Jena2 기반의 효율적인 OWL Ontology 관리를 위한 저장 모델*

신희영[O1] 정동원[2] 김진형[1] 백두권[1]
[1]고려대학교 컴퓨터학과
{shintul, koolmania, baikdk}@korea.ac.kr
[2]국립군산대학교 정보통계학과
djeong@kunsan.ac.kr

Efficient OWL Ontology Storage Model based on Jena2

Heeyoung Shin[O1] Dongwon Jeong[2] Jinhyung Kim[1] Doo-Kwon Baik[1]
[1]Dept. of Computer Science and Engineering, Korea University
[2]Dept. of Informatics & Statistics, Kunsan National University

요 약

W3C에서 표준 온톨로지 언어로 OWL을 채택함에 따라 많은 온톨로지들이 OWL로 기술 및 구현되고 있다. 이에 따라 대용량의 OWL 문서를 효율적으로 저장하고, 검색할 수 있는 모델에 대한 필요성이 제기되고 있으며, Jena, Protégé, Sesame, FacT 등 다양한 프레임워크가 제안되어 활발히 연구가 진행되고 있다. 이 논문에서는 기본적인 Jena2의 저장소 모델이 단일 테이블에 문서의 정보를 저장하여 대용량의 OWL데이터의 처리에 있어 성능이 저하되는 문제점을 해결하여 대용량의 OWL 문서의 효율적인 저장, 관리, 질의 가능한 OWL 온톨로지 관계형 데이터베이스 모델을 제안한다. 또한 OWL 온톨로지 관계형 데이터베이스 모델을 위한 어댑터 및 컨버터를 제안한다.

키워드 : 온톨로지, Jena, OWL, 관계형 데이터베이스 모델

1. 서 론

최근 시맨틱 웹에 대한 관심이 증대되면서 World Wide Web Consortium(W3C)표준으로 규정된 시맨틱 웹 온톨로지 언어(RDF, RDFS, OWL 등) 및 관련 기술에 대한 연구가 활발히 진행되고 있다. 시멘틱 웹 [2]이란 컴퓨터가 인간처럼 웹 상에 존재하는 자원들을 인지하고 논리적 추론을 통해 자원들간의 관계를 파악하여 사용자가 요구하는 정보에 대한 정확한 결과를 찾아주는 차세대 지능형 웹을 의미한다. 시멘틱 웹의 주요 기술로는 온톨로지 기술이 있으며, 온톨로지는 웹 상의 자원들을 정의하고 자원들이 어떻게 연관되어 있는지를 표현한다. 컴퓨터가 이해할 수 있는 형태로 온톨로지 정보를 표현하기 위해서는 온톨로지 언어가 필요하며, 온톨로지를 기술하기 위한 대표적인 언어로는 Resource Description Framework (RDF) [3], Resource Description Framework Schema(RDFS) [4], DAML+OIL [6], Web Ontology Language (OWL) [5] 등이 있다. 또한 OWL, DAML, OIL은 RDF, RDFS와는 달리 추가적으로 추론 기능을 지원한다. 특히 OWL은 RDFS와 DAML+OIL에서 확장된 언어로서 보다 풍부한 단어집을 보유한다. 따라서 다양하고 세밀하게 자원들의 개념 및 관계를 기술할 수 있고, 추론을 지원함으로써 강력한 표현력을 지닌다. W3C가 OWL을 표준 온톨로지 언어로 지정함에 따라 추후 대부분의 온톨로지 데이터들은 OWL로 기술될 전망이다. 이로 인해, 최근 RDF, RDFS, OWL, SPARQL [1]등의 언어를 위한 프로그램 환경에서 규칙 기반 추론 엔진을 포함하고 있는 Java 프레임워크인 Jena [7]에 대한 관심이 높아지고 있다.

