KIPS Tr. Software and Data Eng.
Vol.3, No.9 pp.349~354 pISSN: 2287-5905

http://dx.doi.org/10.3745/KTSDE.2014.3.9.349

Performance Comparison of DW System Tajo Based on Hadoop and Relational DBMS

Liu Chen† · Junghyun Ko†† · Jeongmo Yeo†††

ABSTRACT

Since Hadoop which is the Big-data processing platform was announced, SQL-on-Hadoop is the spotlight as the technique to analyze data using SQL on Hadoop. Tajo created by Korean programmers has recently been promoted to Top-Level-Project status by the Apache in April and has been paid attention all around world. Despite a sensible change caused by Hadoop's appearance in DW market, researches of those performance is insufficient. Thus, this study has been conducted to help choose a DW solution based on SQL-on-Hadoop as progressing the test on comparison analysis of RDBMS and Tajo. It has shown that Tajo based on Hadoop is more superior than RDBMS if it is used with accurate strategy. In addition, open-source project Tajo is expected not only to achieve improvements in technique due to active participation of many developers but also to be in charge of an important role of DW in the filed of data analysis.

Keywords : Tajo, Hadoop, Analyzing Performance of Data, Data Warehouse

하둡 기반 DW시스템 타조와 관계형 DBMS의 성능 비교

유 신† · 고 정 현†† · 여 정 모†††

요 약

빅데이터 처리 플랫폼인 하둡의 등장 이후 SQL을 이용하여 하둡상에서 데이터 분석을 할 수 있는 SQL-on-Hadoop 기술이 주목받고 있다. 그 중에서도 국내 개발자가 주축이 되어 개발하고 올해 4월 아파치 최상위 프로젝트로 선정된 타조(Tajo)가 많은 주목을 받고 있다. SQL-on-Hadoop 기술의 등장으로 DW시장의 변화가 포착되고 있지만 그 성능에 관한 연구는 미미한 실정이다. 그래서 본 연구에서는 타조를 이용하여 관계형 데이터베이스와의 데이터 분석성능 비교에 관한 실험을 진행하여 SQL-on-Hadoop 기반 DW 선택에 도움이 될 연구를 수행하였다. 하둡 기반기술인 타조를 올바른 사용전략을 세워 활용한다면 관계형 데이터베이스보다 우수한 성능을 보인다는 결과를 얻었으며 오픈 소스인 타조는 많은 개발자들의 참여로 인해 점차 기술의 완성도가 높아져 DW 및 데이터 분석분야에서 중요한 축을 담당할 수 있을 것으로 예상한다.

키워드 : 타조, 하둡, 데이터 분석성능, 데이터 웨어하우스

1. 서 론

디지털 경제의 확산으로 규모를 가늠할 수 없을 정도로 많은 정보가 생산되는 빅데이터 시대를 맞이하게 되었고, 이전에는 저장 및 처리할 수 없던 수많은 데이터들에 대한 처리 및 분석의 영역이 하둡(Hadoop)[1]의 등장으로 인해 문제를 해결하고 새로운 가치를 창출할 수 있게 되었다. 하이브(Hive)[2] 또한, 관계형 데이터베이스 사용자에게 친숙한 SQL을 이용하여 하둡 플랫폼상에 저장된 데이터를 쉽게 처리하고 분석할 수 있도록 해주는 도구로 널리 사용되었다. 그러나 하이브는 데이터를 실시간으로 조회하거나 처리하는 측면에서 제약사항이 존재하였다. 그렇기 때문에 하이브의 성능을 최적화하기 위한 연구들이 많이 존재하였으나 맵리듀스[3]를 이용한 하이브에서는 태생적인 문제가 존재하고 있다. 이런 측면을 해결하기 위한 수단으로 최근 주목받고 있는 기술이 SQL-on-Hadoop이라고도 하는 SQL 질의 분석 기술이다. 이 기술은 하둡상에 저장된 대용량 데이터를 대화 형식의 SQL 질의를 통해서 처리하고 분석하는 기술이다.[4]

※ 이 논문은 부경대학교 자율창의학술연구비(2013년)에 의하여 연구되었음.
† 준 회 원 : 부경대학교 컴퓨터공학과 박사과정
†† 준 회 원 : 부경대학교 컴퓨터공학과 석사과정
††† 정 회 원 : 부경대학교 컴퓨터공학과 교수, (주)엔코아 사외이사
Manuscript Received : April 28, 2014
First Revision : June 27, 2014
Accepted : August 1, 2014
* Corresponding Author : Jeongmo Yeo(yeo@pknu.ac.kr)

