1. Ambari

Ambari 는 Apache Hadoop 클러스터의 프로비저닝, 관리, 모니터링 및 보안을 위한 완벽한 오픈소스 관리 플랫폼입니다. Ambari는 Hadoop operating에 관해 추측을 합니다. Hortonworks 데이터 플랫폼의 일부인 Apache Ambari는 기업이 HDP를 계획, 설치 및 안전하게 구성하여 클러스터 크기에 상관없이 지속적인 클러스터 유지관리 및 관리를 쉽게 제공 할 수 있도록합니다.

Ambari는 운영제어를위한 일관되고 안전한 플랫폼을 제공함으로서 Hadoop 관리를 보다 단순하게 만듭니다. Ambari는 직관적인 웹 UI와 강력한 REST API를 제공하며 클러스터 작업 자동화에 특히 유용합니다. Ambari를 통해 Hadoop 사업자는 다음과 같은 핵심 혜택을 얻습니다.

1. 단순화된 설치, 구성 및 관리.

클러스터를 쉽고 효율적으로 생성, 관리 및 모니터링합니다. Smart Configs 및 Cluster Recommendation을 사용하여 구성을 추측 할 수 있습니다. Ambari Blueprints로 반복 가능한 자동 클러스터 생성이 가능합니다.

2. 중앙 집중식 보안 설정.

복잡성을 줄이면 전체 플랫폼에서 클러스터 보안을 관리하고 구성 할 수 있습니다. Kerberos 및 Apache Ranger와 같은 고급 클러스터 보안 기능의 설정 및 구성을 자동화 할 수 있습니다.

3. 클러스터 상태에 대한 완전한 가시성.

전체적인 모니터링 방법을 사용하여 클러스터가 건전하고 사용 가능한 상태인지 확인할 수 있습니다. 클러스터 모니터링을 위한 운영 모범 사례를 기반으로 사전 정의 된 경고를 구성합니다. 분석 및 문제 해결을 위해 Grafana를 사용하여 중요한 운영 메트릭을 캡처하고 시각화합니다. 사전 예방 적 문제 예방 및 해결을 위해 Hortonworks SmartSense와 통합되었습니다.

4. 고도로 확장 가능하고 사용자 정의 가능.

Hadoop을 기업 환경에 완벽하게 적용하십시오. Ambari Stacks에서 관리하에 맞춤 서비스를 제공하고 Ambari Views를 사용하여 Ambari 웹 UI를 사용자 정의 할 수 있습니다.

2. Avro

네트워크 또는 영구 저장장치를 통해 데이터를 전송하려면 데이터를 직렬화해야 합니다. Java와 Hadoop이 제공 한 직렬화 API 이전에는 스키마 기반 직렬화 기술인 Avro라는 특별한 유틸리티가 있습니다.

Avro는 언어 독립적인 직렬화 라이브러리입니다. 이를 위해 Avro는 핵심구성요소 중 하나인 스키마를 사용합니다. 추가 데이터 처리를 위해 파일에 스키마를 저장합니다.

Avro는 Big Data 처리에 가장 적합합니다. 빠른 처리를 위해 Hadoop 및 Kafka에서 널리 사용됩니다.

Avro는 메타 데이터 섹션에 스키마와 함께 데이터를 보관하는 데이터 파일을 만듭니다. 무엇보다 다른 유사한 솔루션보다 대중적인 데이터 구조를 제공합니다.

Avro 데이터가 읽힐 때, 쓰는 데 사용되는 스키마는 항상 존재합니다. 이렇게 하면 값 당 오버헤드가 없고 빠른 속도와 작은 소모로 데이터를 작성할 수 있습니다. 스키마와 함께 데이터가 완벽하게 자체 기술되므로 동적 스크립팅 언어로도 쉽게 사용할 수 있습니다.

Avro 데이터가 파일에 저장되면 해당 스키마는 모든 프로그램에 의해 나중에 처리 될 수 있도록 it와 함께 저장됩니다. 데이터를 읽는 프로그램이 다른 스키마를 기대할 경우 이 스키마는 두 스키마가 모두 있기 때문에 쉽게 해결할 수 있습니다.

RPC에서 Avro가 사용되면 클라이언트와 서버는 연결 핸드 셰이크를 교환합니다. (이것은 대부분의 호출에서 실제로 전달되는 스키마가 없기 때문에 최적화 될 수 있습니다.) 클라이언트와 서버 모두 둘 다 fullschema를 가지므로 동일한 이름 필드, 누락 필드, 추가 필드 등이 모두 쉽게 해결 될 수 있습니다.