Jena2는 OWL 데이터에 대한 저장모델 자동 생성 시 물리적 스키마 구조가 단순하고, 이로 인해 많은 OWL 관련 시스템 개발에 이용되고 있다. 하지만 비정규화된 단일구조로서 단일테이블에 정보들이 저장함에 따라 대용량의 OWL 데이터에 대한 저장 및 질의 시 성능이 저하되는 문제점을 가지고 있다 [10],[11].

이 논문에서는 단순 선택 연산 (Simple Selection)은 물론 조인 연산이 요구되는 질의에 대한 성능이 저하되는 문제를 해결하고, 기본적인 Jena2 API를 그대로 사용하면서 보다 효율적으로 저장 및 질의처리를 제공하는 저장 모델 개발과 이를 위한 어댑터(Adapter)및 컨버터(Converter)를 제안한다.

2. 관련 연구

2.1 Jena2 프레임워크

* 이 연구에 참여한 연구자는 'BK21 2단계 사업'의 지원을 받았음

이 장에서는 Hewlett-Packard 연구소에서 개발된 시멘틱 웹 프레임워크인 Jena2의 기본적 아키텍처와 Jena2 API의 RDBMS 기반 저장 모델에 대해서 기술한다.

2.1.1 Jena2 구조

그림 1은 Jena2의 전체적인 구조를 보여준다.

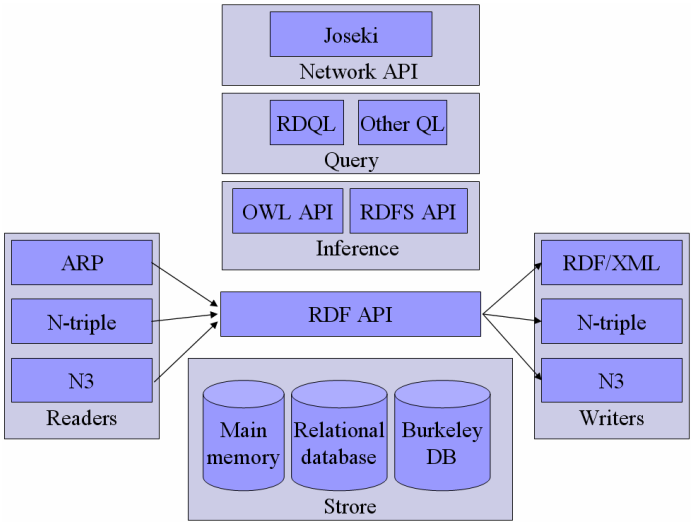

그림 1 Jena2 구조

Jena2는 자바로 구현되어 자바언어가 갖는 모든 이점을 가지고 있으며, 기본적인 RDF 파서도 제공한다. 또한 내부적인 추론은 그래프 매칭 방법을 이용하여 접근한다. Jena2 프레임워크의 가장 주요부분은 RDF API이다. 이 외에 추론 관련 API, 질의 언어, 네트워크 API 및 저장소 등으로 구성되어 있다.

RDF API는 RDF 그래프에 대한 생성, 처리, 그리고 질의를 지원하고 메인 메모리와 관계형 데이터베이스 같은 여러 저장기술들을 지원한다. 그림 1과 같이 Readers와 Writers는 RDF API를 통해서 접근 할 수 있다. ARP는 Jena2에 포함되어 있는 RDF/XML 파서이며, RDF/XML문법으로부터 생성된다. 그림 1에서 RDF API 상위에는 Inference, Query, Network등 3개의 layer가 존재한다. Inference layer는 RDF그래프에 정보를 추가하여 사용된다. 예를 들어 RDF-S와 DAML은 기존의 존재하는 RDF 그래프에 정보를 추가하여 하위클래스의 관계로 구현되어 있다. 질의 계층 (Query Layer)는 RDQL(RDF Data Query Language) [8]이라 불리 우는 RDF그래프를 위한 질의 언어와 다른 질의 언어 등으로 구성된다. 시멘틱 질의 언어의 핵심은 RDF 트리플에 대한 패턴을 기술함으로써 원하는 트리플 정보를 검색할 수 있다. 따라서, 기존의 인덱스는 단일 트리플을 효율적으로 검색하는 데 초점을 두고 있다. 네크워크 계층 (Network Layer)는 원격 RDF데이터베이스를 갱신 시킬 수 있도록 어플리케이션의 질의를 할 수 있도록 지원한다.

