(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2017년03월23일
(11) 등록번호 10-1719399
(24) 등록일자 2017년03월17일

(51) 국제특허분류(Int. Cl.)
G06F 17/30 (2006.01)
(21) 출원번호 10-2014-0132335
(22) 출원일자 2014년10월01일
심사청구일자 2016년09월02일
(65) 공개번호 10-2015-0039118
(43) 공개일자 2015년04월09일
(30) 우선권주장
14/043,753 2013년10월01일 미국(US)
(56) 선행기술조사문헌
US07984043 B1
KR1020140112427 A
최성진 외 3명, "하둡 분산 파일 시스템을 위한 효율적인 데이터 분산 저장 기법", 2011 한국컴퓨터종합학술대회 논문집 Vol 38, No. 1, 2011
JP2009211154 A

(73) 특허권자
크라우데라, 인크.
미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2
(72) 발명자
코르네커, 마셀
미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2
에릭슨, 저스틴
미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2
(뒷면에 계속)
(74) 대리인
강명구, 김현석

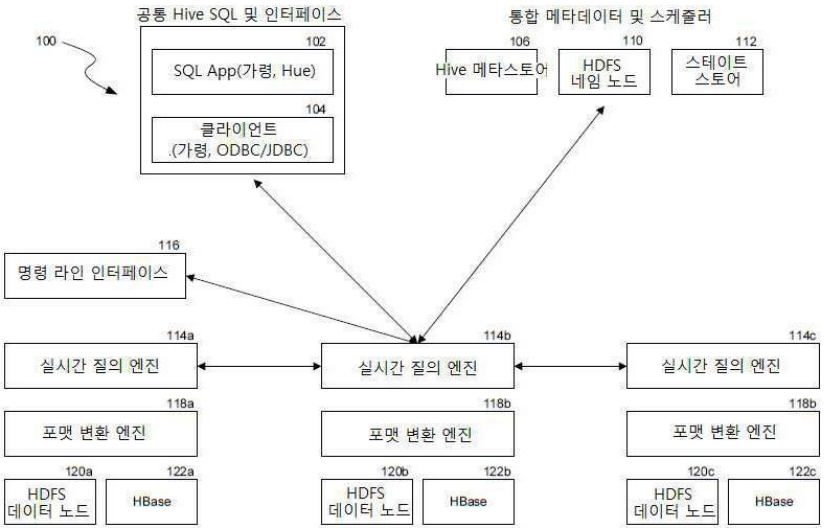
전체 청구항 수 : 총 15 항 심사관 : 경연정

(54) 발명의 명칭 **하둡에서의 개선된 SQL-형 질의를 위한 백그라운드 포맷 최적화**

(57) 요 약

저-대기시간(LL) 질의 엔진에 의해 사용하기 위해 소정 시점에서 데이터를 원본 포맷으로부터 데이터베이스형 포맷으로 포맷 변환하는 아파치 하둡(Apache Hadoop)용 포맷 변환 엔진이 개시된다. 포맷 변환 엔진은 하둡 클러스터 내 각각의 데이터 노드 상에 설치되는 데먼을 포함한다. 데먼은 스케줄러 및 컨버터를 포함한다. 스케줄러는 포맷 변환을 수행할 시기를 결정하고, 시간이 될 때 컨버터에 알린다. 컨버터는 저-대기시간(LL) 질의 엔진에 의해 사용하기 위해 데이터 노드 상에서 데이터를 원본 포맷으로부터 데이터베이스형 포맷으로 포맷 변환한다.

대 표 도



(72) 발명자

리, 농

미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2

커프, 레니

미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2

로빈슨, 헨리 노엘

미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2

초이, 앨런

미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2

벰, 앨렉스

미국, 캘리포니아 94304, 팔로 알토, 1001 페이지 밀 로드 빌딩 2

명 세 서

청구범위

청구항 1

하둡(Hadoop™) 분산 컴퓨팅 클러스터에 저장된 데이터에 대한 질의를 수행하기 위한 시스템으로서, 상기 시스템은 클라이언트로부터 수신된 질의에 대해 피어-투-피어 네트워크를 형성하는 복수의 데이터 노드를 포함하며,

복수의 데이터 노드의 각각의 데이터 노드는 피어-투-피어 네트워크의 피어(peer)로 기능하며, 하둡 클러스터의 구성요소와 대화할 수 있고, 각각의 피어는 메모리에서 실행되는 질의 엔진의 하나의 인스턴스를 가지며, 질의 엔진의 각각의 인스턴스는

클라이언트로부터의 질의를 구문 분석(parse)하고, 데이터 노드에서의 변환된 데이터의 가용성 여부를 기초로 질의 조각(query fragment)을 선택적으로 생성하도록 구성된 질의 플래너(query planner) - 상기 변환된 데이터는 원본 포맷(original format)에서 스키마(schema)에 의해 특정된 목표 포맷(target format)으로 변환되는 질의 연관 데이터이며, 질의는 질의를 수신한 어느 데이터 노드에 의해서라도 처리됨 - ,

질의 조각을 복수의 데이터 노드에 분산시키도록 구성된 질의 코디네이터(query coordinator), 및

질의 실행 엔진

을 가지며, 상기 질의 실행 엔진은

스키마를 기초로, 질의 조각이 생성될 때의 포맷에 대응하는 어느 로컬 데이터라도 인-메모리 튜플(in-memory tuple)로 변환시키도록 구성된 변환 모듈(transformation module), 및

인-메모리 튜플 상의 질의 조각을 실행시켜, 질의 조각을 수신한 타 데이터 노드로부터 중간 결과(intermediate result)를 얻고 상기 중간 결과를 클라이언트를 위해 집성하도록 구성된 실행 모듈

을 포함하는, 질의를 수행하기 위한 시스템.

청구항 2

제1항에 있어서, 상기 목표 포맷은 칼럼나 포맷(columnar format)인, 질의를 수행하기 위한 시스템.

청구항 3

제1항에 있어서, 상기 목표 포맷은 관계 데이터베이스 프로세싱을 위해 최적화되는, 질의를 수행하기 위한 시스템.

청구항 4

제1항에 있어서, 상기 변환된 데이터가 가용할 때, 질의 조각은 목표 포맷에 대해 생성되는, 질의를 수행하기 위한 시스템.

청구항 5

제1항에 있어서, 상기 변환된 데이터가 가용하지 않을 때, 질의 조각은 원본 포맷에 대해 생성되는, 질의를 수행하기 위한 시스템.

청구항 6

하둡(Hadoop™) 분산 컴퓨팅 클러스터 시스템에서 저장된 데이터에 대한 질의를 수행하기 위한 방법으로서, 상기 방법은

클라이언트로부터 수신된 질의에 대해 피어-투-피어 네트워크를 형성하는 복수의 데이터 노드를 구성하는 단계 - 복수의 데이터 노드의 각각의 데이터 노드는 피어-투-피어 네트워크에서의 하나의 피어로서 기능하며 하둡 클러스터의 구성요소와 대화할 수 있고, 각각의 피어는 메모리 내에서 실행되는 질의 엔진의 하나의 인스턴스를 가짐 - , 및

상기 질의 엔진의 각각의 인스턴스를,

클라이언트로부터의 질의를 구문 분석(parse)하고, 데이터 노드에서의 변환된 데이터의 가용성 여부를 기초로 질의 조각(query fragment)을 선택적으로 생성하도록 구성된 질의 플래너(query planner) - 상기 변환된 데이터는 원본 포맷(original format)에서 스키마(schema)에 의해 특정된 목표 포맷(target format)으로 변환되는 질의 연관 데이터이며, 질의는 질의를 수신한 어느 데이터 노드에 의해서라도 처리됨 - ,

질의 조각을 복수의 데이터 노드에 분산시키도록 구성된 질의 코디네이터(query coordinator), 및

질의 실행 엔진

을 포함하도록 구성하는 단계

를 포함하며, 상기 질의 실행 엔진은

스키마를 기초로, 질의 조각이 생성될 때의 포맷에 대응하는 어느 로컬 데이터라도 인-메모리 튜플(in-memory tuple)로 변환시키고,

인-메모리 튜플 상의 질의 조각을 실행시켜, 질의 조각을 수신한 타 데이터 노드로부터 중간 결과(intermediate result)를 얻고 상기 중간 결과를 클라이언트를 위해 집성하는,

질의를 수행하기 위한 방법.

청구항 7

제6항에 있어서, 상기 목표 포맷은 칼럼나 포맷(columnar format)인, 질의를 수행하기 위한 방법.

청구항 8

제6항에 있어서, 상기 목표 포맷은 관계 데이터베이스 프로세싱을 위해 최적화되는, 질의를 수행하기 위한 방법.

청구항 9

제6항에 있어서, 상기 변환된 데이터가 가용할 때, 질의 조각은 목표 포맷에 대해 생성되는, 질의를 수행하기 위한 방법.

청구항 10

제6항에 있어서, 상기 변환된 데이터가 가용하지 않을 때, 질의 조각은 원본 포맷에 대해 생성되는, 질의를 수행하기 위한 방법.

청구항 11

비일시적(non-transitory) 컴퓨터 판독형 매체에 있어서, 상기 매체는, 하나 이상의 프로세서에 의해 실행될 때, 하둡(Hadoop™) 분산 컴퓨팅 클러스터 시스템으로 하여금, 상기 시스템에서 저장된 데이터에 대한 질의를 수행하기 위한 방법을수행하게 하는 복수의 명령을 저장하며, 상기 방법은

클라이언트로부터 수신된 질의에 대해 피어-투-피어 네트워크를 형성하는 복수의 데이터 노드를 구성하는 단계 - 복수의 데이터 노드의 각각의 데이터 노드는 피어-투-피어 네트워크에서의 하나의 피어로서 기능하며 하둡 클러스터의 구성요소와 대화할 수 있고, 각각의 피어는 메모리 내에서 실행되는 질의 엔진의 하나의 인스턴스를 가짐 - , 및

클라이언트로부터의 질의를 구문 분석(parse)하고, 데이터 노드에서의 변환된 데이터의 가용성 여부를 기초로 질의 조각(query fragment)을 선택적으로 생성하며 - 상기 변환된 데이터는 원본 포맷(original format)에서 스키마(schema)에 의해 특정된 목표 포맷(target format)으로 변환되는 질의 연관 데이터이며, 질의는 질의를 수신한 어느 데이터 노드에 의해서라도 처리됨 - ,

질의 조각을 복수의 데이터 노드에 분산시키며,

스키마를 기초로, 질의 조각이 생성될 때의 포맷에 대응하는 어느 로컬 데이터라도 인-메모리 튜플(in-memory tuple)로 변환시키고,

인-메모리 튜플 상의 질의 조각을 실행시켜, 질의 조각을 수신한 타 데이터 노드로부터 중간 결과(intermediate result)를 얻고 상기 중간 결과를 클라이언트를 위해 집성하도록

상기 질의 엔진의 각각의 인스턴스를 구성하는 단계를 포함하는,

컴퓨터 판독형 매체.

