"SQL on Hadoop" 기술 동향

김종욱*

1. 서 론

클라우드 컴퓨팅은 최근 몇 년간 IT 산업의 패러다임을 바꿀 기술로 주목받고 있다. 클라우드 컴퓨팅으로 인한 IT 산업의 패러다임의 변화와 더불어 다양한 응용 분야(예, 소셜 미디어, 센서 네트워크)에서 생산 되고 있는 빅 데이터는 전통적인 데이터 웨어하우스 (Data Warehouse) 산업 전반에 영향을 미치게 되었다. 기존의 서버 기반의 관계형 데이터 웨어하우스로는 다양한 분야에서 기하급수적으로 생산되는 빅 데이터를 처리하는데 한계에 도달함에 따라, 맵리듀스 (MapReduce)을 이용한 데이터 처리 기법이 주요 인터넷 서비스 회사들 (예, Google, Yahoo, Facebook, eBay, LinkedIn)을 중심으로 활발히 사용되고 있다.

맵리듀스 기반 데이터 분석 시스템은 수백, 수천 대의 노드 컴퓨터를 이용하여 데이터를 분산 처리함으로써 서버 기반 시스템으로 대변되던 기존의 관계형 데이터베이스 웨어하우스에 비해 대용량의 데이터를 처리할 수 있으며, 처리해야 할 데이터의 양에 따라 유연성 있게 가용자원을 활용할 수 있다는 장점이 있다. 그러나 맵리듀스 기법은 배치 지향 시스템 (batch oriented system) 내에서 대용량 데이터 처리에 최적화된 기술로서, 실시간으로 사용자 질의를 처리하는데 있어서는 기존 서버 기반 관계형 데이터 웨어하우스보다 상대적으로 낮은 성능을 보이고 있다.

실시간 데이터 처리에 있어서 맵리듀스의 문제점을 해결하기 위한 다양한 연구들이 현재 산업체를 중심으로 활발하게 진행 되고 있다. 이러한 연구들 중에 최근에 가장 주목 받고 있는 기술이 "SQL on Hadoop"이라고 불리는 실시간 데이터 처리 기법이다. SQL on Hadoop는 하둡 분산 파일 시스템 (HDFS)에 저장된 대용량의 데이터들을 대상으로 SQL을 이용하여 사용자의 질의를 실시간으로 처리하는 기술이다. 본 보고서는 빅데이터 실시간 처리로 주목받고 있는 SQL on Hadoop 기술의 최근 동향을 소개 한다.

2. 맵리듀스

대용량의 데이터를 효율적으로 처리하기 위한 방법으로는 맵리듀스 (MapReduce) 기법에 관한 연구가 최근 들어 활발하게 진행되고 있다. 맵리듀스 기법은 배치 지향 시스템 환경 내에서 대용

* 상명대학교, 미디어 소프트웨어학과
(E-mail : jkim@smu.ac.kr)

량의 데이터를 효율적으로 처리하기 위한 방식으로서, 사용자는 맵 (Map)과 리듀스 (Reduce)라고 불리는 두 함수 (function)를 정의 한다 (그림 1). 맵 단계에서는 마스터 노드 (master node)가 전체 입력 데이터를 해시 함수를 이용하여 분할 한 후, 각각의 분할 된 데이터를 워커 노드 (worker node)로 분산 한다. 각각의 워커 노드들은 사용자가 정의한 맵 함수를 수행한 후, 정렬 작업을 거친 후 중간형태의 key/value로 구성 된 결과물을 생성 한다. 리듀스 단계에서는 맵 함수들에 의해 생성된 데이터를 해시함수 (hash function)를 사용하여 각각의 노드에 분산한 후 처리 한다. 리듀스 단계에서 해시함수를 사용하는 주된 목적은 같은 key를 가진 데이터가 동일한 노드에서 처리되어 질 수 있도록 하기 위함이다. 최근 들어 맵리듀스 기법은 데이터 마이닝, 정보처리 같은 다양한 분야에서 성공적으로 사용되고 있다.

맵리듀스 기법은 배치 (batch) 처리에 적합한 기술이므로, 실시간 질의 처리에 있어서는 낮은 성능을 보인다는 문제점을 가지고 있다. 또한 사용자가 직접 맵리듀스 함수를 작성해야하기 때문에 프로그래밍에 대한 지식이 부족한 일반 사용자가 사용하기 불편 한다는 단점을 가지고 있다.