이처럼 Jena2는 시멘틱 웹 환경의 어플리케이션을 개발하기 위한 많은 API들을 제공한다.

2.1.2 Jena2 저장소 구조

Jena2는 관계형 데이터베이스에서 RDF의 영속적 저장에 대해서 제공한다. 그림 2는 특정 OWL파일을 로드 한 후 Relational Database API를 이용하여 상용데이터베이스에 데이터를 저장하는 스키마를 나타낸다. Jena2 프레임워크를 통해서 OWL파일을 저장할 때 총 7개의 테이블만이 생성된다. 문서에 대한 실제 데이터정보는 "jena_gntn_stmt"의 단일 테이블에 저장된다. 읽어 들인 OWL파일의 분석된 데이터 정보들이 subject-property-object의 컬럼으로 구성된 하나의 테이블에 분류되어 저장된다.

"jena_gntn_stmt" 테이블에 저장될 값들 중 일정 크기 이상의 데이터가 들어갈 경우에 테이블 크기가 너무 커지는 것을 막기 위해 리터럴과 URI정보가 저장되는 별도의 테이블인 "jena_long_lit" 와 "jena_long_uri"를 구성하고 이를 참조하는 형태로 테이블이 구성되어 있다. 그리고 공통적으로 사용하는 접두사에 대해서는 "jena_prefix" 테이블에 저장하고 이를 참조하도록 한다. Jena2의 핵심 구조는 노드들의 트리플 집합체인 RDF 그래프이며 "jena_gntn_stmt" 테이블이 Statement의 트리플 데이터를 저장하는 구조를 갖는다.

이는 단일 테이블로서 질의가 표현되는 경우에는 테이블간의 상관관계로 인해 단순정보검색을 위한 질의 시 속도가 저하되고, 조인 연산 시 두 테이블의 크기가 크기 때문에 조인비용이 많이 발생하는 문제점이 야기된다.

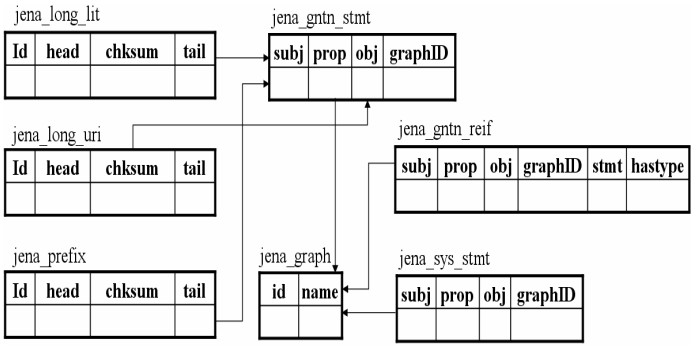

그림 2 Jena2 관계형 데이터베이스 저장 스키마

3. JeSPi

이 장에서는 2장에서 언급한 기존 Jena2 시스템의 문제점을 보완하기 위해 JeSPi (Jena Storage Plug-in for Enhanced Query Processing)을 제안한다. JeSPi는 Jena2에 Plug-in형식으로 추가하여 제안하는 최적화 OWL 온톨로지 관계형 데이터베이스 모델을 생성 하여 효율적으로 저장할 수 있도록 지원한다. JeSPi를 구성하고 있는 최적화의 OWL 온톨로지 관계형 데이터베이스 모델의 구조에 대해서 서술하고 JeSPi의

핵심 컴포넌트인 어댑터와 컨버터의 구조 및 기능에 대해서 구체적으로 기술한다.

