R-DBMS기반 추론 서비스인 OntoThink-K®에서의 SPARQL 질의 지원

이승우° 정한민 성원경
한국과학기술정보연구원 정보시스템연구팀
{swlee°, jhm, wksung}@kisti.re.kr

Supporting SPARQL in OntoThink-K®, an Inference Service based on R-DBMS

Seungwoo Lee° , Hanmin Jung, Won-Kyung Sung
ISRL, Korea Institute of Science & Technology Information

요 약

시맨틱 웹 기술을 이용하는 추론 엔진들이 사용하는 지식은 기본적으로 주어(subject)와 술어(predicate), 목적어(object)로 구성된 트리플(triple)들의 집합이며, 이를 저장하기 위한 구조로 관계형 데이터베이스(R-DB)가 주로 이용된다. 본 논문은 DBMS 기반의 추론 서비스인 OntoThink-K®에서의 트리플 저장 구조와 함께 SPARQL 질의 언어를 지원하기 위한 SPARQL-SQL 매핑에 대해 설명한다. OntoThink-K®는 스키마 무관과 스키마 인지의 두 가지 방식의 트리플 저장 구조를 지원하며, 본 논문에서는 각 저장 구조에 따른 SPARQL-SQL 매핑 방법을 설명하고 실험을 통해 두 방식에서의 추론 속도의 차이를 비교한다. 이 실험 결과로부터 우리는 스키마 인지 방식을 사용함으로써 스키마 무관 방식에 비해 적어도 2배 이상의 속도 향상을 꾀할 수 있음을 알았다.

1. 서론

최근 시맨틱 웹에 대한 관심이 높아지면서 시맨틱 웹을 구현하기 위한 여러 가지 기술들이 연구· 발표 되고 있다. 대표적으로 RDF (Resource Description Framework)[1] 와 RDFS (RDF Schema)[2], OWL (Web Ontology Language)[3], SPARQL(SPARQL Protocol And RDF Query Language)[4] 등의 기술들은 W3C에 추천 혹은 추천 후보 상태에 있으며, 특히 RDF와 RDFS 및 OWL기반으로 지식을 표현하고 RDQL이나 nRQL, SPARQL을 통해 질의하는 추론 엔진들이 이미 개발되었거나 개발 중에 있다[1][2][3][4][5]. 추론 엔진들이 사용하는 지식은 기본적으로 주어(subject)와 술어(predicate), 목적어(object)로 구성된 트리플(triple)들의 집합이며, 이를 저장하기 위한 구조로 관계형 데이터베이스(R-DB)가 주로 이용된다. 관계형 데이터베이스는 이미 오랫동안 연구· 개발되어 왔으며 안정적인 성능을 제공하는 DBMS시스템들을 그대로 이용할 수 있다는 점이 그 주된 이유이다.

본 논문은 DBMS 기반의 추론 서비스인 OntoThink-K® 에서의 트리플 저장 구조와 함께 SPARQL을 질의 언어로 사용할 수 있도록 하는 SPARQL-SQL 매핑에 대해 설명한다. 본 논문이 기반으로 하는 OntoThink-K®는 트리플 저장을 위해 관계형 데이터베이스를 이용하는 추론 서비스의 하나로 연구자 간의 협업을 지원하기 위한 지식 기반 정보유통 플랫폼인 OntoFrame-K®[6]에 적용하여 연구자 네트워크와 연구자 정보, 연구성과 맵, 통계정보, 성과정보, 기관정보 등의 추론 서비스를 제공하는 것을 목적으로 한다[7].

관계형 데이터베이스에 트리플을 저장하는 구조는 크게 세 가지로 구분할 수 있다[8]. 첫째는 RDF 스키마와 무관한 (Schema-Oblivious) 방식으로 주어(Subject)와 술어 (Predicate), 목적어(Object)를 칼럼으로 하는 하나의 테이블에 트리플을 저장한다. 둘째는 스키마의 각 속성과 클래스를 개별 테이블로 구성하는 스키마 인지(Schema-Aware) 방식으로 주어와 목적어를 칼럼으로 하는 각각의 술어 테이블에 트리플을 저장한다. 셋째는 이 두 가지는 혼용한 (Hybrid) 방식으로 범위(range) 유형 별로 개별 테이블을 구성하며, 각 테이블은 주어와 술어, 목적어를 칼럼으로 갖는다. [8]에 따르면, 속도 측면에서 스키마 인지 방식이 스키마 무관 방식에 비해 월등히 우수하며, 혼용 방식은 스키마 인지 방식에 비해 근소하게 앞선다. OntoThink-K®는 이 중 처음 두 가지 방식의 트리플 저장 구조를 지원하며, 본 논문에서는 각 저장 구조에 따른 SPARQL-SQL 매핑과 함께 추론 속도의 차이를 설명하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 관계형 데이터베이스를 기반으로 시맨틱 웹 질의언어를 지원하는 과거 연구를 살펴본다. 3장에서는 OntoThink-K® 추론 서비스에서 요구되는 SPARQL 질의에 대해 기술하고, 트리플 저장 구조에 따라 이를 SQL로 변환하는 과정을 4장에서 설명한다. 5장에서는 실험을 통해 트리플 저장구조에 따른 SQL처리 속도 차이를 비교하고 6장에서 결론을 맺는다.

