DBMS기반 트리플 저장소에서 뷰를 이용한 효율적인 추론

Efficient Reasoning Using View in DBMS-based Triple Store

이승우, 김재한, 류범종
한국과학기술정보연구원

Seungwoo Lee, Jae-Han Kim, Beom-Jong You
Korea Institute of Science and Technology Information (KISTI)

요약

온톨로지가 대용량화되면서 온톨로지 시스템의 성능 향상을 위해 효율적인 추론이 중요해졌다. 본 논문에서는 DBMS 기반의 온톨로지 저장소에서 RDFS 포함관계 함의 규칙 (rdfs7 규칙과 rdfs9 규칙)과 OWL 역관계 규칙(owl:inverseOf)의 추론을 효율적으로 수행할 수 있는 방법으로서, DB 테이블에 대한 뷰(view)를 활용하는 방법을 소개한다. 추론 규칙을 뷰 정의로 대체하고 RDF 트리플을 구조화된 트리플 테이블에 저장하는 것으로 추론이 완료되며 대신 질의 처리과정에서는 그 뷰를 참조하면 된다. 이와 같이 뷰를 정의하는 것으로 추론을 대신함에 따라 추론에 소요되는 시간을 단축할 수 있고 트리플 저장소의 공간 효율성도 얻을 수 있다.

Abstract

Efficient reasoning has become important for improving the performance of ontology systems as the size of ontology grows. In this paper, we introduce a method that efficiently performs reasoning of RDFS entailment rules (i.e., rdfs7 and rdfs9 rules) and OWL inverse rule using views in the DBMS-based triple sotre. Reasoning is performed by replacing reasoning rules with the corresponding view definition and storing RDF triples into the structured triple tables. When processing queries, the views is referred instead of original tables. In this way, we can reduce the time needed for reasoning and also obtain the space-efficiency of the triple store.