3.1. JeSPi 개념구조

그림 3은 기존의 Jena2 API를 그대로 이용하면서 보다 효율적인 저장과 질의를 위한 새로운 관계형 데이터베이스 저장 개념구조 이다.

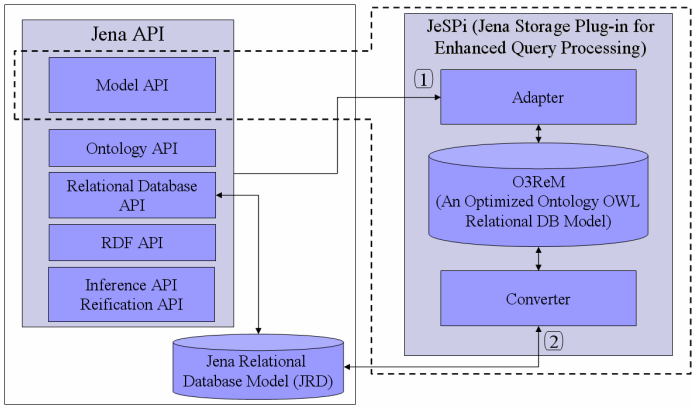

그림 3 JeSPi 개념구조

Jena2 API를 이용한 JeSPi는 두 가지 컴포넌트를 통해서 실행 가능하다. 첫 번째 컴포넌트는 Jena2 API를 이용하여 OWL 문서가 제안하는 어댑터를 거쳐 새로운 개념의 OWL 온톨로지 관계형 데이터베이스 모델 형태로 관계형 데이터베이스에 저장된다. 두 번째 컴포넌트는 Jena2 API를 이용하여 이미 관계형 데이터베이스에 저장된 모델을 마이그레이션(Migration) 하기 위한 컨버터를 거친 후 최적화 OWL 온톨로지 관계형 데이터베이스 모델로 변환한다. 두 가지 컴포넌트 모두 기존의 Jena2 API를 이용하여 제안하는 개념을 Plug-In 형식으로 적용함으로써 최적화 OWL 온톨로지 관계형 데이터베이스 모델을 생성 및 변환할 수 있다.

결과적으로, JeSPi는 새롭게 저장되는 OWL문서와 기존에 이미 저장되었던 OWL문서의 관계형 정보들을 모두 최적화 OWL 온톨로지 관계형 데이터베이스 모델로 수용할 수 있는 장점을 지닌다.

3.1.1 O3ReM 구조

이 절에서는 앞서 언급한 컴포넌트들을 통해 생성할 수 있는 최적화의 OWL 온톨로지 관계형 데이터베이스 모델에 대해서 구체적으로 기술한다.

제안하는 O3ReM (Optimized OWL Ontology Relational Database Model)은 기존의 Jena2 프레임워크에서 단일테이블로 생성되었던 구조를 OWL 문서의 의미론적 입장에서 Class, Property, Individual로 분류하여 구조화된다. 그림 4는 제안하는 O3ReM의 구조를 세 개의 계층으로 분류하여 기술한다.

제안하는 OWL 온톨로지 관계형 데이터베이스 모델의 테이블 구조에 대한 적용 범위는 OWL DL에만 한정한다. OWL Full에서는 각 요소의 관계가 유연하기 때문에 Class와 Property 그리고 Individual간의 관계를 큰 제약조건 없이 표현 할 수 있기 때문이다.