Avro는 스키마에 크게 의존합니다. 스키마에 대한 사전 지식없이 모든 데이터를 작성할 수 있습니다. 그것은 빠르게 직렬화되며 직렬화 된 데이터의 크기는 작아집니다. 스키마는 추가 처리를 위해 Avro 데이터와 함께 파일에 저장됩니다.

Avro는 JSON 형식을 사용하여 데이터 구조를 선언합니다. 현재 Java, C, C ++, C #, Python 및 Ruby와 같은 언어를 지원합니다.

3. Cassandra

Apache Cassandra 데이터베이스는 성능 저하없이 확장 성과 가용성이 필요할 때 올바른 선택입니다. 범용 하드웨어 또는 클라우드 인프라에 대한 선형 확장 성 및 입증 된 내결함성은 완벽한 플랫폼 형성 핵심 데이터입니다. Cassandra는 여러 데이터 센터에 걸쳐 복제를 지원하므로 동급 최강의 제품으로 사용자의 지연 시간을 줄이고 지역 장애를 극복 할 수 있다는 확신을 가질 수 있습니다.

Apache Cassandra는 무료 및 오픈 소스, 분산 형 와이드 컬럼 스토어인 NoSQL 데이터베이스 관리 시스템으로 많은 상용 서버에서 대량의 데이터를 처리 할 수 ​​있도록 설계되어 단일 실패 지점이 없는 고 가용성을 제공합니다. Cassandra는 비동기 마스터리스 복제 기능을 사용하여 모든 클라이언트에 대해 대기 시간이 적은 작업을 허용하는 여러 데이터 센터에 걸친 클러스터에 대한 강력한 지원을 제공합니다

분산

클러스터의 모든 노드는 동일한 역할을 합니다. 단일 실패 지점이 없습니다. 데이터는 클러스터에 분산되므로 각 노드는 다른 데이터를 포함하지만 모든 노드가 모든 요청을 처리 할 수 ​​있으므로 마스터가 없습니다.

복제 및 다중 데이터 센터 복제 지원

복제 전략은 구성 가능합니다. 카산드라는 다수의 데이터 센터에 다수의 노드를 배치하기 위해 분산 시스템으로 설계되었습니다. Cassandra의 분산 아키텍처의 주요 기능은 다중 데이터 센터 배포, 이중화, 장애 조치 및 재해 복구를 위해 특별히 설계되었습니다.

확장성

가동 중단이나 응용 프로그램 중단을 방지하기 위해 읽기 및 쓰기 처리량이 새 기계 추가에 따라 선형적으로 증가하도록 설계되었습니다.

내결함성

데이터는 내결함성을 위해 자동으로 여러 노드에 복제됩니다. 여러 데이터 센터에서 복제가 지원됩니다. 장애가 발생한 노드는 중단 시간없이 교체 될 수 있습니다.

4. Chukwa

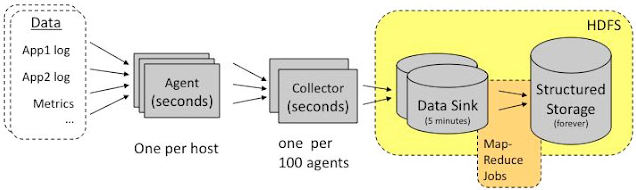
Apache Chukwa는 대규모 분산 시스템을 모니터링하기위한 오픈 소스 데이터 수집 시스템입니다. Apache Chukwa는 Hadoop 분산 파일 시스템 (HDFS) 및 Map / Reduce 프레임 워크 위에 구축되고 Hadoop의 확장 성과 견고성을 계승합니다. Apache Chukwa는 또한 수집된 데이터를 최대한 활용하기 위해 결과를 표시, 모니터링 및 분석하기위한 유연하고 강력한 툴킷을 포함합니다.

구성요소

각 시스템에서 실행되는 에이전트로 다양한 응용 프로그램에서 생성 된 로그를 수집합니다.

에이전트에서 데이터를 수신하여 안정적인 저장소에 기록하는 수집기 - Hadoop의 경우 HDFS

MapReduce 작업 또는 파싱 및 데이터 보관.