청구항 12

제11항에 있어서, 상기 목표 포맷은 컬럼나 포맷(columnar format)인, 컴퓨터 판독형 매체.

청구항 13

제11항에 있어서, 상기 목표 포맷은 관계 데이터베이스 프로세싱을 위해 최적화되는, 컴퓨터 판독형 매체.

청구항 14

제11항에 있어서, 상기 변환된 데이터가 가용할 때, 질의 조각은 목표 포맷에 대해 생성되는, 컴퓨터 판독형 매체.

청구항 15

제11항에 있어서, 상기 변환된 데이터가 가용하지 않을 때, 질의 조각은 원본 포맷에 대해 생성되는, 컴퓨터 판독형 매체.

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

발명의 설명

기 술 분 야

[0001] 본 발명은 데이터 프로세싱 방법, 시스템, 및 기계-판독가능 매체에 관한 것이다.

배 경 기 술

[0002] 아파치 하둡(Aparch Hadoop) 프로젝트(이후 "하둡")는 상품 기계의 클러스터 간에 대형 데이터 세트의 신뢰가능하고 스케일링가능하며 분산되는 소프트웨어를 개발하기 위한 오픈-소스 소프트웨어 프레임웍이다. 하둡은 하둡 분산 파일 시스템(HDFS)으로 알려진, 분산형 파일 시스템을 포함한다. HDFS는 전체 하둡 클러스터에 걸치는 통합 파일 시스템을 형성하기 위해 로컬 노드 상의 파일 시스템들을 함께 연결한다. 하둡은 MapReduce로 알려진 프로그래밍 프레임웍에 의해 사용되는 잡 스케줄링 및 클러스터 리소스 관리를 위한 프레임웍을 제공하는 하둡 YARN을 또한 포함한다. 하둡은 아파치 Hive(이후 "하이브") 및 아파치 HBase(이후 "HBase")를 포함하는 다른 아파치 프로젝트에 의해 또한 보완된다. 하이브는 데이터 요약 및 ad hoc 질의를 제공하는 데이터 웨어하우스 인프러스트럭처다. HBase는 대형 테이블을 위한 구조화된 데이터 스토리지를 지원하는 스케일링가능한, 분산형 NoSQL(No Structured Query Language) 데이터베이스 또는 데이터 스토어다.

[0003] 하둡은 현재 관계 데이터베이스관리 시스템(RDBMS)을 지원하지 않는다. 관계 데이터베이스를 위해, 한 스키마 - 표의 칼럼 사이의 호환성을 보장하는 한 세트의 무결성 제약사항과 함께 특정 칼럼을 갖는 표 내에 데이터를 조

직 - 가 형성될 수 있다. 전형적인 RDBMS는 스키마-온-라이트 모델(schema-on-write model)을 구현하며, 데이터가 데이터베이스에 기록됨에 따라 데이터 상에서 스키마가 집행된다. 구체적으로, 데이터가 데이터베이스에 저장되기 전에 데이터가 무결성 제약사항을 이용하여 재조직 및 필터링된다. 스키마-온-라이트 모델은 알려진 질문에 대답하는데 적합하다. 과거에 알려지지 않은 질문에 답변할 필요가 있을 경우, 새 데이터를 캡처할 필요가 있다. 그러나, RDBMS는 스키마에 대응하는 새 데이터를 수용할 수 없다. 새 데이터를 수용하기 위해, 통상적으로 데이터베이스로부터 구 데이터가 삭제될 필요가 있고, 스키마가 수정될 필요가 있으며, 새 데이터가 분석되어 데이터베이스에 로딩될 필요가 있다. 추가적으로, 데이터 아키텍트는 통상적으로, RDBMS에 연결된 모든 시스템들이 업데이트된 스키마와 함께 작동함을 보장할 필요가 있다. 새 데이터를 수용하는 이러한 프로세스는 긴 시간이 걸릴 수 있다. 그때까지, 새 데이터는 앞서 알려지지 않은 질문에 답변하기 위해 캡처될 수 없다.

[0004] 다른 한편, 하둡은 스키마-온-리드 모델(schema-on-read model)을 따르며, 데이터가 데이터베이스로부터 판독될 때까지 데이터에 대해 스키마가 집행되지 않는다. 이러한 경우에, 스키마는 통상적으로 파일 포맷 측면에서 데이터의 조직화를 명시한다. 그 결과, 데이터의 처리는 데이터의 스토리지로부터 분리될 수 있다. 구체적으로, 하둡의 하위 스토리지 시스템은 원 포맷의 파일(예를 들어, 탭-경계 텍스트 파일, CSV, XML, JSON, 이미지, 등)을 취할 수 있으면서도, 관련 스키마를 후에 설계하여 별도로 저장할 수 있다. 질의에 대한 응답으로, 저장된 데이터는 별도로 저장된 스키마에 따라 메모리 내에서 변환(transformation)된다. 스키마-온-리드 모델 덕분에, 입력 데이터가 신속하게 데이터베이스에서 업데이트될 수 있고, 이는 서로 다른 스키마를 이용하여 실험하는 사용자를 돕는다.

발명의 내용

해결하려는 과제

[0005] 스키마-온-리드 모델 및 스키마-온-라이트 모델은 개개의 장점을 갖는다. 구체적인 필요성 및 요건에 따라 이들을 단독으로 또는 조합하여 이용하는 융통성을 하둡이 사용자에게 제공하는 것이 유용할 것이다.

도면의 간단한 설명

[0006] 도 1은 저-대기시간 질의 엔진 및 포맷 변환 엔진이 전개될 수 있는 예시적 환경의 도면.

도 2는 배치-배향(batch-oriented) 및 실시간 ad-hoc 질의를 지원하는 통합 플랫폼의 예시적 구성요소들을 나타내는 블록도.

도 3은 설치 매니저의 예시적 구성요소들을 나타내는 블록도.

도 4는 하둡 클러스터 내 각각의 데이터 노드 상에 설치되는 저-대기시간(LL) 질의 엔진 데몬(daemon)의 예시적 구성요소들을 나타내는 블록도.

도 5는 하둡 클러스터 내 각각의 데이터 노드 상에 설치되는 포맷 변환 엔진 데몬의 예시적 구성요소들을 나타내는 블록도.

도 6은 일련의 질의 조각들을 실행하기 전에 질의 실행 엔진의 예시적 작동을 나타내는 순서도.

도 7은 여기서 논의되는 방법들 중 하나 이상을 기계로 하여금 실행하게 하기 위한, 한 세트의 명령어를 실행할 수 있는 컴퓨터 시스템의 예시적 형태에서 기계의 도식적 표현.

발명을 실시하기 위한 구체적인 내용

[0007] 다음의 설명 및 도면은 예시적인 것으로서, 제한적으로 간주되어서는 안된다. 수많은 구체적 세부사항들이 발명의 완전한 이해를 돕기 위해 설명된다. 그러나, 소정의 예에서, 잘 알려진 또는 통상적인 세부사항은 설명의 본질이 흐려지는 것을 방지하기 위해 설명되지 않는다. 본 발명의 일 실시예에 대한 언급은 동일한 실시예에 대한 언급일 수 있으나, 반드시 그런 것은 아니며, 이러한 언급은 실시예 중 하나 이상을 의미한다.

[0008] 본 명세서에서 "일 실시예" 또는 "하나의 실시예"에 대한 언급은, 본 실시예와 연계하여 설명되는 특정 특징, 구조 또는 특성이 발명의 적어도 하나의 실시예에 포함됨을 의미한다. 명세서 내 다양한 위치에서 "일 실시예에서"라는 표현의 등장은 반드시 동일한 실시예를 언급하는 것이 아닐 뿐 아니라, 다른 실시예를 상호 배제하는 별도의 또는 대안의 실시예를 언급하는 것도 아니다. 더욱이, 일부 실시예에 의해 나타날 수 있고 다른 실시예에서는 나타나지 않을 수 있는 다양한 특징들이 설명된다. 마찬가지로, 일부 실시예에 대한 요건일 수 있고 다

른 실시예에 대해서는 해당없을 수 있는, 다양한 요건들이 설명된다.

[0009] 본 명세서에 사용되는 용어는 발명의 범주 내에서, 그리고 각각의 용어가 사용되는 구체적 범주 내에서, 당 분야의 통상적인 의미를 갖는 것이 일반적이다. 발명의 설명에 사용되는 소정의 용어가 아래에서 또는 명세서 내 다른 부분에서 논의되어, 발명의 설명에 관해 실시자들에게 추가적인 안내를 제공할 수 있다. 편의상, 소정의 용어는, 예를 들어, 이탤릭체 및/또는 따옴표를 이용하여, 강조될 수 있다. 강조의 이용은 용어의 범위 및 의미에 전혀 영향을 미치지 않고, 용어의 범위 및 의미는 강조 여부에 관계없이, 동일 범주 내에서, 동일하다. 동일 사항이 2개 이상의 방식으로 언급될 수 있다.

[0010] 결과적으로, 대안의 언어 및 동의어가 여기서 논의되는 용어들 중 하나 이상을 위해 사용될 수 있고, 용어가 여기서 논의되거나 자세히 설명되는지 여부에 따라, 어떤 특별한 중요도가 주어지지 않는다. 소정의 용어에 대한 동의어가 제공된다. 하나 이상의 동의어의 구체적 설명이 다른 동의어의 이용을 배제하지 않는다. 여기서 논의되는 용어들의 예를 포함한, 본 명세서 내 임의의 위치에서 예의 이용은 예시적일 뿐이며, 예시되는 용어의 또는 발명의, 범위 및 의미를 추가적으로 제한하고자 함이 아니다. 마찬가지로, 발명은 본 명세서에서 주어지는 다양한 실시예에 제한되지 않는다.

