Journal of Digital Contents Society Vol. 9 No. 2 June. 2008(pp. 285-293)

# RDF/S 데이터의 관리를 위한 그래프 데이터 모델 기반 저장 기법

김연희*, 최재언**, 임해철***

**요 약**

차세대 웹으로 인식되고 있는 시맨틱 웹에서는 정보 리소스의 개념과 의미적 관계를 정의하는 메타데이터와 온톨로지의 역할이 무엇보다 중요시되고 있다. 따라서 RDF와 RDF 스키마 같은 시맨틱 웹 언어로 표현되어 있는 시맨틱 데이터의 효과적인 저장 및 검색 기법이 요구된다. 본 논문에서는 RDF와 RDF 스키마를 함께 고려하여 시맨틱 웹 데이터의 질의 유형을 크게 3가지로 분류하였다. 그리고 RDF와 RDF 스키마가 그래프 데이터 모델로 표현이 가능하고 다양한 질의의 형태가 그래프 모델에서 추출 가능한 경로식으로 표현될 수 있다는데 초점을 맞추고 효율적인 질의 처리를 위한 저장 기법을 제안한다. 본 논문에서 제안한 저장 구조는 RDF와 RDF 스키마에 기본적으로 기술되어 있는 클래스, 프로퍼티, 리소스에 대한 정의나 계층 정보는 물론 추출 가능한 모든 경로 정보를 유지할 수 있다. 본 논문에서 제안한 저장 기법을 통해 일반적인 질의 유형은 물론 그래프 모델 상의 복잡한 경로식으로 표현되는 질의 유형에 대해서도 테이블의 조인 횟수를 최소화하면서 특정 클래스나 프로퍼티로부터 시작하는 모든 경로에 대한 질의 처리가 가능한 장점이 있다.

# Storing Scheme based on Graph Data Model for Managing RDF/S Data

Youn Hee Kim*, Jae Yeon Choi**, Hae Chull Lim***

**Abstract**

In Semantic Web, metadata and ontology for representing semantics and conceptual relationships of information resources are essential factors. RDF and RDF Schema are W3C standard models for describing metadata and ontology. Therefore, many studies to store and retrieve RDF and RDF Schema documents are required. In this paper, we focus on some results of analyzing available query patterns considering both RDF and RDF Schema and classify queries on RDF and RDF Schema into the three patterns. RDF and RDF Schema can be represented as graph models. So, we proposed some strategies to store and retrieve using the graph models of RDF and RDF Schema. We can retrieve entities that can be arrived from a certain class or property in RDF and RDF Schema without a loss of performance on account of multiple joins with tables.

**Keywords : Semantic Web, RDF, RDF Schema, Storing Scheme**

## 1. 서론

현재 웹 환경에서는 정보 리소스간의 단순하고 물리적인 하이퍼링크 관계를 통해 사용자의 검색 요구에 대해 관련된 정보를 쉽게 제공하지만 정확하고 풍부한 정보의 검색과 정보의 의미 있는 해석이 어려운 한계를 가지고 있다. 따라서 사용자의 요구가 점차 다양해지고 보다 정확한 검색이 요구되면서 정보의 의미 있는 관계를 개념적으로 정의하고 추론 등을 통해 새로운 지식의 생성이 가능한 차세대 웹 환경이 요구된다. 이러한 요구에 따라 차세대 웹의 대안으로서 시맨틱 웹이 각광받고 있다.

시맨틱 웹에서는 메타데이터와 온톨로지를 이용하여 정보 리소스의 개념을 정의하고 정보 리

※ 제일저자(First Author) : 김연희
접수일자:2008년02월19일, 심사완료:2008년03월08일
* 부천대학 e-비즈니스과,
ellyblue06@gmail.com
** 홍익대학교 컴퓨터공학과
*** 홍익대학교 컴퓨터공학과
▣ 이 논문은 2006학년도 홍익대학교 학술연구진흥비에 의하여 지원되었음

소스간의 의미적 연관성을 표현함으로써 보다 지능적인 정보 검색은 물론 자동화된 다양한 웹 서비스를 제공할 수 있다[1]. 시맨틱 웹에서 이러한 정보 리소스의 개념과 의미적 관계를 기술한 것을 메타데이터라 하고 이러한 메타데이터의 의미를 보다 정확하게 정의한 것을 온톨로지라 한다. 온톨로지는 메타데이터를 기술할 때 발생할 수 있는 중첩성과 모호성을 없애기 위해 메타데이터를 기술할 때 사용되는 용어의 개념과 용어들 간의 관계를 정형화시킨다.

시맨틱 웹 환경에서 메타데이터와 온톨로지가 담당하고 있는 역할이 중요한 만큼 이러한 시맨틱 웹 데이터를 형식적으로 기술하기 위해 RDF (Resource Description Framework), RDF 스키마, DAML+OIL, OWL(Web Ontology Language)와 같은 다양한 언어들이 소개되어왔다. W3C에서 제안한 RDF와 RDF 스키마는 각각 메타데이터와 온톨로지를 기술하기 위한 기본적인 시맨틱 웹 언어이다[2][3]. 현재는 RDF와 RDF 스키마에 비해 풍부한 표현력과 모델링 요소를 가지고 있는 OWL이 W3C의 권고안으로 채택되어 관심을 받고 있지만 보편화되기까지 많은 시간이 소요될 것으로 예상되기 때문에 RDF와 RDF 스키마를 이용해 기술되어 있는 메타데이터와 온톨로지 데이터를 효율적으로 저장하고 처리하기 위한 연구가 무엇보다 요구된다.