SQL-on-Hadoop의 등장으로 기존 DW(Data Warehouse) 시장의 중심에 있는 업체들의 시장 판세에 커다란 변화가 포착되고 있다. 가트너(Gartner)에서는 2014년 3월 DW 관련 보고서[5]에서 처음으로 빅데이터 및 하둡전문업체 클라우데라(Cloudera)를 포함했다. 또한, 빅데이터를 지렛대로 활용하는 새로운 경쟁자가 떠오르고 있고, 전통적인 DW 벤더도 빅데이터 기술에 투자하고 있다고 발표했다. 그러나 DW 시스템에서 주로 활용되는 관계형 데이터베이스(RDBMS)와 SQL-on-Hadoop 기술들과의 성능 비교에 관한 연구는 부족한 실정이다. 수많은 RDBMS와 SQL-on-Hadoop 기반 DW 솔루션들은 저마다 벤치마크 성능테스트 결과가 우수하다며 발표하고 있지만, 그 결과는 각각의 솔루션들에 대해 하드웨어 및 시스템 환경을 최적화시켜 발표한 결과이며 특정 유사 기술들에 대한 비교결과만을 발표하고 있다. 즉 동일한 하드웨어 환경에서 SQL-on-Hadoop과 RDBMS의 성능비교연구는 미미하다는 것이다. 그리고 오픈 소스인 SQL-on-Hadoop 기술은 기존 DW 시스템 대비 가격경쟁력이 뛰어나고 값싼 하드웨어를 추가하여 유연하게 시스템을 확장해 나갈 수 있는 장점을 가지고 있음에도 불구하고 기존의 DW가 제공하는 모든 이점을 대신해 주지 않기 때문에 DW 사용자들이 어떤 것을 선택해야 할지 고민하고 있다. 그래서 본 논문에서는 SQL-on-Hadoop 기술이 RDBMS와 비교할 때 어느 정도 성능을 보이는지에 관한 연구를 수행하여 DW 선택 판단의 도움이 될 근거를 제공하고자 한다. 본 논문에서는 가장 최근 국내 개발자가 주축이 되어 개발하고 2014년 4월 아파치 최상위 프로젝트(Top Level Project)로 승격되어 많은 주목을 받고 있는 하둡기반 DW 시스템인 타조(Tajo)[6]를 이용하여 관계형 데이터베이스와의 정형 데이터 분석 성능 비교에 관한 연구를 수행하고자 한다. 또한, 동일한 하드웨어 환경을 가진 노드들을 추가해가며 Tajo의 분석 성능이 어느 정도 향상되는지에 관한 연구를 수행하고자 한다.

2. 관련 연구

2.1 하둡과 하이브

하둡은 오픈 소스 개발자인 더그 커팅(Doug Cutting)과 마이크 카파렐라(Mike Cafarella)에 의해서 2006년 발표되었다. 구글의 분산 파일 시스템의 구조와 분산 병렬처리 프레임워크인 맵리듀스(MapReduce) 논문의 아이디어를 이용하여 하둡을 만들게 된다. 이후 하둡은 야후(Yahoo!)의 지원을 받아 급속히 성장하게 되고 2008년 2월 아파치 재단 내 최고 수준 오픈 소스 프로젝트로 격상하게 되면서 점차 빅데이터 처리 표준 기술의 중심축을 담당하게 되었다. 그러나 하둡의 분산 병렬 처리 프레임워크인 맵리듀스가 대용량의 데이터를 빠르고 효과적으로 처리할 수 있는 방법을 제공하였지만, 데이터 처리 요청마다 맵리듀스 코드를 작성해야 한다는 단점 때문에 실제 데이터 분석가들이 하둡을 활용하는데 문제점이 존재하였다. 이러한 문제점을 개선하기 위해 아파치 하이브 프로젝트가 등장한다. 하이브는 페이스북에 의해 개발되었고 사용자가 SQL을 통해 질의하면 하이브 엔진에 의해 자동으로 맵리듀스 코드로 변환되어 하둡상에서 실행할 수 있도록 한다. 그래서 사용자가 맵리듀스 코드를 작성하지 않더라도 하이브를 이용하여 사용자들에게 친숙한 SQL로 질의하여 결과를 볼 수 있는 환경을 제공하였다. 그러나 맵리듀스 프로그래밍을 바탕으로 한 하이브는 개발자의 역량에 성능이 의존적이며 버그 가능성이 높았다. 또한, 맵리듀스와 하이브는 작업 스케줄링, 그리고 맵에서 리듀스로 넘어가는 과정에서 발생하는 데이터 교환 오버헤드가 존재했고 SQL과 유사하지만 많은 부분이 다르고 SQL 표준을 지원하지 않는 불편함이 존재했다.