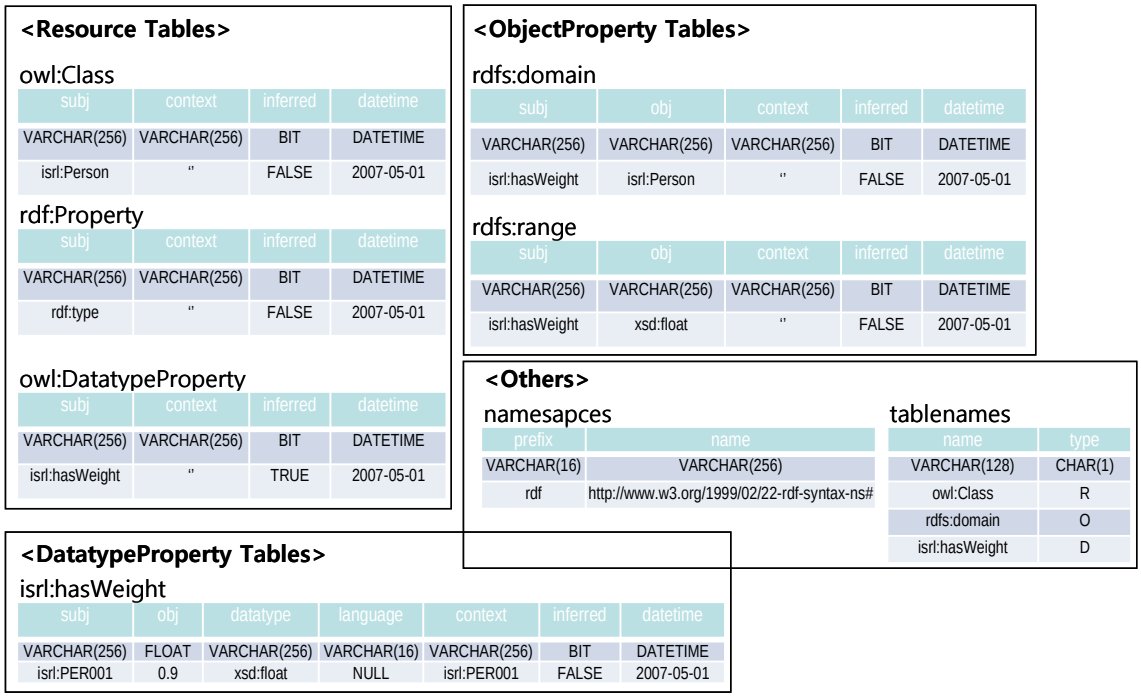
Ⅰ. 서론

시맨틱 웹(Semantic Web)의 발전 과정에서 수많은 다양한 시맨틱 웹 관련 시스템들이 제 각각의 특성을 갖고 개발되고 확장되어 왔다. 온톨로지 또한 초기에는 장난감 수준의 소규모 온톨로지(예를 들면, Wine Ontology)에 지나지 않았으나, 최근에는 여러 기관들에서 대용량 온톨로지들이 개발 및 구축되고 실제 서비스에도 활용되는 사례가 속속 발견되고 있다. 이러한 추세에서 온톨로지를 저장하고 추론하는 시스템의 규모성(scalability)과 실용성(practicality)에 관한 이슈가 최근에 전문으로 부상하고 있다. 이는 최근에 독일 칼스루에(Karlsruhe)에서 개최된 ISWC2008의 시맨틱 웹 챌린지(Semantic Web Challenge)[1]에 BTC(Billion Triple Challenge) 트랙이 새롭게 추가된 사실로도 확인할 수 있다.

1. 규모성 (Scalability)

시맨틱 웹의 발전 초기에 온톨로지 규모는 작게는 수만에서 많아야 수백만 트리플에 지나지 않았다. 이 정도의 작은 규모의 온톨로지를 대상으로 하는 시맨틱 웹 서비스 시스템들은 모든 온톨로지를 메모리에 올려서 처리하는 것도 가능했기 때문에 규모성 문제를 신경 쓸 필요가 없었다. 그러나 현재는 수억에서 수십억 트리플의 시대로 온톨로지의 규모가 방대해졌으며, 머지않아 기존 웹의 규모로까지 확대될 전망이다.

이와 같은 상황에서 시맨틱 웹 시스템, 특히 온톨로지 저장 및 추론을 위한 시스템의 규모성 문제는 더 이상 뒤로 미룰 수 없는 중요한 사안이 되었다.



<Resource Tables>

owl:Class

subj	context	inferred	datetime
VARCHAR(256)	VARCHAR(256)	BIT	DATETIME
isrl:Person	''	FALSE	2007-05-01

rdf:Property

subj	context	inferred	datetime
VARCHAR(256)	VARCHAR(256)	BIT	DATETIME
rdf:type	''	FALSE	2007-05-01

owl:DatatypeProperty

subj	context	inferred	datetime
VARCHAR(256)	VARCHAR(256)	BIT	DATETIME
isrl:hasWeight	''	TRUE	2007-05-01

<ObjectProperty Tables>

rdfs:domain

subj	obj	context	inferred	datetime
VARCHAR(256)	VARCHAR(256)	VARCHAR(256)	BIT	DATETIME
isrl:hasWeight	isrl:Person	''	FALSE	2007-05-01

rdfs:range

subj	obj	context	inferred	datetime
VARCHAR(256)	VARCHAR(256)	VARCHAR(256)	BIT	DATETIME
isrl:hasWeight	xsd:float	''	FALSE	2007-05-01

<Others>

namesapces

prefix	name
VARCHAR(16)	VARCHAR(256)
rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#

tablenames

name	type
VARCHAR(128)	CHAR(1)
owl:Class	R
rdfs:domain	O
isrl:hasWeight	D

<DatatypeProperty Tables>

isrl:hasWeight

subj	obj	datatype	language	context	inferred	datetime
VARCHAR(256)	FLOAT	VARCHAR(256)	VARCHAR(16)	VARCHAR(256)	BIT	DATETIME
isrl:PER001	0.9	xsd:float	NULL	isrl:PER001	FALSE	2007-05-01

▶▶ 그림 1. 온톨로지 저장소 DB 테이블 스키마

2. 실용성 (Practicality)