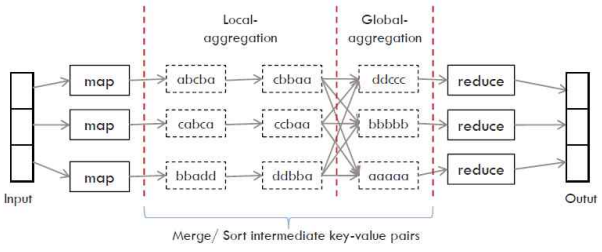

그림 1. 맵리듀스

3. SQL on Hadoop 최근 기술

본 장에서는 실시간 빅데이터 분석 및 처리를 위한 SQL on Hadoop 기술의 최근 동향을 소개한다.

3.1 Facebook Hive

Facebook에 의해 개발된 Hive는 사용자가 SQL과 유사한 HiveQL 질의를 사용하여 대용량의 데이터를 분석할 수 있는 하둡 기반의 데이터 웨어하우(Data Warehouse) 이다.

사용자에 의해 작성된 HiveQL 질의는 Hive Driver에 의해 맵리듀스 코드로 변환되어 하둡 상에서 실행된다. HiveQL은 다음과 같은 SQL 형태의 질의를 지원하고 있다.

- equi join, outer join, left semi join
- 합집합 (union)
- 서브쿼리 (subquery)
- 관계 연산 (relational operator), 산술 연산 (arithmetic operator), 논리 연산 (logical operator), 복합 형식 연산 (complex type operator)
- 산술 함수 (arithmetic function), 문자열 함수 (string function), 날짜 함수 (date function), XPath 함수 (XPath function), 사용자 정의 함수 (user-defined function)

Hive의 데이터는 관계형 데이터베이스 시스템과 같이 테이블 (table) 형태로 표현된다. Hive의 테이블은 HDFS 상의 디렉토리로 맴핑되며, 이러한 맵핑 정보는 메타스토어 (Metastore)에 저장·관리 된다. 가령, Hive에서 “test_table” 이라는 임의의 테이블을 생성하면. HDFS 상에서는 “<root_directory>/test_table”이라는 디렉토리가 생성된다. 또한, Hive의 테이블은 컬럼지향 형태의 데이터베이스 시스템 (column-oriented DBMS)처럼 컬럼에 의해 분할 될 수 있으며, 각각의 분할 된 데이터들은 테이블 디렉토리의 서브

디렉토리에 저장된다.

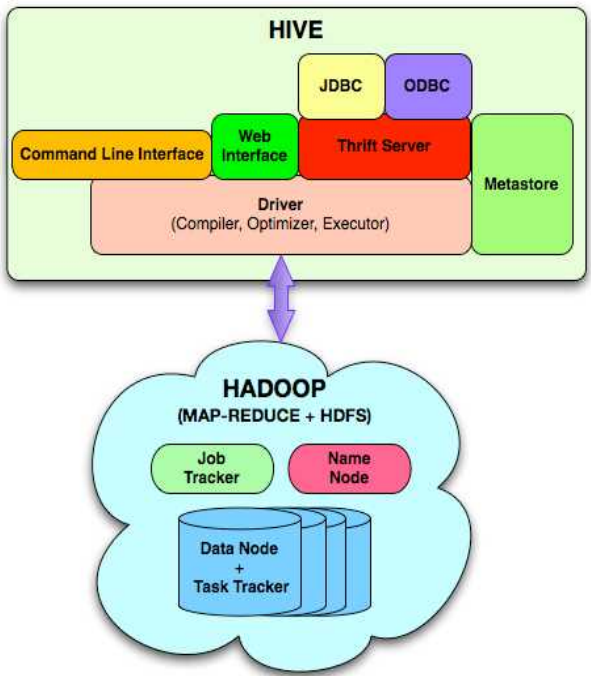

그림 2. Hive 시스템 구성도 (출처: Thusoo et al. [1])

Hive의 시스템 구성도는 그림 2에서 보는 바와 같이 크게 메타 스토어 (Meta Store), 질의 컴파일러 (Query Compiler), 실행 엔진 (Execution Engine), 클라이언트 모듈 (Client Component)로 구성 되어있다. 각각의 모듈이 맡고 있는 주요 임무는 다음과 같이 요약 되어 진다.

· 메타 스토어: Hive의 테이블과 HDFS 디렉토리 사이의 맴핑 정보를 저장·관리 하고 있다. 또한, 각 테이블의 관한 정보들도 (예, 분할, 컬럼) 메타 스토어에 저장된다.