Chukwa Agents는 로그를 Hadoop으로 전송해야하는 모든 시스템에서 실행됩니다. 에이전트는 어댑터를 사용하여 응용 프로그램 계층에서 생성 된 로그를 수집합니다. 하나의 에이전트는 많은 어댑터를 가질 수 있으며, 각각은 로그를 수집하는 별도의 태스크를 수행합니다.

수집가는 로그 수집의 실제 작업을 수행합니다. 위 그림에서 설명한 것처럼 로그를 생성하는 모든 응용 프로그램에는 고유 한 어댑터가 있습니다. 어댑터의 작업은 로그를 에이전트로 보내고 에이전트는 콜렉터로 전달합니다. 수집기는 다양한 에이전트에서 수집 한 모든 로그를 HDFS의 데이터 싱크 파일에 저장합니다. 추가 작업은 Chukwa 측에서 수행되므로 대용량의 로그를 성공적으로 처리 할 수 ​​있습니다. 임계 값 이후에 데이터 싱크의 이름이 변경되고 이동되며 다음 데이터 싱크 파일에 로그가 순차적으로 기록됩니다.

5. HBase

HBase는 Google의 Bigtable을 모델로 하고 Java로 작성된 오픈 소스 비관계형 분산 데이터베이스입니다. Apache Software Foundation의 Apache Hadoop 프로젝트의 일부로 개발되었으며 HDFS (Hadoop Distributed File System) 또는 Alluxio에서 실행되어 Hadoop에 Bigtable과 같은 기능을 제공합니다. 즉, 대량의 sparse 데이터 (20 억 개의 레코드 그룹에서 50 개의 가장 큰 항목을 찾거나, 2 억 개의 레코드 그룹을 찾는 것과 같이 빈 또는 중요하지 않은 데이터의 대량 수집에 소량의 정보를 저장하는 내결함성있는 방법을 제공합니다. 거대한 수집의 0.1 % 미만을 나타내는 0이 아닌 항목).

HBase는 원래의 Bigtable 논문에서 설명한대로 열 단위로 압축, 메모리 내 작업 및 Bloom 필터를 제공합니다. HBase의 테이블은 Hadoop에서 실행되는 MapReduce 작업의 입력 및 출력 역할을 할 수 있으며 Java API뿐만 아니라 REST, Avro 또는 Thrift 게이트웨이 API를 통해 액세스 할 수도 있습니다. HBase는 열 기반의 키 - 값 데이터 저장소이며 Hadoop 및 HDFS의 계보로 인해 널리 채택되었습니다. HBase는 HDFS를 기반으로 하며 높은 처리량과 낮은 입력 / 출력 대기 시간을 갖는 대형 데이터 세트에서 더 빠른 읽기 및 쓰기 작업에 적합합니다.

HBase는 고전적인 SQL 데이터베이스를 직접적으로 대체하는 것은 아니지만 Apache Phoenix 프로젝트는 다양한 분석 및 비즈니스 인텔리전스 애플리케이션과 통합 할 수있는 JDBC 드라이버뿐만 아니라 HBase 용 SQL 레이어를 제공합니다. Apache Trafodion 프로젝트는 HBase를 스토리지 엔진으로 사용하는 여러 구문, 테이블 및 행에 대해 ODBC 및 JDBC 드라이버와 분산 ACID 트랜잭션 보호 기능을 갖춘 SQL 쿼리 엔진을 제공합니다.

HBase는 현재 몇 가지 데이터 기반 웹 사이트를 제공하고 있지만 페이스 북의 메시징 플랫폼은 최근 HBase에서 MyRocks로 마이그레이션했습니다. 관계형 및 기존 데이터베이스와 달리 HBase는 SQL 스크립팅을 지원하지 않습니다. 대신 MapReduce 응용 프로그램과 유사성을 사용하여 Java로 작성되었습니다.

HBase는 Hadoop 분산 파일 시스템 (HDFS)에서 실행되는 열 기반 데이터베이스 관리 시스템입니다. 많은 데이터 사용 사례에서 흔히 볼 수 있는 sparse 데이터 세트에 적합합니다.

HBase 시스템은 테이블 세트로 구성됩니다. 각 테이블에는 기존 데이터베이스와 마찬가지로 행과 열이 포함되어 있습니다. 각 테이블에는 기본 키로 정의 된 요소가 있어야하며 HBase 테이블에 대한 모든 액세스 시도는이 기본 키를 사용해야합니다.

