(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0058357
(43) 공개일자 2022년05월09일

(51) 국제특허분류(Int. Cl.)
G06F 16/25 (2019.01) *G06F 16/17* (2019.01)
G06F 16/174 (2019.01) *G06F 16/182* (2019.01)
G06F 16/22 (2019.01) *G06F 16/27* (2019.01)
(52) CPC특허분류
G06F 16/25 (2019.01)
G06F 16/1727 (2019.01)
(21) 출원번호 10-2021-0009564
(22) 출원일자 2021년01월22일
심사청구일자 2021년01월22일
(30) 우선권주장
1020200143681 2020년10월30일 대한민국(KR)

(71) 출원인
충남대학교산학협력단
대전광역시 유성구 대학로 99 (궁동, 충남대학교)
(72) 발명자
이규철
대전광역시 유성구 대학로 99 충남대학교
이경하
대전광역시 유성구 대학로 245 한국과학기술정보연구원
김인아
대전광역시 유성구 대학로 99 충남대학교
(74) 대리인
이은철, 김재문

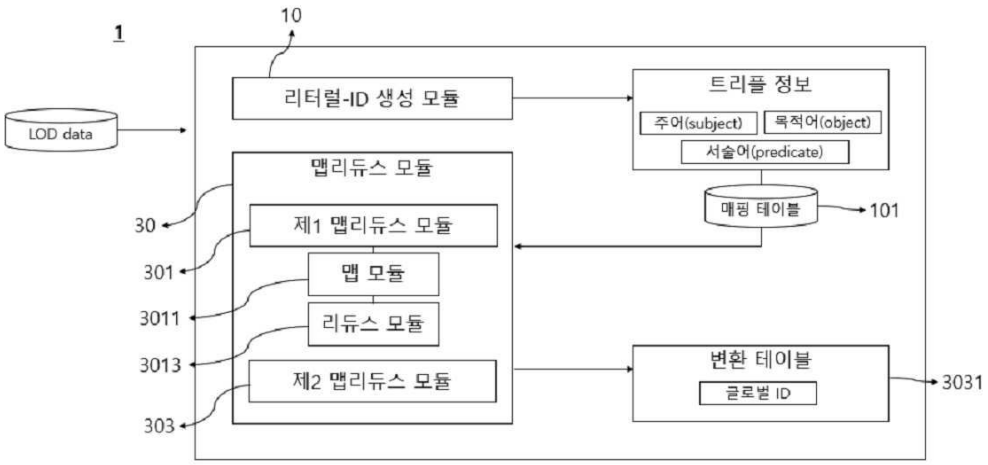
전체 청구항 수 : 총 5 항

(54) 발명의 명칭 **대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템 및 방법**

(57) 요 약

본 발명은, 대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 시스템에 있어서; 상기 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축하기 위해, 상기 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 상기 리터럴과 정수값의 ID를 매핑한 매핑 테이블을 구축하는 리터럴-ID 생성 모듈; 및 상기 리터럴-ID 생성 모듈에서 구축된 상기 매핑 테이블에서, 상기 ID의 정수값이 부여된 리터럴을 중복 없는 리터럴로 추출하여, 추출된 리터럴을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환하는 맵리듀스 모듈;을 포함하여, 상기 맵리듀스 모듈은, 상기 LOD 데이터의 RDF 포멧을 상기 글로벌-ID로 압축 변환된 변환 테이블을 구축하는 것을 일 특징으로 한다.

대 표 도 - 도2



(52) CPC특허분류
G06F 16/1744 (2019.01)
G06F 16/182 (2019.01)
G06F 16/2255 (2019.01)
G06F 16/27 (2019.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711101951
과제번호 G-19-GT-CU07
부처명 과학기술정보통신부
과제관리(전문)기관명 국가과학기술연구회
연구사업명 국가과학기술연구회연구운영비지원(R&D)(주요사업비)
연구과제명 AI기술을 활용한 공공데이터 기반 지역현안 솔루션 개발 및 실용화(안전·안심사회 실현을 위한 실증연구 중심으로)
기 여 율 1/1
과제수행기관명 한국과학기술정보연구원
연구기간 2019.12.01 ~ 2020.11.30

명 세 서

청구범위

청구항 1

대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 시스템에 있어서;

상기 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축하기 위해, 상기 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 상기 리터럴과 정수값의 ID를 매핑한 매핑 테이블을 구축하는 리터럴-ID 생성 모듈; 및

상기 리터럴-ID 생성 모듈에서 구축된 상기 매핑 테이블에서, 상기 ID의 정수값이 부여된 리터럴을 중복 없는 리터럴로 추출하여, 추출된 리터럴을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환하는 맵리듀스 모듈;을 포함하여,

상기 맵리듀스 모듈은,

상기 LOD 데이터의 RDF 포멧을 상기 글로벌-ID로 압축 변환된 변환 테이블을 구축하는 것을 특징으로 하는 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템.

청구항 2

제 1 항에 있어서,

상기 리터럴-ID 생성 모듈은,

자원의 고유 식별을 위한 주어(subject), 목적어(object), 서술어(predicate)의 요소가 각각 다수의 문자가 포함된 리터럴의 형태로 URI 또는 문자열이 구성된 주어(subject), 목적어(object), 및 서술어(predicate)에 각각 정수값의 ID로 부여하는 것을 특징으로 하는 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템.

청구항 3

제 1 항에 있어서,

상기 리터럴-ID 생성 모듈은,

해시 테이블을 기반으로 상기 매핑 테이블을 구축하는 것을 특징으로 하는 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템.

청구항 4

제 1 항에 있어서,

상기 맵리듀스 모듈은,

추출된 중복 없는 리터럴에 부여된 상기 글로벌-ID와, 상기 글로벌-ID에 매칭되는 리터럴 정보를 밸류(value)로 생성한 제1 맵리듀스 모듈; 및

상기 제1 맵리듀스 모듈에서 생성한 상기 밸류의 정보 중 특정 RDF 블록에 속하는 리터럴의 상기 글로벌-ID 정보만을 저장하여 상기 변환 테이블을 구축하는 제2 맵리듀스 모듈;을

포함하는 것을 특징으로 하는 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템.