2.2 SQL-on-Hadoop 기술

하둡과 하이브가 등장하면서 과거 투자 대비 저렴한 가격으로 대용량 데이터 처리를 할 수 있게 되었다는 사실에 만족하였으나 사용자들은 점차 보다 높은 처리 성능 및 빠른 반응을 요구하였고, 빠른 반응속도로부터 데이터 분석의 생산성을 높여 빠른 의사결정을 가능하게 할 수 있는 시스템이 필요하다는 요구가 증대되었다. 이러한 사용자들의 요구 사항 변화로 인해 하둡 기반의 차세대 분석 엔진으로 일컬어지는 SQL-on-Hadoop 기술이 등장하게 된다. SQL-on-Hadoop 기술은 맵리듀스 프레임워크가 아닌 새로운 분산 처리 프레임워크를 기반으로 HDFS에 저장된 데이터에 대한 SQL 처리를 제공하는 시스템을 말한다. 또한, SQL 표준을 지원하여 기존 시스템과 통합 또는 대체가 용이하고 맵리듀스의 한계를 극복하는 분산 처리 프레임워크로 보다 높은 처리 성능을 보장한다.[7]

구글에서 2011년 클라우드 기반의 빅데이터 서비스인 빅쿼리(Bigquery)[8] 서비스를 대중에게 공개하였는데 이 서비스는 사용자가 비정형 데이터들을 업로드 한 후, 테이블을 생성하고 SQL 질의를 하는 것과 같은 방식으로 조회 및 분석이 가능한 서비스를 제공한다. 빅쿼리의 내부 플랫폼으로 사용된 핵심 SQL-on-Hadoop 기술이 드레멜[9]이다. 구글은 드레멜 기술에 관련된 논문을 발표한 후, 드레멜과 유사한 시스템을 오픈 소스로 개발하려는 시도들이 많았다. 맵알(MapR) 주도로 개발중인 아파치 드릴(Drill), 호튼웍스(Hortonworks)에서 개발을 주도하는 아파치 스팅거(Stringer), 인 메모리 기반 시스템 샤크(Shark), 클라우데라의 임팔라(Impala), 2013년 11월 페이스북에서 개발하여 사용중인 프레스토(Presto), 그리고 본 연구에서 다루고 고려대학교와 국내 빅데이터 전문 회사 그루터(Gruter) 개발자들이 주축을 이뤄 개발중인 타조(Tajo)가 대표적인 SQL- on-Hadoop 기술들이다.

2.3 Tajo

다양한 오픈 소스 기반의 SQL 질의 기술들이 대중에게 공개되었는데, 가장 최근에 공개된 기술이 타조이다. 타조는 국내 개발자들이 주도하는 프로젝트로서 2013년 3월 아파치 인큐베이션(Incubation) 프로젝트로 선정된 이후 1년 만인 2014년 4월 최상위 프로젝트로 선정돼 관련 업계의 큰 관심을 받고 있다. 타조는 하둡 기반의 데이터웨어하우징 시스템이며 HDFS 및 다양한 소스의 대용량 데이터에 대한 집계, 연산, 조인, 정렬 등의 분석을 제공한다. 타조 역시 다른

SQL-on-Hadoop 기술들과 유사하게 맵리듀스 프레임워크 대신 자신의 쿼리 실행 엔진을 가진다. 타조는 Fig. 1과[10, 11] 같이 Master-Worker 모델의 아키텍처로 구성된다. Tajo Master는 클라이언트 및 애플리케이션의 요청을 처리하며 쿼리 실행 계획과 Worker들의 작업을 조정하는 역할을 한다. Master에서 테이블의 스키마, 물리적인 정보, 각종 통계 등 카탈로그 정보를 가지고 있으며, Query 파서, 플래너, 최적화기, 클러스터 자원 관리, Query Master 등을 관리한다. Query Master는 사용자 질의별 동작을 결정하며 질의 실행 단계(Execution Block)를 제어하고 각 태스크들을 스케줄링하는 역할을 한다. Tajo Worker는 로컬에서 Master에 의해 할당된 작업을 수행하며 Master에게 진행 상태를 주기적으로 보고한다.

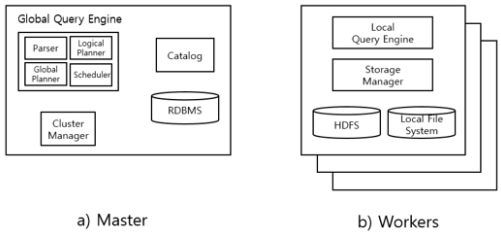

Fig. 1. Systems architecture of Tajo

2.4 벤치마크 성능

데이터베이스의 성능을 검증하는 도구로 벤치마크 시험이 사용되고 있다. 대용량의 실 데이터(real data)를 사용하는 것이 효과적이나, 대용량의 실 데이터를 얻기 위해서는 많은 시간과 비용이 소모된다. 따라서 효과적인 데이터베이스 벤치마크 시험을 위해서는 데이터 생성기를 이용하거나 실 데이터와 유사하게 생성하여 성능 테스트를 한다. 대량의 데이터시험 및 Ad-hoc query의 성능 시험에 사용되는 벤치마크는 주로 TPC-H[12]를 사용한다.