· 질의 컴파일러: 사용자에 의해 작성된 HiveQL 질의를 일련의 맵리듀스 작업으로 구성된 방향 그래프 (directed acyclic graph)로 전환한다.

· 실행 엔진: 질의 컴파일러에 의해 생성된 일련의 맵리듀스 작업을 Hadoop 위에서 실행하는 역할을 담당한다.

· 클라이언트 모듈: 명령어 인터페이스 (CLI, Command line interface), 웹 사용자 인테페이스 (web UI), JDBC/ODBC 드라이버와 같이 사용자 혹은 응용프로그램이 Hive와 인터렉션 할 수 있는 방법을 제공해 준다.

3.2 Cloudera Impala

Hive는 사용자에 의해 작성된 HiveQL 질의를 맵리듀스 작업으로 변환 화여 하둡 상에서 실행한다. 일반적으로 하둡 상에서 실행하는 맵리듀스 작업은 맵과 리듀스 작업 사이의 셔플 (shuffle) 오버헤드로 인하여 대용량의 데이터를 대상으로 하는 복잡한 질의 수행에 있어서는 전통적인 병렬 데이터베이스 시스템 (Parallel DBMS)보다 속도가 매우 느리다는 단점을 가지고 있다. 이러한 단점을 극복하기 위해 현재 Cloudera에서 개발 중인 Impala는 HDFS 상에 데이터를 저장하기는 하지만, 질의 처리는 병렬 데이터베이스 시스템 (예, Teradata, Oracle Exadata, IBM Netezza)에서 사용하는 MPP (Massively Parallel Processing) 모델을 사용하고 있다. 따라서 Hive와 달리 Impala는 SQL 형태의 질의를 맵리듀스 코드로 변환하여 하둡 상에서 실행 시키지는 않는다 [2,3].

Impala의 시스템 구성도는 그림 3과 같이 크게 Impalad와 Impala State Store로 구성 되어 있다. Impalad는 각각의 HDFS 데이터 노드위에 존재하며, 질의에 대한 계획을 세우고 각각의 데이터 노드 상에 저장된 로컬 데이터를 대상으로 질의를 실행 하는 역할을 담당한다. 또한 노드들 간의 중간 결과 값들을 공유할 필요가 있을 때 (예, 조인

질의 수행 시), Impalad는 노드들 사이에 중간 결과 값을 전송해주는 역할도 담당한다. Impala State Store는 Impalad 대한 메타데이터를 저장·관리하는 역할을 담당한다.

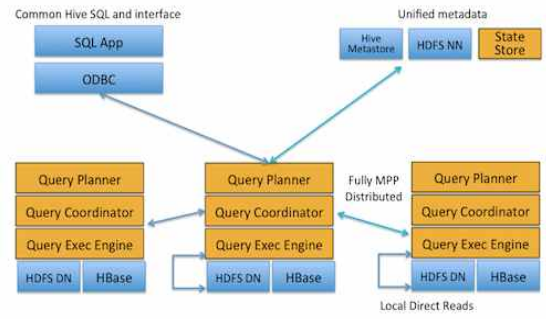

그림 3. Impela 시스템 구성도 (출처: [2])

Impala 시스템에서의 질의처리는 다음과 같이 수행 된다.

1. ODBC, JDBC, Hue GUI, Shell를 통해 제출된 사용자 질의는 시스템에 의해 선택된 특정 Impalad에 전달된다.
2. 사용자 질의를 전달받은 특정 Impalad는 Query Planner를 통하여 사용자 질의를 각각 다른 HDFS 데이터 노드 상에서 수행할 수 있도록 질의 계획을 작성한다. 작성된 질의 계획은 Query Coordinator에 의해서 각각의 HDFS 데이터 노드위에 존재하는 Impalad에 보내진다.
3. 질의 계획을 전달 받은 Impalad들은 HDFS 데이터 노드에 저장된 로컬 데이터를 대상으로 질의를 수행한다. 각각의 Impalad에 의해 실행된 질의의 중간 결과 값들은 스트리밍을 통하여 Impalad들 간에 공유 된다.
4. 각 Impalad에서 얻어진 질의의 최종 결과 값들은 최초로 질의를 전달받은 특정 Impalad에 의해 통합되어 사용자에게 전달된다.
5. 위에 설명한 바와 같이 Cloudera Impala은 병렬 데이터베이스 시스템에서 사용하는 질의 수행 방식과 매우 유사 기법을 사용함으로써, 맵리듀스 작업으로 인한 성능저하 문제점을 해결하고 있다.