청구항 5

대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 방법에 있어서,

(a) 상기 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축하기 위해, 상기 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 상기 리터럴과 정수값의 ID를 매핑한 매핑 테이블을 구축하는 단계;

(b) 상기 (a)단계에서 구축된 상기 매핑 테이블에서, 상기 ID의 정수값이 부여된 리터럴을 중복 없는 리터럴로 추출하여, 추출된 리터럴을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환하는 단계;를 포함하여,

상기 (b)단계는,

상기 LOD 데이터의 RDF 포멧을 상기 글로벌-ID로 압축 변환된 변환 테이블을 구축하는 것을 특징으로 하는 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템 및 방법에 관한 것으로서, 특히 맵리듀스 기법을 적용하여 LOD의 RDF 데이터를 압축하여 변환하는 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템 및 방법에 관한 것이다.

배 경 기 술

[0003] LOD(Linked Open Data)는 데이터를 웹에 개방함으로써 다른 도메인의 데이터와 연결이 될 수 있는 링크 기반 개방형 데이터를 의미한다. LOD는 연결된 데이터에 대한 정보를 기술하기 위해 RDF(Resource Description Framework) 포멧을 사용하여 데이터를 컴퓨터가 이해할 수 있는 형태로 저장한다. 따라서, LOD 데이터는 웹 자원에 대한 메타데이터 정보를 기계가 읽을 수 있도록 표현하기 위한 언어인 RDF 포맷을 사용하여 데이터를 저장한다. RDF는 각 자원을 URI(Uniform Resource Identifier)로 식별함으로써 웹상에 있는 데이터 간 연결을 가능하게 하며, 주어(Subject), 서술어(Predicate), 목적어(Object)로 구성된 가장 작은 단위인 트리플(Triple)이 서로 연결된 구조를 가지고 있다.

[0004] LOD의 트리플 데이터가 가지는 가장 주요한 특징은 자원에 대한 고유 식별을 위해 URI(Uniform Resource Identifier)를 가진다는 것이며, 그 외에 문자열(String), 빈 노드(Blank node)를 가질 수 있다. 트리플 데이터의 각 요소를 구성하는 URI와 문자열은 결국 많은 수의 문자(Character)를 가진 Literal 형태로 저장된다. 이는 저장 비용, 조인 비용, 그리고 분산 환경에서의 통신 비용 증가를 야기하는 주요 요소이다.

[0005] RDF는 각 자원을 URI(Uniform Resource Identifier)로 식별함으로써 웹상에 있는 데이터 간 연결을 가능하게 하며, 주어(Subject), 서술어(Predicate), 목적어(Object)로 구성된 가장 작은 단위인 트리플(Triple)이 서로 연결된 구조를 가지고 있다.

[0006] RDF가 가지는 가장 주요한 특징은 자원에 대한 고유 식별을 위해 각 자원에 URI를 부여하는 것이며, 때문에 트리플의 각 요소인 주어, 서술어, 목적어 자리에는 URI, 문자열(String), 빈 노드(Blank node)가 위치할 수 있다.

[0007] 트리플을 구성하는 URI와 문자열은 많은 수의 문자(Character)를 가진 'Literal' 형태로 저장된다. 이는 데이터의 저장 비용, 조인 비용, 그리고 분산 환경에서는 통신 비용을 발생시키는 주요 요소가 된다. 따라서, RDF의 URI와 문자열을 압축할 필요성이 있다.

[0008] 도 1은 종래의 RDF 트리플 정보를 예시한 도면이다.

[0009] 도 1을 참조하면, RDF의 URI로 “<http://www.Department0.University0.edu/Lee>”와 문자열 “lee@cnu.ac.kr” 모두 많은 수의 문자로 구성된 것을 확인할 수 있다. 도 2와 같이 종래에 존재하는 RDF 관리 시스템들은 이

러한 문제를 고려하여 RDF 데이터를 정수 형태의 값으로 변환하는 여러 가지 압축 방법을 제시한다. 예를 들어, RDF 관리 기술 중 하나인 RDF-3X는 데이터의 Literal과 정수 형태의 ID를 연결하는 ID 매핑 테이블(ID Mapping Table)을 만들어 데이터를 변환함으로써, 저장공간을 압축하고 적은 비용의 쿼리 프로세싱을 지향한다.

[0010] 또한, 다른 기술 중 하나인 트리플비트(TripleBit)는 트리플 데이터의 문자열을 Prefix ID, Suffix로 나누어 Prefix-ID 매핑 테이블, Suffix-ID 매핑 테이블을 생성하여 데이터를 압축한다. 데이터의 Literal과 ID를 매핑하는 매핑 테이블은 중복이 없는(Distinct) 문자열에 대해 고유한 ID를 매핑하여 저장해야 하며, 이를 위해 동일한 테이블에 해당 정보들을 저장해야 한다.

[0011] 그러나 변환하고자 하는 RDF 데이터의 크기가 일정 이상으로 증가할 경우, 언급한 종래 기술들은 ID 매핑 테이블을 제한된 메모리 공간에 유지할 수가 없어 메모리 부족 문제가 발생한다.

[0012] 이에 본 출원인은 대량 LOD 저장을 위한 데이터 변환 방법으로 분산 데이터 처리 프레임워크인 맵리듀스(MapReduce)를 사용하여 대량의 RDF 데이터를 압축하여 변환하는 새로운 모델을 고안하게 되었다.

선행기술문헌

특허문헌

[0014] (특허문헌 0001) 대한민국 등록특허 제10-1928529호

발명의 내용

해결하려는 과제

[0015] 본 발명이 이루고자 하는 기술적 과제는 종래의 단점을 해결한 것으로서, 대량의 LOD 데이터 저장시, RDF(Resource Description Framework) 데이터를 압축하여 변환할 때 발생하는 메모리 부족 문제를 해결하고자 하는데, 그 목적이 있다.

과제의 해결 수단

[0017] 상기 목적을 달성하기 위하여 본 발명은, 대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 시스템에 있어서; 상기 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축하기 위해, 상기 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 상기 리터럴과 정수값의 ID를 매핑한 매핑 테이블을 구축하는 리터럴-ID 생성 모듈; 및 상기 리터럴-ID 생성 모듈에서 구축된 상기 매핑 테이블에서, 상기 ID의 정수값이 부여된 리터럴을 중복 없는 리터럴로 추출하여, 추출된 리터럴을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환하는 맵리듀스 모듈;을 포함하여, 상기 맵리듀스 모듈은, 상기 LOD 데이터의 RDF 포멧을 상기 글로벌-ID로 압축 변환된 변환 테이블을 구축하는 것을 일 특징으로 한다.