6. Hive

Apache Hive는 데이터 쿼리 및 분석을 제공하기 위해 Apache Hadoop 위에 구축된 데이터 웨어 하우스 소프트웨어 프로젝트입니다. Hive는 Hadoop과 통합되는 다양한 데이터베이스 및 파일 시스템에 저장된 데이터를 쿼리하는 SQL과 같은 인터페이스를 제공합니다. 분산 데이터에 대해 SQL 응용 프로그램 및 쿼리를 실행하려면 MapReduce Java API에서 기존 SQL 쿼리를 구현해야 합니다. Hive는 저수준 Java API에서 쿼리를 구현할 필요없이 SQL과 유사한 쿼리 (HiveQL)를 기본 Java에 통합하기 위해 필요한 SQL 추상화 기능을 제공합니다. 대부분의 데이터 웨어 하우징 응용 프로그램은 SQL 기반 쿼리 언어로 작동하므로 Hive는 SQL 기반 응용 프로그램의 Hadoop으로의 이식을 지원합니다. 페이스 북이 처음 개발한 Apache Hive는 넷플릭스 (Netflix) 및 금융 산업 규제 당국 (FINRA)과 같은 다른 회사에서 사용하고 개발했습니다. Amazon은 Amazon Web Services에서 Amazon Elastic MapReduce에 포함된 Apache Hive의 소프트웨어 포크를 관리합니다.

Apache Hive는 Hadoop의 HDFS에 저장된 대규모 데이터 세트와 Amazon S3 파일 시스템 및 Alluxio와 같은 호환 가능한 파일 시스템의 분석을 지원합니다. HiveQL 이라고 불리는 SQL과 유사한 질의 언어를 스키마와 함께 제공하고 MapReduce, Apache Tez 및 Spark 작업으로 쿼리를 투명하게 변환합니다. 세 가지 실행 엔진은 하둡의 자원 협상가 인 YARN (또 다른 자원 협상가)에서 실행될 수 있습니다. 쿼리를 가속화하기 위해 비트 맵 인덱스를 포함하여 인덱스를 제공합니다. Hive의 다른 기능은 다음과 같습니다.

가속화를 제공하는 인덱싱, 0.10에서 압축 및 비트 맵 인덱스를 포함한 인덱스 유형, 더 많은 인덱스 유형이 계획됩니다.

일반 텍스트, RCFile, HBase, ORC 및 기타와 같은 다양한 저장소 유형

관계형 데이터베이스 관리 시스템의 메타 데이터 저장소로 쿼리 실행 중 의미 확인을 수행하는 시간이 크게 줄어 듭니다.

DEFLATE, BWT, snappy 등의 알고리즘을 사용하여 Hadoop 에코 시스템에 저장된 압축 데이터를 조작합니다.

날짜, 문자열 및 기타 데이터 마이닝 도구를 조작하는 내장 사용자 정의 함수 (UDF). Hive는 내장 함수가 지원하지 않는 유스 케이스를 처리하도록 UDF 세트를 확장하는 것을 지원합니다.

암시 적으로 MapReduce 또는 Tez 또는 Spark 작업으로 변환되는 SQL 유사 쿼리 (HiveQL).

7. Mahout

Apache Mahout은 주로 협업 필터링, 클러스터링 및 분류의 영역에 초점을 둔 분산 또는 기타 확장 가능한 기계 학습 알고리즘의 무료 구현을 생성하는 Apache Software Foundation의 프로젝트입니다. 구현의 대부분은 Apache Hadoop 플랫폼을 사용합니다. Mahout은 또한 일반적인 수학 연산 (선형 대수와 통계에 초점을 맞춘)과 원시 자바 컬렉션을 위한 Java 라이브러리를 제공합니다. Mahout은 진행중인 작업입니다. 구현된 알고리즘의 수가 급속히 증가했지만, 다양한 알고리즘은 여전히 ​​빠져있습니다.

클러스터링, 분류 및 배치 기반 협업 필터링을 위한 Mahout의 핵심 알고리즘은 map / reduce 패러다임을 사용하여 Apache Hadoop 위에 구현되지만 Hadoop 기반 구현에 대한 기여는 제한하지 않습니다. 단일 노드 또는 Hadoop이 아닌 클러스터에서 실행되는 컨트리뷰션도 환영합니다. 예를 들어 Mahout의 'Taste'협업 필터링 추천 구성 요소는 원래 별개의 프로젝트였으며 Hadoop없이 독립 실행 형으로 실행할 수 있습니다.