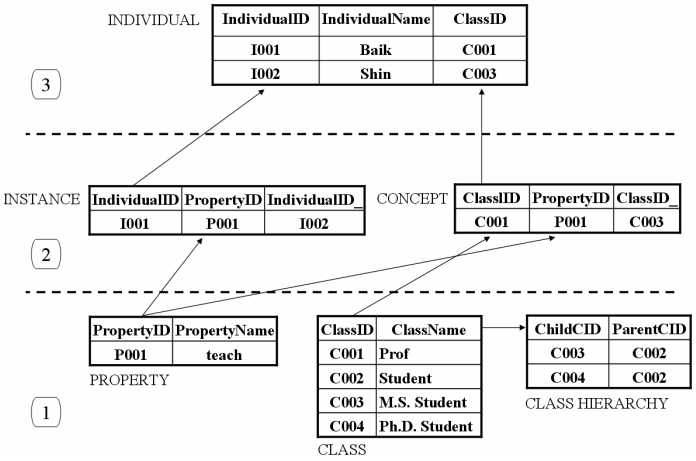

그림 4 최적화 OWL 온톨로지 관계형 데이터베이스 모델 스키마

그림 4는 데이터 정보들이 단일 테이블로 저장되었던 기존 Jena2에서의 저장구조와는 다르게 여러 개의 테이블로 구성되어 있다. OWL 문서의 의미를 Class, Property, Individual로 분류하여 각각의 데이터 정보들을 테이블에 저장한다. 1계층은 Property와 Class의 해당 정보에 대한 테이블로 구성되며, 각 테이블은 주 키 값인 ID와 데이터 값인 Name 컬럼으로 구성된다. 또한 각 클래스 간의 부모에 해당하는 Class와 부모에게 포함되어 있는 자식관계에 대해서는 CLASS HIERARCHY를 통하여 표현할 수 있다. 이는 질의에 포함되어 있는 특정 클래스의 하위 클래스까지 추론하여 보다 정확한 질의결과를 생성할 때 빠른 연산을 가능하게 한다. 이 외에도 OWL에서 정의하고 있는 다양한 제약조건 (Restrictions, Axioms)을 정의하고 있다. 이러한 조건에 대한 판단이 요구되는 질의를 보다 빠르게 처리할 수 있도록 하기 위해서는 이러한 조건들을 추가적인 테이블을 통해 정의하여 관리할 필요가 있다. 그러나 이 논문에서는 rdfs:subClassOf 조건만을 다루고 이 외의 조건들에 대해서는 향후 연구 내용을 남겨 둔다. 2계층에서는 Individual-Property-Individual 형태의 Instance와 Class-Property-Class형태의 Concept이 각각의 테이블로 구성되어 있다. 마지막으로 3계층은 Class와 Individual의 관계를 매핑 시킬 수 있는 테이블로 구성된다.

예를 들어 1계층에 클래스 테이블에는 “Prof”와 “Student”의 데이터 레코드가 저장되고 프로퍼티 테이블에는 “teach”의 데이터 레코드가 저장되어 있다. 이때 2계층에서는 1계층에서 삽입된 레코드정보를 이용하여 클래스와 클래스간의 관계를 표현할 수 있다. 이어서 3계층의 테이블에서는 Individual의 데이터 레코드인 “baik”과 “shin”을 Is-a관계의 Class와 함께

저장됨으로써 Individual-Property-Class의 관계를 표현한다. 즉, individual의 레코드 데이터 "baik"과 class의 레코드 데이터 "Prof"와의 관계를 파악할 수 있으며, individual의 레코드 데이터 "shin"과 class의 레코드데이터 "student"와 관계 역시 파악할 수 있다. 마지막으로 2계층에서의 Individual간의 관계를 Individual-Property-Individual의 테이블 구조로 표현하여 "baik"-"teach"-"shin"의 관계와 의미를 알 수 있다.

3.1.2 어댑터 (Adapter)

그림 5는 그림 3에서 Jena2 API를 이용하여 JeSPi를 구성할 수 있도록 제시한 첫 번째 컨포넌트인 어댑터 구조이다. JeSPi에 로드된 OWL Data Sets은 Jena2 API에 의해 기본적인 온톨로지 모델을 생성한다. 온톨로지 모델은 어댑터의 추출 및 변환 단계를 거쳐 최적화 OWL 온톨로지 관계형 데이터베이스 모델을 변환된다. OWL 온톨로지 관계형 데이터베이스 모델을 기반으로 관계형 데이터베이스 테이블을 구조화하여 생성한 후 데이터들을 지정된 규칙기반으로 저장한다.