[0011] 발명의 범위를 추가적으로 제한하려는 의도없이, 본 발명의 실시예에 따른 기기, 장치, 방법 및 관련 결과의 예들이 아래에서 주어진다. 제목 또는 부제는 독자의 편의를 위해 예시로 사용될 수 있고, 이는 어떤 방식으로도 발명의 범위를 제한해서는 안된다. 달리 명시하지 않을 경우, 여기서 사용되는 모든 기술적 및 과학적 용어는 발명이 속하는 분야에서 통상의 지식을 가진 자에 의해 통상적으로 이해되는 것과 동일한 의미를 갖는다. 상충시에, 정의를 포함하는 본 문서가 이를 조율할 것이다.

[0012] 본 발명의 실시예는 하둡용 포맷 변환 엔진을 포함한다. 본 발명의 실시예는 하둡에 저장된 데이터에 대한 포맷 변환(conversion)을 실시간 또는 준-실시간으로 수행하기 위한 시스템 및 방법을 또한 포함한다.

[0013] 일 실시예에서, 포맷 변환 엔진은 데이터를 쉽게 질의가능한 포맷으로 만듦으로써 고속 검색을 가능하게 하는 메커니즘을 제공한다. 다른 실시예에서, 포맷 변환 엔진은 데이터를 신속하게 업데이트할 수 있고 안정화된 데이터와 함께 효율적으로 작업할 수 있는 융통성을 사용자에게 제공한다.

[0014] 도 1은 저-대기시간(LL) 질의 엔진 및 포맷 변환 엔진이 전개될 수 있는 예시적 환경(100)을 나타내는 도면을 도시한다. 환경(100)은 하둡 클러스터를 포함하는 복수의 데이터 노드(120a-c)를 포함한다. 데이터 노드(120a-c) 중 일부는 HDFS만을 구동시킬 수 있고, 다른 데이터 노드는 HBase 영역 서버(122a-c)를 구동시킬 수 있다.

[0015] 환경(100)은 하둡 클러스터에 연결 및/또는 액세스하기 위한 API 및 기타 툴을 제공하는, 자바 데이터베이스 커넥티비티(JDBC) 클라이언트, 오픈 데이터베이스 커넥티비티(ODBC) 클라이언트, 등과 같은 클라이언트(104)를 포함한다. Hue와 같은 SQL 애플리케이션(102)은 하둡으로 하여금 질의 또는 잡을 실행시키고, HDFS를 브라우징하며, 워크플로를 생성하게 하는, 등등을 위한 사용자 인터페이스를 제공한다. 환경(100)은 질의를 발급하기 위한 명령 라인 인터페이스(116)를 또한 포함한다. 일 실시예에서, 클라이언트(104), SQL 애플리케이션(102), 및 명령 라인 인터페이스(116)는 각각 또는 함께, 클라이언트로 불릴 수 있다.

[0016] 저-대기시간(LL) 질의 엔진 데먼(114a-c)은 각각의 데이터 노드 상에서 실행된다. 저-대기시간(LL) 질의 엔진 데먼은 질의를 조율하고 실행하는 긴 실행 프로세스(long running process)다. 각각의 저-대기시간(LL) 질의 엔진 데먼(114a-c)은 클라이언트(102/104)를 통해 수신되는 질의를 수신, 플랜, 및 조율할 수 있다. 예를 들어, 저-대기시간(LL) 질의 엔진 데먼은 병렬 실행을 위해 추가적인 저-대기시간(LL) 질의 엔진 데먼을 실행하는 원격 노드들 간에 분산되는 조각들로 질의를 나눌 수 있다. 질의는 HDFS(가령, 120a-c) 및/또는 HBase(가령, 122a-c) 상에서 직접 실행된다.

[0017] 포맷 변환 엔진 데먼(118a-c)은 각각의 데이터 노드 상에서 또한 실행된다. 포맷 변환 엔진 데먼(118a-c)은 칼럼나 포맷(columnar format) Parquet와 같이, 관련 데이터베이스 프로세싱에 더 좋은 요약 포맷(condensed format)으로 데이터를 원본 포맷으로부터 포맷 변환하는 긴 실행 프로세스(long running process)다. 포맷 변환은 하나 이상의 시점에서 수행될 수 있다. 포맷 변환된 데이터는 포맷 변환되지 않은 원본 데이터와 함께 데이터 노드 상에 저장되며, 두 데이터 모두 저-대기시간(LL) 질의 엔진에 모두 가용하다.

[0018] 환경(100)은 Hive 메타스토어(106), HDFS 네임 노드(110), 및/또는 스테이트 스토어(112)와 같은 통합 메타데이터 구성요소들을 더 포함한다. Hive 메타스토어(106)는 환경(100) 내의 다양한 엔진에 가용한 데이터에 관한 정보를 포함한다. 구체적으로, Hive 메타스토어(106)는 데이터 노드(120a-c) 상에 저장되는 데이터에 대한 스키마를 포함한다. HDFS 네임 노드(NN)(110)는 로컬 판독을 최적화시키기 위해 데이터 노드(120a-c) 간에 파일의 분

산의 세부사항을 포함한다. 일 구현예에서, 네임 노드(110)는 개별 노드 상에서, 파일이 안착하는 디스크 볼륨에 관한 정보를 포함할 수 있다.

[0019] 스테이트 스토어(112)는 클러스터 내 단일 노드 상에서 실행되는 전역 시스템 리포지터리(repositary)다. 스테이트 스토어(112)는 일 구현예에서, 네임 서비스로 사용될 수 있다. 모든 저-대기시간(LL) 질의 엔진 데먼은 시작 시에, 스테이트 스토어에 멤버임을 등록할 수 있고, 클러스터 상에서 실행되는 모든 저-대기시간(LL) 질의 엔진 데먼을 명시하는 기존 멤버십 정보를 얻을 수 있다. 스테이트 스토어(112)는, 추가적인 구현예에서, 질의 실행을 위한 메타데이터의 제공에 사용될 수 있다. 스테이트 스토어(112)는 메타데이터를 캐싱할 수 있고, 시작 시에 또는 다른 시기에, 저-대기시간(LL) 질의 엔진 데먼에 메타데이터를 분배할 수 있다. 스테이트 스토어가 고장날 때, 나머지 시스템이 스테이트 스토어로부터 수신한 마지막 정보에 기초하여 계속 작동할 수 있다. 추가적인 구현예에서, 스테이트 스토어는 하둡 클러스터의 기능 및/또는 성능을 개선시키는데 사용될 수 있는 로드 정보, 진단 정보, 등과 같은 다른 시스템 정보를 저장 및 분산시킬 수 있다.

[0020] 도 2는 배치-배향 및 실시간 ad-hoc 질의를 지원하는 통합 하둡 플랫폼(212)의 예시적 구성요소들을 나타내는 블록도를 도시한다. 통합 하둡 플랫폼(212)은 분산된 프로세싱 및 분산된 스토리지를 지원한다. 통합 하둡 플랫폼(212)은 사용자 인터페이스(214), 스토리지(220), 및 메타데이터(222) 구성요소를 포함한다. 사용자 인터페이스(214)는 ODBC 드라이버, JDBC 드라이버, Hue Beeswax, 등과 같은 Hive 인터페이스를 포함한다. 사용자 인터페이스(214)는 SQL 서포트를 또한 포함한다. 사용자 인터페이스(214)를 통해, 질의가 발급될 수 있고, 데이터가 스토리지(200)로부터 판독되거나 스토리지(200)에 기록될 수 있고, 등등이다. 스토리지(220)는 HDFS 및/또는 HBase 스토리지를 포함한다. HDFS는 텍스트 파일, 시퀀스 파일, RC 파일, Avro, 등을 포함한, 그러나 이에 제한되지 않는, 다양한 파일 포맷을 지원할 수 있다. 스내피, gzip, 디플레이트(deflate), bzip, 등을 포함하는 다양한 압축 코덱이 또한 지원될 수 있다. 메타데이터(222)는 예를 들어, 테이블, 그 파티션, 칼럼, 타입, 테이블/블록 위치, 등에 관한 정보를 포함할 수 있다. 메타데이터(222)는 HBase 테이블의 매핑을 포함하는 기존 Hive 메타스토어를 레버리지할 수 있고, 이러한 Hive 메타스토어는 시작/정지 로우에 매핑되는 로우 키(row key)에 대해 예측하고, 단일 칼럼 값 필터, 등에 매핑되는 다른 칼럼에 대해 예측한다.

[0021] 기존 하둡 플랫폼은 하둡 데이터의 배치 프로세싱(216)을 위해 배치-배향 질의 엔진(즉, MapReduce)을 이용한다. MapReduce의 배치 프로세싱 기능은 통합 하둡 플랫폼(2120 내 실시간 액세스 구성요소(218)에 의해 보완된다. 실시간 액세스 구성요소(218)는 저-대기시간을 위해 최적화된 분산형 저-대기시간(LL) 질의 엔진을 통해 통합 스토리지(220)에 대해 직접 실시간 ad hoc SQL 질의를 수행하게 한다. 따라서, 실시간 액세스 구성요소(218는 빅 데이터에 대한 질의 및 분석을 모두 지원할 수 있다.