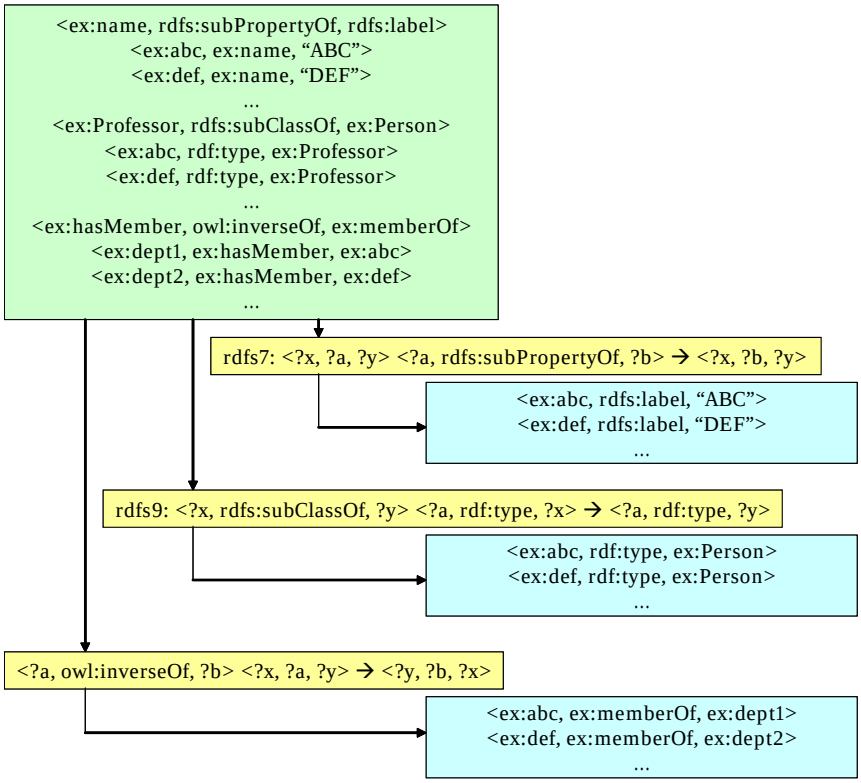
대용량의 온톨로지를 처리하는 데 있어서, 규모성과 함께 필연적으로 해결해야 할 문제가 실용성이다. 온톨로지 저장 및 추론 시스템이 아무리 대용량의 온톨로지를 처리할 수 있다 하더라도, 실용성 측면을 무시하고는 실제 응용에 적용되어 사용될 여지는 적다. 기존의 정보검색 시스템이 수많은 사용자를 확보할 수 있었던 것은 정보의 홍수인 웹의 엄청난 데이터를 빠른 시간 내에 처리하여 응답할 수 있었던 규모성과 실용성 측면에서의 능력을 발판으로 하고 있었기 때문이다. 시맨틱 웹 응용 시스템들도 이와 같은 맥락에서 볼 때, 웹에 근접한 수준의 대용량 온톨로지를 빠른 시간에 저장하고 추론하여, 사용자의 질의에 짧은 시간 내에 응답할 수 있는 실용성을 갖춰야만 사용자들의 사랑을 받는 시스템으로 발전할 수 있을 것이다.

이와 같이 규모성과 실용성에 초점을 맞춰, 본 논문에서는 DBMS 기반의 온톨로지 저장소에 뷰를 이용하여 추론을 대체하고 저장 공간의 효율성을 높이는 방법을 소개한다.

Ⅱ. DBMS 기반의 온톨로지 저장소

온톨로지 규모가 커짐에 따라, 메모리에 온톨로지 전부를 유지하는 것이 불가능해지면서, 물리적인 디스크에 온톨로지를 효율적으로 저장할 필요성이 생겼다. 이에 온톨로지 저장에 특화된 자체 저장소(native store) (eg., Sesame Native[2], Virtuoso[3], AllegroGraph[4])를 이용하기도 하지만, 만족스러운 성능을 얻기 위해서는 많은 노력과 비용이 들어가기 때문에, 많은 온톨로지 저장 시스템들은 이미 많은 테스트와 사용 과정을 통해 대용량의 데이터를 처리할 수 있는 능력을 검증받은 DBMS를 백엔드(back-end) 저장소(eg., Jena SDB[5], DLDB[6], 3store[7])로 사용하고 있다.

본 논문에서도 DBMS를 온톨로지 저장소로 사용하고 있으며, 그림 1과 같은 DB 테이블 스키마를 사용한다. 이 스키마 설계는 스키마-인지(schema-aware) 방식[8]에 해당하지만, URI를 표현하는데 있어서, URI 방식이나 ID 방식이 아닌, URI의 네임스페이스(namespace)를 접두사(prefix)로 대체한 prefixed URI 방식을 고안하여 적용하였다.



