韓國컴퓨터情報學會 論文誌
第16卷 第7號, 2011. 7.

2011-16-7-1-4

관계형 데이터베이스를 OWL 2 웹 온톨로지로 사용하기 위한 OWL/관계형 매핑 규칙

최 지 웅* 김 명 호**

OWL/Relational Mapping Rules to Use Relational Databases as OWL 2 Web Ontologies

Ji Woong Choi* Myung Ho Kim**

요 약

본 논문은 관계형 데이터베이스로부터 OWL 온톨로지를 자동으로 생성할 수 있는 규칙을 제안한다. 이 규칙의 목적은 기존 관계형 데이터베이스 데이터를 데이터베이스 스키마 변형과 데이터 마이그레이션과 같은 별도의 과정을 거치지 않고도 시맨틱 웹 환경에서 사용할 수 있도록 하는 것이다. 즉, 이 규칙은 RDBMS가 웹 온톨로지 저장소 역할을 동시에 수행하는 것을 돕는다. 그러나 기존의 관계형 데이터베이스와 OWL 사이의 매핑 규칙들을 본 연구의 목적을 위하여 사용하고자 할 경우 다음과 같은 문제가 발생한다. 첫째, 특정 구조의 테이블이 존재하는 데이터베이스로부터는 OWL 온톨로지를 생성할 수 없다. 둘째, 하나의 개체 추출을 위하여 높은 비용의 데이터베이스 조인 연산 혹은 여러 개의 SQL 질의가 불필요하게 수반된다. 반면에 본 논문에서 제안하는 규칙은 이러한 문제들을 방지하도록 설계되었으며, 데이터베이스 스키마로부터 OWL 클래스와 프로퍼티를 생성하며, 데이터베이스 인스턴스로부터 OWL 개체를 생성한다. 또한 이 규칙을 적용하여 생성한 OWL 온톨로지는 OWL 2 DL 사양을 만족시킨다.

▸ Keyword : 웹 온톨로지 언어, 관계형 데이터베이스, 시맨틱 웹, 매핑 규칙

Abstract

This paper proposes a set of rules to automatically generate OWL ontologies from relational databases. The purpose of the rules is to allow semantic access to existing RDB data without any database schema transformation and data migration process. In other words, the rules help a RDBMS play as a web ontology repository as well. However, the use of the mapping rules between RDB and OWL proposed by other studies for the objective causes troubles as follows. First, databases including the tables with a specific structure can't be translated into OWL. Second, the process for extracting an OWL individual unnecessarily lead to database join operations, or several

• 제1저자 : 최지웅 교신저자 : 김명호
• 투고일 : 2010.11.19, 심사일 : 2011.03.10, 게재확정일 : 2011.03.22.
* 숭실대학교 IT대학 컴퓨터학과(Dept. of Computer Science, Soongsil University)
** 숭실대학교 IT대학 컴퓨터학부(School of Computer Science and Engineering, Soongsil University)

SQL queries. On the other hand, our rules is designed to prevent these problems, can generate OWL classes and properties from database schemas and can generate OWL individuals from the database instances. In addition, an ontology generated by our rules is an OWL 2 DL ontology.

▸ Keyword : Web Ontology Language, Relational Database, Semantic Web, Mapping Rules

Ⅰ. 서 론

관계형 데이터베이스는 웹 애플리케이션을 통해 웹으로 데이터를 제공하는 주요 데이터 원천 역할을 담당하고 있다. 이와 유사하게 OWL 온톨로지는 시맨틱 웹을 위한 지식 베이스이다. Tim Berners-Lee[1]에 의하면 시맨틱 웹은 현재의 웹의 확장이기 때문에 데이터베이스 기술과 온톨로지 기술을 결합하기 위한 다양한 시도 및 연구가 이미 존재한다. 이러한 목적을 위한 기존 연구들의 접근법은 크게 두 가지로 분류될 수 있다.

첫 번째 접근법은 DBMS를 단순히 온톨로지 구성요소들을 저장하기 위한 온톨로지 저장소로서 사용하고자 하는 접근법이다. 이는 DBMS의 이미 검증된 안정된 대량 데이터 관리 능력을 온톨로지를 위하여 사용하고자 하는 것이다. 그 예로는 Virtuoso[2], D2RQ[3], Triplify[4], RDB2OWL[5]의 연구가 있다. 그러나 현재 이러한 연구들이 갖는 한계는 데이터베이스의 관계형 모델과 OWL 모델 사이의 차이로 인해 RDF 혹은 RDFS 수준의 온톨로지만을 지원한다는 것이다. 즉, 온톨로지 저장소로서의 데이터베이스는 RDF 트리플(triple)들을 저장하기 위한 트리플 저장소로써 사용됨을 의미하며 이를 위해 데이터베이스 스키마 또한 트리플을 저장하기 적합한 구조를 취한다. 시맨틱 질의 또한 SPARQL과 같은 RDF 기반의 시맨틱 질의어를 지원하며 온톨로지를 대상으로 수행되는 추론 기능 또한 RDFS 기반의 비교적 간단한 추론만을 제공함을 뜻한다.