[0022] 도 3은 통합 스토리지 계층 바로 위에 대화형 실시간 SQL 질의를 제공하기 위해 하둡 클러스터 내 다양한 엔진의 구성요소들을 설치하기 위한 설치 매니저(302)의 예시적 구성요소들을 나타내는 블록도다. 설치 매니저(302)는 다양한 엔진을 자동적으로 설치, 구성, 관리, 및 모니터링할 수 있다. 대안으로서, 엔진이 수동으로 설치될 수 있다. 설치 매니저(302)는 저-대기시간(LL) 질의 엔진 데먼(304), 스테이트 스토어 데먼(306), 저-대기시간(LL) 질의 엔진 쉘(308) 및 포맷 변환 엔진 데먼(310)을 포함한 4개의 바이너리를 설치한다. 상술한 바와 같이, 저-대기시간(LL) 질의 엔진 데먼(304)은 HDFS 및/또는 HBase 데이터에 대한 질의를 플랜 및 실행하는 서비스 또는 프로세스다. 이는 클러스터 내 각각의 데이터 노드 상에 설치된다. 포맷 변환 엔진 데먼은 원본 포맷으로부터 요약 포맷으로 데이터를 포맷 변환하는 서비스 또는 프로세스다. 이 또한 클러스터 내 각각의 데이터 노드 상에 설치된다. 스테이트 스토어 데먼(306)은 클러스터 내 모든 저-대기시간(LL) 질의 엔진 데먼의 위치 및 상태를 추적하는 네임 서비스다. 스테이트 스토어 데먼(306)는 또한 일부 구현예에서 메타데이터 및/또는 다른 진단 정보를 제공하기 위한 메타데이터 스토어일 수 있다. 저-대기시간(LL) 질의 엔진 쉘(308)은 저-대기시간(LL) 질의 엔진 데먼에 질의를 발급하기 위한 명령 라인 인터페이스이고, 클라이언트 상에 설치된다.

[0023] 도 4는 하둡 클러스터 내 각각의 데이터 노드 상에 설치되는 저-대기시간(LL) 질의 엔진 데먼의 예시적 구성요소들을 나타내는 블록도다. 저-대기시간(LL) 질의 엔진 데먼은 일 실시예에서 질의 플래너(316), 질의 코디네이터(query coordinator)(318), 및 질의 실행 엔진(320)을 포함한다.

[0024] 질의 플래너(316)는 저장된 스키마에 기초하여, 클라이언트로부터의 질의 요청을 일련의 플랜 조각으로 전환하고, 플랜 조각을 질의 코디네이터(318)에 제공한다. 질의 플래너(316)는 Hive 메타스토어, 스테이트 스토어, API, 등과 같은 하둡 환경의 나머지와 상호작용을 돕기 위해, 자바 또는 다른 적절한 언어로 기록한 저-대기시간(LL) 질의 엔진 데먼의 프론트엔드를 구성할 수 있다. 질의 플래너(316)는 질의 플랜을 구성하기 위해, Scan, HashJoin, HashAggregation, Union, TopN, Exchange, 등과 같은 다양한 오퍼레이터를 이용할 수 있다. 각각의

오퍼레이터는 소정의 방식으로 데이터를 구체화(materialization)하거나 발생시킬 수 있고 또는 데이터를 조합할 수 있다. 일 구현예에서, 예를 들어, 질의 플래너는 (가령, 수동으로 또는 옵티마이저를 이용하여) 하나 이상의 오퍼레이터의 좌측(lefty) 플랜 또는 트리를 생성할 수 있다. 스캔 오퍼레이터는 스캔 라인 또는 경계를 따라 플랜을 분리할 수 있다. 서로 다른 스토리지 매니저에 대해 특화된 스캔 노드들이 존재할 수 있다. 예를 들어, HDFS 스캔 노드 및 HBase 스캔 노드가 존재할 수 있고, 각각은 서로 다른 파일 포맷에 대해 서로 다른 프로세스를 내부적으로 이용할 수 있다. 일부 플랜은 해시 테이블을 채울 수 있는, 그리고 통합 결과를 출력할 수 있는, 해시 통합(hash aggregation)을 위해 데이터를 조합한다. 통합 오퍼레이터는 서로 다른 플랜 조각들로부터 출력을 병합할 수 있다. TopN 오퍼레이터는 한계 값을 가진 채로 순서 정렬의 등가치일 수 있다. 교환 오퍼레이터는 서로 다른 2개의 노드 상에서 구동되는 2개의 플랜 조각들 사이의 데이터 교환을 취급할 수 있다.

[0025] 질의 코디네이터(318)는 질의에 관련된 모든 저-대기시간(LL) 질의 엔진 데먼 간에 플랜 조각들의 실행을 개시한다. 질의 코디네이터(318)는 질의 플랜 조각을 실행하기 위한 저-대기시간(LL) 질의 엔진 데먼을 결정 또는 식별하기 위해, HDFS 네임 노드로부터 데이터 블록에 대한 위치 정보 및/또는 스테이트 스토어로부터 멤버십 정보를 이용한다. 일 구현예에서, 질의 코디네이터(318)는 질의로부터 임의의 술어(predicates)를 또한 적용하여, 플랜 조각이 구동되어야 하는 한 세트의 파일 및 블록까지 좁혀져 내려간다. 질의 코디네이터(318)는 원격 데이터 노드 상의 저-대기시간(LL) 질의 엔진 데먼으로부터 데이터의 최종 통합 또는 병합을 또한 수행할 수 있다. 일 구현예에서, 저-대기시간(LL) 질의 엔진 데먼은 데이터의 일부를 미리 통합할 수 있고, 따라서, 데이터 노드 간에 통합을 분배하고 질의 프로세싱을 가속시킨다.

[0026] 질의 실행 엔진(320)은 HDFS 및 HBase 상에서 국부적으로, 플래닝된 질의 조각들을 실행한다. 예를 들어, 스캔 및/또는 그외 다른 질의 오퍼레이터를 구동한다. 질의 실행 엔진(320)은 C++로 기록되지만, 자바와 같은, 다른 적절한 언어로 또한 기록될 수 있다. 질의 실행 엔진(320)은 MapReduce로부터 분리된 실행 엔진이다. 질의 실행 엔진(320)이 데이터(가령, HDFS 및 HBase)를 제공하는 인프러스트럭처에 액세스할 때, 잡 트래커(job tracker) 및 태스크 트래커(task tracker)와 같이, 맵 감소를 지원하는 인프러스트럭처를 이용하지 않는다.

[0027] 일 실시예에서, 최초에, 데이터가 들어와, HDFS 데이터 노드 상의 원본 포맷에 저장된다. 사용자 또는 관리자에 의해 생성될 수 있는, 데이터가 저장되는 파일 포맷에 대한 정보를 포함하는 하나 이상의 연관 스키마가, 데이터 저장과 동시에 또는 나중에, Hive 메타스토어(106)에 별도로 저장된다. 일 실시예에서, 질의가 제출된 후, 소정의 플래닝된 질의 조각을 실행해야할 질의 실행 엔진(320)이 스키마에 따라 데이터 노드 상의 파일을 국부적으로 우선 변환시킨다. 구체적으로, 질의 실행 엔진(320)은 스키마를 판독하고, 이러한 스키마는 예를 들어, Hive 메타스토어로부터 파일을 위한, 로우 및 칼럼 엔딩에 대한 정보를 지닌다. 그 후 데이터 노드로부터 파일을 판독하고, 스키마에 명시된 파일 포맷에 따라 파일을 분석하며, 스키마 내 추가 정보에 따라 분석된 데이터를 일련의 인-메모리 튜플로 변환한다. 이 때, 질의 실행 엔진(320)은 변환 결과에 대해 국부적으로, 플래닝된 질의 조각을 실행할 준비가 되어 있다.

[0028] 일 실시예에서, 질의 실행 엔진(320)은 중앙 처리 유닛(CPU)에 의해 효율적으로 실행될 수 있는 포맷으로 해석 코드를 변환하기 위해, 런-타임 코드 발생을 위해, 로우 레벨 가상 기계(LLVM)(322), 옵티마이저, 또는 다른 컴파일러 인프러스트럭처를 포함할 수 있다. 전형적인 RDBMS는, 예를 들어, 인덱스 등으로부터 데이터를 추출하기 위해 수식 평가를 위한 해석 코드를 갖는다. 질의 실행 엔진(320)은 코드를 하드웨어와 더욱 밀접하게 연결하기 위해 로우 레벨 가상 기계(LLVM)를 이용함으로써 이 문제를 취급한다. 예를 들어, 일 질의에서 A = B/(A+B) = C라는 수식은 3개의 함수 호출을 통해 평가될 수 있다. 3개의 함수 호출을 행하는 대신에, LLVM은 수식 평가 및 속도 이득 실현을 위해 CPU가 제공하는 연산을 이용한다.

[0029] 추가 실시예에서, 저-대기시간(LL) 질의 엔진 데먼은 예를 들어, 텍스트 프로세싱 및/또는 다른 리소스-집약적 프로세스를 수행하기 위해, 전용 CPU 명령어를 또한 이용할 수 있다. 다른 예를 들자면, 해시 값 연산은 전용 사이클릭 리던던시 체크(CRC32) 명령어를 이용하여 속도 이득을 실현하도록 수행될 수 있다.

[0030] 일 실시예에서, 저-대기시간(LL) 질의 엔진은 Hive 및 MapReduce의 기존 배치 프로세싱 프레임웍을 이용하여 가능한 것보다 더 빠른 속도로 사용자가 많은 볼륨의 데이터를 질의할 수 있고 답변을 얻을 수 있는 저-대기시간의 장점을 제공한다. 추가 실시예에서, 실시간 질의 엔진은 많은 볼륨의 데이터에서 감추어진 인사이트(hidden insights)를 검색하는데 사용되는 스키마를 적용함에 있어서 융통성을 제공한다.

[0031] 하나의 질의 실행 엔진이 서로 다른 파일 포맷으로 데이터를 분석 및 변환하는데 서로 다른 시간이 걸린다. 일반적으로, 파일 포맷이 SQL-형 질의에 대한 응답으로 관련 데이터베이스 프로세싱에 더 좋을 때, 시간이 감소한다. 따라서, 포맷 변환 엔진은 런타임 시에 질의 프로세싱의 효율을 증가시키기 위해 백그라운드에서 데이터를

이러한 파일 포맷으로 포맷 변환한다. 도 5는 하둡 클러스터 내 각각의 데이터 노드 상에 설치되는 포맷 변환 엔진 데먼의 예시적 구성요소들을 나타내는 블록도다. 일 실시예에서, 포맷 변환 엔진 데먼은 스케줄러(412) 및 컨버터(414)를 포함한다. 스케줄러(412)는 관리자 또는 사용자에 의한 입력에 기초하여 포맷 변환을 수행할 시기를 결정하고, 시간이 되었을 때 컨버터에 알린다. 일 예에서, 스케줄러(412)는 주기적으로 또는 차후 소정의 시기에 포맷 변환을 수행하기 위해 타이머를 이용한다. 차후의 소정의 시기는 생성, 데이터의 최초 업데이트, 또는 데이터의 최종 업데이트와 같이, 이벤트의 발생으로부터 측정될 수 있다. 다른 예에서, 포맷 변환은 데이터가 업데이트되었을 때, 검색되었을 때, 동일 질의로 검색되었을 때, 등에 소정의 시간 동안 수행된다. 따라서, 스케줄러(412)는 업데이트의, 모든 질의의, 특정 질의의, 구분된 질의의, 등의 총 개수의 카운터를 유지하여, 이러한 수치를 포함한 기준에 부합될 때 포맷 변환이 수행될 수 있다. 추가적인 예에서, 데이터 노드에 대한 리소스 이용의 상태는, 포맷 변환의 스케줄링시 고려된다.