RDF는 정보 리소스에 대한 메타데이터를 표현하기 위한 가장 기본적인 시맨틱 웹 언어로 정보 리소스의 의미와 다른 정보 리소스와의 의미적 관계를 기술하고 자동화된 추론을 지원하는데 목적을 두고 있다[2]. RDF 스키마는 온톨로지를 생성하기 위한 시맨틱 웹 언어로 특정 도메인에서 활용될 RDF를 기술하는데 사용될 어휘들을 정의하고 어휘들 간의 계층적인 관계를 구성하여 RDF로 기술된 메타데이터를 해석하는데 있어 모호성이나 중첩성을 없애는 역할을 담당한다[3].

시맨틱 웹에서 진보된 정보 검색 서비스를 제공하기 위해서는 RDF와 RDF 스키마로 기술된 메타데이터와 온톨로지 정보를 함께 이용해야 한다. 따라서 시맨틱 웹을 기술적인 측면에서 지원하기 위해서는 RDF와 RDF 스키마 정보를 효율적으로 관리하기 위한 연구가 필요하다. 그러나 지금까지 RDF와 RDF 스키마 정보 관리를 위한 대부분의 연구들은 트리플 문장 구조를 활용하여 저장과 검색을 단순화시키는 경우가 많았다. 하지만 RDF와 RDF 스키마는 그래프 모델로서 표현이 가능하고 사용자가 시맨틱 웹 환경에 적합한 다양한 질의 형태의 처리를 요구함에 따라 단순 트리플 문장 구조로 표현되는 기본적인 질의뿐만 아니라 그래프 모델 상에서 복잡한 경로로 표현되는 질의의 효율적인 처리가 가능한 관리 시스템의 개발이 요구된다.

따라서 본 논문에서는 RDF와 RDF 스키마를 위한 질의 형태를 분류하고 트리플의 기본 구조에 기반한 질의뿐만 아니라 RDF와 RDF 스키마의 그래프 모델에서 복잡한 경로로 표현되는 질의 처리를 지원하기 위한 저장 구조를 제안한다. 특히, 본 논문에서 제안한 저장 구조는 경로 정보를 이용하여 보다 다양한 질의 형태를 지원할 수 있다.

본 논문은 다음과 같이 구성된다. 2장에서는 RDF와 RDF 스키마의 데이터 모델과 기존 RDF와 RDF 스키마 데이터 관리 기법을 소개하고 비효율적인 질의 처리에 대한 문제점을 분석한다. 3장에서는 RDF와 RDF 스키마의 그래프 모델에 기반하여 추출가능한 모든 경로 정보와 계층 정보를 저장할 수 있도록 본 논문에서 제안한 저장 구조의 스키마를 설명한다. 4장에서 질의 처리 성능 평가 결과를 소개하고 결론을 맺는다.

## 2. 관련연구

### 2.1 RDF/S의 데이터 모델

RDF는 정보 리소스의 의미와 다른 정보 리소스와의 의미적 관계, 즉 메타데이터를 기술하는 시맨틱 웹 언어로 W3C에서 제안하였다[2]. RDF 데이터 모델을 구성하는 기본 단위는 주어(subject), 서술어(predicate), 목적어(object)의 트리플 구조로 구성된 문장(statement)이다. 주어는 정보 리소스를 의미하고 정보 리소스의 URI(Uniform Resource Identifier)로 표현된다. 정보 리소스는 웹 사이트나 웹 페이지의 전체 혹은 부분이 될 수도 있고 텍스트, 이미지, 동영상 등 다양한 객체가 될 수도 있다. 서술어는 주어로 표현된 정보 리소스가 가지는 고유의 속성이나 다

른 정보 리소스와의 관계를 의미하는 프로퍼티로 표현된다. 목적어는 프로퍼티의 실제 값으로 리터럴 데이터나 관계를 맺고 있는 다른 정보 리소스의 URI로 표현된다. RDF 데이터 모델은 노드와 간선에 모두 레이블을 가지고 있는 방향성있는 그래프로 표현할 수 있으며 그래프 표현에서 노드는 리소스를, 간선은 프로퍼티를 의미한다.

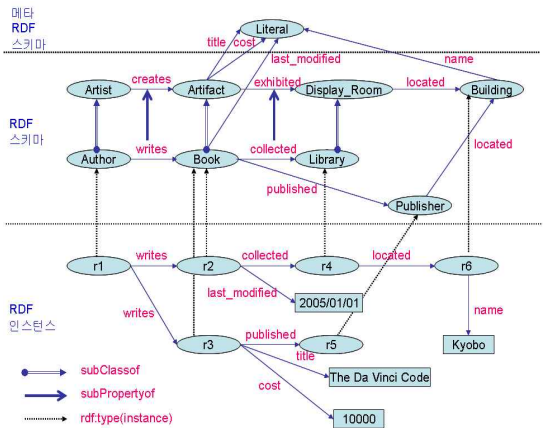
RDF 스키마는 RDF를 기술하기 위해 필요한 어휘들의 개념을 클래스로 정의하고 어휘들의 의미적인 관계인 프로퍼티를 정의하는 온톨로지 기술 언어이다[3]. 그리고 정의된 클래스들 간의 계층 정보와 정의된 프로퍼티들 간의 계층 정보도 함께 정의한다. RDF 스키마는 내부적으로 RDF 문법 구조를 그대로 사용하여 작성되므로 주어/서술어/목적어의 트리플 문장으로 구성된다. 그리고 RDF 스키마의 데이터 모델 또한 RDF와 마찬가지로 노드와 간선에 레이블된 방향성 있는 그래프로 표현이 가능하며 그래프 표현에서 노드가 클래스를, 간선이 프로퍼티를 의미하고 계층 관계를 나타내는 부가적인 간선이 존재한다.

