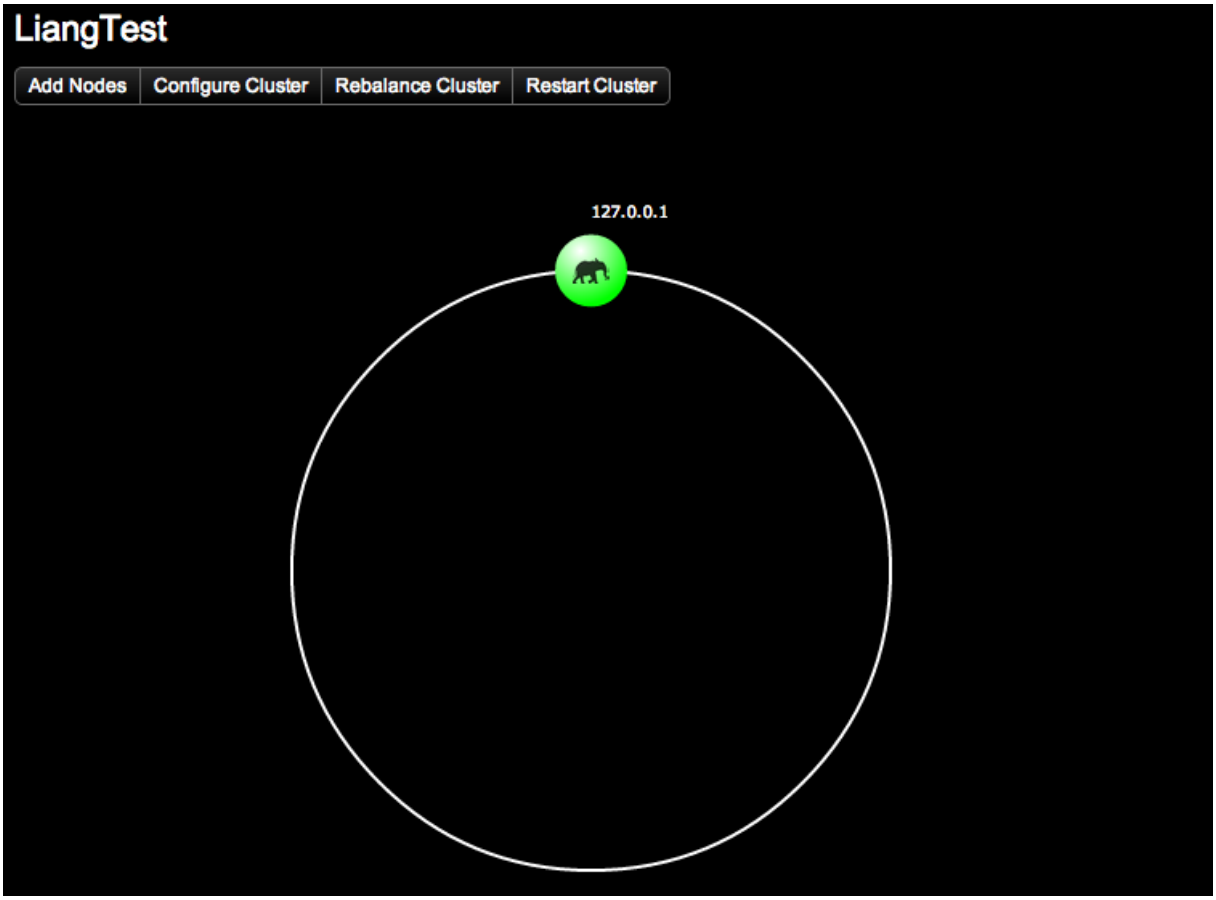
Because of limited memory on my machine, I operated OpsCenter on a single-node cluster. I learned that this software monitors the cassandra server, and Datastax Enterprise is bridged to OpsCenter. Following is a screen shot of the monitoring dash board on the demo operation:



I learned that OpsCenter makes cluster management such an easy job. It allows users to add and connect clusters in simple steps. OpsCenter made using local resources and linking to clouds, such as Amazon EC2, an easy and time-saving task.

Following are screen shots of Hadoop job and node(s) tracking:





OpsCenter provides intuitive visualization of database management. Also, it allows users to quickly catch cluster and node operations, which can avoid cluster collapses.