2. 관련 연구

관계형 데이터베이스를 기반으로 시맨틱 웹 질의언어를

[1] RDF, http://www.w3.org/RDF/
[2] RDFS, http://www.w3.org/TR/rdf-schema/
[3] OWL, http://www.w3.org/TR/owl-features/
[4] SPARQL, http://www.w3.org/TR/rdf-sparql-query/

지원하는 지금까지의 연구는 크게 두 가지로 나눠 볼 수 있다. 첫째는 기존의 관계형 데이터베이스로 표현된 데이터들을 SQL이 아닌 RDQL이나 SPARQL 등의 시맨틱 웹 질의언어를 사용하여 질의할 수 있도록 하는 방식이며[9][10][11], 둘째는 RDF 트리플 데이터를 관계형 데이터베이스에 저장하고 이를 시맨틱 웹 질의언어로 질의할 수 있도록 하는 방식이다[12][13]. 두 가지 모두 시맨틱 웹 질의언어를 SQL로 변환하는 과정을 포함한다는 점에서 공통점이 있지만, 전자는 데이터가 원래 관계형 데이터베이스로 구축되어 있는 경우이며 후자는 트리플 형태의 데이터의 저장구조로 안정적인 관계형 데이터베이스를 이용한다는 점에서 차이가 있다.

[9]와 [10]은 비RDF 데이터의 관계형 데이터베이스에 RDQL 질의를 사용할 수 있도록 하기 위해 관계형 데이터베이스 스키마를 RDFS/OWL 온톨로지로 매핑하는 과정을 포함한다. 그러나 이러한 매핑은 해당 응용에 제한적이다. [11]은 MySQL 서버 측에서 SPARQL을 지원하려는 시도로 질의를 변환하는 과정을 없앰으로써 효율성을 높이고자 하였다. 관계형 데이터를 RDF로 매핑하는 과정은 각 테이블의 주키 속성 값과 그 외 속성명 및 속성값을 RDF 트리플로 간주하는데, 이러한 매핑은 둘 이상의 칼럼이 주키 혹은 외래키를 구성할 때에는 적용될 수 없으며, SPARQL질의를 처리하기 위해 질의 구조로부터 주키를 추정해야 한다.

[12]와 [13]은 트리플 구조의 데이터베이스에 SPARQL 질의를 사용할 수 있도록 SPARQL-SQL 매핑을 지원한다. [12]는 사용하는 스키마에 무관한 Jena의 트리플 저장구조를 이용하며, [13]도 마찬가지로 스키마 무관 방식을 가정하고 있다. 이와 달리, 본 논문에서는 보다 효율적인 스키마 인지 방식으로 트리플을 저장하며 이에 따른 SPARQL-SQL 매핑 과정을 설명한다.

현재 SPARQL은 W3C에서 추천 후보 상태에 있으며, 우리가 아는 범위에서는 아직까지 완전한 SPARQL 명세를 지원하는 예는 없다. 본 논문에서도 OntoThink-K®에서 요구되는 간단한 SPARQL표현에 대해서만 다룬다.

3. OntoThink-K® 추론 서비스의 SPARQL 질의

OntoThink-K®[7]는 연구자 간의 협업을 지원하기 위한 지식 기반 정보유통 플랫폼 상에서 온톨로지와 URI서버로부터 트리플을 생성 및 확장하여 스키마 무관(Schema-Oblivious)과 스키마 인지(Schema-Aware)의 두 가지 방식으로 RDBMS에 저장하고, SPARQL질의를 받아들여 6가지의 추론 서비스를 제공한다. 여기에는 연구자 네트워크와 연구자 정보, 연구성과 맵, 통계정보, 성과정보, 기관정보 등이 포함된다. 연구자 네트워크는 다시 세부적으로 성과물 기반의 전문가 추천과 공저자 및 인용 관계를 통한 그룹핑 및 네트워크 생성 서비스를 포함한다. 연구성과 맵은 지역별로 성과물의 분포를 보여 주며, 통계정보는 특정 연구자들의 연도별 성과물 건수를 수치로 비교해 준다. 연구자 정보와 성과 정보, 기관 정보는 각각 조건에 맞는 연구자, 성과물, 기관의 목록을 제시해 준다.