(그림 1)은 도서관에 소장되어 있는 책과 그 책의 저자에 대한 정보를 기술한 RDF와 RDF 스키마를 그래프 데이터 모델로 표현한 예이다. (그림 1)에서 사각형으로 표현된 노드는 리터럴 데이터를 의미한다. (그림 1)의 RDF 스키마 부분은 8개의 클래스와 11개의 프로퍼티를 정의하고 있으며 클래스들 간의 계층 관계와 프로퍼티들 간의 계층 관계를 모두 표현하고 있다.

RDF와 RDF 스키마가 (그림 1)과 같은 그래프 형태의 데이터 모델로 표현될 수 있고 다양한 질의 형태가 가능하기 때문에 RDF와 RDF 스키마에 대해 그래프 모델에 기반한 저장 및 질의 처리 기법을 고려하는 것이 필요하다. 즉, RDF와 RDF 스키마의 그래프 모델에서 추출 가능한 모든 경로 정보를 기본 단위로 하는 저장 및 질의 처리 기법이 요구된다.

## 2.2 RDF/S의 관리 기법

시맨틱 웹 데이터를 위한 기존 저장 및 질의 처리에 대한 연구들은 RDF 스키마를 고려하지 않고 RDF만을 고려하는 경우가 많았다. 그러나 보다 정확하고 진화된 형태의 다양한 질의 처리를 지원하기 위해서는 RDF와 RDF 스키마를 모두 고려할 필요가 있다. RDF와 RDF 스키마를 위한 저장 및 질의 처리와 관련된 몇몇 연구들이 이미 제안되었다[4][5][6][7][8][9].




(그림 1) RDF/S의 그래프 데이터 모델 예

이러한 연구 결과의 대부분은 RDF와 RDF의 트리플 기본 구조를 저장 및 질의의 기본 단위로 정하고 상용 데이터베이스가 가지는 여러 장점을 그대로 활용하기 위해 관계형 데이터베이스나 객체-관계 데이터베이스를 이용한다. 이러한 기존 연구 결과들의 저장 방식은 크게 두 가지로 분류해볼 수 있다.

첫 번째는 단일 테이블로 저장하는 방식이다. 이 방식은 RDF와 RDF 스키마가 가지고 있는 근본적인 차이는 고려하지 않고 RDF와 RDF 스키마를 주어/서술어/목적어의 트리플 문장의 집합으로 이해하고 테이블에 모든 정보를 함께 저장한다. 따라서 테이블은 주어 컬럼, 서술어 컬럼, 목적어 컬럼으로 구성되어 있다. 이러한 방식은 데이터 관리가 용이한 장점이 있으나 질의 처리 측면에서 볼 때 항상 테이블 전체를 접근해야 하므로 다양한 질의 유형을 처리할 때 많은 자기 조인이 발생하는 비효율적인 문제가 있다.

두 번째는 RDF 스키마와 RDF의 근본적인 차이점을 인식하고 저장 스키마를 설계할 때 RDF 스키마에 정의된 내용을 고려하는 방식이다. 즉, RDF 스키마에 정의된 클래스, 프로퍼티, 클래스 계층 정보, 프로퍼티 계층 정보를 위한 테이블을

모두 독립적으로 유지하고 RDF 데이터들을 관련된 테이블에 나누어 저장한다. 그러나 이러한 저장 방식 또한 트리플 구조를 저장의 기본 단위로 하고 있기 때문에 RDF와 RDF 스키마의 그래프 데이터 모델에서 복잡한 경로식으로 표현되는 다양한 사용자 질의 형태를 처리할 때 많은 수의 조인이 발생하는 비효율적인 문제가 존재한다. 그러므로 RDF와 RDF 스키마의 근본적인 차이점을 저장 구조에 반영하면서도 단순 트리플 구조로 표현되는 기본 질의뿐만 아니라 복잡한 경로로 표현되는 다양한 질의를 효율적으로 처리할 수 있는 저장 구조가 필요하다.

## 3. RDF/S를 위한 저장 구조

### 3.1 RDF/S의 질의 유형

RDF와 RDF 스키마에 대한 질의 유형은 크게 구조적 질의와 키워드 기반 질의로 나뉠 수 있다. 구조 기반 질의는 RDF와 RDF 스키마의 그래프 데이터 모델에서 추출 가능한 경로 정보를 이용한 질의 형태이다. 키워드 기반 질의는 RDF와 RDF 스키마를 일반적인 텍스트 문서로 이해하고 질의 조건으로 주어진 키워드를 모두 포함하는 RDF와 RDF 스키마 정보를 검색하는 질의 형태이다. 본 논문에서는 구조적 질의 처리만을 고려한다.