[0018] 바람직하게, 상기 리터럴-ID 생성 모듈은, 자원의 고유 식별을 위한 주어(subject), 목적어(object), 서술어(predicate)의 요소가 각각 다수의 문자가 포함된 리터럴의 형태로 URI 또는 문자열이 구성된 주어(subject), 목적어(object), 및 서술어(predicate)에 각각 정수값의 ID로 부여할 수 있다.

[0019] 바람직하게, 상기 리터럴-ID 생성 모듈은, 해시 테이블을 기반으로 상기 매핑 테이블을 구축할 수 있다.

[0020] 바람직하게, 상기 맵리듀스 모듈은, 추출된 중복 없는 리터럴에 부여된 상기 글로벌-ID와, 상기 글로벌-ID에 매칭되는 리터럴 정보를 밸류(value)로 생성한 제1 맵리듀스 모듈; 및 상기 제1 맵리듀스 모듈에서 생성한 상기 밸류의 정보 중 특정 RDF 블록에 속하는 리터럴의 상기 글로벌-ID 정보만을 저장하여 상기 변환 테이블을 구축하는 제2 맵리듀스 모듈;을 포함할 수 있다.

[0021] 또한, 본 발명은 대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 방법에 있어서, 상기 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축

하기 위해, 상기 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 상기 리터럴과 정수값의 ID를 매핑한 매핑 테이블을 구축하는 (a)단계; 상기 (a)단계에서 구축된 상기 매핑 테이블에서, 상기 ID의 정수값이 부여된 리터럴을 중복 없는 리터럴로 추출하여, 추출된 리터럴을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환하는 (b)단계;를 포함하여, 상기 (b)단계는, 상기 LOD 데이터의 RDF 포멧을 상기 글로벌-ID로 압축 변환된 변환 테이블을 구축하는 것을 다른 특징으로 한다.

발명의 효과

[0023] 본 발명에 따르면, 대량의 RDF 형식으로 저장된 데이터를 효과적으로 압축 저장함으로써, 저장공간 비용, 데이터 검색 비용, 분산 환경에서의 통신 비용 등을 줄일 수 있는 효과가 있다. 또한, RDF 데이터로 구성되는 지식베이스 데이터의 저장에 활용할 수 있는 효과가 있다.

도면의 간단한 설명

[0025] 도 1은 종래의 RDF 트리플 정보를 예시한 도면이다.

도 2는 본 발명의 실시예에 따른 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템 구성도를 나타낸다.

도 3은 도 2의 실시예에 따른 맵리듀스 기반 데이터 변환 시스템의 전체 데이터 처리 구성도를 나타낸다.

도 4a 내지 도 4d는 본 발명의 실시예에 따른 매핑 테이블을 나타낸다.

도 5는 본 발명의 실시예에 따른 트리플 정보의 변환 예시를 나타낸다.

도 6은 본 발명의 실시예에 따른 리터럴-ID 생성 모듈이 구축한 매핑 테이블의 구축 예시이다.

도 7은 본 발명의 실시예에 따른 LOD 데이터 변환 및 저장 과정을 나타내는 도면이다.

도 8은 본 발명의 실시예에 따른 제1 맵리듀스 모듈에서 수행하는 글로벌(Global) ID 생성 맵리듀스 잡의 동작 과정을 나타내는 도면이다.

도 9는 본 발명의 실시예에 따른 제2 맵리듀스 모듈에서 수행하는 트리플 변환 잡 동작 과정을 나타내는 도면이다.

발명을 실시하기 위한 구체적인 내용

[0026] 이하, 첨부된 도면들에 기재된 내용들을 참조하여 본 발명을 상세히 설명한다. 다만, 본 발명이 예시적 실시 예들에 의해 제한되거나 한정되는 것은 아니다. 각 도면에 제시된 동일 참조부호는 실질적으로 동일한 기능을 수행하는 부재를 나타낸다.

[0027] 본 발명의 목적 및 효과는 하기의 설명에 의해서 자연스럽게 이해되거나 보다 분명해 질 수 있으며, 하기의 기재만으로 본 발명의 목적 및 효과가 제한되는 것은 아니다. 또한, 본 발명을 설명함에 있어서 본 발명과 관련된 공지 기술에 대한 구체적인 설명이, 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략하기로 한다.

[0028] 도 2는 본 발명의 실시예에 따른 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템(1) 구성도를 나타낸다.

[0029] 도 2를 참조하면, 대량의 LOD 저장을 위한 맵리듀스 기반 데이터 변환 시스템(1)은 리터럴-ID 생성 모듈(10), 및 맵리듀스 모듈(30)을 포함할 수 있다.

[0030] 리터럴-ID 생성 모듈(10)은 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축하기 위해, 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 리터럴과 정수값의 ID를 매핑한 매핑 테이블(101)을 구축할 수 있다. 리터럴-ID 생성 모듈(10)은 해시 테이블을 기반으로 매핑 테이블(101)을 구축할 수 있다.

[0031] 리터럴-ID 생성 모듈(10)은 자원의 고유 식별을 위한 주어(subject), 목적어(object), 서술어(predicate)의 요

소가 각각 다수의 문자가 포함된 리터럴의 형태로 URI 또는 문자열이 구성된 주어(subject), 목적어(object), 및 서술어(predicate)에 각각 정수값의 ID로 부여할 수 있다.

[0032] 리터럴-ID 생성 모듈(10)은 RDF 데이터를 줄이기 위해, 데이터의 주어, 서술어, 목적어에 위치한 리터럴(Literal)을 정수(Integer) 값으로 변경한다. 정수 값은 문자열에 비해 적은 공간과 통신 비용을 소비하며, 이후 스파클(SPARQL) 쿼리 요청 또는 RDF 데이터에 대한 검색 요청에서 조인 연산 처리 시 두 데이터 사이의 비교 비용을 줄일 수 있다는 장점을 가진다.

[0033] 본 실시예로, 리터럴-ID 생성 모듈(10)은 데이터 변환을 위해 Subject/Object(주어 및 목적어)-ID 매핑 테이블, Predicate-ID(서술어) 매핑 테이블, 2개의 ID 매핑 테이블을 생성할 수 있다. 도 4a 내지 도 4d는 본 발명의 실시예에 따른 매핑 테이블을 나타낸다. 도 4a는 Subject/Object-ID 매핑 테이블과 ID-Subject/Object 테이블을 나타내는 도면이고, 도 4b는 Predicate-ID 매핑 테이블과 ID-Predicate 테이블을 나타내는 도면이다. 또한, 도 4c는 Subject/Object-ID 테이블과 ID-Subject/Object 테이블을 나타내는 도면이고, 도 4d는 Predicate-ID 테이블과 ID-Predicate 테이블을 나타내는 도면이다.