릴리스 0.10.0부터이 프로젝트는 "Samsara"라는 백엔드 독립적 프로그래밍 환경 구축에 초점을 옮겼습니다. 환경은 대수 백엔드 독립 옵티마이저와 메모리 및 분산 대수 연산자를 통합하는 대수 스칼라 DSL로 구성됩니다.

Apache Mahout은 Hadoop MapReduce를 기반으로 하는 강력하고 확장 가능한 기계 학습 라이브러리입니다. 기계 학습은 시스템이 데이터만으로 학습할 수 있게 해주는 인공 지능의 원칙으로 더 많은 데이터가 처리되면서 지속적으로 성능을 향상시킵니다. 기계 학습은 일상 생활의 일부인 많은 기술의 기초입니다.

최근까지 데이터 과학자들은 사용하고 있는 컴퓨팅 프레임 워크에 기계 학습 알고리즘을 수동으로 구현하고 커스터마이징 해야 했기 때문에 많은 양의 작업이 필요했습니다. 이제 Hadoop 및 Mahout을 사용하여 데이터 과학자는 사전 정의된 여러 알고리즘을 참조하는 MapReduce 작업을 작성하여 이러한 종류의 응용 프로그램을 쉽게 구축할 수 있습니다.

Mahout의 알고리즘은 Hadoop 위에 작성되므로 분산 환경에서 잘 작동합니다. Mahout은 Apache Hadoop 라이브러리를 사용하여 클라우드에서 효과적으로 확장합니다.

Mahout은 대량의 데이터에서 데이터 마이닝 작업을 수행할 수 있도록 즉시 사용할 수 있는 프레임 워크를 코더에 제공합니다.

Mahout을 사용하면 응용 프로그램이 많은 양의 데이터를 효과적이고 신속하게 분석 할 수 있습니다.

8. Pig

Apache Pig 는 Apache Hadoop에서 실행되는 프로그램을 작성하기위한 고급 플랫폼입니다. Pig는 MapReduce, Apache Tez 또는 Apache Spark에서 Hadoop 작업을 실행할 수 있습니다. Pig Latin은 Java MapReduce 관용구의 프로그래밍을 관계형 데이터베이스 관리 시스템의 SQL과 비슷한 MapReduce 프로그래밍을 높은 수준으로 만드는 표기법으로 추상화합니다. Pig Latin은 사용자가 Java, Python, JavaScript, Ruby 또는 Groovy 로 작성할 수 있는 사용자 정의 함수 (UDF)를 사용하여 확장한 다음 언어에서 직접 호출할 수 있습니다.

Apache Pig는 MapReduce의 추상화입니다. 데이터 흐름에 따라 더 큰 데이터 세트를 분석하는 데 사용되는 도구 / 플랫폼입니다. Pig는 일반적으로 Hadoop과 함께 사용됩니다. Apache Pig를 사용하여 Hadoop에서 모든 데이터 조작 작업을 수행할 수 있습니다.

데이터 분석 프로그램을 작성하기 위해 Pig는 Pig Latin으로 알려진 고급 언어를 제공합니다. 이 언어는 프로그래머가 데이터 읽기, 쓰기 및 처리를 위한 자체 기능을 개발할 수 있는 다양한 연산자를 제공합니다.

Apache Pig를 사용하여 데이터를 분석하려면 프로그래머는 Pig Latin 언어를 사용하여 스크립트를 작성해야 합니다. 이러한 모든 스크립트는 내부적으로 Map 및 Reduce 작업으로 변환됩니다. Apache Pig에는 Pig Latin 스크립트를 입력으로 받아들여 해당 스크립트를 MapReduce 작업으로 변환하는 Pig Engine이라는 구성 요소가 있습니다.

Java에 능숙하지 않은 프로그래머는 일반적으로 Hadoop과의 작업에 어려움을 겪었습니다. 특히 MapReduce 작업을 수행하는 동안 어려움을 겪었습니다. Apache Pig는 그러한 프로그래머 모두에게 이익입니다.

Pig Latin을 사용하여 프로그래머는 복잡한 코드를 Java로 입력하지 않고도 MapReduce 작업을 쉽게 수행 할 수 있습니다.