3.3 Hortonworks Stinger

Hortonworks에서 개발 중인 Stinger는 기존 Hive의 성능을 개선하기 위한 프로젝트로서, 내부적으로는 MapReduce를 사용하지 않고 질의를 수행할 수 있도록 질의 처리 프로세싱을 개선하고 있다[4,5]. Stringer 프로젝트는 3단계로 구성되어 있으며, 현재 마지막 단계를 개발하고 있는 중이다. 각 단계의 주요 개발사항은 다음과 같이 요약할 수 있다[4].

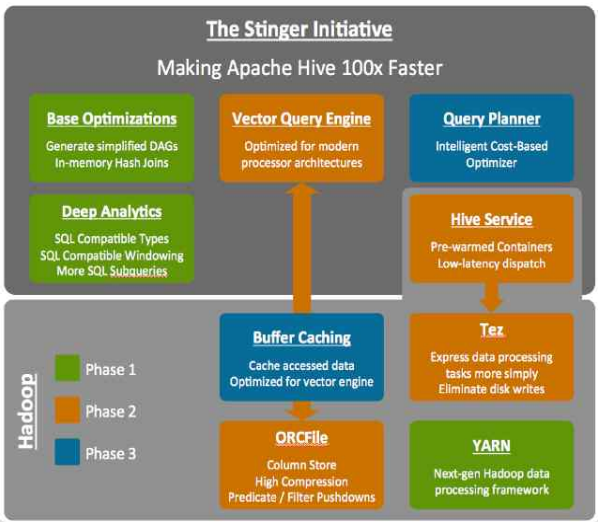

그림 4. Stinger Initiative At A Glance (출처: [5])

· Phase One: Hive 0.11에 포함된 1단계의 주요 개선사항들은 다음과 같다. 조인 질의의 성능을 높이기 위해서 in-memory hash join, sort-merge-bucket join 같은 질의 최적화 기법을 구현 하였다. 또한 1단계에서는 Hive에서 사용하는 SQL의 호환성을 개선하였으며, 사용자가 다양한

종류의 서브 질의를 사용할 수 있도록 개선하였다. 성능면에 있어서는 ORCFile이라고 불리는 컬럼 스토어 기술을 이용하여 질의 처리 성능을 Hive보다 35 ~ 40배 빠르게 향상 시켰다.

- Phase Two: Hive 0.12에 포함된 2단계에서는 vectorized query engine를 도입함으로써 다양한 종류의 질의에 대한 처리속도를 5 ~ 10배 향상 시켰다. 또한 2 단계부터는 YARN[6]을 사용하여 질의를 처리할 수 있도록 시스템을 개선하였다. SQL 측면에서는 VARCHAR와 DATE 타입을 지원함으로써 SQL 상호 운영성을 높였다.
- Phase Three: 현재 개발 진행 중인 3단계에서는 실행엔진으로 Tez를 사용하여, 기존 MapReduce 기반 실행엔진의 문제점을 해소 하였다. 또한 2단계에서 개발한 vectorized query engine의 성능을 개선하고, 다양한 종류의 SQL 함수 및 연사자를 제공할 계획이다.

3.4 Teradata Aster's SQL-H

SQL-H는 상용 데이터 웨어하우스 업계 선두주자인 Teradata에서 개발한 SQL on Hadoop 기술에 해당한다. 그림 5에서 보는바와 같이 SQL-H는 Apache HCatalog를 사용하여 사용자가 HDFS에 저장된 데이터를 접근할 수 있는 방법을 제공해 준다. SQL-H의 특징은 다음과 같이 요약할 수 있다[7].