구조적 질의 유형은 RDF 스키마에 관한 질의, RDF 데이터에 관한 질의, RDF 스키마/RDF 혼합 질의의 세 가지 형태로 세분화될 수 있다. 이러한 세 가지 질의 유형 모두 트리플의 단순 경로 또는 여러 트리플 구조가 반복하는 복잡한 경로로서 표현이 가능하다.

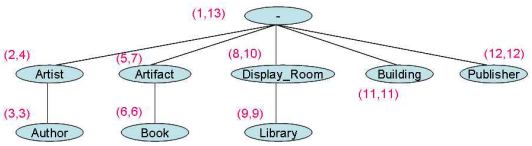
### 3.2 식별자 부여 기법

본 논문에서는 RDF와 RDF 스키마에서 저장의 대상이 되는 리소스, 클래스, 프로퍼티 요소들을 식별할 수 있도록 레이블링 기법을 사용한다[10].

본 논문에서는 RDF에 존재하는 모든 리소스에 고유의 RID(Resource ID)를 부여한다. RID는 r1, r2, r3와 같이 RDF 문서 내 출현 순서대로 부여된다.

RDF 스키마에 정의된 모든 클래스는 고유의 CID(Class ID)로 식별된다. 특히 CID는 클래스 간의 계층 관계를 표현할 수 있도록 2가지 정보로 구성된다. 각 클래스의 CID는 먼저 RDF 스키마에 정의된 클래스들을 이용해 트리를 구성한 후 트리 구조 내에서의 각 클래스 노드의 포함 관계에 따라 시작 위치 정보와 끝 위치 정보로 결정된다. 시작 위치는 깊이 우선 방법을 이용해서 클래스로 구성된 트리를 순회했을 때 각 클래스의 첫 번째 방문 순서를 의미한다. 끝 위치는 깊이 우선 방법을 이용해서 클래스로 구성된 트리를 순회했을 때 각 클래스의 마지막 방문 순서를 의미한다.

(그림 2)는 (그림 1)의 RDF 스키마 부분에서 정의한 클래스들을 이용해 트리를 구성한 예이다. 트리에 루트 노드에 해당하는 클래스가 존재하지 않을 경우는 임의의 루트 노드를 가정하고 특정 클래스의 서브 클래스로 정의된 클래스는 자식 노드로 표현함으로써 클래스의 계층 구조를 트리로 표현한다.




(그림 2) CID를 부여한 클래스 트리의 예

CID를 통해 각 클래스를 식별할 수 있을 뿐만 아니라 간단한 비교 연산을 통해 클래스들간의 계층 관계를 쉽게 판별할 수 있다. 예를 들어 클래스 노드 C1의 CID가 (S1, E1) 이고 또 다른 클래스 노드 C2의 CID가 (S2, E2)일 때, S1 < S2 이고 E2 < E1 이면 클래스 C2는 클래스 C1의 서브 클래스라 할 수 있다.

RDF 스키마에 정의된 프로퍼티의 경우도 클래스와 같은 방법으로 PID(Property ID)를 부여한다. PID 또한 프로퍼티로 구성한 트리를 이용하여 프로퍼티 간의 계층 관계를 쉽게 판별할 수 있도록 한다.

RDF와 RDF 스키마 내에 요소들에 부여된 레이블 정보는 본 논문에서 제안한 저장 구조에 유지되며 3.3절에서 자세히 설명한다.