OntoThink-K®에서 제공하는 6 가지 추론 서비스에 대한 SPARQL질의 예는 부록A 에 나와 있다. 각 추론 서비스에 공통적으로 적용되는 4가지 파라메타[5]는 SPARQL에서 다음과 같은 WHERE 절로 표현될 수 있다.

```
WHERE {
  ?outcome rdf:type T .
  ?outcome isrl:isClassifiedBy ?S . FILTER (?S=S1 || ?S=S2) .
  ?outcome isrl:isCategorizedBy ?C . FILTER (?C=C1 || ?C=C2) .
  ?outcome isrl:hasPublicationYear ?Y . FILTER (?Y>=Y1 && ?Y<=Y2) .
}
```

그림 1 공통제약에 대한 SPARQL 표현

여기서, T는 지정된 성과물 유형을, S1과 S2는 선택된 성과물 주제를, C1과 C2는 선택된 성과물 분야를, Y1과 Y2는 제한된 성과물의 연도범위를 가리킨다. 이들 파라메타가 특정한 값으로 지정되지 않은 경우에는 해당 트리플 및 FILTER 절은 생략된다.

공통 제약 외에 각 추론 서비스에 따라 적용할 수 있는 조건들이 있다. 예를 들어, 부록A의 ①은 전문가 추천을 위해 WHERE절에 조건에 맞는 성과물에 대한 저자(URI와 이름)와 가중치를 얻기 위한 트리플 조건을 표현한다. 마지막의 FILTER절은 특정URI가 할당되지 않은 저자는 결과에서 제외시키기 위한 조건이다. 다른 예로, ②는 공저자 중심의 그룹을 생성하기 위해, 공저자 관계에 있는 저자 쌍을 얻는데, 저자 순서를 달리 하여 중복된 쌍이 발생하는 것을 막기 위해 "FILTER (?order1<?order2)" 조건이 추가되었다. 또한 "FILTER (?CI1!=?CI2)"는 공저자 관계의 두 저자는 서로 달라야 함을 나타낸다. ④에서는 "FILTER (?creator=isrl:PER_5910092603)"를 통해 특정 연구자로 제약하며, ⑤의 FILTER절은 URI가 할당되지 않은 지역을 결과에서 제외시킨다. ⑦에서는 성과물을 유형별로 구분하기 위해 성과물 유형(?T)을 SELECT한다. 이때, 이 유형은 성과물 개체가 직접적으로 속한 클래스이지만 subClassOf 관계에 의한 확장을 통해 상위 클래스(isrl:Outcome) 또한 유형으로 얻어지게 된다. 이를 막기 위해 "FILTER (?T=isrl:Article || ?T=isrl:Patent || ?T=isrl:Report)"와 같이 성과물 유형을 명시적으로 제한다. "FILTER REGEX(?oTitle, '정보')"는 키워드('정보')를 통해 성과물의 제목을 부분 매칭으로 검색할 수 있도록 정규 표현식 조건을 가리킨다. ⑧의 "FILTER (?Y >= 2004 && ?Y<=2006)"는 성과물의 연도범위를 제약하기 위한 조건이다.

추론의 결과로 얻고자 하는 튜플(tuple)은 SELECT 절로 표현되며 중복된 튜플을 제거하기 위해 DISTINCT 제한자를 가질 수 있다. 또한 추론 결과 튜플들을 특정 필드를 기준으로 정렬하기 위해 ORDER BY 절을 사용된다. SPARQL질의 표현의 간결성을 위해 긴 namespace는 짧은 prefix로 대신하여 표현될 수 있다.

사용자로부터 SPARQL질의를 직접 받아들일 수도 있지만, 실제로는 GUI를 통해 사용자가 파라메타를 선택하면 이로부터 SPARQL질의가 자동으로 생성되며 SQL로의 변환을 거쳐 RDBMS로부터 질의에 대한 결과를 얻는다. 이 결과는 각 추론 서비스에 따른 후처리 과정을 거쳐 XML형식으로 시각화 인터페이스에 전달된다.