타조의 연구결과[6,10]에서는 벤치마크 테스트 성능이 하이브나 임팔라에 비해 더 빠르다고 발표하고 있다. 타조의 100G 데이터 대상 TPC-H 벤치마크 성능 테스트 결과는 Fig. 2와[10,11] 같다. 대부분의 질의 문장에서 하이브에 비해 뛰어나며 임팔라와 비교하여도 유사하거나 뛰어난 결과를 보였다.

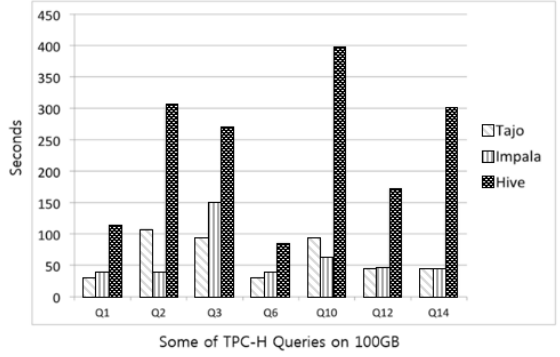

Fig. 2. Some of TPC-H Queries on 100GB

3. RDBMS와 Tajo 분석 성능 비교

```
CPU : (intel i5) 4Core
RAM : 4G
HDD : 500G * 4, 7200RPM
OS : Windows7 64bit
DBMS: ORACLE 11g
```

a) RDBMS

```
CPU : (intel i5) 4Core
RAM : 2G
HDD : 500G, 7200RPM
OS : CentOS 6.4
SW: Hadoop 2.0.5-alpha, Tajo_0.2.0,
Sqoop1.4.4
```

b) Hadoop node

Fig. 3. Specifications of test computers

다양한 SQL-on-Hadoop 기술들이 타조와 마찬가지로 하이브와 비교하여 성능 테스트 결과를 발표하고 있지만 동일하거나 유사한 하드웨어 환경에서 RDBMS와의 성능을 비교하는 연구는 극히 드물다. 그래서 본 연구에서는 최근에 가장 주목받고 있는 SQL-on-Hadoop 기술인 타조를 이용하여 RDBMS와 분석 성능을 비교하는 연구를 수행하였다. 테스트에 사용된 서버의 환경은 Fig. 3과 같다.

Table 1. Information of Tables

Table Name	Size(GB)	Records
Big_orders	103.5668	1,677,721,600
Big_employees	146.359	1,677,721,600
Big_order_items	256.5738	11,156,848,640
Big_customers	569.6716	5,033,164,800

성능 테스트는 Table 1와 같이 4개의 테이블, 총합 1T사이즈의 테스트 데이터를 생성하여 분석성능을 측정하였다. DW에서 주로 자주 사용되는 질의문의 형태는 전체로우를 대상으로 그룹별 집계결과를 출력하는 형태이기 때문에 Fig. 4와 같이 집계결과를 Result_Tab 테이블에 삽입하는 질의문을 테스트 대상 질의문으로 선정했다.

```
INSERT INTO Result_Tab(col, cnt)
SELECT col_name, AGGRIGATE FUNCTION
FROM tab_name
GROUP BY col_name ;
```

a) RDBMS SQL query

```
INSERT OVERWRITE INTO result_Tab(col, cnt)
SELECT col_name, AGGRIGATE FUNCTION
FROM tab_name
GROUP BY col_name ;
```

b) Tajo SQL query

Fig. 4. SQL statements

Table 2. Results of RDBMS's test

Table Name	Column Name	Column type	No option(h)	Parallel 4 (h)
Big_orders	Order_mode	Varchar2(8)	0.306	0.266
Big_employees	Phone_number	Varchar2(20)	0.461	0.388
Big_order_items	Quantity	Number(22)	1.670	1.549
Big_customers	Date_of_birth	Date	3.679	2.968

3.1 RDBMS 분석성능

테스트 대상 테이블에서 임의의 컬럼을 선정한 후 Fig. 4의 RDBMS 테스트 질의문장에 대해서 병렬처리 옵션의 여부에 따라 두 번 수행하였다. PL/SQL을 이용하여 질의 문장을 수행하기 전후의 시간을 기록하였고 분석 성능 결과는 Table 2과 같다. 병렬처리 옵션을 사용하지 않은 경우가 사용한 경우보다 다소 많은 시간이 걸렸으며 테이블의 전체 레코드 수와 상관없이 사이즈가 클수록 분석을 수행하는데 많은 시간이 소요됐다.