[0034] <Subject/Object-ID 매핑 테이블>

[0035] Subject/Object-ID 테이블은 RDF 데이터에서 주어(Subject)와 목적어(Object)에 위치하는 리터럴(Literal)과 이에 대응하는 정수 값을 저장하는 ID 매핑 테이블을 나타낸다. 동일한 자원이 주어와 목적어 두 곳에 모두 위치할 수 있으므로, 주어와 목적어에 위치하는 모든 리터럴(Literal)을 하나의 테이블에 저장한다.

[0036] 해당하는 데이터를 빠르게 검색하기 위해, 매핑 테이블은 해시 테이블로 구성된다. 주어/서술어 Literal을 테이블의 Key로, 각 Literal에 대응하는 정수 ID를 Value로 저장한다. 이때, 데이터의 변환과 매핑 테이블 생성은 동시에 발생한다. 변환하고자 하는 주어/서술어 Literal을 Subject/Object-ID 테이블에 입력하면, 이에 대응하는 ID 값을 검색한다.

[0037] 또한, 이후 ID 값으로 변환된 데이터의 복원을 위해 주어/서술어 ID를 테이블의 Key로, 각 ID에 대응하는 실제 Literal을 Value로 저장하는 ID-Subject/Object 테이블을 동시에 생성한다.

[0038] 도 4a에서 도시된 Subject/Object ID 매핑 테이블은 현재 데이터 압축에 바로 사용이 되며, ID-Subject/Object 매핑 테이블은 최종 검색 결과를 다시 복원하는 데에 사용된다. 도 4a에서 상단에 위치한 Subject/Object-ID 테이블의 Hash Key는 Subject/Object 리터럴(Literal)들이 입력되어 있으며, 해당 리터럴(Literal)들을 테이블에 입력하여 바로 대응되는 Hash Value, 즉 정수 ID를 추출할 수 있다.

[0039] 도 4c의 Subject/Object-ID 테이블은 LOD 트리플 데이터에서 Subject와 Object에 위치하는 Literal과 이에 대응하는 정수 ID를 저장하는 매핑 테이블을 나타낸다. Literal은 Subject와 Object 두 곳에 모두 위치할 수 있으므로, Subject와 Object 두 곳의 Literal을 하나의 테이블에 저장한다. 또한, 빠른 데이터의 변환을 위해 테이블은 해시 테이블로 Litera과 ID를 저장한다.

[0040] LOD 트리플의 Subject/Object Literal을 테이블의 Key로, 각 데이터에 대응하는 정수 ID를 Value로 저장한다. 다른 트리플의 Subject/Object Literal을 Subject/Object-ID 테이블에 입력하면, 이에 대응하는 해당하는 ID 값을 검색한다. 만약 테이블에 해당하는 ID 값이 없는 경우 테이블의 가장 높은 ID 값을 증가시키고 새로운 ID 값을 부여하여 저장한다.

[0041] 또한, 이후 ID 값으로 변환된 Literal의 복원을 위해 Subject/Object ID를 테이블의 Key로, 각 ID에 대응하는 실제 Literal을 Value로 저장하는 ID-Subject/Object 테이블을 동시에 생성한다.

[0042] 도 3은 도 2의 실시예에 따른 맵리듀스 기반 데이터 변환 시스템의 전체 데이터 처리 구성도를 나타낸다. 도 3을 참조하면 본 실시예에 따른 대량 LOD 저장 시스템(1)은 LOD를 저장하는 Data Loader와 쿼리에 대한 기본 정보를 생성하는 Query Loader로 구성된다.

[0043] 대량 LOD 저장을 위한 맵리듀스 기반 데이터 변환 방법은 도 3에서 도시된 LOD 데이터 저장 시스템(1)에서 파란색으로 표시된 “Triple Loading with MapReduce jobs“에서 수행되는 과정을 의미할 수 있다.

[0044] 이하, 리터럴-ID 생성 모듈(10)의 매핑 테이블(101) 구축 과정부터 상세히 설명한다.

[0045] <Predicate-ID 매핑 테이블>

[0046] 도 4b에 도시된 Predicate-ID 매핑 테이블은 RDF 데이터에서 서술어(Predicate)에 위치하는 리터럴(Literal)과

이에 대응하는 정수 값을 저장하는 매핑 테이블을 나타낸다. 해당하는 데이터를 빠르게 검색하기 위해, 매핑 테이블(101)은 해시 테이블로 구성된다.

[0047] 서술어 리터럴(Literal)을 테이블의 Key로, 각 리터럴(Literal)에 대응하는 정수 ID를 Value로 저장한다. 변환하고자 하는 서술어 리터럴(Literal)을 Predicate-ID 테이블에 입력하면, 이에 대응하는 해당하는 ID 값을 검색할 수 있다.

[0048] 또한, 이후 ID 값으로 변환된 리터럴(Literal)의 복원을 위해 서술어 ID를 테이블의 Key로, 각 ID에 대응하는 실제 데이터를 Value로 저장하는 ID-Predicate 테이블을 동시에 생성한다. Predicate-ID 매핑 테이블은 현재 데이터 압축에 바로 사용이 되며, ID-Predicate 매핑 테이블은 최종 검색 결과를 다시 복원하는 데에 사용된다.

[0049] 도 4b의 상단에 위치한 Predicate-ID 테이블의 Hash Key는 Predicate Literal들이 입력되어 있으며, 해당 리터럴(Literal)들을 테이블에 입력하여 바로 대응되는 Hash Value, 즉 정수 ID를 추출할 수 있다. 생성한 해시 기반 ID-mapping 테이블을 사용하여 RDF 데이터를 정수 값으로 변환한다.

[0050] 도 4d의 Predicate-ID 테이블은 LOD 트리플 데이터에서 Predicate에 위치하는 리터럴(Literal)과 이에 대응하는 정수 ID를 저장하는 매핑 테이블을 나타낸다. 빠른 데이터의 변환을 위해 테이블은 해시 테이블로 리터럴(Literal)과 ID를 저장한다.

[0051] LOD 트리플의 Predicate 리터럴(Literal)을 테이블의 Key로, 각 리터럴(Literal)에 대응하는 정수 ID를 Value로 저장한다. 다른 트리플의 Predicate 리터럴(Literal)을 Predicate-ID 테이블에 입력하면, 이에 대응하는 해당하는 ID 값을 검색한다. 만약 테이블에 해당하는 ID 값이 없는 경우 테이블의 가장 높은 ID 값을 증가시키고 새로운 ID 값을 부여하여 저장한다.