[5] 연구 성과물에 대한 주제와 분야, 유형(보고서, 특허, 논문), 창작연도 범위를 공통제약으로 가지며, 각 추론 서비스 별로 추가적인 제약(e.g., 통계정보의 경우 연구자URI 리스트)을 가질 수 있다.

4. SPARQL-SQL 변환

여기서는 앞장에서 설명한 SPARQL질의에서 사용된 각 구성요소가 SQL 질의의 어떤 부분으로 변환되는지를 설명한다. 이 변환은 트리플의 저장구조가 스키마 무관 방식인지, 스키마 인지 방식인지에 따라 달라진다. 스키마 무관 방식에서는 Property의 종류에 상관없이 트리플이 하나의 테이블에 저장되기 때문에, 이 테이블의 주어와 술어, 목적어 칼럼의 값을 검사하는 데에 반해, 스키마 인지 방식에서는 Property 종류별로 테이블을 하나씩 가지기 때문에, 트리플의 술어에 따라 해당 테이블의 주어와 목적어 칼럼의 값을 검사하게 된다[6].

먼저, SPARQL의 prefix표현은 SQL에서 full namespace 표현으로 대체된다. 이는 트리플 데이터가 DB에 full namespace 표현으로 저장되어 있기 때문이다.

다음으로, WHERE 절은 SQL에서는 FROM 절과 WHERE 절의 결합으로 변환될 수 있다. WHERE 절의 트리플 조건 하나는 SQL에서는 하나의 테이블에 대한 접근을 필요로 한다. 즉, 트리플 조건의 수만큼의 테이블을 접근해야 하며, 트리플 조건에서 공유된 변수는 해당 테이블 사이의 내부 조인(INNER JOIN)으로 표현될 수 있다. 예를 들어, 그림 1의 처음 두 트리플 조건은 변수(?outcome)를 서로 공유하므로 두 개의 테이블(rdf_type과 isrl_isClassifiedBy)에 대한 내부 조인으로 표현될 수 있는데, 다음의 두 가지 SQL로 변환될 수 있다. 트리플 조건의 변수(e.g., ?outcome)는 테이블의 해당 칼럼(e.g., T1.Subject)으로 매핑되며, 상수(e.g., 'T')는 테이블의 해당 칼럼의 값에 대한 제약(e.g., T1.Object='T')으로 표현된다.

```
FROM rdf_type AS T1 INNER JOIN isrl_isClassifiedBy AS T2
  ON T2.Subject=T1.Subject
WHERE T1.Object='T'
```

```
FROM rdf_type AS T1, isrl_isClassifiedBy AS T2
WHERE T1.Object='T'
  AND T2.Subject=T1.Subject
```

그림 2 내부 조인의 두 가지 표현

그림 2는 스키마 인지 방식의 경우이고 스키마 무관 방식에 해당하는 SQL 예는 그림 3과 같이 하나의 테이블(ETriples)만이 사용되며 술어(Predicate)의 값이 항상 검사된다.

```
FROM ETriples AS T1, ETriples AS T2
WHERE
  T1.Predicate='http://www.w3.org/1999/02/22-rdf-syntax-ns#type'
  AND T1.Object='T'
  AND T2.Subject=T1.Subject
  AND T2.Predicate='http://www.kisti.re.kr/isrl#isClassifiedBy'
```

그림 3 스키마 무관 방식에서의 SQL 변환 예

WHERE 절 내의 FILTER 표현식은 SQL에서는 사용된 변수에 대한 테이블 칼럼의 값에 대한 비교 조건으로 WHERE 절에 추가된다. 예를 들어, 그림 1의 "FILTER (?S=S1 || ?S=S2)"는 WHERE 절에 "(T2.Object = 'S1' OR T2.Object = 'S2')"로 추가된다.

FILTER 표현식 중 부분 매칭을 지원하는 정규 표현식 제약은 SQL에서는 'LIKE' 구문으로 변환될 수 있다. 예를 들어, "FILTER REGEX(?oTitle, '정보')"는 "(T6.Object LIKE '%정보%')"로 WHERE 절에 추가된다.

SPARQL의 SELECT 절은 SQL의 SELECT 절로 바로 매핑될 수 있다. 다만, SPARQL의 변수는 WHERE 절에 나타나는 그 변수에 대응하는 테이블 칼럼으로 치환되어야 한다. 예를 들어, 부록A의 ①에서 SELECT 절의 ?creator는 WHERE절의 둘째 트리플의 목적어이면서 넷째 트리플의 주어에 해당하므로 T2.Object 혹은 T4.Subject로 치환된다. ORDER BY 절도 SELECT 절의 경우와 마찬가지로 간단히 매핑된다.