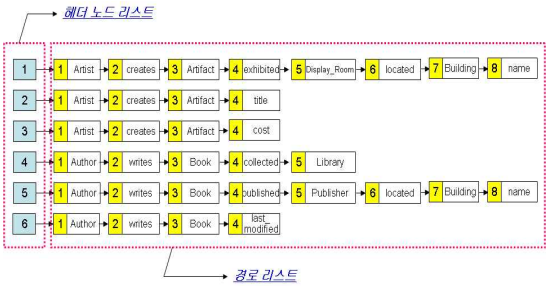
### 3.3 제안 저장 스키마

RDF와 RDF 스키마의 데이터 모델은 노드와

간선에 모두 레이블된 그래프로 표현될 수 있기 때문에 RDF 와 RDF 스키마에 대한 모든 질의 형태는 그래프에서 추출 가능한 경로 형식으로 표현할 수 있다. 그런데 기존에 연구된 RDF와 RDF 스키마를 위한 저장 모델들이 트리플 구조 기반의 단순 경로로 표현되는 질의 처리에는 효율적이지만 복잡한 경로로 표현되는 질의를 처리하는데 있어 조인 횟수가 급격히 증가되어 질의 처리의 성능이 저하되는 문제를 가지고 있다. 따라서 본 논문에서는 복잡한 경로식으로 표현되는 질의 처리를 지원하기 위한 저장 구조를 제안한다. 본 논문에서 제안한 저장 구조는 상용 데이터베이스의 장점을 그대로 활용하기 위해 관계형 데이터베이스의 테이블을 이용하며 RDF와 RDF가 정의하는 클래스, 프로퍼티, 리소스에 대한 정보는 물론 RDF와 RDF 스키마가 표현될 수 있는 모든 경로 정보를 저장하는데 목적을 두고 있다. 본 논문에서 제안한 저장 구조를 구성하기 위해서는 먼저 RDF 스키마의 그래프 데이터 모델에서 가장 긴 경로 표현식을 추출한 후 PList(Path List)에 저장한다. PList는 특정 클래스나 프로퍼티로부터 시작되는 모든 경로식에 대한 정보를 유지하며 각 경로식을 구별하는 헤더 노드와 각 경로식을 표현하는 리스트로 구성되어 있다. 각 경로식을 표현한 리스트는 해당 경로를 구성하는 클래스와 프로퍼티들에 대한 정보를 가지고 있는 노드들을 순서대로 유지한다. PList를 이용하면 특정 클래스나 프로퍼티로부터 시작하는 복잡한 경로 정보를 쉽게 검색할 수 있을 뿐만 아니라 특정 프로퍼티를 속성으로 포함하는 클래스를 의미하는 도메인 클래스와 특정 프로퍼티가 값으로 가질 수 있는 값의 클래스 타입을 의미하는 레인지 클래스를 쉽게 검색할 수 있는 장점이 있다. PList 내에 가장 긴 경로 표현식만을 유지하는 이유는 그래프 모델의 특성 상 긴 경로 표현식이 내부적으로 더 짧은 경로식들을 모두 포함하고 있기 때문이다. 그리고 RDF 스키마로부터 추출한 경로 정보만을 유지하는 이유는 RDF가 RDF 스키마의 인스턴스 데이터라 할 수 있기 때문에 RDF로부터 추출된 경로 정보가 RDF 스키마로부터 추출된 경로 정보에 근간을 두고 있기 때문이다.

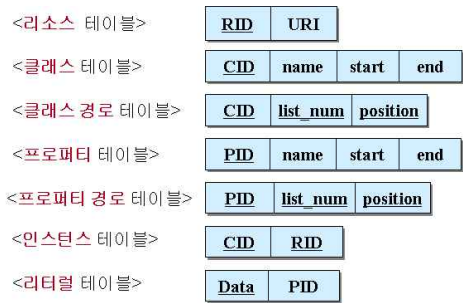
(그림 3)은 (그림 1)의 RDF 스키마 부분에서 추출한 가장 긴 경로 정보를 유지하는 PList의 예를 보여준다.

본 논문에서는 RDF와 RDF 스키마 내의 요소들을 식별하기 위한 레이블 정보와 PList에 유지되고 있는 경로 정보와 연계하여 관계형 데이터베이스에 기반한 7개의 테이블 구조로 RDF와 RDF 스키마 정보를 저장한다.




(그림 3) PList 구성 예

(그림 4)는 본 논문에서 제안한 7개의 테이블로 구성된 저장 스키마를 보여준다. (그림 4)에서 리소스 테이블은 RDF 문서 내에 존재하는 모든 리소스에 대한 RID와 URI 정보를 저장한다. URI 자체가 리소스를 구별함에도 RID를 별도로 유지하는 이유는 URI의 길이가 대부분 길어 저장 및 질의를 처리할 때 추가적인 저장 공간 및 처리에 드는 오버헤드를 줄이기 위함이다.




(그림 4) 경로 정보를 이용한 저장 스키마

클래스 테이블은 RDF 스키마에 정의된 모든 클래스에 대한 정보를 유지하기 위해 클래스의 CID와 클래스 이름 정보를 저장한다. 클래스의 CID는 3.2절에서 설명한대로 클래스 트리 내에서의 시작 위치와 끝 위치로 구성되기 때문에 시작 위치는 클래스 테이블에서 start 필드에 끝 위치는 클래스 테이블에서 end 필드에 저장된다.

단, 테이블 간의 조인 연산을 쉽게 처리하기 위해서 클래스 테이블의 기본키는 간단히 순차적으로 부여한다.

프로퍼티 테이블은 RDF 스키마에 정의된 프로퍼티에 대한 PID와 이름 정보를 클래스 테이블과 같은 방법으로 저장한다.

클래스 경로 테이블은 특정 클래스로부터 시작하는 경로에 대한 정보를 PList와 연계하여 저장한다. 경로를 시작하는 클래스의 CID와 PList에서 그 클래스가 나타나는 경로 리스트의 번호를 list_num 필드에 저장하고 경로 리스트내의 위치 정보를 position 필드에 저장한다. 예를 들어, 2번 클래스 “Artist”는 PList에서 1번과 2번, 그리고 3번 경로 리스트의 시작 클래스로 나타나기 때문에 클래스 경로 테이블에 <2, 1, 1>, <2, 2, 1>, <2, 3, 1>의 정보가 저장된다. 그러므로 “Artist“ 클래스로부터 시작되는 경로식으로 표현되는 질의를 처리할 때 쉽게 이용될 수 있다.

프로퍼티 경로 테이블도 클래스 경로 테이블과 같은 방식으로 특정 프로퍼티로부터 시작하는 경로에 대한 정보를 저장한다.

리터럴 테이블은 리터럴 데이터를 값으로 가지는 프로퍼티에 대한 정보를 저장한다. 그리고 인스턴스 테이블은 RDF 내에서 “rdf:type” 프로퍼티를 이용하여 특정 클래스 타입의 인스턴스로 정의된 리소스에 대한 정보를 저장한다.