[0052] 또한, 이후 ID 값으로 변환된 리터럴(Literal)의 복원을 위해 Predicate ID를 테이블의 Key로, 각 ID에 대응하는 실제 리터럴(Literal)을 Value로 저장하는 ID-Predicate 테이블을 동시에 생성한다.

[0053] 맵리듀스 모듈(30)은 리터럴-ID 생성 모듈(10)에서 구축된 매핑 테이블(101)에서, ID의 정수값이 부여된 리터럴(Literal)을 중복 없는 리터럴(Literal)로 추출하여, 추출된 리터럴(Literal)을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환할 수 있다. 이 과정으로, 맵리듀스 모듈(30)은 LOD 데이터의 RDF 포멧을 글로벌-ID로 압축 변환된 변환 테이블(3031)을 구축할 수 있다.

[0054] 도 5는 본 발명의 실시예에 따른 트리플 정보의 변환 예시를 나타낸다. 도 6은 본 발명의 실시예에 따른 리터럴-ID 생성 모듈이 구축한 매핑 테이블(101)의 구축 예시이다.

[0055] 도 5 및 도 6를 참조하면, Subject, Predicate, Object로 구성된 트리플 데이터를 앞서 생성한 ID-mapping 테이블을 사용하여 변환하는 과정을 확인할 수 있다. 보다 상세하게, 변환 실시예를 설명하면 Subject에 위치한 URI “<http://www.Department0.University0.edu/Lee>”는 Subject/Object-ID 매핑 테이블에서 URI 리터럴(Literal)에 대응되는 ‘1’이라는 정수 ID 값을 획득한다. 따라서 Subject Literal은 정수 ID ’ 1’로 변환된다.

[0056] 마찬가지로, Predicate에 위치한 URI “<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>“는 Predicate-ID 매핑 테이블에서 해당 URI Literal에 대응되는 ‘1’이라는 정수 ID를 획득한다.

[0057] 마지막으로, Object에 위치한 URI “<http://www.Department0.University0.edu/Professor>“를 Subject/Object-ID 매핑 테이블에서 검색한다. 또한, 이에 대응하는 ‘2’라는 정수 ID로 변환한다. 이러한 과정의 결과로, 트리플 데이터는 최종적으로 “1, 1, 2 .”로 변환된다.

[0058] 맵리듀스 모듈(30)은 제1 맵리듀스 모듈(301) 및 제2 맵리듀스 모듈(303)을 포함할 수 있다. 제1 맵리듀스 모듈(301) 및 제2 맵리듀스 모듈(303)은 모두 맵 모듈(3011) 및 리듀스 모듈(3013)을 포함할 수 있다.

[0059] 앞에서 본 발명은 RDF 데이터 압축을 위한 ID 매핑 테이블(101)에 대해 설명하였다. 그러나 이는 상대적으로 작은 양의 RDF 데이터에 대해서는 문제없이 동작하지만, 만약 변환하고자 하는 RDF 데이터의 규모가 특정 규모 이상으로 클 경우, RDF의 리터럴(Literal)과 ID를 저장하는 매핑 테이블의 크기가 제한된 메모리 크기를 넘을 수 있다. 이 경우 메모리 부족 현상이 발생하여 RDF 데이터 변환을 하지 못하는 문제가 발생하게 된다. 때문에 종래의 RDF 저장소에서 ID 매핑 테이블(101)을 사용하여 압축하는 기술을 많이 사용하고 있으나, 대량의 RDF 데이터에 대해서는 변환하지 못하는 문제점을 가지고 있다.

[0060] 따라서 본 발명은 앞서 설명한 ID-매핑 테이블(101)을 사용한 RDF 데이터 변환 과정을 분산 데이터 처리 프레

임워크인 맵리듀스(MapReduce) 모듈(30)로 가져와, 맵리듀스 모듈(30)에서 분산으로 처리한다. 도 7은 본 발명의 실시예에 따른 LOD 데이터 변환 및 저장 과정을 나타내는 도면이다. 도 7에 도시된 바와 같이 각 트리플을 상술한 과정을 통해 변환하고, 변환된 트리플을 하둡 기반 분산 파일 저장소인 HDFS(Hadoop Distributed File System)에 저장한다. 맵리듀스 모듈(30)은 대용량 데이터를 처리하기 위한 분산 처리 프레임워크로, 맵(Map)과 리듀스(Reduce) 두 단계를 사용하여 데이터를 배치(Batch)로 처리한다. 본 발명에서는 두 개의 맵리듀스 잡(Job)을 생성하여 대용량의 RDF 데이터를 변환하는 과정을 수행한다.

[0061] 제1 맵리듀스 모듈(301)은 추출된 중복 없는 리터럴(Literal)에 부여된 글로벌-ID와, 글로벌-ID에 매칭되는 리터럴 정보를 밸류(value)로 생성할 수 있다. 제1 맵리듀스 모듈(301)이 수행하는 글로벌 ID 생성 맵리듀스 잡은 다음과 같다. 도 8은 본 발명의 실시예에 따른 제1 맵리듀스 모듈에서 수행하는 글로벌(Global) ID 생성 맵리듀스 잡의 동작 과정을 나타내는 도면이다.

[0062] <글로벌 ID 생성 맵리듀스 잡(1st MapReduce Job)>

[0063] 첫 번째 수행되는 맵리듀스 잡은 글로벌 ID 생성 잡으로, 전체 RDF 트리플에 포함된 주어, 서술어, 목적어를 중복 없는 리터럴(Literal)로 추출하여, 각 리터럴(Literal)을 구별할 수 있는 정수 자료형의 글로벌 ID를 부여하는 작업이다.

[0064] 맵 모듈(3011)은 입력 RDF 트리플 데이터를 일정한 크기의 블록으로 나누어 입력받는다. 각 맵 모듈(3011)은 입력받은 트리플 데이터를 주어, 서술어, 목적어 단위로 분할하고, 각 리터럴(Literal)을 중간 결과의 Key로, 매퍼의 태스크 ID를 중간 결과의 Value로 작성한다. 또한, 입력받은 데이터 블록에 매퍼의 태스크 ID를 부여하여 HDFS에 바로 쓰기를 수행한다.