Apache Pig는 다중 쿼리 방식을 사용하므로 코드 길이가 줄어 듭니다. 예를 들어 Java에서 200 줄의 코드 (LoC)를 입력해야 하는 작업은 Apache Pig에 10 LoC 미만으로 입력하면 쉽게 수행할 수 있습니다. 궁극적으로 Apache Pig는 개발 시간을 거의 16 배 단축합니다.

9. Spark

Apache Spark는 빠르고 일반적인 범용 클러스터 컴퓨팅 시스템입니다. Java, Scala, Python 및 R의 고급 API와 일반 실행 그래프를 지원하는 최적화된 엔진을 제공합니다. 또한 SQL 및 구조화된 데이터 처리를 위한 Spark SQL, 기계 학습을 위한 MLlib, 그래프 처리를 위한 GraphX ​​및 Spark Streaming을 비롯한 다양한 고급 도구 세트를 지원합니다.

Apache Spark는 아키텍처 기반으로, 장애 허용 방식으로 유지 관리되는 컴퓨터 클러스터에 분산된 데이터 항목의 읽기 전용 다중 세트 인 RDB (resilient distributed dataset)를 가지고 있습니다. Dataframe API는 RDD 위에 추상화 된 다음 Dataset API로 출시되었습니다. Spark 1.x에서 RDD는 주요 응용 프로그래밍 인터페이스 (API) 였지만 Spark 2.x에서는 RDD API가 더 이상 사용되지 않더라도 Dataset API를 사용하는 것이 좋습니다. RDD 기술은 여전히 ​​Dataset API의 기초를 둡니다.

Spark와 RDD는 MapReduce 클러스터 컴퓨팅 패러다임의 한계에 부응하여 2012 년에 개발되었습니다. Map 패밀리는 분산 프로그램에서 특정 선형 데이터 흐름 구조를 강제합니다. MapReduce 프로그램은 디스크에서 입력 데이터를 읽고, 데이터에서 함수를 매핑하고, 결과를 줄입니다. 맵핑 및 저장 결과를 디스크에 저장합니다. Spark의 RDD는 (일부러) 제한된 형식의 분산 공유 메모리를 제공하는 분산 프로그램의 작업 집합으로서의 기능을 합니다.

Spark은 반복적으로 데이터 세트를 여러 번 방문하는 반복 알고리즘과 데이터의 반복적 인 데이터베이스 스타일 쿼리와 같은 양방향 / 탐색 데이터 분석을 용이하게 합니다. 이러한 애플리케이션의 대기 시간은 Apache Hadoop MapReduce 구현에 비해 몇 배 정도 감소할 수 있습니다. 반복 알고리즘 중 클래스에는 Apache Spark 개발을 위한 초기 추진력을 형성하는 기계 학습 시스템 용 학습 알고리즘이 있습니다.

Apache Spark에는 클러스터 관리자와 분산 저장 시스템이 필요합니다. 클러스터 관리를 위해 Spark는 독립 실행 형 (기본 Spark 클러스터), Hadoop YARN 또는 Apache Mesos를 지원합니다. 분산 스토리지를 위해 Spark는 Alluxio, Hadoop 분산 파일 시스템 (HDFS), MapR File System (MapR-FS), Cassandra, OpenStack Swift, Amazon S3, Kudu , 또는 맞춤 솔루션을 구현할 수 있습니다. Spark은 또한 pseudo-distributed 로컬 모드를 지원합니다. 일반적으로 개발 또는 테스트 용도로만 사용되며 분산 저장 장치가 필요하지 않고 로컬 파일 시스템을 대신 사용할 수 있습니다. 이러한 시나리오에서 Spark는 CPU 코어 당 하나의 실행 프로그램이있는 단일 시스템에서 실행됩니다.

10. Tez

하둡의 YARN 기반 데이터 처리 응용 프로그램을 위한 프레임 워크

Apache ™ Tez은 Apache Hadoop의 YARN이 조정 한 고성능 배치 및 대화식 데이터 처리 응용 프로그램을 구축하기위한 확장 가능한 프레임 워크입니다. Tez는 MapReduce 패러다임의 속도를 대폭 향상시키고 MapReduce의 페타바이트 단위로 확장할 수 있는 기능을 유지합니다. Apache Hive 및 Apache Pig와 같은 중요한 Hadoop 생태계 프로젝트는 Apache Tes를 사용합니다. 더 광범위한 Hadoop 생태계를 위해 개발된 타사 데이터 액세스 응용 프로그램이 늘어나고 있습니다.