본 논문에서 제안한 PList의 구조와 7개의 테이블 구조를 이용하면 테이블간의 조인 횟수를 최소화하면서 특정 클래스나 프로퍼티, 또는 리소스로부터 시작하는 질의를 쉽게 처리할 수 있는 장점이 있다.

## 4. 성능 평가

### 4.1 실험 환경과 실험 데이터

본 논문에서 제안한 RDF와 RDF 스키마를 위한 저장 시스템은 펜티엄4 3.0GHz의 CPU와 1GB의 메모리를 가지고 Linux 9.1을 운영 체제로 이용하는 컴퓨터 시스템 상에 구현하였다. 그리고 저장 구조에 활용한 관계형 데이터베이스는 오라클 9i를 이용하여 구축하였으며 C 언어를 이용하여 전체 시스템을 개발하였다.

실험 데이터는 Wordnet에서 제공하는 RDF와 RDF 스키마 문서를 사용하였다. Wordnet은 온라인 어휘 참조 시스템으로 영어 단어의 품사를 명사, 동사, 형용사, 부사로 나누고 그들이 원래 가지고 있는 의미에 따라 동의어, 하위어 등의 관계를 정의한 것이다[11]. Wordnet의 RDF 스키마는 6개의 클래스와 5개의 프로퍼티를 정의하고 있다. Wordnet의 RDF 스키마를 그래프 데이터 모델로 표현하면 사이클이 존재하는데 본 논문에서는 고려하지 않는다. Wordnet에서 제공하는 RDF 데이터를 크기가 다른 4개의 파일로 나누어 실험을 진행하였고 각 실험 파일에는 서로 다른 수의 트리플 문장이 존재한다.

실험 결과를 평가하기 위해 3개의 구조 기반 질의를 사용하였고 크게 RDF 스키마만을 대상으로 하는 질의와 RDF와 RDF 스키마를 모두 대상으로 하는 질의로 나눌 수 있다. RDF 문서만을 대상으로 하는 질의의 경우는 처리 과정이 RDF와 RDF 스키마를 모두 대상으로 하는 경우와 거의 유사하기 때문에 본 논문에서는 생략하였다.

<표 1>은 실험 데이터에 대한 추가적인 설명을 보여주고 있다.

<표 1> 실험 데이터로 사용된 4개의 파일

| | Dataset1 | Dataset2 | Dataset3 | Dataset4 |
|---|---|---|---|---|
| RDF 스키마 문서의 수 | 1 | 1 | 1 | 1 |
| RDF 문서의 수 | 4 | 4 | 4 | 4 |
| 트리플 문장 수 | 4,198 | 15,059 | 149,081 | 447,243 |
| RDF 스키마 문서의 크기 | 4KB | 4KB | 4KB | 4KB |
| RDF 문서의 크기 | 587KB | 2,125KB | 21,199KB | 63,598KB |

실험에 사용된 3개의 구조 기반 질의는 다음과 같다.

- RDF 스키마만을 대상으로 하는 질의
  (1) 클래스 계층 정보에 대한 질의

| 트리플 표현 |
| --- |
| (LexicalConcept subClassOf ?) |
| **의미** |
| “LexicalConcept” 클래스의 서브 클래스를 검색하라. |

▪ RDF 스키마와 RDF를 모두 대상으로 하는 질의

(1) 경로 정보에 기반한 질의

| 트리플 표현 |
| --- |
| (LexicalConcept type ?X)<br>(Lexical Concept type ?Y)<br>(?X hyponymOf ?Y) |
| **의미** |
| "LexicalConcept.hyponymOf.LexicalConcept" 경로 상에 존재하는 모든 리소스를 검색하라. |

(2) 클래스와 리소스의 타입 관계에 대한 질의

| 트리플 표현 |
| --- |
| (? type Noun) |
| **의미** |
| “Noun" 클래스 타입으로 정의된 리소스를 검색하라. |

## 4.2 실험 결과

본 논문에서 제안한 저장 기법에 따라 구성된 저장 시스템의 성능을 평가하기 위해서 RDF와 RDF 스키마의 저장 시스템에 많이 이용되는 저장 기법을 비교 대상으로 선정하였다. 비교 대상으로 선택한 일반적인 시맨틱 웹 데이터 저장 기법은 RDF와 RDF 스키마를 구별하지 않고 주어/서술어/목적어의 트리플 필드 구조로 이루어진 단일 테이블에 저장하는 방식이다.

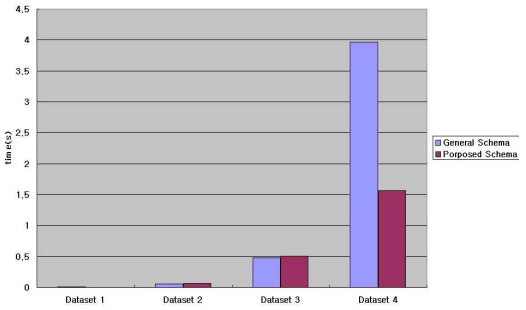
4.1절에서 제시한 3가지 질의 유형에 대해 검색 시간을 기준으로 제안 저장 시스템과 RDF와 RDF 스키마를 구별하지 않는 단일 테이블 구조의 저장 시스템을 비교하였으며 검색 시간의 단위는 초로 지정하였다.