지금까지 설명한 변환 과정을 거쳐, 그림 1의 SPARQL은 그림 4의 SQL로 변환되며, 6가지의 추론 서비스 각각의 SPARQL질의에 대한 SQL 변환 예는 부록에 실려 있다.

```
FROM rdf_type AS T1, isrl_isClassifiedBy AS T2,
isrl_isCategorizedBy AS T3, isrl_hasPublicationYear AS T4,
isrl_createdByLocation AS T5
WHERE T1.Object='T'
 AND T2.Subject=T1.Subject
 AND T3.Subject=T1.Subject
 AND T4.Subject=T1.Subject
 AND T5.Subject=T1.Subject
 AND ( T2.Object = 'S1' OR T2.Object = 'S2' )
 AND ( T3.Object = 'C1' OR T3.Object = 'C2' )
 AND ( T4.Object >= '2004' AND T4.Object <= '2006' )
```

그림 4 공통제약에 대한 변환된 SQL

5. 실험

관계형 데이터베이스에 트리플을 저장하는 두 가지 구조(스키마 무관 vs. 스키마 인지)에 추론 서비스 모듈의 효율성을 비교하기 위해 각각의 SQL 실행 속도를 측정해 보았다.

실험에 사용된 RDF 트리플 수는 1,112,100개로, KISTI 내부 성과 정보와 외부 성과 정보를 수집하고 가공하여 생성되었다[7].

연구자 네트워크의 3 가지 세부 서비스를 포함하여 모두 8가지의 추론 서비스에 대해 SQL소요 시간이 각각 가장 길도록 파라메타를 설정하였다. 이는 두 가지 방식의 처리 소요 시간 차이를 보다 두드러지게 보이기 위함이다. 실험 결과는 표 1와 같다. 이로부터 스키마 인지 방식은 스키마 무관 방식에 비해 적어도 2배의 속도 향상을 가져옴을 알 수 있다. 이는 [8]의 결과와도 일치한다.

표 1 트리플 저장 구조에 따른 SQL 처리 속도 비교

추론 서비스	소요 시간 (초)		비율
	스키마 무관	스키마 인지	
연구자네트워크 – 전문가 추천	9	2	0.2
연구자네트워크 – 공저자	14	5	0.4
연구자네트워크 – 인용	1	0.4	0.4
연구자 정보	6	2	0.3
연구성과 맵	0.7	0.2	0.3
통계 정보	0.016	0.008	0.5
성과 정보	14	5	0.4
기관 정보	2	0.8	0.4

6. 결론

시맨틱 웹 기술을 이용하는 추론 엔진들이 사용하는 지식은 기본적으로 주어(subject)와 술어(predicate), 목적어(object)로 구성된 트리플(triple)들의 집합이며, 이를 저장

[6] Prefix가 결합된 술어(Predicate)를 테이블 이름으로 사용하였다.

하기 위한 구조로 관계형 데이터베이스(R-DB)가 주로 이용된다.

본 논문은 DBMS 기반의 추론 서비스인 OntoThink-K® 에서의 트리플 저장 구조와 함께 SPARQL 질의 언어를 지원하기 위한 SPARQL-SQL 매핑에 대해 설명하였다. OntoThink-K®는 스키마 무관과 스키마 인지의 두 가지 방식의 트리플 저장 구조를 지원하며, 본 논문에서는 각 저장 구조에 따른 SPARQL-SQL 매핑 방법을 설명하고 실험을 통해 두 방식에서의 추론 속도의 차이를 비교하였다. 이를 통해, 우리는 스키마 인지 방식을 사용함으로써 스키마 무관 방식에 비해 적어도 2배 이상의 속도 향상을 꾀할 수 있음을 알았다.

현재의 SPARQL-SQL 매핑은 OntoThink-K®에서 요구하는 표현만을 다루었으나, 앞으로는 범용의 추론 서비스에서도 사용할 수 있도록 다양한 SPARQL 표현을 지원해 나갈 예정이다.

참고문헌

[1] KAON2, http://kaon2.semanticweb.org/

[2] RacerPro, http://www.franz.com/products/racer/

[3] OntoBroker, http://ontobroker.semanticweb.org/

[4] Pellet, http://www.mindswap.org/2003/pellet/

[5] Jena2, http://jena.sourceforge.net/

[6] 정한민, 이미경, 성원경, 박동인, "OntoFrame-K: 연구자 간 협업 지원 서비스를 위한 시멘틱 웹 기반 정보 유통 플랫폼", 한국컴퓨터종합학술대회, 2006.