▶▶ 그림 2. 대학교를 모델링한 온톨로지의 일부로 RDFS 포함관계 함의 규칙과 OWL 역관계 규칙을 적용함으로써 추론 확장되는 트리플의 예

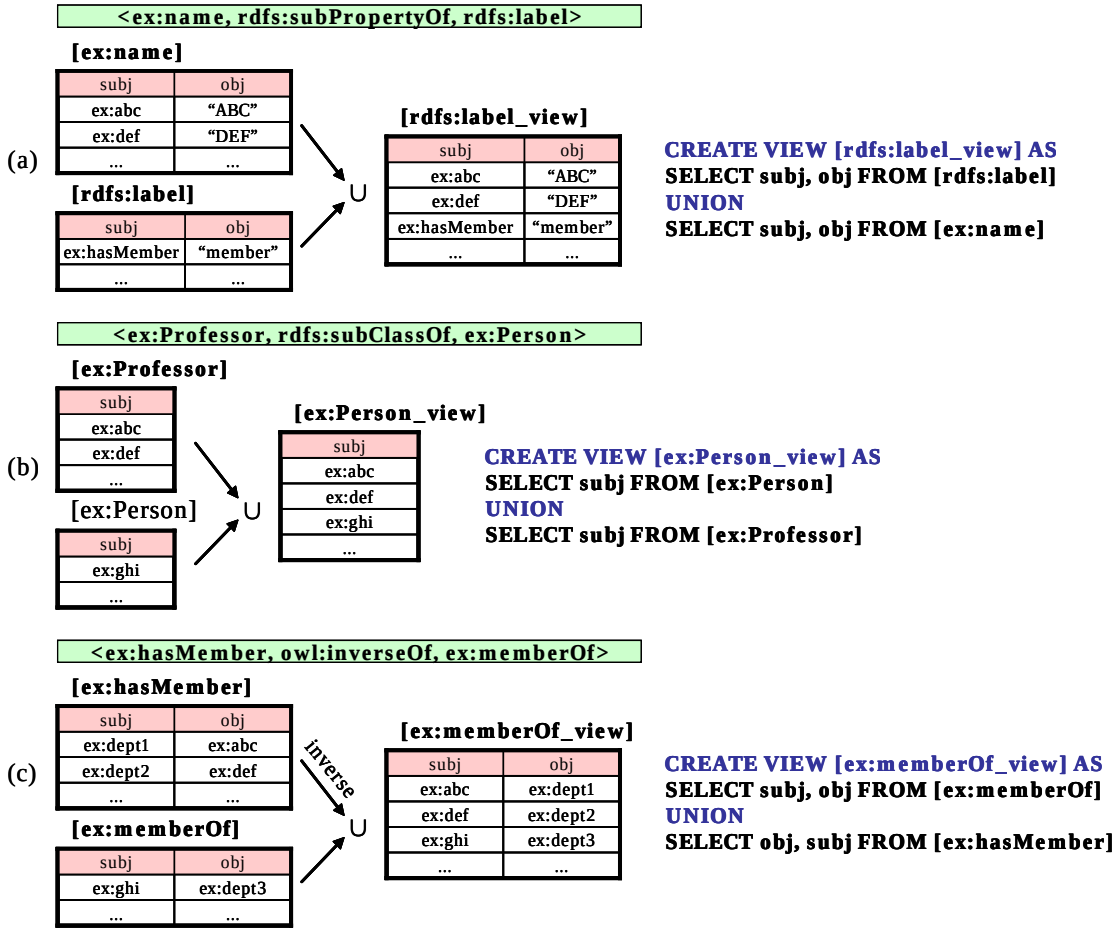
URI는 보통 긴 문자열로 구성되기 때문에 URI를 그대로 저장하는 것은 저장 공간의 낭비를 가져올 뿐만 아니라, 저장과 검색의 효율성도 저하시킬 수 있다. 이 때문에 사전 인코딩(dictionary encoding)을 통해 긴 URI를 정수 형태의 ID로 사상시켜 ID로 구성된 트리플을 저장하는 방식을 사용하기도 한다. 하지만, 이 경우, 저장시에 URI를 ID로 매핑하고, 질의응답시에 ID를 다시 URI로 매핑해야 하는데, 온톨로지 규모가 커질수록 URI-ID 매핑을 메모리에 유지하는 것이 불가능해지기 때문에, 저장 공간 측면에서는 효율적이나 속도 측면에서의 효율이 반감되는 단점이 있다. 반면, 보통 온톨로지에서 사용되는 네임스페이스의 수는 그리 많지 않기 때문에 접두사-네임스페이스 매핑을 메모리에 유지하는 것이 가능하여 매핑을 빠르게 처리할 수 있다. 또한, prefixed URI는 원래의 URI에 비해 문자열 길이를 상당히 줄여서 저장 공간의 낭비도 어느 정도 줄일 수 있다.