(그림 5)는 4.1절에서 제시한 첫 번째 질의 유형에 대한 실험 결과를 보여준다. (그림 5)의 실험에 사용된 첫 번째 질의는 “LexicalConcept” 클래스의 모든 서브 클래스를 검색하는 것이다. 단일 테이블 저장 기법을 사용하는 경우, “LexicalConcept” 클래스의 서브 클래스로 직접 정의되어 있는 서브 클래스들은 테이블의 자기 조인이 필요없이 빠르게 질의 결과를 검색할 수 있다. 그러나 “LexicalConcept” 클래스의 서브 클래스로서 직접 정의되어 있는 클래스의 서브 클래스로 정의된 “LexicalConcept”의 간접적인 서브 클래스들을 검색하기 위해서는 테이블의 자기 조인이 많이 요구되기 때문에 질의 처리 시 큰 오버헤드를 가지게 된다. 본 논문에서 제안한 저장 스키마를 이용하는 경우는 클래스들마다 고유하게 부여된 CID에 대해 간단한 비교 연산을 적용하여 직접 정의된 서브 클래스뿐만 아니라 다른 클래스를 통해 간접적으로 정의된 서브 클래스의 경우도 쉽게 계층 구조를 판단할 수 있다. 따라서 저장된 트리플 문장의 수가 많을수록 본 논문에서 제안한 방식으로 시맨틱 웹 데이터를 저장하는 것이 단일 테이블 저장 기법을 이용하는 것보다 검색 성능이 우수한 것을 (그림 5)에서 확인할 수 있다. 특히, 클래스 계층 구조의 깊이가 깊은 경우 단일 테이블 저장 방식은 몇 번의 자기 조인을 수행해야 직·간접적으로 정의된 모든 서브 클래스들을 검색할 수 있는지 미리 알 수 없어 정확한 검색 결과를 유도하기 어려운 문제가 있다.

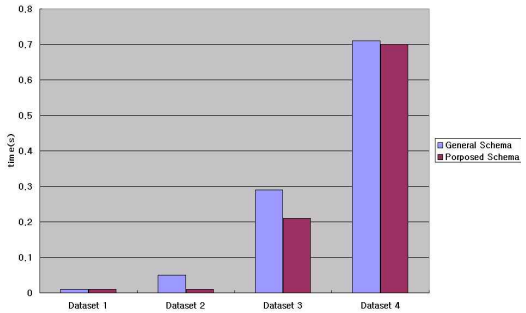

(그림 5) 첫 번째 질의에 대한 실험 결과

(그림 6)은 4.1절에서 제시한 두 번째 질의 유형에 대한 실험 결과를 보여준다. (그림 6)의 실험에서 사용된 두 번째 질의는 “LexicalConcept. hyponymOf.LexicalC-oncept” 경로 상에 존재하는 모든 리소스의 쌍을 검색하는 것으로 결과적으로는 하위어로 의미적 관계를 맺고 있는 리소

스의 쌍을 검색하게 된다. 단일 테이블 저장 기법은 RDF와 RDF 스키마의 역할을 분리하지 않고 모두 트리플 문장의 집합으로 해석하기 때문에 두 번째 질의처럼 RDF 스키마에서 추출할 수 있는 경로식에 기반하여 RDF에 기술된 리소스 정보를 검색하고자 한다면 단일 테이블에 대한 많은 수에 자기 조인이 발생하게 되어 성능이 저하될 수밖에 없다. 그러나 본 논문에서 제안한 저장 기법의 경우 RDF 스키마에서 추출 가능한 모든 경로 정보를 저장한 PList를 통해 특정 클래스나 프로퍼티로부터 시작하는 경로 정보를 질의 처리 시 쉽게 이용할 수 있어 효과적이다. 따라서 (그림 6)의 결과에서 확인할 수 있는 바와 같이 실험 데이터의 크기가 클수록 제안 저장 구조를 이용한 검색 시간이 단일 테이블 저장 기법을 이용하는 경우보다 검색 시간 측면에서 우수하다. 특히 (그림 6)에서는 간단한 경로식으로 표현할 수 있는 질의를 처리한 실험 결과를 보여주고 있지만 만약 더욱 복잡한 경로식으로 표현되는 질의를 처리하는 경우는 성능의 차가 더욱 커질 것으로 예상된다.




(그림 6) 두 번째 질의에 대한 실험 결과




(그림 7) 세 번째 질의에 대한 실험 결과

그리고 본 논문에서 제안한 PTable은 RDF 스키마에서 직·간접적으로 추출할 수 있는 모든 경로 정보를 저장하고 있기 때문에 질의에서 직접 제시하고 있는 경로뿐만 아니라 의미적으로 관련된 모든 경로 정보를 이용한 보다 정확하고 풍부한 검색 결과를 유도할 수 있다.

(그림 7)은 4.1절에서 제시한 세 번째 질의 유형에 대한 처리 결과를 보여준다. (그림 7)의 실험에서 사용된 질의는 "Noun" 클래스 타입으로 정의된 리소스들을 검색하는 것이다. 본 논문에서 제안한 저장 구조의 경우 인스턴스 테이블 내에서 클래스 타입과 리소스간의 관계 정보를 저장하고 있고 클래스마다 부여된 CID를 이용할 수 있다. 따라서 직·간접적으로 정의된 모든 서브 클래스를 질의 처리 시 쉽게 이용할 수 있기 때문에 단일 테이블 저장 기법을 이용하는 것보다 빠른 검색 결과를 보인다. 그러나 실험 데이터가 커지면 검색 대상이 되는 리소스의 양이 크게 증가하기 때문에 제안 저장 기법을 이용한 검색 시간이 일반 저장 기법을 이용한 경우보다 성능이 크게 향상되지는 않는다.