[0065] 맵 모듈(3011)이 작성한 중간 결과는 같은 Key, 즉 같은 리터럴(Literal)끼리 그룹핑되어 리듀서 모듈(3013)에게 전달된다. 리듀서 모듈(3013)은 중복이 없는 리터럴(Literal)을 입력받으며, 자신에게 전달된 리터럴(Literal)에 대한 로컬 정수 ID를 생성한다. 로컬 정수 ID는 다른 리듀서 모듈(3013)에 배정된 리터럴(Literal)과 중복이 될 가능성이 높다.

[0066] 따라서, 리듀서 모듈(3013)의 태스크 ID (1byte)와 로컬 정수 ID (7byte)를 결합하고 중복되지 않는 글로벌 정수 ID를 생성하여 최종적으로 리터럴(Literal)에 부여한다. 글로벌 ID를 생성한 후, 리듀서 모듈(3013)은 리터럴(Literal)이 전송된 맵 모듈(3011)의 ID 리스트, 글로벌 ID, 리터럴(Literal)을 최종 결과로 쓴다.

[0067] 본 실시예에 따르면 맵리듀스 잡을 사용함으로써, 대량의 RDF 데이터에 대해 중복이 없는 리터럴(Literal)을 빠르고 간단하게 추출한다. 이에 대해서 각 리터럴(Literal)을 구별하기 위해 앞에 1byte를 각 리듀서 모듈(3013)의 ID를 부여함으로써 리터럴(Literal)의 중복이 없도록 한다. 이렇게 생성된 첫 번째 잡의 결과는 이후 제2 맵리듀스 모듈(303)에서 활용된다.

[0068] 도 9는 본 발명의 실시예에 따른 제2 맵리듀스 모듈(303)에서 수행하는 트리플 변환 잡 동작 과정을 나타내는 도면이다. 도 9를 참조하면, 제2 맵리듀스 모듈(303)은 제1 맵리듀스 모듈(301)에서 생성한 밸류의 정보 중 특정 RDF 블록에 속하는 리터럴의 글로벌-ID 정보만을 저장하여 변환 테이블(3031)을 구축할 수 있다. 두 번째 수행되는 맵리듀스 잡은 트리플 변환 잡으로, 첫 번째 맵리듀스 잡에서 생성한 결과, 즉 글로벌 ID 값을 사용하여 RDF 트리플 데이터를 변환하는 작업이다.

[0069] <변환 테이블을 생성하는 두 번째 맵리듀스 잡(2nd MapReduce Job)>

[0070] 맵 모듈(3011)은 첫 번째 잡의 결과를 입력받는다. 각 맵 모듈(3011)은 입력받은 리터럴(Literal)의 Key로 레이블 되어있는 맵 모듈(3011) ID의 리스트를 분할하고, 각 맵 모듈(3011) ID를 중간 결과의 Key로, 글로벌 ID와 Literal을 중간 결과의 Value로 작성한다.

[0071] 맵 모듈(3011)들이 작성한 중간 결과는 같은 Key, 즉 같은 매퍼 ID(첫 번째 잡의 매퍼 ID)끼리 그룹핑되어 리듀스 모듈(3013)에게 전달된다. 리듀서 모듈(3013)은 맵 모듈(3011) ID를 가진 여러 개의 글로벌 ID와 리터럴(Literal)을 입력받는다.

[0072] 해당 리터럴(Literal)들은 이전 잡에서 같은 맵 모듈(3011)이 읽은 RDF 데이터에 포함되는 리터럴(Literal)을 의미한다. 따라서, 리듀서 모듈(3013)은 해당 리터럴(Literal)과 글로벌 ID을 기반으로 매핑 테이블을 생성한다. 리듀스 모듈(3013)가 생성하는 ID 매핑 테이블은 앞에서 설계한 해시 테이블 기반의 매핑 테이블을 나타낸다.

[0073] 그 후, 리듀스 모듈(3013)은 이전 단계에서 HDFS에 쓰기를 수행한 RDF 데이터 블록 중 리듀서 모듈이(3013) 읽은 Key와 동일한 블록 ID를 가진 블록을 읽으며, 블록에 저장된 트리플 데이터를 생성한 매핑 테이블을 사용하여 변환한다. 변환된 RDF 데이터는 리듀서에 의해 최종 결과로 쓰여지게 된다.

[0074] 두 번째 맵리듀스 잡을 통해서, 각 리듀스 모듈(3013)이 존재하는 분산된 노드는 전체 리터럴(Literal)에 대한 모든 글로벌 ID 정보를 메모리에 저장할 필요 없이, 첫 번째 잡의 맵 모듈(3011)이 생성한 특정 RDF 블록에 속하는 리터럴(Literal)의 글로벌 ID 정보만을 저장한다.

[0075] 따라서, 대량의 RDF 데이터에 대해서 메모리 문제가 발생하지 않고, 분산으로 RDF 데이터를 나누어 변환함으로써 빠르게 데이터 변환이 가능하다.

[0076] 이와 같이 본 발명의 실시예에 따른 대량 LOD 저장을 위한 맵리듀스 기반 데이터 변환 방법은 대량의 RDF 데이터를 효과적으로 압축 및 저장하는 것으로, RDF 데이터를 저장하는 관리 시스템에 활용될 수 있다. 또한, 많은 리터럴(Literal) 값을 가지는 그래프 형태의 데이터를 저장하는 관리 시스템에 활용될 수 있다.

[0077] 따라서 본 발명의 실시예에 따른 대량 LOD 저장을 위한 맵리듀스 기반 데이터 변환 방법은 매핑 테이블을 여러 노드에서 생성 및 관리함으로써, 대량의 RDF 데이터를 변환할 때 종래 발생하던 메모리 부족 문제를 해결할 수 있는 효과가 있다.

[0078] 본 발명의 다른 실시예로, 대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 방법은 전술한 도 2의 실시예에 따른 대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 시스템에서 수행되는 수행 단계가 될 수 있다. 보다 상세하게, 대량의 LOD(Linked Open Data) 저장을 위한 맵리듀스 기반 데이터 변환 방법은 LOD 데이터의 RDF(Resource Description Framework) 포맷의 자원 중 식별을 위한 트리플(Triple) 정보를 압축하기 위해, 트리플 정보의 리터럴(literal)에 정수값의 ID를 부여하여 리터럴과 정수값의 ID를 매핑한 매핑 테이블(101)을 구축하는 (a)단계를 포함할 수 있다. (a)단계는 전술한 리터럴-ID 생성 모듈(10)에서 수행되는 동작일 수 있다.