- 현재 SQL on Hadoop에 해당하는 시스템들은 질의 최적화 방식에 있어서, 간단한 경험적 규칙을 이용하는 규칙기반 질의 최적화 (rule-based query optimization) 기법을 사용하고 있다. 규칙기반 질의 최적화 기법은 일반적으로 복잡한 형태의 질의에 대해 효율적인 질의 수행 계획을 세울 수 없다는 문제점을 가지고 있다. 이와 달리, Teradat Aster에서 개발한 SQL-H는 기존의 테라데이터 시스템에서 사용하고 있는 비용기반 질의 최적화 (cost-based query optimization) 기법을 사용하여, OLAP과 같은 복잡한 형태의 질의를 처리하는데 있어서 다른 시스템보다 높은 성능을 보이고 있다.
- SQL-H는 HFDS에 저장된 비정형화된 데이터와 기존 관계형 데이터베이스에 저장된 정형화된 데이터 사이에 조인 연산을 가능하게 함으로써, 사용자가 보다 복잡한 형태의 분석을 수행할 수 있게 한다.
- 다른 SQL on Hadoop 기술과는 달리 다양한 종류의 비즈니스 인텔리전스 (BI, Business Intelligence) 소프트웨어들이 SQL-H를 통하여 HDFS에 저장된 데이터를 사용할 수 있도록 인터페이스를 제공해 주고 있다.

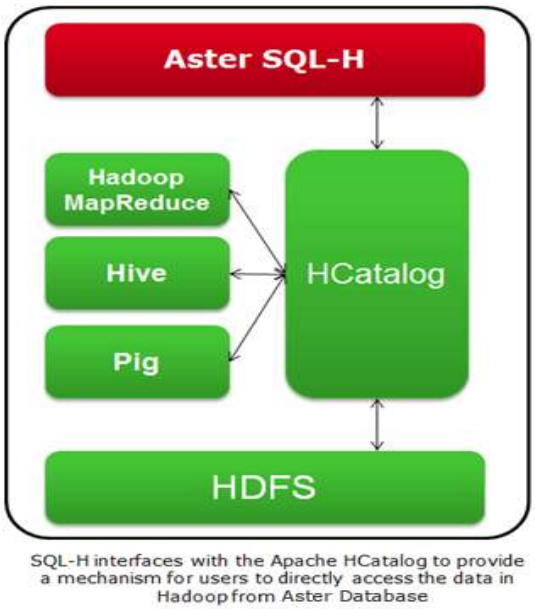

그림 5. Teradata Aster SQL-H Integration with Hadoop Catalog (출처: [7])

3.5 Hadapt

Hadapt[9]은 HadoopDB[8] 프로젝트의 상용화 제품에 해당한다. Hadapt은 데이터를 저장하기 위해 그림 6과 같이 하이브리드(hybrid) 방식을 사용 한다. 관계형 데이터 베이스를 기반으로한 데이터베이스 노드는 구조화된 데이터들은 저장하며, 하둡 분산 파일 시스템 (HDFS)는 비 구조화된 데이터 저장소로 사용된다. Hadapt은 이러한 하이브리드 구조를 이용하여 MapReduce의 장점인 확장성과 효율적인 배치 프로세싱은 그대로 유지하면서, MapReduce의 단점인 실시간 질의 처리는 데이터베이스 시스템의 기술에 의존하고 있다. 현재 Hadap은 PostgreSQL를 사용하여 실시간 질의를 처리하고 있다.

4. 결론

본 보고서에서는 현재 주목받고 SQL on Hadoop 기술의 최근 동향을 소개했다. 기존 관계형 데이터 베이스 시스템에 의해 독식되어온 데이터 웨어 하우스 시장은 앞으로 하둡 분산 파일 시스템을 기반으로 한 SQL on Hadoop 기술에 의해 분산될 것으로 예상된다.

참 고 문 헌

[1] A. Thusoo, J.S. 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 Proceedings of the 26th International Conference on Data Engineering, 2010

[2] Cloudera Impala: Real-Time Queries in Apache Hadoop, For Real, http://blog.cloudera.com/blog/2012/10/cloudera-impala-real-time-queries-in-apache-hadoop-for-real/

[3] Hadoop에서의 실시간 SQL 질의: Impala, http://helloworld.naver.com/helloworld/246342

[4] Stinger: Interactive Query for Apache Hive, http://hortonworks.com/labs/stinger/

[5] Stinger Initiative: Introduction, http://2013.adattarhazforum.hu/letoltes/2013_dw_forum/adattarhaz_forum_2013_hortonworks_chris_harris_stinger_initiative.pdf

[6] Apache Hadoop NextGen MapReduce (YARN), http://hadoop.apache.org/docs/r2.3.0/hadoop-yarn/hadoop-yarn-site/YARN.html

[7] Teradata Aster's SQL-H, http://www.asterdata.com/sqlh/

[8] A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz and A.Rasin, HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads, In Proceedings of the VLDB, 2009.

[9] Hadapt, http://hadapt.com/assets/
Hadapt-Product-Overview1.pdf