[7] 정한민, 강인수, 이미경, 이승우, 성원경, "OntoThink-K: DBMS 기반 추론 서비스", 한국정보과학회 추계학술대회, 2006. (제출 중)

[8] Y. Theoharis, V. Christophides, G. Karvounarakis, "Benchmarking Database Representation of RDF/S Stores", In Proceedings of the 4th International Semantic Web Conference, 2005.

[9] C. Bizer, A. Seaborne, "D2RQ – Treating Non-RDF Databases as Virtual RDF Graphs", In Proceedings of the 3rd International Semantics Web Conference (ISWC2004), Hiroshima, Japan, November 2004.

[10] C. P. de Laborda, S. Conrad, "Querying Relational Databases with RDQL", In Proceedings of Berliner XML Tage 2005, Berlin, Germany, September 2005.

[11] SPASQL: SPARQL Support In MySQL, http://www.w3.org/2005/05/22-SPARQL-MySQL/XTech#features

[12] sparql2sql – a query engine for SPARQL over Jena triple stores, http://jena.sourceforge.net/sparql2sql

[13] A. Chebotko, S. Lu, H. M. Jamil, F. Fotouhi, "Semantics Preserving SPARQL-to-SQL Query Translation for Optional Graph Patterns", Technical Report TR-DB-052006-CLJF, May 2006.

부록

A. 추론 서비스 별 SPARQL 질의 및 변환된 SQL 질의 예

① 연구자 네트워크 – 전문가 추천

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT ?creator ?cName ?score
WHERE {
  ?outcome isrl:hasCreationInformation ?CI .
  ?CI isrl:hasCreator ?creator .
  ?CI isrl:contributionWeightOfCreator ?score .
  ?creator isrl:nameOfPerson ?cName .
  FILTER (?creator!='http://www.kisti.re.kr/isrl#PER_0000000000[7]') .
}
ORDER BY ?creator
```

```
SELECT T2.Object AS creator, T4.Object AS cName, T3.Object AS score
FROM isrl_hasCreationInformation AS T1, isrl_hasCreator AS T2, isrl_contributionWeightOfCreator AS T3, isrl_nameOfPerson AS T4
WHERE T2.Subject=T1.Object AND T3.Subject=T1.Object
 AND T4.Subject=T2.Object
 AND ( T2.Object != 'http://www.kisti.re.kr/isrl#PER_0000000000' )
ORDER BY T2.Object
```

② 연구자 네트워크 – 공저자 중심 그룹 및 네트워크

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT ?creator1 ?cName1 ?creator2 ?cName2
WHERE {
  ?outcome isrl:hasCreationInformation ?CI1 .
  ?CI1 isrl:hasCreator ?creator1 . ?CI1 isrl:orderOfCreator ?order1 .
  ?outcome isrl:hasCreationInformation ?CI2 . FILTER (?CI1!=?CI2) .
  ?CI2 isrl:hasCreator ?creator2 . ?CI2 isrl:orderOfCreator ?order2 .
  FILTER (?order1<?order2) .
  ?creator1 isrl:nameOfPerson ?cName1 .
  ?creator2 isrl:nameOfPerson ?cName2 .
  FILTER (?creator1!='http://www.kisti.re.kr/isrl#PER_0000000000') .
  FILTER (?creator2!='http://www.kisti.re.kr/isrl#PER_0000000000') .
}
```

```
SELECT T2.Object AS creator1, T7.Object AS cName1, T5.Object AS creator2, T8.Object AS cName2
FROM isrl_hasCreationInformation AS T1, isrl_hasCreator AS T2, isrl_orderOfCreator AS T3, isrl_hasCreationInformation AS T4, isrl_hasCreator AS T5, isrl_orderOfCreator AS T6, isrl_nameOfPerson AS T7, isrl_nameOfPerson AS T8
WHERE T2.Subject=T1.Object AND T3.Subject=T1.Object
 AND T4.Subject=T1.Subject AND T5.Subject=T4.Object
 AND T6.Subject=T4.Object AND T7.Subject=T2.Object
 AND T8.Subject=T5.Object AND ( T1.Object != T4.Object )
 AND ( T3.Object < T6.Object )
 AND ( T2.Object != 'http://www.kisti.re.kr/isrl#PER_0000000000' )
 AND ( T5.Object != 'http://www.kisti.re.kr/isrl#PER_0000000000' )
```