[0032] 일 실시예에서, 스케줄러(412)는 데이터 노드 상의 데이터의 각 조각, 각각의 원본 포맷, 각각의 목표 포맷, 각 쌍의 원본 포맷 및 표적 포맷, 등에 대해 하나씩 스케쥴을 관리한다. 다른 실시예에서, 스케줄러(412)는 포맷 변화를 수행할 시기의 결정과 유사할 수 있는, 데이터 노드로부터 포맷 변환 결과를 삭제할 시기를 결정하고, 시간이 될 때 컨버터(414)에 알린다. 포맷 변환 엔진 데먼의 스케줄러(412)가 독립적으로 작용할 수 있지만, 복수의 데이터 노드에 걸쳐 또는 심지어 전체 클러스터에 걸쳐 계통적 방식으로 포맷 변환을 수행하기 위해 다른 포맷 변환 엔진 데먼의 스케줄러와 협력할 수도 있다.

[0033] 컨버터(414)는 스케줄러(412)로부터 통지를 수신하면 포맷 변환을 수행한다. 일 실시예에서, 컨버터(414)는 하나 이상의 목표 포맷의 리스트를 유지한다. 컨버터는 관리자 또는 사용자에 의한 입력에 기초하여 데이터 노드 상의 데이터를 목표 포맷의 데이터로 포맷 변환하고, 데이터 노드 상에 포맷 변환된 데이터를 원본 데이터와 함께 저장한다. 예를 들어, 컨버터(414)는 데이터 노드로부터 CSV 포맷의 파일을 메모리로 판독하고, CSV 포맷에 따라 파일을 분석하며, 이를 선택된 Parquet 포맷으로 포맷 변환하고, CSV 포맷의 파일과 함께 데이터 노드 상에 Parquet 포맷의 파일을 저장한다. 일 실시예에서, 포맷 변환은 가능하다면, Hive 메타스토어에 저장된 구체적 스키마에 기초하여, 소정의 원본 포맷과 목표 포맷 사이에서 완전하게 자동화될 수 있다. 예를 들어, CSV 파일 내 모든 필드는 Parquet 파일 내 일 칼럼으로 자동적으로 포맷 변환될 수 있다. 포맷 변환은 입력 파일을 동일한 목표 포맷 또는 서로 다른 목표 포맷의 복수의 출력 파일로 포맷 변환함을 결정할 수 있는 관리자 또는 사용자에 의해 맞춤화될 수 있고, 각각의 출력 파일은 예를 들어, 특정 순서로 배열되는 입력 파일의 선택 필드들을 갖는다. 다른 실시예에서, 컨버터(414)는 스케줄러(412)로부터 통지를 수신할 때, 소정의 포맷 변환 결과를 또한 삭제한다.

[0034] 목표 포맷이 통상적으로 관계 데이터베이스 프로세싱에 좋은 요약 포맷이기 때문에, 목표 포맷 준비가 된 데이터를 갖는 것은 SQL-형 질의의 프로세싱을 가속시킨다. 포맷 변환이 백그라운드에서 조심스럽게 선택된 시점에서 수행되기 때문에, 데이터 노드 상의 다른 작동과의 간섭 및 리소스 이용을 최소화시키는 경향이 있다.

[0035] 포맷 변환 엔진 데먼을 이용하여, 일 실시예에서, 질의가 제출된 후, 질의 플래너가 플랜 조각을 설정하여, 포맷 변환된 데이터가 가용함을 표시할 것이다. 데이터 노드 상의 질의 실행 엔진은 더 이상 데이터 노드 상에서 데이터의 복잡한 변환을 수행할 필요가 없다. 데이터 노드로부터 포맷 변환된 데이터를 단순히 판독할 수 있고, 이는 본질적으로 튜플 형태일 것이다. 포맷 변환 엔진 데먼은 따라서, 데이터가 업로드 및 업데이트될 때 긴 프로세싱 시간을 필요로하는 모델의 일부 비용에 대한 문제점없이, 데이터가 질의 프로세싱에 사용될 때 프로세싱 시간을 감소시킴으로써 스키마-온-라이트 모델의 일부 이점을 제공한다.

[0036] 도 6은 포맷 변환 엔진의 존재시 질의 플래닝 및 실행의 예시적 작동을 나타내는 순서도다. 단계(602)에서, 질의 플래너가 질의를 수신한다. 단계(603)에서, 질의 플래너는 데이터 저장을 위한 가용 파일 포맷을 식별하기 위해 관련 스키마 정보를 리뷰한다. 그러나 포맷 변환된 목표 포맷의 데이터가 또한 가용할 경우, 단계(606)에서, 질의 플래너는 목표 포맷을 위한 목표 조각을 형성한다.

[0037] 일련의 플래닝된 질의 조각을 수신하면, 단계(608)에서, 데이터 노드 상의 질의 실행 엔진이 데이터 노드로부터 적절한 파일 포맷의 데이터를 판독한다. 단계(610)에서, 질의 실행 엔진은 스키마 정보에 따라 데이터를 일련의 인-메모리 튜플(a series of in-memory tuples)로 변환한다. 단계(612)에서, 질의 실행 엔진은 인-메모리 튜플을 이용하여 일련의 플래닝된 질의 조각들을 실행한다. 이러한 특징들 덕분에, 사용자는 데이터세트로부터 특정 인사이트를 효율적인 방식으로 추출가능하면서 데이터 업로드 및 업데이트시 큰 오버헤드 발생없이 서로 다른 구조를 갖는 데이터세트로 실험할 수 있는 융통성을 부여받는다.

[0038] 도 7은 여기서 논의되는 방법들 중 하나 이상을 기계로 하여금 수행하게 하기 위한, 한 세트의 명령어를 실행할

수 있는 컴퓨터 시스템 형태의 기계의 도식적 표현을 도시한다.

[0039] 도 7의 예에서, 컴퓨터 시스템(700)은 프로세서, 메모리, 비휘발성 메모리, 및 인터페이스 장치를 포함한다. 다양한 공통 구성요소(가령 캐시 메모리)가 예시적 단순화를 위해 생략되었다. 컴퓨터 시스템(700)은 도 1의 예에 묘사되는 구성요소들(및 본 명세서에서 설명되는 다른 구성요소들)을 구현할 수 있는 하드웨어 장치를 예시하도록 구성된다. 컴퓨터 시스템(700)은 임의의 적용가능한 알려진 또는 편의적인 형태의 것일 수 있다. 컴퓨터 시스템(700)의 구성요소들은 버스를 통해 또는 그외 다른 알려진 또는 편의적 장치를 통해 함께 연결될 수 있다.

[0040] 프로세서는 예를 들어, 인텔 펜티엄 마이크로프로세서 또는 모토롤라 파워 pc 마이크로프로세서와 같은 기존의 마이크로프로세서일 수 있다. 당 업자는 "기계-판독가능 (저장) 매체" 또는 "컴퓨터-판독가능 (저장) 매체"라는 용어가, 프로세서에 의해 액세스가능한 임의의 타입의 장치를 포함함을 이해할 것이다.

[0041] 메모리는 예를 들어, 버스에 의해, 프로세서에 연결된다. 메모리는 예를 들자면, 그러나 제한없이, 동적 RAM(DRAM) 및 정적 RAM(SRAM)과 같은 랜덤 액세스 메모리(RAM)를 포함할 수 있다. 메모리는 로컬 타입, 원격 타입, 또는 분산형일 수 있다.

[0042] 버스는 비휘발성 메모리 및 구동 유닛에 프로세서를 또한 연결한다. 비휘발성 메모리는 종종 자기 플라피 디스크 또는 하드 디스크, 자기-광학 디스크, 광학 디스크, 읽기-전용 메모리(ROM), 예를 들어, CD-ROM, EPROM, 또는 EEPROM, 자기 또는 광학 카드, 또는 다른 형태의 대량 데이터 스토리지다. 이 데이터 중 일부가 컴퓨터(800) 내 소프트웨어의 실행 중 종종 직접 메모리 액세스 프로세스에 의해 메모리에 기록된다. 비휘발성 스토리지는 로컬형, 원격형 또는 분산형일 수 있다. 비휘발성 메모리는 선택적인 사항인데 왜냐하면, 시스템이 메모리 내 가용한 모든 적용가능 데이터로 생성될 수 있기 때문이다. 전형적인 컴퓨터 시스템은 통상적으로 적어도 프로세서, 메모리, 및 프로세서에 메모리를 연결하는 장치(가령, 버스)를 포함할 것이다.

[0043] 소프트웨어는 통상적으로 비휘발성 메모리 및/또는 구동 유닛에 저장된다. 게다가, 대형 프로그램의 경우, 메모리에 전체 프로그램을 저장하는 것이 심지어 가능하지 않을 수 있다. 그럼에도 불구하고, 소프트웨어의 실행을 위해, 필요할 경우, 소프트웨어가 프로세싱을 위해 적절한 컴퓨터 판독가능 위치로 이동하며, 예시적 용도로, 이 위치가 본 문헌에서 메모리로 불린다. 소프트웨어가 실행을 위해 메모리로 이동할 때에도, 프로세서는 통상적으로, 소프트웨어와 연관된 값을 저장하기 위해 하드웨어 레지스터와, 이상적으로, 실행을 가속시키는 기능을 하는 로컬 캐시를 이용할 것이다. 여기서 사용되는 바와 같이, 소프트웨어 프로그램이 "컴퓨터-판독가능 매체에서 구현되는" 것으로 언급될 때 (비휘발성 스토리지로부터 하드웨어 레지스터로) 임의의 알려진 또는 편리한 위치에 소프트웨어 프로그램이 저장된다고 가정한다. 프로세서는, 프로그램과 연관된 적어도 하나의 값이 프로세서에 의해 판독가능한 레지스터에 저장될 때 "프로그램을 실행하도록 구성된다"고 간주된다.