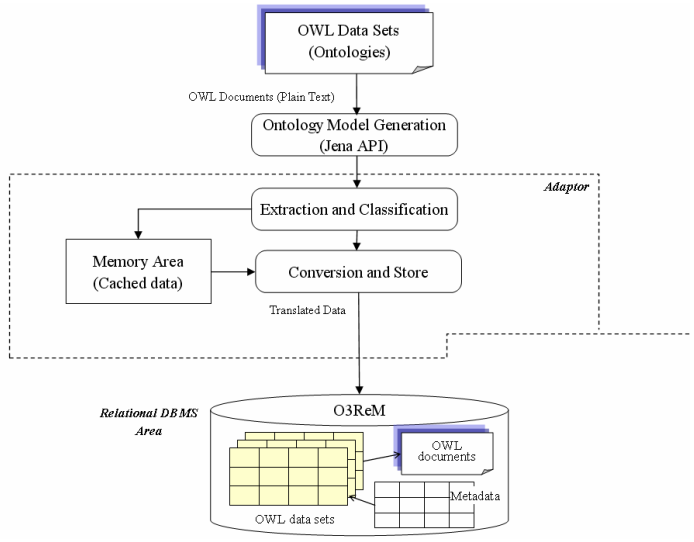

그림 5 어댑터 구조

어댑터는 OWL 데이터 정보에 대해 추출하고 분류하는 부분과 데이터를 임시 저장할 수 있는 메모리 부분, 그리고 관계형 데이터베이스에 변환하여 저장할 수 있도록 지원하는 부분 등 세 부분으로 구성되어 있다.

OWL 데이터 정보에 대해서 추출하고 분류하는 부분에서는 Jena2 프레임워크의 API를 이용하여 읽어들인 OWL Data Sets의 의미를 분석한 후 요구하는 정보들을 그림 4의 최적화의 OWL 온톨로지 관계형 데이터베이스 모델의 테이블 구조로 저장될 수 있도록 정보를 분류한다. 분류된 데이터 정보들을 임시로 저장할 경우 Memory Area 데이터를 보관한다. 최적화 OWL 온톨로지 관계형 데이터베이스 모델을 생성하고 테이블을 생성하는 과정을 거쳐 분류된 데이터 정보들을 테이블에 저장한다. 이와 같이 OWL Data Sets을 어댑터의 저장 프로세스를 통해 최적화된 OWL 온톨로지 관계형 데이터베이스에 저장하게 되면 기존의 Jena2 프레임워크를 그대로 사용하며, 기존의 단일화로 구성된 저장모델의 단점을 보완할 수 있다.

3.1.3 컨버터 (Converter)

그림 6은 그림 3에서 Jena2 API를 이용하여 JeSPi를 구성할 수 있도록 제시한 두 번째 컨포넌트인 컨버터 구조이다. 컨버터 기존 Jena2 시스템에 의해 이미 관계형 데이터베이스에 저장된 데이터들에 대해 정해진 규칙을 적용하여 새롭게 제안하는 OWL 온톨로지 관계형 데이터베이스 모델로 변환한다.