③ 연구자 네트워크 – 인용 중심 그룹 및 네트워크

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT ?citingOutcome ?citingCreator ?citingName ?citedCreator ?citedName
WHERE {
  ?citingOutcome isrl:hasCitationOfOutcomes ?citedOutcome .
  ?citingOutcome isrl:createdByPerson ?citingCreator .
  ?citedOutcome isrl:createdByPerson ?citedCreator .
  ?citingCreator isrl:nameOfPerson ?citingName .
  ?citedCreator isrl:nameOfPerson ?citedName .
  FILTER
(?citingCreator!='http://www.kisti.re.kr/isrl#PER_0000000000') .
  FILTER
(?citedCreator!='http://www.kisti.re.kr/isrl#PER_0000000000') .
}
ORDER BY ?citedCreator ?citingOutcome
```

```
SELECT T1.Subject AS citingOutcome, T2.Object AS citingCreator, T4.Object AS citingName, T3.Object AS citedCreator, T5.Object AS citedName
FROM isrl_hasCitationOfOutcomes AS T1, isrl_createdByPerson AS T2, isrl_createdByPerson AS T3, isrl_nameOfPerson AS T4, isrl_nameOfPerson AS T5
WHERE T2.Subject=T1.Subject AND T3.Subject=T1.Object
 AND T4.Subject=T2.Object AND T5.Object=T3.Object
 AND ( T2.Object != 'http://www.kisti.re.kr/isrl#PER_0000000000' )
```

[7] 특정 URI가 할당되지 않은(unknown) 연구자는 제외시킨다.

```
AND ( T3.Object != 'http://www.kisti.re.kr/isrl#PER_0000000000' )
ORDER BY T3.Object, T1.Subject
```

④ 연구자 정보 (특정 연구자 제약)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT
DISTINCT ?creator ?cName ?institution ?iName ?department ?dName
WHERE {
   ?creator isrl:hasInstitutionOfPerson ?institution .
   ?creator isrl:hasDepartmentOfPerson ?department .
   ?creator isrl:nameOfPerson ?cName .
   ?institution isrl:nameOfInstitution ?iName .
   ?department isrl:nameOfDepartment ?dName .
   FILTER (?creator=isrl:PER_5910092603) .
}
ORDER BY ?creator
```

```
SELECT DISTINCT T1.Subject AS creator, T3.Object AS cName,
T1.Object AS institution, T4.Object AS iName, T2.Object AS department,
T5.Object AS dName
FROM isrl_hasInstitutionOfPerson AS T1, isrl_hasDepartmentOfPerson
AS T2, isrl_nameOfPerson AS T3, isrl_nameOfInstitution AS T4,
isrl_nameOfDepartment AS T5
WHERE T2.Subject=T1.Subject AND T3.Subject=T1.Subject
AND T4.Subject=T1.Object AND T5.Subject=T2.Object
AND ( T1.Subject = 'http://www.kisti.re.kr/isrl#PER_5910092603' )
ORDER BY T1.Subject
```

⑤ 연구성과 맵 (특정 주제, 특정 유형 제약)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT ?zipcode ?outcome
WHERE {
   ?outcome rdf:type isrl:Article .
   ?outcome isrl:isClassifiedBy ?S . FILTER (?S=isrl:TOP_GE0006956) .
   ?outcome isrl:createdByLocation ?zipcode .
   FILTER (?zipcode!='http://www.kisti.re.kr/isrl#LOC_000000'[8]) .
}
```

```
SELECT T3.Object AS zipcode, T1.Subject AS outcome
FROM rdf_type AS T1, isrl_isClassifiedBy AS T2, isrl_createdByLocation
AS T3
WHERE T1.Object='http://www.kisti.re.kr/isrl#Article'
AND T2.Subject=T1.Subject AND T3.Subject=T1.Subject
AND ( T2.Object = 'http://www.kisti.re.kr/isrl#TOP_GE0006956' )
AND ( T3.Object != 'http://www.kisti.re.kr/isrl#LOC_000000' )
```

⑥ 통계 정보 (연구자 3명)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT DISTINCT ?creator ?cName ?Y ?outcome
WHERE {
   ?outcome isrl:hasPublicationYear ?Y .
   ?creator isrl:creatorOf ?outcome .
   FILTER (?creator=isrl:PER_0000003725
|| ?creator=isrl:PER_5610072143 || ?creator=isrl:PER_7010186243) .
   ?creator isrl:nameOfPerson ?cName .
}
ORDER BY ?creator ?Y
```

```
SELECT DISTINCT T2.Subject AS creator, T3.Object AS cName,
T1.Object AS Y, T1.Subject AS outcome
FROM isrl_hasPublicationYear AS T1, isrl_creatorOf AS T2,
isrl_nameOfPerson AS T3
WHERE T2.Object=T1.Subject AND T3.Subject=T2.Subject
AND ( T2.Subject = 'http://www.kisti.re.kr/isrl#PER_0000003725' OR
T2.Subject = 'http://www.kisti.re.kr/isrl#PER_5610072143' OR
T2.Subject = 'http://www.kisti.re.kr/isrl#PER_7010186243' )
ORDER BY T2.Subject, T1.Object
```