3.2 Tajo 분석성능

타조 성능 테스트를 수행하기에 앞서 RDBMS의 테스트 데이터를 하둡 파일시스템(HDFS)으로 적재하는 도구인 아파치 스쿱(Sqoop)[13]을 사용하여 HDFS의 텍스트 파일로 적재한다. 하둡 클러스터는 1대의 Master 노드와 3대의 Slave 노드로 구성하였고, Slave 노드를 1대씩 추가하여 5대의 노드가 될 때까지 구성하며 반복적으로 테스트 하였다. 분석에 걸린 시간 측정은 타조의 로그를 추적하여 기록하였다. 분석성능 결과는 Table 3와 같다.

Table 3. Results of Tajo's test

Table Name	Column Name	Column type	3 Node	4 Node	5 Node
Big_orders	Order_mode	Varchar2(8)	0.273	0.300	0.408
Big_employees	Phone_number	Varchar2(20)	0.683	0.685	1.141
Big_order_items	Quantity	Number(22)	1.241	0.713	0.777
Big_customers	Date_of_birth	Date	8.207	5.731	6.375

일반적으로 Slave 노드를 추가하면 하드웨어 자원이 늘어나기 때문에 분석 성능이 선형적으로 향상되어야 함에도 불구하고 Table 3의 결과에서는 오히려 성능이 나빠진 경우가 발생했다. Big_order_items 테이블의 결과만 비교적 양호한 결과가 도출되었고 Big_orders 테이블과 Big_employees 테이블의 경우는 노드수가 늘어남에 따라 오히려 시간이 증가하였다. 또한 Big_customers 테이블은 크기에 비해 과도한 시간이 걸렸으며 대부분의 경우 좋은 결과를 얻지 못하였다.

3.3 분석성능 비교

RDBMS에서의 성능과 타조에서의 성능을 비교한 결과는 Fig. 5와 같다. 꺾은선 그래프는 RDBMS에서의 Parallel 옵션 여부에 따른 분석성능 시간이며, 막대그래프는 하둡의 Slave 노드수별 타조의 분석성능 결과이다. 테이블이 가장 작은 Big_orders 테이블의 경우 RDBMS와 타조 모두 0.27시간으로 동일한 성능을 보였다. Big_employees 테이블의 경우는 RDB-MS에서 0.39시간, 타조 3노드에서 0.68시간으로 RDBMS의 성능이 더 뛰어났으며, Big_customers 테이블의 경우 RDBMS에서 2.97시간, 타조 4노드에서 6.37시간으로 타조에서 비교적 많은 시간이 소요되었다. Big_order _items 테이블의 경우는 RDBMS에서 1.55시간, 타조 4노드에서 0.71시간으로 타조가 RDBMS보다 분석성능이 더 빠르게 나타났다. 전반적으로 RDBMS의 성능이 타조보다 좋은 결과를 보였다.

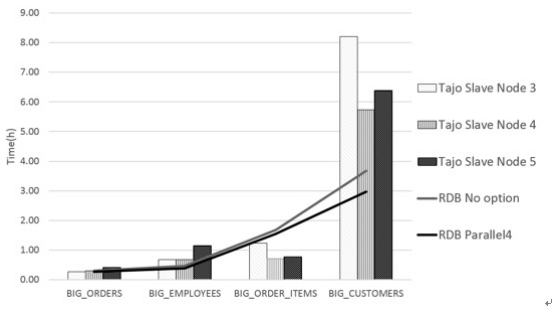

Fig. 5. Results of comparison test

3.4 Tajo 분석성능 개선 방안

타조에서 RDBMS와 분석 성능을 비교하였을 때 보다 좋은 성능을 보이지 못한 이유는 하둡 클러스터의 모든 하드웨어 자원을 효율적으로 사용하지 못한 데 이유가 있다. 앞선 분석 성능 결과는 스쿱을 통해 RDBMS의 데이터를 하둡 클러스터상으로 적재를 할 때 적재시간을 단축하기 위해서 하둡의 복제 정책을 사용하지 않고 원본만 저장하는 전략을 사용한 결과다. 스쿱은 RDBMS에 저장된 데이터를 하둡 파일시스템으로 쉽게 옮길 수 있도록 도와주는 오픈 소스 도구이다. 스쿱은 적절한 내부 알고리즘에 의해 작업을 분산하여 하둡 클러스터상으로 데이터를 옮기는데 데이터가 모든 노드에 균등하게 분배된다는 보장은 없다.