본 논문에서 스키마-인지 방식의 설계를 적용하는 이유는 4장에서 설명될 효율적인 추론과도 관련이 있다. 추론의 효율성을 높이기 위해 DBMS 뷰를 이용하여 추론 규칙을 대신할 수 있는데, 트리플을 클래스 및 속성 단위로 나눠서 저장함으로써 이를 가능하게 하는 것이다.

Ⅲ. 온톨로지 추론

온톨로지에 대한 추론은 보통 규칙 기반 추론과 기술논리(description logic) 기반의 추론으로 구분되는데, 추론이 일어나는 시점을 기준으로 볼 때는, 전방 추론(forward reasoning)과 후방 추론(backward reasoning)으로 나눠볼 수 있다. 전방 추론은 질의가 들어오기 전에 추론이 발생하므로 오프라인 추론(off-line reasoning)이라 말할 수 있는데, 질의 응답의 속도가 빠르지만 상대적으로 저장 공간을 많이 차지하며 온톨로지 적재시에 더 많은 시간을 요구한다. 반면, 후방추론은 질의가 들어오는 순간에 추론이 일어나므로 온라인 추론(online reasoning)이라 말할 수 있는데, 적재 속도는 빠르고 추론에 의한 추가적인 저장 공간을 필요로 하지 않지만, 질의 응답의 속도가 상대적으로 느려진다.

온톨로지 저장소의 경우와 마찬가지로, 온톨로지 추론에 대해서도 수십억 이상의 트리플을 다뤄야 함에 있어서, 규모성과 실용성은 중요한 이슈이며, 따라서, 후방 추론에서와 같이 빠른 적재 속도와 저장공간의 효율성을 유지하면서 전방 추론에서와 같이 빠른 질의 응답 속도를 얻을 수 있는 방안이 필요하다.



▶▶ 그림 3. RDFS 포함관계 함의 규칙 및 OWL 역관계 규칙의 추론을 대체할 수 있는 DBMS 뷰 정의

DBMS를 활용함으로써 대용량의 트리플을 저장하는 것은 쉽게 달성될 수 있지만, 문제는 추론 과정에서 발생하는 불필요한 중복으로 인한 저장소의 비효율성과 이로 인한 추론 속도의 저하이다. 특히, RDFS 포함관계 함의 규칙인 rdfs7 규칙과 rdfs9 규칙, OWL 역관계(owl:inverseOf) 규칙의 경우, 규칙이 적용되는 회수도 작고 필요 이상으로 많은 수의 트리플을 생성하게 됨으로써 저장 공간을 차지할뿐더러 추론의 효율성도 떨어뜨린다.

예를 들어, 그림 2는 대학교를 모델링한 온톨로지의 일부로, ex:name 속성은 rdfs:label 속성의 하위 속성이기 때문에 ex:name 속성을 갖는 모든 주어(subject), 목적어(object)는 rdfs:label 속성의 주어 및 목적어로 확장된다. ex:Professor 클래스와 ex:Person 클래스의 경우도, 속성이 아닌 클래스라는 점만 다를 뿐 상하위 관계에 의한 확장은 동일하다. 마지막으로, ex:hasMember 속성과 ex:memberOf 속성은 owl:inverseOf 관계에 있기 때문에 ex:hasMember 속성의 모든 주어 및 목적어는 각각 ex:memberOf 속성의 목적어 및 주어로 확장된다. 이러한 확장은 추론 시스템의 질의 응답 측면에서 완전성(completeness)를 위해 반드시 필요하지만, 트리플의 술어만 대체되거나 혹은 추가로 주어와 목적어만 뒤바뀔 뿐, 여러 트리플 사이의 복잡한 조합을 필요로 하지 않기 때문에 단순하게 트리플을 복제하여 저장하는 것은 온톨로지 저장 및 추론 시스템의 공간과 시간 측면에서의 비효율성을 초래한다.

결국, 저장 공간과 추론의 효율성을 높이기 위해서는 지나치게 많은 저장 공간과 추론의 규칙의 적용을 줄일 수 있는 장치가 필요하며, 물론, 이러한 장치가 RDF 트리플의 저장소에 대한 질의 응답의 효율성을 저해해서는 안 된다. 부분적이나마 이에 대한 한 가지 해결 방안으로서 DBMS 기반의 온톨로지 저장소에서 뷰를 활용하는 방법을 다음 장에서 소개한다.

Ⅳ. 뷰를 이용한 효율적인 추론

이 장에서는 DBMS 기반의 온톨로지 저장소에서 RDFS 포함관계 함의 규칙과 OWL 역관계 규칙을 추론함에 있어서 뷰를 이용하여 저장 공간과 함께 추론 속도 측면에서 효율성을 높이는 방안을 소개한다.