[0079] 맵리듀스 기반 데이터 변환 방법은 (a)단계에서 구축된 매핑 테이블(101)에서, ID의 정수값이 부여된 리터럴을 중복 없는 리터럴로 추출하여, 추출된 리터럴을 구별할 수 있는 정수 자료형의 글로벌-ID를 부여하는 분산 데이터 처리를 수행하여 대용량의 RDF 데이터를 변환하는 (b)단계를 포함할 수 있다. (b)단계는, LOD 데이터의 RDF 포멧을 글로벌-ID로 압축 변환된 변환 테이블을 구축할 수 있다. (b)단계는 전술한 맵리듀스 모듈(30)에서 수행되는 동작일 수 있다.

[0080] 이상에서 대표적인 실시예를 통하여 본 발명을 상세하게 설명하였으나, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 상술한 실시예에 대하여 본 발명의 범주에서 벗어나지 않는 한도 내에서 다양한 변형이 가능함을 이해할 것이다. 그러므로 본 발명의 권리 범위는 설명한 실시예에 국한되어 정해져서는 안 되며, 후술하는 특허청구범위뿐만 아니라 특허청구범위와 균등 개념으로부터 도출되는 모든 변경 또는 변형된 형태에 의하여 정해져야 한다.

부호의 설명

[0082] 1: LOD 저장 시스템

10: 리터럴-ID 생성 모듈

101: 매핑 테이블

30: 맵리듀스 모듈

301: 제1 맵리듀스 모듈

3011: 맵 모듈

3013: 리듀스 모듈

303: 제2 맵리듀스 모듈

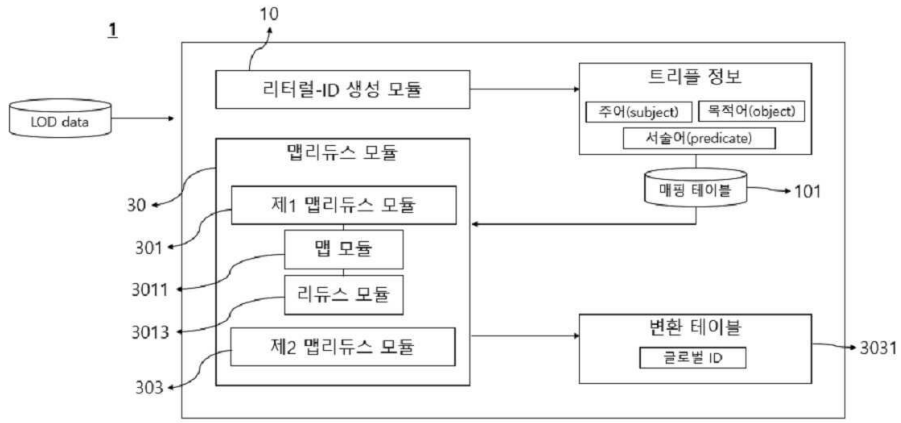
3031: 변환 테이블

도면

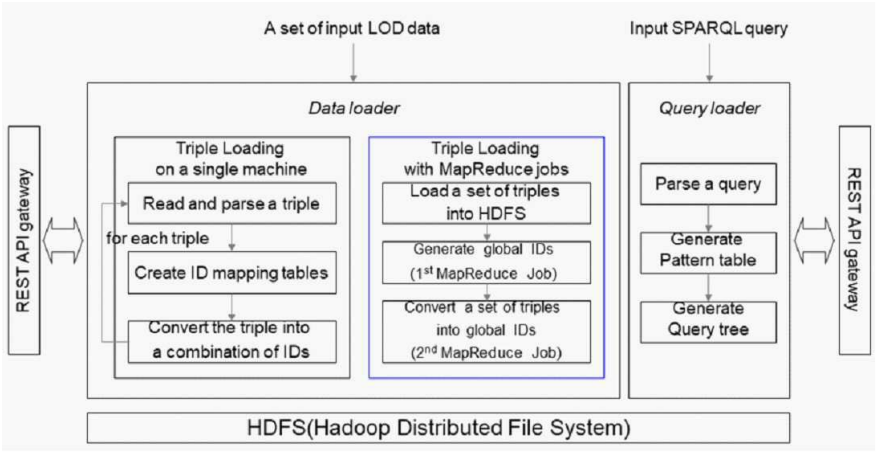
도면1



도면2



도면3



도면4a

[Subject/Object-ID 테이블]

Hash Key (Subject/Object Data)	Hash Value (Subject/Object ID)
http://www.Department0.University0.edu/Lee	1
http://www.Department0.University0.edu/Professor	2
...	...

[ID-Subject/Object 테이블]

Hash Key (Subject/Object ID)	Hash Value (Subject/Object Data)
1	http://www.Department0.University0.edu/Lee
2	http://www.Department0.University0.edu/Professor
...	...

도면4b

[Predicate-ID 테이블]

Hash Key (Predicate Data)	Hash Value (Predicate ID)
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>	1
http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#emailAddress	2
...	...

[ID-Predicate 테이블]

Hash Key (Predicate ID)	Hash Value (Predicate Data)
1	<http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
2	http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#emailAddress
...	...

도면4c

[Subject/Object-ID 테이블]

Hash ID (Subject/Object Data)	Hash Value (Subject/Object ID)
1#Lee	1
1#Professor	2
...	...

[ID-Subject/Object 테이블]

Hash ID (Subject/Object ID)	Hash Value (Subject/Object Data)
1	1#Lee
2	1#Professor
...	...

도면4d

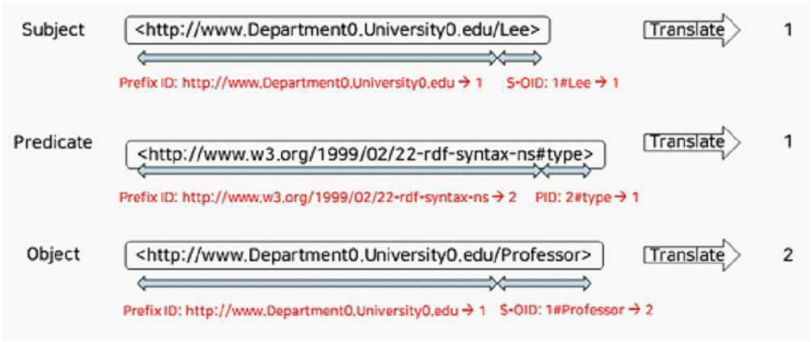
[Predicate-ID 테이블]

Hash ID (Predicate Data)	Hash Value (Predicate ID)
2#type	1
3#emailAddress	2
...	...