Apache Tez은 개발자 API와 프레임 워크를 제공하여 대화 형 작업과 일괄 작업의 범위를 연결하는 원시 YARN 응용 프로그램을 작성합니다. 이를 통해 데이터 액세스 응용 프로그램은 수천 개의 노드에서 페타바이트 단위의 데이터로 작동할 수 있습니다. Apache Tez 구성 요소 라이브러리를 사용하면 개발자는 기본적으로 Apache Hadoop YARN과 통합되는 Hadoop 응용 프로그램을 만들 수 있으며 혼합된 작업 부하 클러스터에서 잘 수행할 수 있습니다.

Tez는 확장 가능하고 내장 가능하기 때문에 MapReduce 및 Apache Spark와 같은 최종 사용자 지향 엔진에 비해 최적화된 데이터 처리 응용 프로그램을 표현할 수 있는 목적에 맞는 자유를 제공합니다. 또한 Tez은 사용자가 복잡한 계산을 데이터 흐름 그래프로 표현할 수 있는 사용자 지정 가능한 실행 아키텍처를 제공하여 데이터 및 프로세스 요구 사항에 대한 실제 정보를 기반으로 동적 성능 최적화를 허용합니다.

Tez는 MapReduce 패러다임을 계산을 데이터 흐름 그래프로 표현하여 보다 강력한 프레임 워크로 일반화합니다. Tez은 최종 사용자를 위한 것이 아닙니다. 실제로 개발자는 성능과 유연성이 훨씬 뛰어난 최종 사용자 응용 프로그램을 만들 수 있습니다. Hadoop은 전통적으로 대량의 데이터를 처리하는 일괄 처리 플랫폼이었습니다. 그러나 거의 실시간으로 쿼리 처리를 수행하는 유스케이스가 많이 있습니다. MapReduce 패러다임에 적합하지 않은 기계 학습과 같은 여러 작업 부하가 있습니다. Tez은 Hadoop이 이러한 사용 사례를 해결하는 데 도움이 됩니다.

Tez 프로젝트는 사람들이 일을 할 수 있는 방법을 벗어나지 않으면 서 광범위한 유스케이스를 충족시킬 수 있도록 고도의 사용자 정의가 가능한 것을 목표로 합니다.

11. ZooKeeper

Apache ZooKeeper는 Apache Software Foundation의 소프트웨어 프로젝트입니다. 이것은 본질적으로 분산 시스템을 위한 계층 형 키 - 값 저장소에 대한 중앙 집중식 서비스로서 대규모 분산 시스템을 위한 분산 구성 서비스, 동기화 서비스 및 명명 레지스트리를 제공하는 데 사용됩니다. ZooKeeper는 Hadoop의 하위 프로젝트였지만 현재는 최상위 수준의 Apache 프로젝트입니다.

ZooKeeper의 아키텍처는 중복 서비스를 통해 고 가용성을 지원합니다. 따라서 클라이언트는 첫 번째 응답에 실패하면 다른 ZooKeeper 리더에게 요청할 수 있습니다. ZooKeeper 노드는 파일 시스템이나 트리 데이터 구조와 마찬가지로 계층 적 이름 공간에 데이터를 저장합니다. 클라이언트는 노드에서 읽고 쓸 수 있으며 이러한 방식으로 공유 구성 서비스를 갖습니다. ZooKeeper는 원자력 방송 시스템으로 볼 수 있으며 이를 통해 업데이트가 완전히 주문됩니다. ZooKeeper Atomic Broadcast (ZAB) 프로토콜은 시스템의 핵심입니다.

ZooKeeper는 Rackspace, Yahoo!, Odnoklassniki, Reddit, NetApp SolidFire, Facebook, Twitter 및 eBay뿐만 아니라 Solr과 같은 오픈 소스 엔터프라이즈 검색 시스템을 포함하는 회사에서 사용됩니다.

ZooKeeper는 원래 ZooKeeper 서버의 로컬 로그 파일에 상태를 저장하여 Big Data 클러스터에서 실행중인 프로세스를 간소화하기 위해 Yahoo에서 개발되었습니다. 이 서버는 클라이언트 시스템과 통신하여 정보를 제공합니다. ZooKeeper는 분산 형 대형 데이터 응용 프로그램을 배포하는 동안 발생한 버그를 수정하기 위해 개발되었습니다. Apache ZooKeeper의 주요 기능 중 일부는 다음과 같습니다.