먼저, RDF 트리플이 입력되면, 이 트리플이 RDFS 포함관계 규칙이나 OWL 역관계 규칙에 해당하는지를 검사한다. 다시 말해, 트리플의 술어가 rdfs:subPropertyOf와 rdfs:subClassOf, owl:inverseOf 중의 하나에 해당하는지를 검사한다. 해당될 경우, 그에 따른 뷰를 정의한다. 그림 3을 예로 들어 설명하면, (a)의 경우, 화살표 왼쪽과 같이 트리플이 입력되어 저장된 상태에서 트리플 〈ex:name, rdfs:subPropertyOf, rdfs:label〉이 입력되면 [ex:name] 테이블과 [rdfs:label] 테이블을 합집합(union)한 것을 [rdfs:label_view]로 정의하고 질의 응답에서는 이 뷰를 사용하도록 함으로써 rdfs:subPropertyOf에 의한 포함관계 함의 규칙의 추론을 대신할 수 있다.

rdfs:subClassOf에 의한 포함관계 함의 규칙 추론의 경우도 그림 3의 (b)에서 보는 바와 같이 [ex:Person] 테이블과 [ex:Professor] 테이블을 합집합(union)한 것을 [ex:Person_view]로 정의하고 질의 응답에서 이 뷰를 사용하도록 함으로써 해결된다. 마지막으로, owl:inverseOf에 의한 역관계 규칙 추론의 경우(c)도 주어와 목적어를 뒤바꾸는 과정이 추가로 들어가는 것을 제외하면 동일한 방법으로 해결된다.

이와 같이 DBMS 뷰는 물리적으로 저장하는 것이 아닌, 단지 정의만을 갖고 있기 때문에, 추론 확장에 의한 트리플을 저장하는 데 소요되는 시간을 단축할 수 있을 뿐만 아니라, 저장 공간의 효율성도 추구할 수 있다. 다만, 질의 응답시에 뷰를 통해 접근함에 따른 추가적인 시간 소요가 발생할 수 있으나, 이는 중복이 거의 없는 합집합(union) 연산에 해당되기 때문에, 적재 시에 얻어지는 효율성에 비해 무시할 수 있는 수준이라 할 수 있다.

Ⅴ. 결론

온톨로지가 대용량화되면서 온톨로지 시스템의 규모성 및 실용성에 대한 요구가 점차 커져가고 있다. 이런 상황에서 DBMS는 대용량 온톨로지 저장소를 위한 백엔드(back-end) 저장소로 많이 활용되고 있으며, 본 논문에서는 DBMS 기반의 온톨로지 저장소에서 RDFS 포함관계 함의 규칙 (rdfs7 규칙과 rdfs9 규칙)과 OWL 역관계 규칙(owl:inverseOf)의 추론을 효율적으로 수행할 수 있는 방법으로서, DB 테이블에 대한 뷰(view)를 활용하는 방법을 소개하였다. 추론 규칙을 뷰 정의로 대체하고 RDF 트리플을 구조화된 트리플 테이블에 저장하는 것으로 추론이 완료되며 대신 질의 처리과정에서는 그 뷰를 참조하면 된다. 이와 같이 뷰를 정의하는 것으로 추론을 대신함에 따라 추론에 소요되는 시간을 단축할 수 있고 트리플 저장소의 공간 효율성도 얻을 수 있다.

■ 참 고 문 헌 ■

[1] ISWC2008 Semantic Web Challenge, http://challenge.semanticweb.org/

[2] Sesame, http://www.openrdf.org/

[3] Virtuoso, http://virtuoso.openlinksw.com/wiki/main/Main/

[4] AllegroGraph, http://www.franz.com/agraph/allegrograph/

[5] Jena SDB, http://jena.hpl.hp.com/wiki/SDB

[6] Z. Pan and J. Heflin, "DLDB: Extending Relational Databases to Support Semantic Web Queries", Workshop on Practical and Scalable Semantic Systems, ISWC2003, 2003.

[7] S. Harris and N. Gibbins, "3store: Efficient Bulk RDF Storage", In Proceedings of the 1st International Workshop on Practical and Scalable Semantic Web Systems, 2003.

[8] Y. Theoharis, V. Christophides, and G. Karvounarakis, "Benchmakring Database Representations of RDF/S Stores", LNCS 3729 (ISWC2005), 2005.