⑦ 성과 정보 (성과물 제목 키워드 제약)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT DISTINCT ?outcome ?oTitle ?T ?Y ?order ?creator ?cName
WHERE {
   ?outcome rdf:type ?T .
   ?outcome isrl:hasPublicationYear ?Y .
   ?outcome isrl:hasCreationInformation ?CI .
   ?CI isrl:orderOfCreator ?order .
   ?CI isrl:hasCreator ?creator .
   ?outcome isrl:nameOfOutcomes ?oTitle .
   ?creator isrl:nameOfPerson ?cName .
   FILTER (?T=isrl:Article || ?T=isrl:Patent || ?T=isrl:Report) .
   FILTER REGEX(?oTitle, '정보') .
}
ORDER BY ?T ?outcome ?order
```

```
SELECT DISTINCT T1.Subject AS outcome, T6.Object AS oTitle,
T1.Object AS T, T2.Object AS Y, T4.Object AS order, T5.Object AS
creator, T7.Object AS cName
FROM rdf_type AS T1, isrl_hasPublicationYear AS T2,
isrl_hasCreationInformation AS T3, isrl_orderOfCreator AS T4,
isrl_hasCreator AS T5, isrl_nameOfOutcomes AS T6,
isrl_nameOfPerson AS T7
WHERE T2.Subject=T1.Subject AND T3.Subject=T1.Subject
AND T4.Subject=T3.Object AND T5.Subject=T3.Object
AND T6.Subject=T1.Subject AND T7.Subject=T5.Object
AND ( T1.Object = 'http://www.kisti.re.kr/isrl#Article' OR T1.Object =
'http://www.kisti.re.kr/isrl#Patent' OR T1.Object =
'http://www.kisti.re.kr/isrl#Report' )
AND (T6.Object LIKE '%정보%' )
ORDER BY T1.Object, T1.Subject, T4.Object
```

⑧ 기관 정보 (특정 분야, 특정 연도범위 제약)

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX isrl: <http://www.kisti.re.kr/isrl#>
SELECT DISTINCT ?institution ?iName ?zipcode ?address
WHERE {
   ?institution isrl:hasLocationOfInstitution ?zipcode .
   ?institution isrl:nameOfInstitution ?iName .
   ?zipcode isrl:addressOfLocation ?address .
   ?outcome isrl:isCategorizedBy ?C . FILTER (?C=isrl:CAT_010100) .
   ?outcome isrl:hasPublicationYear ?Y .
   FILTER (?Y>=2004 && ?Y<=2006) .
   ?outcome isrl:createdByInstitution ?institution .
   FILTER (?institution!='http://www.kisti.re.kr/isrl#INS_000000'[9]) .
}
ORDER BY ?institution
```

```
SELECT DISTINCT T1.Subject AS institution, T2.Object AS iName,
T1.Object AS zipcode, T3.Object AS address
FROM isrl_hasLocationOfInstitution AS T1, isrl_nameOfInstitution AS
T2, isrl_addressOfLocation AS T3, isrl_isCategorizedBy AS T4,
isrl_hasPublicationYear AS T5, isrl_createdByInstitution AS T6
WHERE T2.Subject=T1.Subject AND T3.Subject=T1.Object
AND T5.Subject=T4.Subject AND T6.Subject=T4.Subject
AND T6.Object=T1.Subject
AND ( T4.Object = 'http://www.kisti.re.kr/isrl#CAT_010100' )
AND ( T5.Object >= '2004' AND T5.Object <= '2006' )
AND ( T1.Subject != 'http://www.kisti.re.kr/isrl#INS_000000' )
ORDER BY T1.Subject
```

[8] 특정 URI가 할당되지 않은(unknown) 지역은 제외시킨다.

[9] 특정 URI가 할당되지 않은(unknown) 기관은 제외시킨다.