[0044] 버스는 프로세서를 네트워크 인터페이스 장치에 또한 연결한다. 인터페이스는 모뎀 또는 네트워크 인터페이스 중 하나 이상을 포함할 수 있다. 모뎀 또는 네트워크 인터페이스는 컴퓨터 시스템의 일부분으로 간주될 수 있다. 인터페이스는 아날로그 모뎀, ISDN 모뎀, 케이블 모뎀, 토큰 링 인터페이스, 위성 전송 인터페이스(가령, "다이렉트 PC"), 또는 컴퓨터 시스템을 다른 컴퓨터 시스템에 연결하기 위한 다른 인터페이스를 포함할 수 있다. 인터페이스는 하나 이상의 입력 및/또는 출력 장치를 포함할 수 있다. I/O 장치는 예를 들자면, 그러나 제한없이, 키보드, 마우스 또는 다른 포인팅 장치, 디스크 드라이브, 프린터, 스캐너, 및 다른 입력 및/또는 출력 장치(디스플레이 포함)를 포함할 수 있다. 디스플레이 장치는 예를 들자면, 그러나 제한없이, 음극관(CRT), 액정 디스플레이(LCD), 또는 그외 다른 적용가능한 알려진 또는 편리한 디스플레이 장치를 포함할 수 있다. 단순화를 위해, 도 8의 예에 묘사되지 않은 장치들의 컨트롤러가 이러한 인터페이스에 존재한다고 가정한다.

[0045] 작동시, 컴퓨터 시스템(800)은 디스크 운영 체제와 같은 파일 관리 시스템을 포함하는 운영 체제 소프트웨어에 의해 제어될 수 있다. 관련 파일 관리 시스템 소프트웨어를 갖는 운영 체제 소프트웨어의 일례는 미국, 워싱턴주, Redmond에 소재한 마이크로소프트사의 Windows®로 알려진 운영 체제 계열이다. 관련 파일 관리 시스템 소프트웨어를 갖는 운영 체제 소프트웨어의 다른 예는 리눅스 운영 체제 및 관련 파일 관리 시스템이다. 파일 관리 시스템은 통상적으로 비휘발성 메모리 및/또는 구동 유닛에 저장되고, 프로세서로 하여금 운영 체제에 의해 요구되는 다양한 작용들을 실행하게 하여, 데이터를 입/출력하고 데이터를 메모리에 저장하며, 비휘발성 메모리 및/또는 구동 유닛 상에 파일을 저장할 수 있다.

[0046] 상세한 설명 중 일부분은 컴퓨터 메모리 내 데이터 비트에 대한 작동의 부호 표현 및 알고리즘 측면에서 제시될 수 있다. 이러한 알고리즘 형태의 설명 및 표현은 당 업계의 다른 자들에게 작업의 본질을 가장 효과적으로 전달하기 위해 데이터 프로세싱 분야의 숙련자에 의해 사용되는 수단이다. 알고리즘은, 여기서, 일반적으로, 요망

결과를 유도하는 작동들의 일관성있는 시퀀스로 인식된다. 작동은 물리적 양의 물리적 조작을 필요로하는 작동이다. 통상적으로, 반드시 그런 것은 아니지만, 이러한 양은 저장, 전송, 조합, 비교, 및 그렇지 않을 경우 조작될 수 있는 전기 또는 자기 신호의 형태를 취한다. 주로 공통 사용의 이유로, 이러한 신호들을 비트, 값, 요소, 심벌, 문자, 용어, 번호, 등으로 호칭하는 것이, 가끔은 편리하다는 것이 입증되었다.

[0047] 그러나, 모든 이러한 그리고 유사한 용어들은 적절한 물리적 양과 연관되어야 하고 이러한 양에 적용되는 편리한 라벨일 뿐임을 명심하여야 한다. 다음의 논의로부터 명백해지듯이 달리 구체적으로 언급하지 않을 경우, 설명 전체를 통해, "프로세싱" 또는 "컴퓨팅" 또는 "결정" 또는 "디스플레이", 등과 같은 용어를 이용하는 논의는, 컴퓨터 시스템의 레지스터 및 메모리 내의 물리적(전자적) 양으로 표현되는 데이터를, 컴퓨터 시스템 메모리 또는 레지스터 또는 다른 이러한 정보 스토리지, 전송, 또는 디스플레이 장치 내의 물리적 양으로 유사하게 표현되는 다른 데이터로 변환 및 조작하는, 컴퓨터 시스템 또는 유사 전자 컴퓨팅 장치의 작용 및 프로세스를 의미한다.

[0048] 여기서 제시되는 알고리즘 및 디스플레이는 어떤 특별한 컴퓨터 또는 다른 장치에 내재적으로 관련되는 것이 아니다. 여기서의 가르침에 따라 다양한 범용 시스템이 사용될 수 있고, 또는, 일부 실시예의 방법을 수행하기 위해 더욱 전용화된 장치를 구성하는 것이 편리함을 입증할 수 있다. 다양한 이러한 시스템들에 대해 요구되는 구조는 아래의 설명으로부터 나타날 것이다. 추가적으로, 그 기술들은 어떤 특정 프로그래밍 언어를 참조하여 설명되는 것이 아니며, 따라서, 다양한 실시예가 다양한 프로그래밍 언어를 이용하여 구현될 수 있다.

[0049] 대안의 실시예에서, 기계는 독립형 장치로 작동하고, 또는, 다른 기계에 연결(가령, 네트워크화)될 수 있다. 네트워크화 전개에서, 기계는 클라이언트-서버 네트워크 환경에서 서버 또는 클라이언트 기계의 용량으로, 또는, 피어-투-피어(또는 분산형) 네트워크 환경에서 피어 기계로 작동할 수 있다.

[0050] 기계는 서버 컴퓨터, 클라이언트 컴퓨터, 개인용 컴퓨터(PC), 태블릿 PC, 랩탑 컴퓨터, 셋-탑 박스(STB), 개인용 디지털 보조기기(PDA), 셀룰러 전화, 아이폰, 블랙베리, 프로세서, 전화, 웹 기기, 네트워크 라우터, 스위치 또는 브리지, 또는, 상기 기계에 의해 수행될 작용을 명시하는 한 세트의 명령어(순차적, 등)를 실행할 수 있는 임의의 기계일 수 있다.

[0051] 기계-판독가능 매체 또는 기계-판독가능 저장 매체가 예시적 실시예에서 단일 매체로 도시되지만, "기계-판독가능 매체" 및 "기계-판독가능 저장 매체"라는 용어는 한 세트 이상의 명령어를 저장하는 단일 매체 또는 복수 매체(가령, 중앙집중형 또는 분산형 데이터베이스 및/또는 관련 캐시 및 서버)를 포함하도록 간주되어야 한다. "기계-판독가능 매체" 및 "기계-판독가능 저장 매체"라는 용어는 현재 개시되는 기술 및 혁신의 방법 중 하나 이상을 기계로 하여금 수행하게 하면서, 기계에 의한 실행을 위한 한 세트의 명령어를 저장, 인코딩, 또는 운반할 수 있는 임의의 매체를 포함하도록 또한 취급되어야 한다.

[0052] 일반적으로, 발명의 실시예를 구현하도록 실행되는 루틴은, 운영 체제 또는 특정 애플리케이션, 구성요소, 프로그램, 객체, 모듈, 또는 "컴퓨터 프로그램"으로 불리는 명령어의 시퀀스의 일부분으로 구현될 수 있다. 컴퓨터 프로그램은 컴퓨터 내 다양한 메모리 및 스토리지 장치에서 다양한 시기에 하나 이상의 명령어 세트를 포함하며, 컴퓨터 내 하나 이상의 프로세싱 유닛 또는 프로세서에 의해 판독 및 실행될 때, 컴퓨터는 발명의 여러 형태를 포함하는 요소들을 실행하기 위한 작동을 수행한다.

[0053] 더욱이, 실시예가 컴퓨터 및 컴퓨터 시스템의 완전한 기능 실현의 범주에서 설명되었으나, 당 업자는 다양한 실시예가 다양한 형태로 프로그램 프로덕트로 분산될 수 있음을 이해할 것이고, 분배에 실제 영향을 미치는데 사용되는 특정 타입의 컴퓨터 또는 컴퓨터-판독가능 매체에 관계없이 발명이 동등하게 적용됨을 또한 이해할 것이다.

[0054] 기계-판독가능 저장 매체, 기계-판독가능 매체, 또는 컴퓨터-판독가능 (저장) 매체의 추가적인 예는 다른 것들 중에서도, 휘발성 및 비휘발성 메모리 장치, 플라피 및 기타 제거가능 디스크, 하드 디스크 드라이브, 광학 디스크(가령, 컴팩트 디스크 읽기-전용 메모리(CD ROM), 디지털 다용도 디스크(DVD), 등)을 포함한다.

[0055] 명세서 및 청구범위 전체를 통해, "포함한다", "포함하는", 등의 용어는 배타적인 형태가 아니라 포괄적인 측면에서, 즉, "포함하지만, 이에 제한되지 않는"의 의미로, 간주되어야 한다. 여기서 사용되는 바와 같이, "연결되는", "결합되는", 등의 용어는 2개 이상의 요소들 사이의 직접적 또는 간접적인 임의의 연결 또는 결합을 의미하며, 요소들 사이의 연결의 결합은 물리적, 논리적, 또는 이들의 조합일 수 있다. 추가적으로, "여기서", "위에서", "아래에서" 및 유사 용어는 본 출원에서 사용될 때, 본 출원의 어떤 특정 부분을 의미하는 것이 아니라, 본 출원 전체를 의미한다. 단수 또는 복수를 이용한 위 상세한 설명의 용어는 각각 복수 또는 단수를 또한 포함

할 수 있다. 2개 이상의 아이템의 리스트를 참조할 때 "또는"이라는 용어는 리스트 내 임의의 아이템, 리스트 내 모든 아이템, 리스트 내 아이템들의 임의의 조합들로 이루어지는, 용어의 모든 해석을 포함한다.