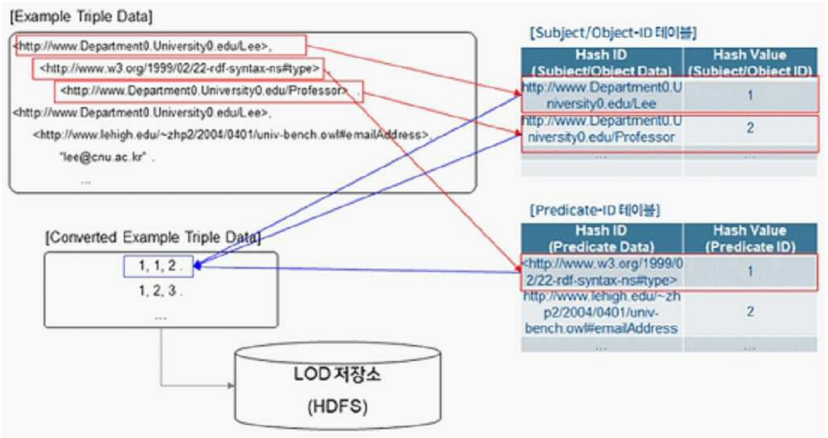
[ID-Predicate 테이블]

Hash ID (Predicate ID)	Hash Value (Predicate Data)
1	2#type
2	3#emailAddress
...	...

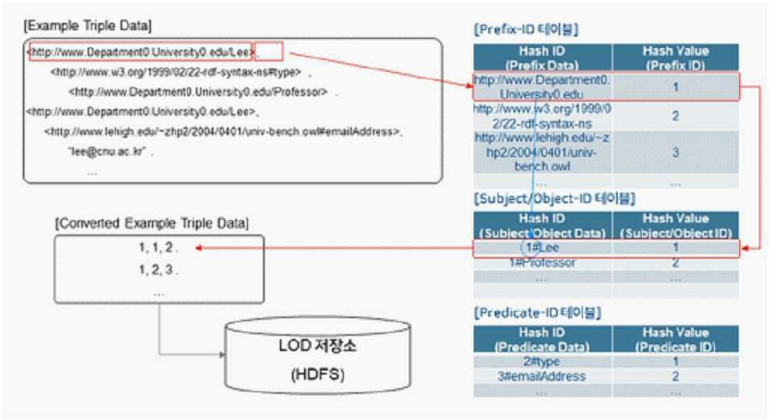
도면5



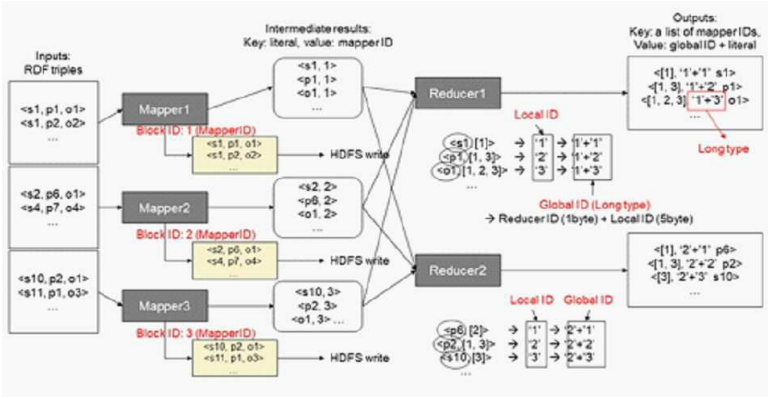
도면6



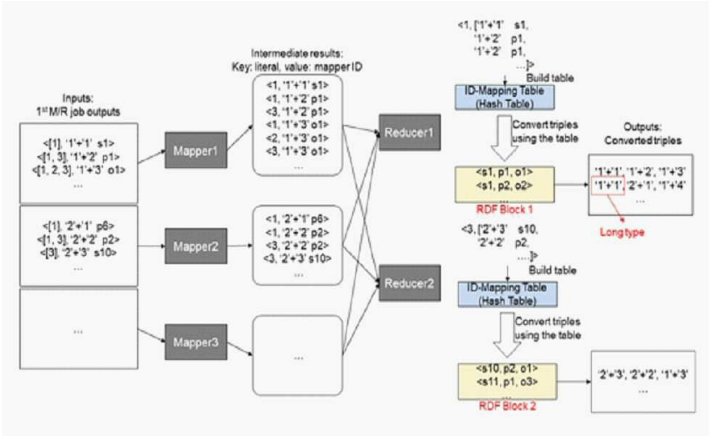
도면7


[Example Triple Data]
<http://www.Department0.University0.edu/Lee>.
<http://www.w3.org/1999/02/22-rdf-syntax-ns#type> .
<http://www.Department0.University0.edu/Professor> .
<http://www.Department0.University0.edu/Lee>,
<http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl#emailAddress>,
"lee@cnu.ac.kr" .
[Converted Example Triple Data]
1, 1, 2 .
1, 2, 3 .
LOD 저장소
(HDFS)
[Prefix-ID 테이블]
Hash ID (Prefix Data)
Hash Value (Prefix ID)
http://www.Department0.University0.edu
1
http://www.w3.org/1999/02/22-rdf-syntax-ns
2
http://www.lehigh.edu/~zhp2/2004/0401/univ-bench.owl
3
[Subject/Object-ID 테이블]
Hash ID (Subject/Object Data)
Hash Value (Subject/Object ID)
1#Lee
1
1#Professor
2
[Predicate-ID 테이블]
Hash ID (Predicate Data)
Hash Value (Predicate ID)
2#type
1
3#emailAddress
2

도면8


Inputs:
RDF triples
Intermediate results:
Key: literal, value: mapperID
Outputs:
Key: a list of mapperIDs,
Value: globalID + literal
Mapper1
Mapper2
Mapper3
Block ID: 1 (MapperID)
Block ID: 2 (MapperID)
Block ID: 3 (MapperID)
HDFS write
Reducer1
Reducer2
Local ID
Global ID (Long type)
→ ReducerID (1byte) + LocalID (5byte)
Long type

도면9


Inputs:
1st M/R job outputs
Intermediate results:
Key: literal, value: mapperID
Mapper1
Mapper2
Mapper3
Reducer1
Reducer2
Build table
ID-Mapping Table (Hash Table)
Convert triples using the table
Outputs:
Converted triples
RDF Block 1
RDF Block 2
Long type