원본 데이터가 각 노드에 균등하게 적재되어 있지 않은 상태에서 데이터 분석을 수행한다면 올바른 분석 성능 효과를 이끌어 낼 수 없다. Fig. 6에서와 같이 특정 노드(Node 1~Node 6)에만 데이터들이 더 많이 쌓여 있다면 그 노드에서만 많은 작업을 수행하게 된다. Fig. 5의 결과는 작업량의 불균형으로 인해 하둡 클러스터 내의 모든 자원을 효과적으로 사용하지 못하였기 때문에 나타난 결과로 볼 수 있다. 그렇기 때문에 클러스터의 모든 데이터 노드에서 작업을 균등하게

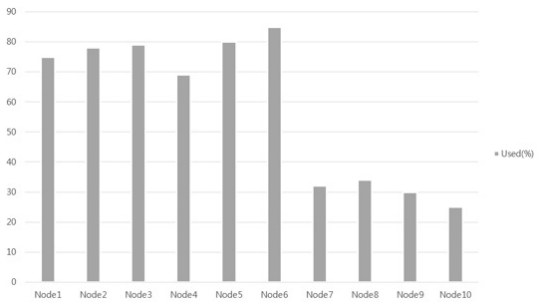

Fig. 6. Datanode usage rate

처리할 수 있게 사용자가 적절한 전략을 설정할 필요가 있다.

작업을 균등하게 처리하게 하도록 부하를 분산시키는 방법(Load Balancing)에는 두 가지가 있다. 첫째 하둡에서 제공하는 Balancer를 통해 데이터를 균등하게 분배시켜주는 방법이다. Balancer는 각 노드들에 저장된 데이터들을 균등하게 분배시켜주는 역할을 한다. 그러나 Balancer를 통하여 균등하게 분배하였다고 하더라도 처리하고자 하는 데이터는 특정 노드에 몰려 있을 수 있다. Fig. 7에서와 같이 데이터가 균등하게 분배되어 있다 하더라도 EMP 테이블의 데이터들은 노드 1, 2, 4에 저장되어 분석을 수행하는데 클러스터의 모든 자원을 모두 사용할 수 없게 된다.

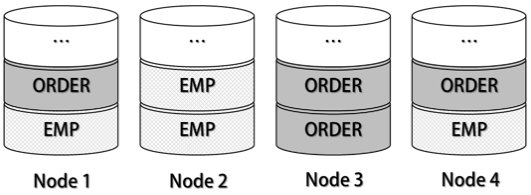

Fig. 7. State distributed equally

두 번째 방법은 적절한 하둡 복제 정책을 설정하는 것이다. 하둡은 내 고장성(Fault tolera nce)과 고가용성(High Availability)을 위해 효율적인 데이터 복제(Replication) 알고리즘을 가지고 있다. 이 복제 정책을 이용하여 분석하고자 하는 데이터가 여러 노드에 복제되어 분산되어 있다면 하둡은 각각의 노드들이 가지고 있는 데이터들의 메타정보를 이용하여 적절하게 작업을 분배할 수 있게 된다.

사용자가 적절한 복제 정책 전략을 설정하여 알맞은 복제 레벨을 지정해 주어야 한다. 복제 레벨은 실제 사용되는 데이터노드의 수보다 작은 값을 설정해야 한다. 레벨이 높을수록 많은 복제본을 생성하기 때문에 저장공간을 많이 차지하지만 높은 고가용성을 지원한다. 사용자의 시스템 사양이 충분하다면 성능을 위해서 데이터노드 수에 가까운 레벨을 설정하는 것이 성능 및 안정성에 유리하다. 하둡의 복제 정책 레벨의 기본값은 3이다. 원본 데이터에 대해 추가 2개의 데이터를 복제하여 분산시킨다. 본 연구의 실험에서는 복제 정책 레벨을 기본값으로 설정하였고, Balancer를 통한 데이터 분배 후 데이터 분석, 복제 정책을 사용한 후 데이터 분석 결과 비교는 Fig. 8과 같다.

복제 정책을 사용하지 않은 환경에서의 분석 성능에 비해 Balancer를 이용한 데이터 밸런싱 후의 분석 성능이 보다 빠르며, 복제 정책을 사용한 후의 분석 성능이 밸런싱 후의 분석 성능 보다 더 빨랐다. Big_order_items 테이블에서는 비교적 적은 성능 향상을 보였는데 이 테이블의 경우는 초기 분석시에도 RDBMS보다 좋은 성능을 보였다. 이는 스쿱을 통해 적재를 수행할 때 분석할 데이터들이 균등한 비율로 분산되어 있어 좋은 성능을 보였다는 것을 추측할 수 있다.