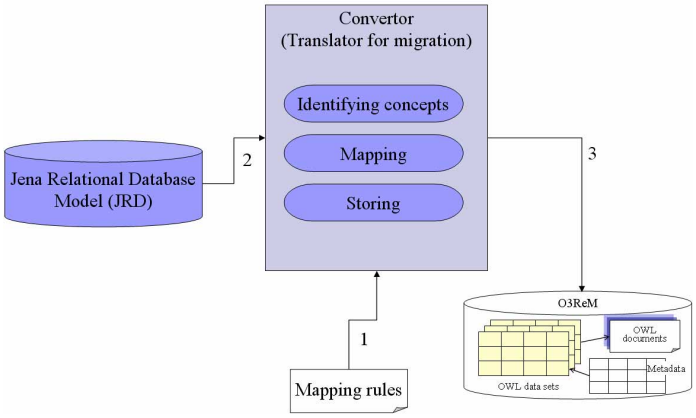

그림 6 컨버터 구조

변환되는 전체적인 과정을 살펴보면, Jena2를 이용하여 관계형 데이터베이스 모델에 이미 정의된 구조를 최적의 OWL 온톨로지 관계형 데이터베이스 모델로 재생성 하기위해서 마이그레이션 할 수 있는 매핑 룰(Mapping rules)을 적용한다. 그 후에 기존의 Jena2 관계형 데이터베이스 모델의 데이터들을 컨버터의 세부프로세스를 통해 기존 저장된 데이터 정보에 대해서 분석 및 분류작업을 한다. 분류된 데이터 정보들은 새롭게 제안하는 최적화 OWL 온톨로지 관계형 데이터베이스 모델로 변환한다. 컨버터의 세부 과정에 대해서 조금 더 구체적으로 기술하면 다음과 같다. 먼저 기존에 저장된 Jena2 관계형 데이터베이스 모델에 저장된 구조와 데이터들을 분석하여 각 구성의 관계를 확인한다. 그 이후 Jena2 관계형 데이터베이스 모델의 데이터들을 최적화 OWL 온톨로지 관계형 데이터베이스 모델로 어떻게 재구성 될지에 대한 세부적인 매핑 정보들을 적용한다. 마지막으로 매핑 정보를 바탕으로 최적화 OWL 온톨로지 관계형 데이터베이스 모델로 재생성한다. 이는 Jena2를 이용하여 이미 저장된 단일 테이블의 관계형 데이터베이스 구조의 단점을 보완할 수 있다

4. 분석 및 비교평가

Jena2 API를 이용한 관계형 데이터베이스 저장 모델과 제시한 최적화 OWL 온톨로지 관계형 데이터베이스 모델의 성능 및 비교평가는 표 1과 같다.

표 1 비교 평가

비교항목	Jena 저장 모델	제안모델
검색 (질의) 속도	느림	빠름
데이터 중복성	높음	낮음
데이터 이해의 용이성	낮음	높음
질의 모델링의 편의성	낮음	높음
모델 변환의 용이성	낮음	높음

Jena1에서는 트리플 데이터(subject-predicate-object)를 저장하기 위해 정규화된 데이터베이스 스키마를 사용하였기 때문에 저장 크기 면에서는 효율적인 장점이 있으나, 질의 시 Statement 테이블과 리터럴 테이블 그리고 리소스 테이블간에 많은 횟수의 조인연산을 필요로 해야 하는 단점이 발견된다. 그러나 Jena2의 경우 Jena1의 문제점을 해결하기 위해 의도적으로 비정규화를 시킴에 따라 Jena1과 비교하여 데이터의 검색 시간과 조인 연산의 횟수는 줄일 수 있는 장점이 있지만, 하나의 테이블 공간이 커지는 단점이 있고 이는 단순 선택 연산 시 성능이 저하되는 문제점을 지닌다. 또한 비정규화가 높아 여전히 조인 연산 시 불필요한 정보를 액세스하게 되고 이에 따른 성능 저하를 가져온다.