[0056] 발명의 실시예의 위 상세한 설명은 앞서 개시한 정확한 형태로 발명을 제한하고자 하는 것이 아니다. 발명의 구체적 실시예 및 dAP는 예시적인 용도로 위에서 설명되었으며, 관련 분야의 숙련자들이 인지하는 한, 다양한 등가 변형예가 발명의 범위 내에서 가능하다. 예를 들어, 프로세스 또는 블록들이 주어진 순서로 제시될 때, 대안의 실시예는 단계를 갖는 루틴을 또는 블록을 갖는 시스템을, 다른 순서로 수행할 수 있고, 일부 프로세스 또는 블록들이 삭제, 이동, 추가, 추가분할, 조합, 및/또는 변형되어, 대안 또는 세부조합을 제공할 수 있다. 이러한 프로세스 또는 블록들 각각은 서로 다른 다양한 방식으로 구현될 수 있다. 또한, 프로세스 또는 블록들이 때때로 차례로 수행되는 것으로 보이지만, 이러한 프로세스 또는 블록들이 병렬로 수행될 수 있고, 또는 서로 다른 시기에 수행될 수 있다. 여기서 언급되는 어떤 특정 수치도 단지 예에 불과하고, 대안의 구현예는 다른 값 또는 다른 범위를 이용할 수 있다.

[0057] 여기서 제공되는 발명의 가르침은 반드시 상술한 시스템에만이 아니라, 다른 시스템에도 적용될 수 있다. 앞서 설명한 다양한 실시예의 원소 및 작용들이 조합되어 추가 실시예를 제공할 수 있다.

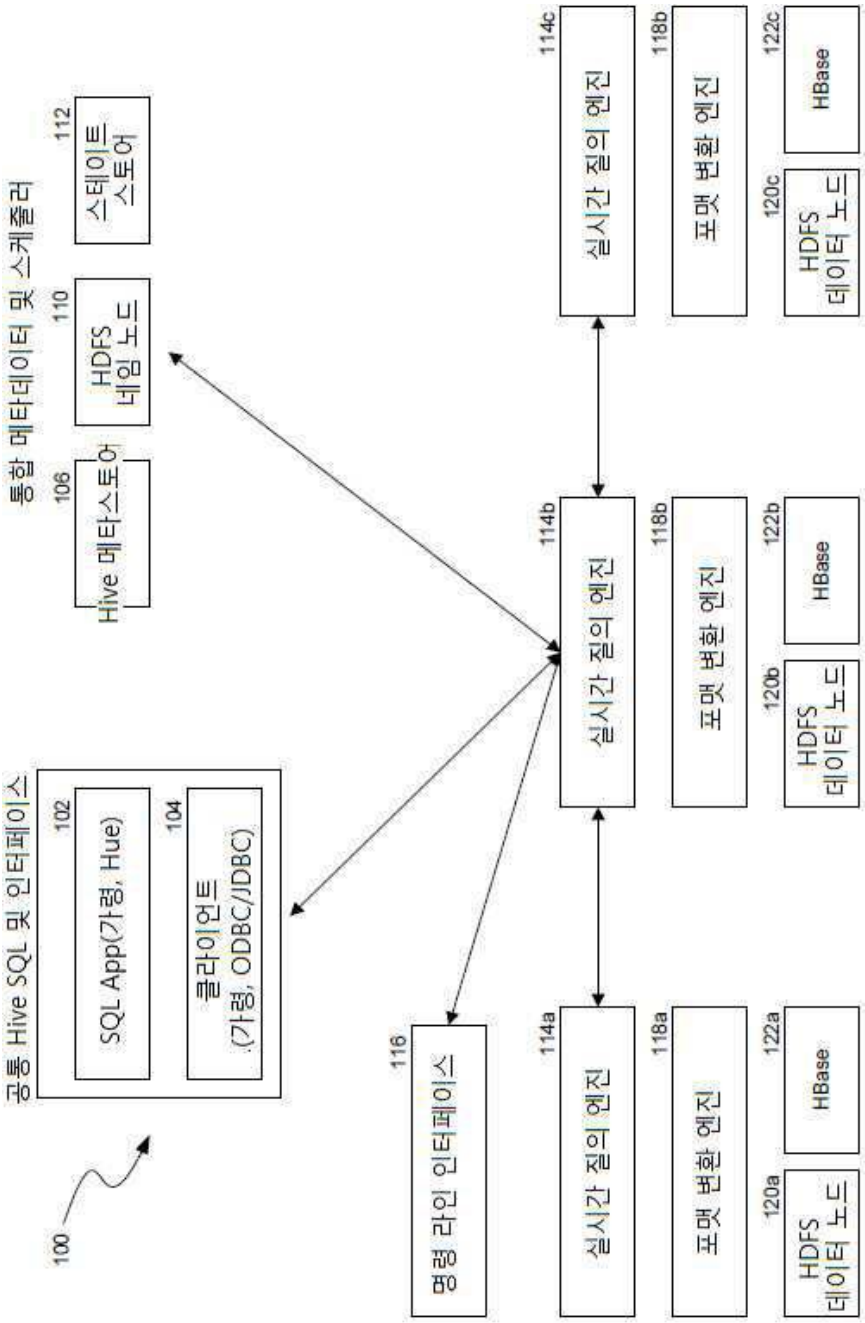
[0058] 첨부 제출 문서에 나열될 수 있는 것을 포함한, 앞서 언급한 임의의 특허 및 특허출원 및 그외 다른 참고문헌은 참고자료로 여기에 포함된다. 발명의 형태들은 앞서 설명한 다양한 참고문헌의 시스템, 기능, 및 개념을 이용하여 변형되어, 발명의 추가 실시예를 제공할 수 있다.

[0059] 이러한 변화 및 다른 변화가, 위 상세한 설명에 비추어 발명에서 이루어질 수 있다. 위 설명이 발명의 소정의 실시예를 설명하고 고려되는 최적 모드를 설명하지만, 위에서 얼마나 세부적으로 설명되었는지에 관계없이, 가르침은 많은 방식으로 실시될 수 있다. 시스템의 세부사항은 여기서 개시되는 대상에 의해 여전히 포함되면서도, 구현 세부사항 측면에서 폭넓게 변화할 수 있다. 상술한 바와 같이, 발명의 소정의 특징 또는 형태를 설명할 때 사용되는 특정 용어는, 해당 용어가 해당 용어와 관련된 발명의 임의의 구체적인 특성, 특징, 또는 형태에 제한되도록 여기서 재규정됨을 의미하는 것으로 간주되어서는 안된다. 일반적으로, 다음의 청구범위에서 사용되는 용어는, 위 상세한 설명 부분이 이러한 용어를 명백하게 규정하지 않을 경우, 명세서 내에서 개시되는 구체적 실시예로 발명을 제한하는 것으로 간주되어서는 안된다. 따라서, 발명의 실제 범위는 개시되는 실시예를 포괄하는 것이 아니라, 청구범위 하의 발명의 실시 또는 구현의 모든 등가 방식들을 포괄한다.

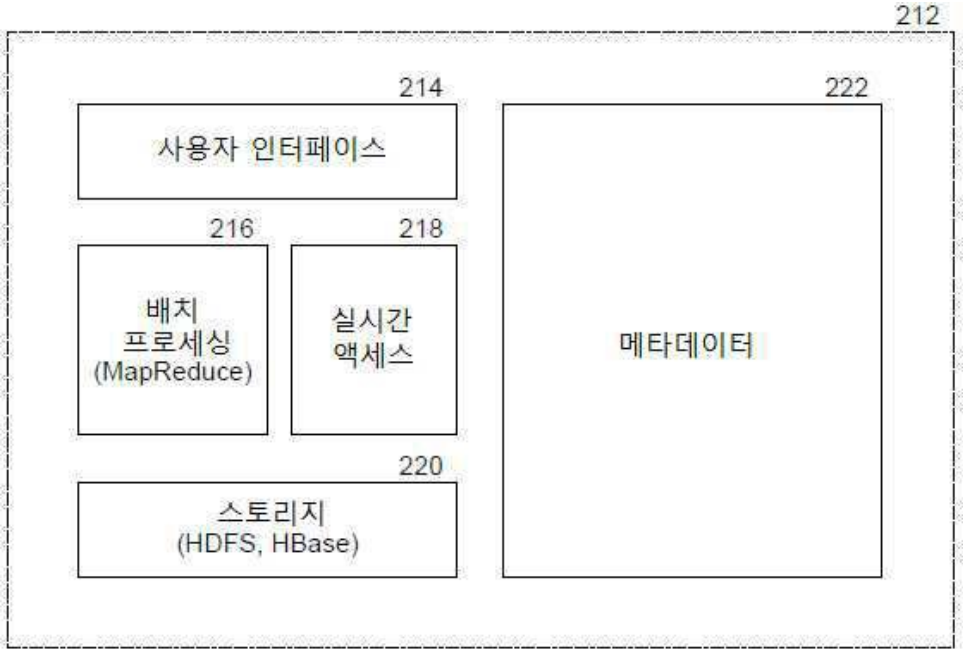
[0060] 발명의 소정의 형태들이 소정의 청구항 형태로 아래에 제시되지만, 발명자는 임의의 개수의 청구항 형태로 발명의 다양한 형태를 고려한다. 예를 들어, 발명의 일 형태만이 35 U.S.C.§112, ∏13 하에 평균-플러스-기능 청구항으로 언급되지만, 다른 형태가, 마찬가지로, 평균-플러스-기능 청구항으로, 또는, 컴퓨터-판독가능 매체로 실시되는 것과 같이, 다른 형태로 실시될 수 있다. (35 U.S.C.§112, ∏13 하에 처리되는 임의의 청구항들이 "~하기 위한 수단"의 용어로 시작될 것이다). 따라서, 출원인은 발명의 다른 형태를 위한 추가적인 청구항 형태를 추구하도록 출원 후 추가 청구항을 추가할 권리를 갖는다.