Testing Scenario

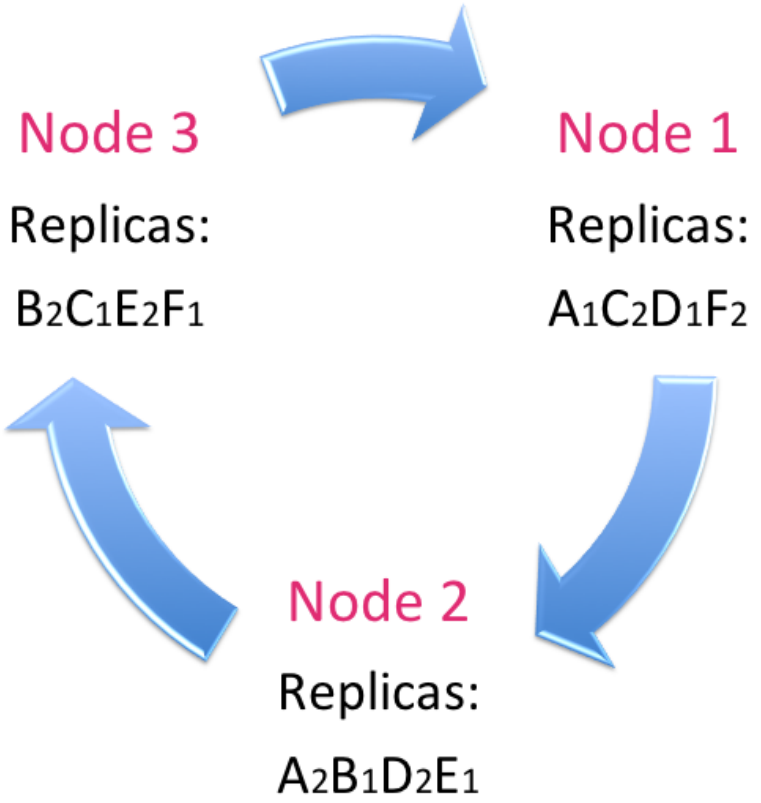
Methodology

For the scenario of a three-node cluster that has one failed, Cassandra is architected to continue to operate the remaining two nodes. Since there is not a master node, all three nodes are peer nodes, and they all serve the same functions while splitting up the replicas.

Assuming there are two replicas of each set of data in this three-node cluster, which follows the general rule of consistency level less than the total number of nodes. To validate that the failed node was able to correctly repair data that was inserted while it was down, we can match replicas row keys. We would find the node that contains the replica of the set that had been written on and then extract the replicas. Next, we would match the extracted replica with the repaired replica. If they match perfectly, then we conclude that we have successfully repaired the failed node.

Demonstration

For example, if we have datasets A, B, C, D, E and F, they will be stored in three nodes as the following diagram:



In this scenario, we can assume that Node 1 fails, and the datasets A_1 and D_1 were the ones that would have been revised. After this node is repaired, we cans extract the replicas of A_2 and D_2 from Node 2. And then, we will match the elements based on row keys in A_2 and D_2 to those in A_1 and D_1, respectively. A perfect match in those files would indicate a correct repair.

Coding in Python

With a given system log file, I was supposed to write a script to generate an aggregated list of exceptions. In other words, this is to filter a file that contains various information. I first read the file into Python as list of strings. Then, I identified the indices of lines that start with "Caused by", which means they are exceptions - let us call them the "exception indices"; this allows me to placemark the exceptions. There are lines following the those exceptions that gives extra information about he exceptions. Using the exception indices, I then identified the consecutive lines starting with an indent that follow by each exception index. Having identified these lines, I wrote them into a new file called Exceptions.txt

(Attached).

Notice there are two types of exceptions - one is originated from a JavaRuntimeException and the other from HiveRuntimeException. To separate those exceptions, I used a similar technique. I read in the Exceptions file and searched for exceptions notes containing the pattern"HiveException". Then, I wrote those lines along with their extra information lines into a new file called HiveRuntimeException.txt (Attached). Similar, for Java runtime exceptions, I searched for exceptions containing the pattern "java.lang.ClassCastException" (See attached file JavaRuntimeException.txt).

Conclusion

This technical screen is interesting because I had a chance to learn, explore, and understand DataStax's productions. With a brief exercise, I was only able to learn the basics of DataStax. I desire an opportunity to meeting with the experts at DataStax and learn more about this company and this industry.