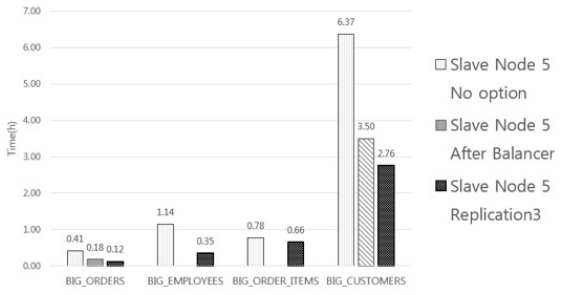

Fig. 8. Comparison and result after load balancing

3.5 Tajo 분석성능 개선 결과

이와 같은 결과에 근거하여 복제 정책을 사용하여 부하를 분산 시킨 뒤 올바른 분석 성능을 내도록 수정환경을 만든 후 분석 결과는 Fig. 9과 같다.

개선 후의 성능은 5대의 노드에서 최대 2배 이상의 성능 향상을 보였고, RDBMS와 비교해 보더라도 3노드에서부터 더 나은 성능을 보인다. 또한, 노드 수의 증가에 따라 선형적으로 성능이 빨라짐을 알 수 있다.

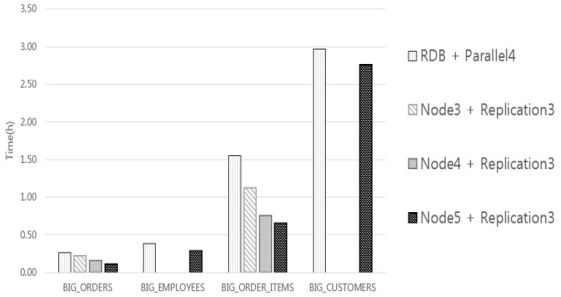

Fig. 9. Comparison and result after improvement

또한 개선을 시킨 환경에서 Fig. 10과 같이 추가적인 SQL 질의문을 대상으로 각 테이블을 대상으로 테스트를 진행하였는데 Fig. 9와 유사한 결과를 보였고 추가적인 3개의 질의문을 수행하는 데 걸린 시간에 대한 평균적인 시간을 나타낸 그림은 Fig. 11과 같다. Fig. 11의 결과에서도 볼 수 있듯이 RDBMS에서의 질의수행시간 보다 타조의 노드수 증가에 따른 수행시간이 더 빠른 것을 볼 수 있다.

```
SELECT col_name1, AVG(col_name2), SUM(col_name3*0.7)
FROM tab_name
GROUP BY col_name1
```

a) Query 1

```
SELECT col_name, COUNT(*), MAX(CASE WHEN col_name2
>= 'value' THEN col_name2 + value ELSE 0 END)
FROM tab_name
GROUP BY col_name1
```

b) Query 2

```
SELECT col_name, COUNT(*), MAX(CASE WHEN col_name2
>= 'value' THEN col_name2 + value ELSE 0 END), MIN(CASE
WHEN col_name2 <= 'value' THEN col_name2 + value ELSE
9999 END), SUM(col_name3 * (1 - col_name4))
FROM tab_name
GROUP BY col_name1
```