도면

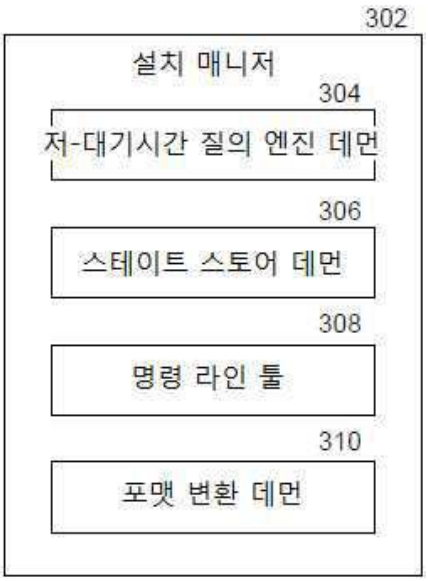
도면1


100
공통 Hive SQL 및 인터페이스
102
SQL App(가령, Hue)
104
클라이언트
(가령, ODBC/JDBC)
통합 메타데이터 및 스케줄러
106
Hive 메타스토어
110
HDFS
네임 노드
112
스테이트
스토어
116
명령 라인 인터페이스
114a
실시간 질의 엔진
118a
포맷 변환 엔진
120a
HDFS
데이터 노드
122a
HBase
114b
실시간 질의 엔진
118b
포맷 변환 엔진
120b
HDFS
데이터 노드
122b
HBase
114c
실시간 질의 엔진
118b
포맷 변환 엔진
120c
HDFS
데이터 노드
122c
HBase

도면2


212
214
사용자 인터페이스
222
216
배치
프로세싱
(MapReduce)
218
실시간
액세스
메타데이터
220
스토리지
(HDFS, HBase)

도면3


302
설치 매니저
304
저-대기시간 질의 엔진 데먼
306
스테이트 스토어 데먼
308
명령 라인 툴
310
포맷 변환 데먼

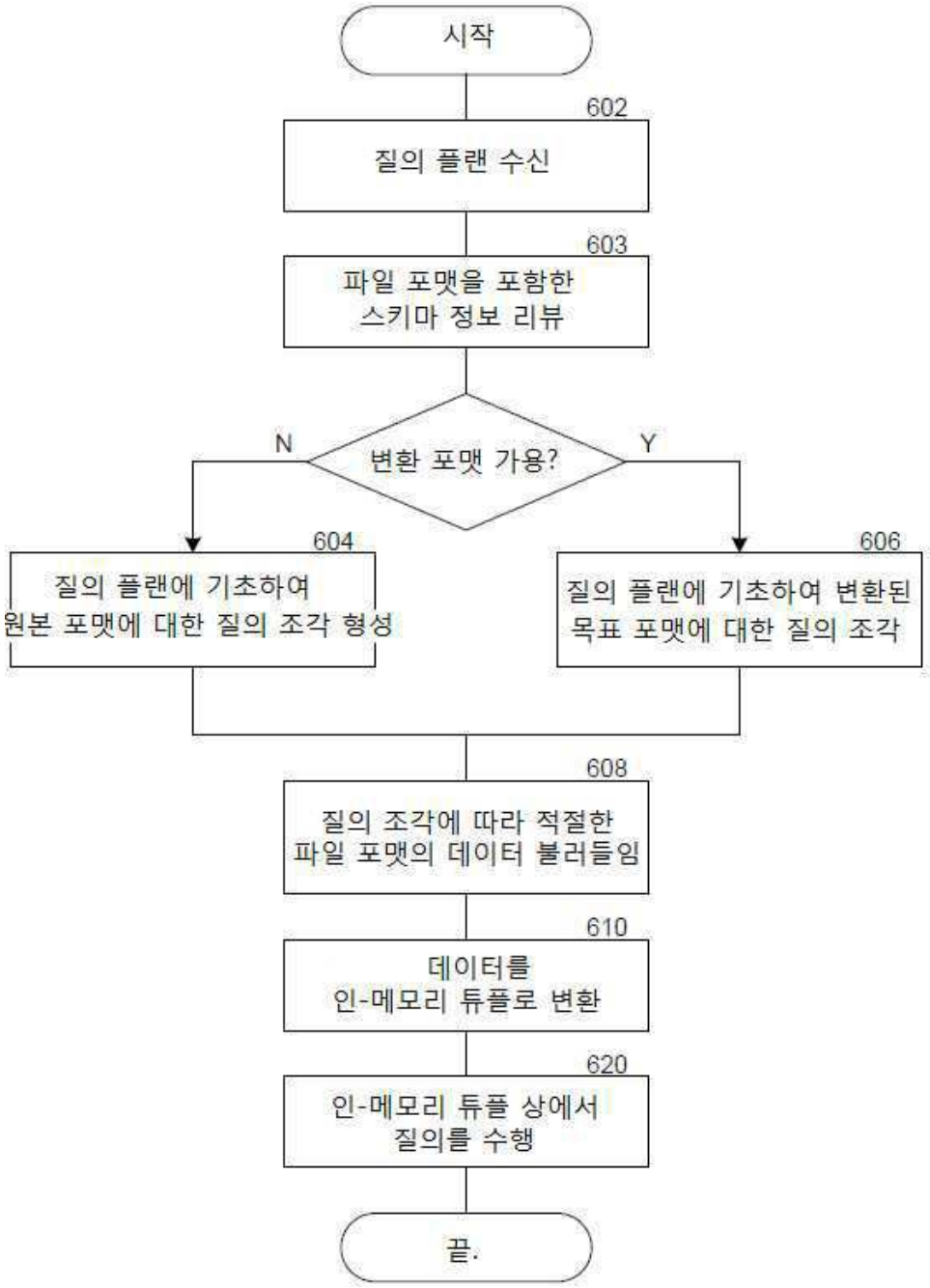
도면4


At 데이터노드
저-대기시간 질의 엔진 데먼
316
질의 플래너
318
질의 코디네이터
320
질의 실행 엔진
322
LLVM

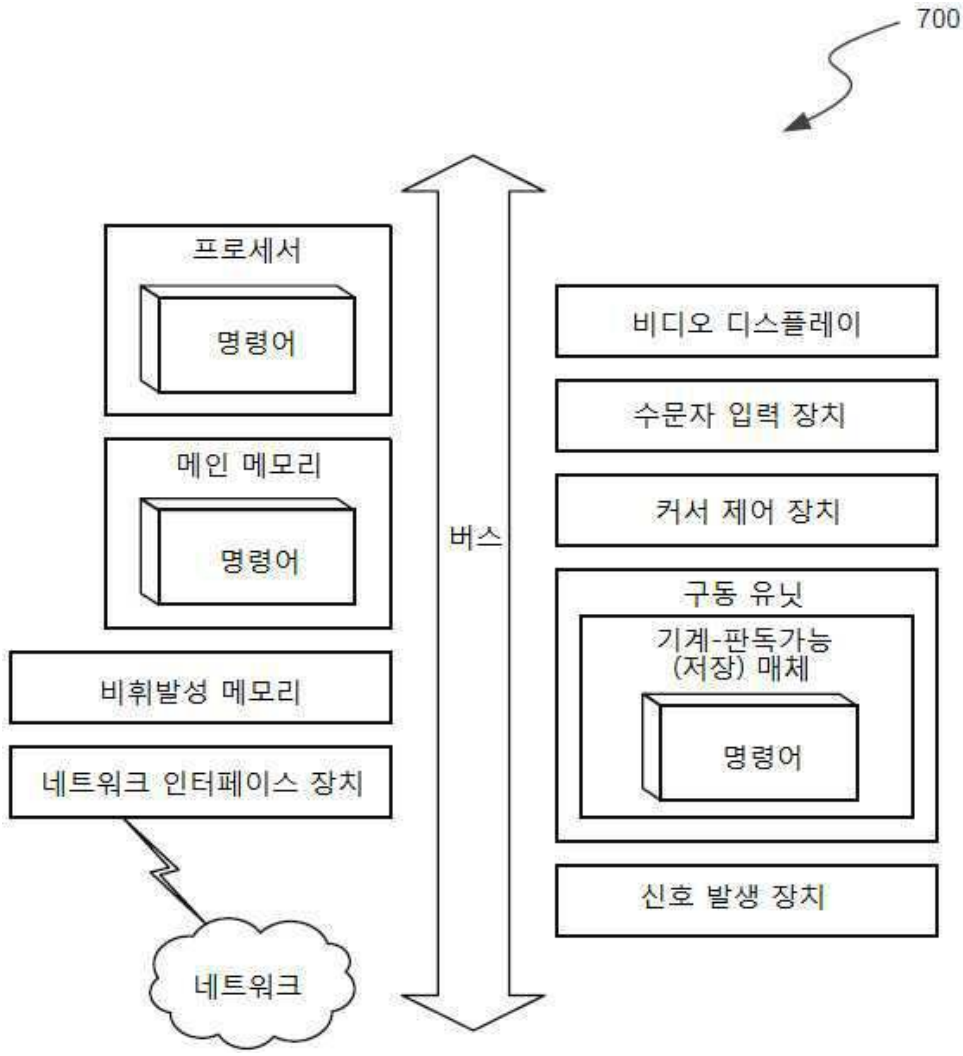
도면5


At 데이터노드
포맷 변환 엔진
412
스케줄러
414
컨버터

도면6


시작
602
질의 플랜 수신
603
파일 포맷을 포함한
스키마 정보 리뷰
변환 포맷 가용?
N
Y
604
질의 플랜에 기초하여
원본 포맷에 대한 질의 조각 형성
606
질의 플랜에 기초하여 변환된
목표 포맷에 대한 질의 조각
608
질의 조각에 따라 적절한
파일 포맷의 데이터 불러들임
610
데이터를
인-메모리 튜플로 변환
620
인-메모리 튜플 상에서
질의를 수행
끝.

도면7


700
프로세서
명령어
메인 메모리
명령어
비휘발성 메모리
네트워크 인터페이스 장치
네트워크
버스
비디오 디스플레이
수문자 입력 장치
커서 제어 장치
구동 유닛
기계-판독가능
(저장) 매체
명령어
신호 발생 장치