신뢰할 수 있는 시스템: 이 시스템은 노드가 실패하더라도 계속 작동하므로 매우 안정적입니다.

간단한 아키텍처 : ZooKeeper의 아키텍처는 프로세스를 조정하는 데 도움이 되는 공유 된 계층 적 네임 스페이스가 있으므로 매우 간단합니다.

빠른 처리 : ZooKeeper는 특히 "읽기 우위"작업 부하 (즉, 쓰기가 읽기보다 훨씬 일반적인 작업 부하)에서 빠릅니다.

확장 성 : 노드를 추가하여 ZooKeeper의 성능을 향상시킬 수 있습니다.

문제1) Ambari는 Apache Hadoop의 어떤 부분을 관리하는가? : 클러스터

문제2) Ambari를 사용함으로서 얻는 장점은 무엇인가? : 클러스터를 쉽고 효율적으로 생성, 관리 및 모니터링합니다

문제3) Avro를 사용하는 이유는 무엇인가? : 데이터를 직렬화 하기 위해서

문제4) 데이터를 직렬화 하는 이유는 무엇인가? : 네트워크 또는 영구 저장장치를 통해 데이터를 전송하려면 데이터를 직렬화해야 합니다

문제5) Cassandra를 어떤 역할을 수행하는가? : 분산형 Data storage를 구성하기 위해 사용된다.

문제6) Cassandra는 가동 중단이나 응용 프로그램 중단이 일어났을 때 어떤 대응책이 있는가? : 읽기 및 쓰기 처리량이 새 기계 추가에 따라 선형적으로 증가하도록 설계, 데이터는 내결함성을 위해 자동으로 여러 노드에 복제됩니다

문제7) Chukwa를 사용하는 이유는 무엇인가? : 대규모 분산 시스템을 모니터링하기위한 오픈 소스 데이터 수집을 위해

문제8) Chukwa는 실시간 분석이 가능하다 : O

문제9) HBase의 역할은 무엇인가? : 하둡 기반의 분산 데이터베이스로 빅데이터를 저장하기 위해서 사용한다.

문제10) HBase의 데이터 저장규칙은 무엇인가? : NoSQL로 분류되는데, 많은 NoSQL 솔류션들이 그렇듯이 스키마 변경없이 자유롭게 데이터를 저장 할 수 있다.

문제11) Hive를 사용하는 이유는 무엇인가? : Hive를 사용하면 데이터의 요약, 쿼리 및 분석을 수행할 수 있습니다.

문제12) Hive는 어떤 언어를 사용하여야 하는가? : 암시 적으로 MapReduce 또는 Tez 또는 Spark 작업으로 변환되는 SQL 유사 쿼리 (HiveQL).

문제13) Mahout 은 어떤 역할을 수행하는가? : Apache Software Foundation(ASF)에서 오픈소스로 제공하는 기계학습 Framework 이다.

문제14) Mahout 의 기능 세가지는 무엇인가? : 협업필터링, 클러스터링, 분류

문제15) Pig는 어떤 역할을 수행하는가? : 대용량 데이터셋을 좀 더 고차원적으로 다루기 위한 스크립트 언어

문제16) Pig는 적은양의 데이터를 처리하는데도 효율적인가? : 대용량 데이터셋에 적합하므로 적은양의 데이터를 쿼리하면 비효율적임

문제17) Spark의 역할은 무엇인가? : 빅데이터 처리에 주로 사용되는 분산처리 시스템

문제18) Spark의 구성요소는 무엇인가? : 스파크는 RDD라고 불리는 자바가상머신 객체들의 집합으로 구성되어있다.

문제19) Tez는 무엇인가? : 하둡의 YARN 기반 데이터 처리 응용 프로그램을 위한 프레임 워크

문제20) Tez는 사용자에게 복잡한 계산을 쉽게 보여주기 위해 어떤 방법을 사용하는가? : 데이터 흐름 그래프

문제21) Zookeeper란 무엇인가? : 분산처리 시스템에서 일괄적으로 관리해주는 시스템을 뜻합니다.

문제22) Zookeeper는 서버를 효율적으로 관리하기위해 어떤 기능을 수행하는가? : 하나의 서버에만 서비스가 집중되지 않도록, 서비스를 알맞게 분산하여 동시에 처리하게 함