c) Query 3

Fig. 10. Additional SQL Queries

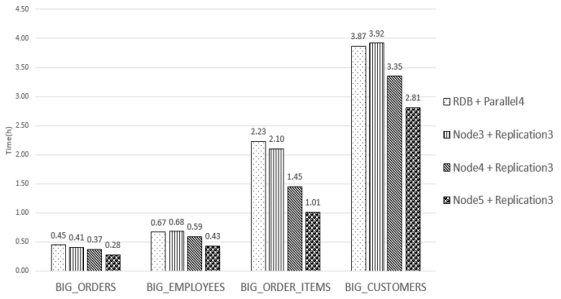

Fig. 11. Average Time about Additional SQL Queries

4. 결 론

하둡의 등장 이후 다양한 형태의 대용량 데이터를 효율적으로 처리하고 분석하는 표준 기술로 채택되고 있지만 아직도 다양한 분야에서 관계형 데이터베이스 시스템을 이용하여 데이터 분석을 수행하고 있다. 이는 하둡 기반 솔루션들의 성능이 관계형 데이터베이스에 비해 얼마나 나은 결과를 보이는지에 관한 연구가 미미하기 때문이다. 본 연구에서는 최근 주목받고 있는 SQL-on-Hadoop 기술의 일종인 타조를 이용하여 관계형 데이터베이스와의 성능 비교를 수행하였다. 최초 분석성능을 비교한 결과에서는 RDBMS에서 타조보다 나은 성능을 보였다. 또한, 하둡 노드수 증가에 따른 성능향상결과도 볼 수 없었는데, 이는 모든 하드웨어 자원에 부하를 분산시키지 못했기 때문이다. 복제정책을 사용하여 부하를 분산시킨 후 향상된 분석성능 결과는 하둡 3노드에서부터 RDBMS의 성능보다 뛰어난 결과를 나타냈다. 하둡 기반 솔루션인 타조를 올바른 전략을 세워 분석을 수행한다면 관계형 데이터베이스 보다 나은 성능을 얻을 수 있음을 본 연구에서 실험적으로 보인 것이다. 물론 타조는 운영 시스템에 적용하기 위해서는 사용자 함수 지원 및 여러 가지로 개선해야 할 사항들이 아직 많은 것이 분명하다. 하지만 적은 비용으로 투자대비 큰 효과를 낼 수 있는 장점도 존재한다. 그리고 2014년 4월 타조가 아파치의 최상위 프로젝트로 승격되면서 그 기술의 성숙도가 높다는 점을 인정받았고 이로 인해 국내외 다양한 업체의 더 많은 개발자들이 개발에 참여하여 다양한 기능이 추가되고 안정성이 개선될 것으로 보인다. 타조는 이처럼 점차 기술의 완성도가 높아져 향후 빅데이터 분석에 중요한 축을 담당할 것이며 관계형 데이터베이스가 주도하는 DW 시장에서도 주목받고 활용 될 수 있을 것으로 예상된다.

References

[1] Apache Hadoop [Internet]. http://hadoop.apache.org
[2] A. Thusoo, J. Sarma, N. Jain, Z. Shao, P. Chakka, N. Zhang, S. Antony, H. Liu, and R. Murthy, “Hive-a petabyte scale data warehouse using Hadoop,” in Data Engine-ering(ICDE), 2010 IEEE 26thInternational Conferenceon, March, 2010, pp.996-1005.
[3] J. Dean and S. Ghemawat, “MapReduce: Simplified data procession on large clusters,” in Usenix OSDI, Vol.51, No.1. ACM, 2004.
[4] Takgil Sim, “Trend of SQL-on-hadoop tech-nology based on Open-source,” The Korea Society of Computer & Information, Vol.21, No.1, Jun., 2013.
[5] Mark A. Beyer and Roxane Edjlali, “Magic Quadrant for Data Warehouse Database Management Systems,” Gartner, 2014.
[6] Hyunsik Choi, Jihoon Son, Haemi Yang, Hyoseok Ryu, Byungnam Lim, Soohyung Kim, and Yon Dohn Chung, “Tajo: A Distributed Data Warehouse System on Large Clusters,” in IEEE ICDE Conference, 2013.
[7] Jaehwa Jung, “SQL-on-Hadoop with Apache Tajo, and application case of SK Telecom”, in Deview, 2013.
[8] Googole BigQuery [Internet], http://developers.google.com/bigquery/
[9] Sergey Melnik, Andrey Gubarev, Jing Jing Long, Geoffrey Romer, Shiva Shivakumar, Matt Tolton, and Theo Vassilakis, “Dremel: Interactive Analysis of Web-Scale Datasets”, Proc. of the 36th Int'l Conf on Very Large Data Bases, pp.330-339, 2010.
[10] Hyunsik Choi “SQL-on-Hadoop and Tajo”, Tech Planet, 2013.
[11] Apache Tajo [Internet]. http://tajo.incu bator.apache.org
[12] Transaction Processing Performance Coouncill [Internet]. http://www.tpc.org/tpch/
[13] Apache Sqoop [Internet]. http://sqoop. apache.org

유 신
e-mail : 6chen.cn@gmail.com
2011년 부산외국어대학교 E-Business학과 (학사)
2013년 부경대학교 컴퓨터공학과(석사)
2013년~현 재 부경대학교 컴퓨터공학과 박사과정
관심분야 : 데이터 아키텍처, 데이터 모델링, 빅 데이터 분석

고 정 현
e-mail : jhko9120@pknu.ac.kr
2013년 부경대학교 컴퓨터멀티미디어공학과 (학사)
2013년~현 재 부경대학교 컴퓨터공학과 석사과정
관심분야 : 데이터 아키텍처, 데이터 모델링, 빅 데이터 분석

여 정 모
e-mail : yeo@pknu.ac.kr
1980년 동아대학교 전자공학과(학사)
1982년 부산대학교 전자공학과(공학석사)
1993년 울산대학교 전자 및 전산기공학과 (공학박사)
1986년~현 재 부경대학교 컴퓨터공학과 교수, (주)엔코아 사외이사
관심분야 : 데이터 아키텍처, ITA/EA