## 5. 결론

차세대 웹의 대안으로 인식되고 있는 시맨틱 웹에서는 정보 리소스의 의미와 개념적 관계를 정의하는 메타데이터와 온톨로지의 역할이 무엇보다 중요시되고 있다. 따라서 RDF와 RDF 스키마와 같은 시맨틱 웹 언어로 표현되는 데이터의 효과적인 저장과 검색의 방법이 필요하다.

기존의 많은 연구들이 RDF와 RDF 스키마의 트리플 구조에 따라 데이터를 저장하고 검색하는 방법을 제안하였다. 그러나 이러한 대부분의 연구들이 RDF와 RDF 스키마의 근본적인 차이점은 간과하고 관계형 데이터베이스 내에서 단일 테이블에 RDF와 RDF 스키마를 함께 저장하는 방식을 이용한다. 그러나 이러한 기존 저장 방식들은 관리가 용이한 장점이 있지만 복잡하고 다양한 질의 유형의 처리 시 비효율적이라는 문제가 존재한다.

따라서 본 논문에서는 RDF와 RDF 스키마에 대한 질의 형태를 분석하여 모든 질의 형태가 RDF와 RDF 스키마의 그래프 데이터 모델에 기

반한 경로식으로 표현이 가능하다는데 초점을 맞추고 RDF와 RDF 스키마의 근본적인 차이점을 반영하는 저장 구조를 제안한다.

본 논문에서 제안한 저장 구조는 RDF와 RDF 스키마에서 추출 가능한 모든 경로 정보는 물론 클래스, 프로퍼티, 리소스에 대한 정의와 클래스나 프로퍼티의 계층 정보도 함께 저장할 수 있도록 관계형 데이터베이스에 기반한 7개의 테이블로 구성된다. 따라서 본 논문에서 제안한 저장 구조를 이용하면 RDF와 RDF 스키마에 대한 다양한 형태의 질의 처리가 가능하다.

## 참 고 문 헌


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

[2] World Wide Web Consortium, Resource Description Framework(RDF) Model and Syntax Specification, 2004

[3] World Wide Web Consortium, Resource Description Framework(RDF) Schema Specification 1.0, 2004

[4] Akiyoshi Matono, Toshiyuki Amagasa, Masatoshi Yoshikawa, and Shunsuke Uemura. "An Indexing Scheme for RDF and RDF Schema based on Suffix Arrays''. First International Workshop on Semantic Web and Databases (SWDB) co-located with 29th International Conference on Very Large Data Bases (VLDB2003), pp. 151-168, Berlin, Germany, September 7-8, 2003.

[5] A. Reggiori, D. van Gulik, and Z. Bjelogrlic. "Indexing and retrieving Semantic Web resources: the RDF Store model". In Proceedings of SWAD-Europe Workshop on Semantic Web Storage and Retrieval, 2003.

[6] Stanislav Barton. "Designing Indexing Structure for Discovering Relationships in RDF Graphs". In Proceedings of the Dateso 2004 Annual International Workshop on DAtabases, TExts, Specifications and Objects, Desna, pages 7-17, Czech Republic, April 14-16, 2004.

[7] S. Alexaki et al., "The RDFSuite: Managing Voluminous RDF Description Bases", In Proc. of the 2nd International Workshop on the Semantic Web, 2001.

[8] J. Broekstra et al., "Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema", In Proc. of the First International Semantic Web Conference, 2002.

[9] Andreas Harth and Stefan Decker. "Optimized Index Structures for Querying RDF from the Web". 3rd Latin American Web Congress, Buenos Aires - Argentina, 2005.

[10] V. Christophides, D. Plexousakis, M. Scholl, and S. Tourtounis, "On labeling schemes for the semantic web", In Proc. of the 12th international conference on World Wide Web, 2003.

[11] Wordnet, http://www.semanticweb.org/library/.



**김 연 희**

2000년 : 홍익대학교 컴퓨터공학과 졸업(학사)
2002년 : 홍익대학교 컴퓨터공학과 대학원 졸업(석사)
2006년 : 홍익대학교 컴퓨터공학과 대학원 졸업(박사)
2007년~현 재 : 부천대학 e-비즈니스과 강의전담교수
관심분야 : 시맨틱 웹, XML, 분산 데이터베이스, 모바일 데이터베이스

**최 재 언**

2001년 : 대전대학교 컴퓨터공학과 졸업(학사)
2003년 : 대전대학교 컴퓨터공학과 졸업(석사)
2005년~현 재 : 홍익대학교 컴퓨터공학과 박사과정
관심분야 : 시맨틱 웹

**임 해 철**

1976년 : 서울대학교 계산통계학과 졸업(이학사)
1978년 : 한국과학기술원 전자계산학과 졸업(이학석사)
1988년 : 서울대학교 컴퓨터공학과 졸업(공학박사)
1981년~현 재 : 홍익대학교 컴퓨터공학과 교수
관심분야 : 시맨틱 웹, XML, 멀티미디어 데이터베이스