제안 모델은 OWL에서 정의한 Class, Individual, Property의 개념을 이용하여 OWL 문서의 데이터를 관계형 데이터베이스에 저장함으로써 단순정보검색 질의 시 Jena2에서 비 정규화된 테이블 구조로 저장할 때보다 질의 응답 속도를 향상 시킬 수 있다. 또한 조인 연산 시 두 테이블의 크기로 인하여 조인비용이 발생하는 문제점을 해결함으로써 빠른 검색 및 질의 속도를 보장할 수 있다. 기존 모델의 경우, 데이터 정보가 저장되는 주요 테이블이 단일로 구성 되어 있기 때문에 데이터에 대한 중복성이 높지만 제안된 모델의 경우에는 데이터의 속성을 분류함으로써 중복을 제거하여 데이터의 중복성이 낮다. 데이터 이해의 용이성이란 저장소의 데이터 구조가 쉽게 이해할 수 있도록 구성되어있는 정도를 의미한다. Jena2의 관계형 데이터베이스 모델은 단일 테이블로 저장되기 때문에 데이터에 대해서 직관적으로 파악하기 쉽지 않아서 사용자로 하여금 분류를 거친 후에 사용자가 이해할 수 있다. 하지만 제안되는 모델에서는 의미상으로 이해하기 편리하게 구조화 되어있기 때문에 사용자로 하여금 직관적으로 파악할 수 있도록 지원하며, 질의 모델링을 하는 개발자로 하여금 편리하게 모델링 할 수 있도록 제공된다. 또한 문서들에 대해서 의미상으로 직관적으로 분류되어 있기 때문에 다른 모델로 변환할 때에도 역시 용이하다. 마지막으로 제안한 저장 모델은 rdfs:subClassOf 조건을 별개의 테이블에 저장함으로써 계층 구조에 대한 조건 판단이 요구되는 질의에 대한 처리 속도를 향상시킬 수 있다.

5. 결론 및 향후 과제

이 논문에서는 Jena2의 관계형 데이터베이스 모델의 효율성을 높이기 위해 최적화 OWL 온톨로지 관계형 데이터베이스 모델을 제안하였다. 시맨틱 웹 온톨로지 언어로 기술된 OWL 문서의 저장, 관리, 질의처리 기법을 높이는 측면에서 많은 장점을 제공한다.

향후 연구 과제로는 특정 시스템이나 환경에 국한하여 저장 모델을 생성하는 방법이 아닌 다양한 환경에 따라 임의 저장 모델을 직접 정의하고 생성하며 관리할 수 있는 방법에 대한 연구가 요구되며, 현재의 Jena2 저장소 모델과의 정량적 비교평가 작업이 요구된다[9].

참고문헌

[1] SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query, Mar. 2007.

[2] Tim Berners-Lee, James Hendler, Ora Lassila, The Semantic Web, Scientific American, May. 2001.

[3] RDF/XML Syntax Specification, http://www.w3.org/TR/rdf-syntax-grammar, Feb. 2004.

[4] RDF Vocabulary Description Language 1.0: RDF Schema, http://www.w3.org/TR/rdf-schema, Feb. 2004.

[5] OWL Web Ontology Language Reference,http://www.w3.org/TR/owl-ref, Feb. 2004.

[6] DAML+OIL Reference Description W3C Note, http://www.w3.org/TR/daml+oil-reference, Dec. 2001.

[7] Jena - A Semantic Web Framework for Java, http://jena.sourceforge.net/

[8] RDQL - A Query Language for RDF, http://www.w3.org/Submission/RDQL, Jan. 2004.

[9] Jeen Broekstra, Arjohn Kampman, Frank van Harmelen, "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema," Lecture Notes In Computer Science (LNCS), Vol. 2342, pp. 54-68, Jun. 2002.

[10] Hak Soo Kim, Hyun Seok Cha, Jungsun Kim, and Jin Hyun Son, "Development of the Efficient OWL Document Management System for the Embedded Applications," Lecture Notes In Computer Science (LNCS), Vol. 3597, pp.75-84, Jul. 2005.

[11] Myung-Jae Park and Chin-Wan Chung, "Property Based OWL Storage Schema in Relational Databases," in Technical Report, CS/TR-2005-247, Div. of Computer Science, KAIST, Dec. 2005.