두 번째 접근법은 데이터베이스가 이미 저장하고 있는 대량의 정제된 형태의 데이터를 시맨틱 웹 환경에서도 온톨로지 형태를 빌어 제공하고자 하는 접근법이다. 그 예로는 DB2OWL[6] 그리고 [7-10]의 연구가 있다. 이러한 접근법에 기반한 연구들은 RDF보다는 높은 수준의 시맨틱 서비스가 가능한 OWL 기반의 온톨로지를 데이터베이스로부터 직접 생성하고자 한다. 이를 위해서는 데이터베이스의 구성 요소들을 온톨로지 구성 요소로 재구성 및 표현할 수 있어야 한다. 그러나 현재 이러한 연구들의 한계는 아직 데이터베이스 스키마 정보만을 기반으로 하여 클래스와 프로퍼티 만으로 구성된 OWL 온톨로지 스키마만을 생성하는 수준에 머물러 있다. 따라서 이러한 관계형 모델-OWL 맵핑 규칙들은 OWL 개체 추출 단계까지 고려하지 않았기 때문에 이에 기반하여 데이터베이스 인스턴스로부터 OWL 개체를 추출하고자 할 경우 특정 구조의 테이블들로부터는 전혀 OWL 개체를 추출해 낼 수 없는 문제점과 하나의 개체 추출이 값비싼 비용의 조인 연산 혹은 여러 개의 SQL 질의문을 필요로 하는 문제를 가지고 있다.

따라서 본 논문의 접근법은 첫 번째 접근법과 비교하여 그림 1과 같이 데이터베이스 스키마 변경 및 데이터 마이그레이션 과정을 발생시키지 않으면서 이미 구축되어 있는 관계형 데이터베이스로부터 OWL 2 DL 온톨로지를 생성할 수 있는 시스템 구조를 목적으로 한다. 또한 이러한 구조를 위하여 고안한 본 논문의 매핑 규칙은 두 번째 접근법에 해당하는 기존 매핑 규칙들을 개선하여 OWL 개체 추출 시 불필요한 조인 연산의 발생을 억제시킬 수 있다.

본 논문의 구성은 다음과 같다. 제2장에서는 관계형 데이터베이스로부터 OWL 온톨로지를 생성하는 것을 목적으로 하는 기존 연구들의 문제점을 분석하여 이를 방지하기 위한 규칙의 요구 사항을 추출한다. 제3장에서는 본 논문이 제안하는 매핑 규칙을 기술한다. 제4장은 결론에 해당한다.

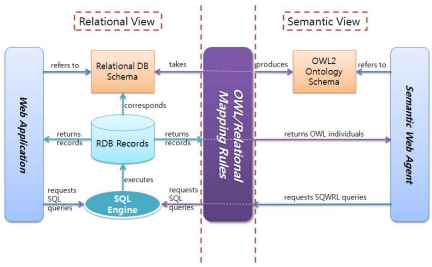

그림 1. OWL/관계형 매핑 규칙 프레임워크 아키텍처
Fig. 1. OWL/Relational Mapping Rules framework architecture

Ⅱ. 관련 연구

이 장은 관계형 데이터베이스로부터 OWL 온톨로지를 생성하는 것을 목적으로 하는 기존 연구들[6-10]의 매핑 규칙

을 분석하여 이들의 문제점과 이를 해소할 수 있는 새로운 규칙을 위한 요구 사항을 도출한다.

1. 기존 연구들의 매핑 규칙

관계형 데이터베이스로부터 OWL 온톨로지를 생성하는 것을 목적으로 하는 기존 연구들의 공통된 두 모델 요소 사이의 사상 관계는 다음과 같다.

- composite primary key를 갖는 테이블을 제외한 하나의 테이블은 하나의 클래스로 매핑된다.
- 하나의 foreign key 컬럼은 하나의 오브젝트 프로퍼티로 매핑된다. 이 때, 그 오브젝트 프로퍼티의 정의역(domain)은 그 컬럼을 포함하는 테이블로부터 유도된 클래스이며 치역(range)은 그 컬럼이 참조하는 테이블로부터 유도된 클래스이다.
- foreign key가 아닌 하나의 컬럼은 하나의 데이터 프로퍼티로 매핑되며 그 컬럼의 데이터 타입은 그것에 대응하는 XML schema data type[11]에 정의되어 있는 하나의 데이터 타입으로 매핑된다.

표 1. 기존 연구들의 RDB와 OWL 사이의 매핑
Table 1. mapping between RDB and OWL in existing studies

Relational Database		OWL
database schema	a table	a named class
	a foreign key column	an object property
	a non-foreign key column	a data property, an XSD data type
database instance	a row	not explicitly specified or anonymous individual
	a non-foreign key column	a literal

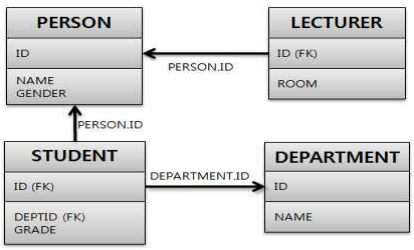

그림 2. College 데이터베이스 스키마 다이어그램
Fig. 2. College Database schema diagram

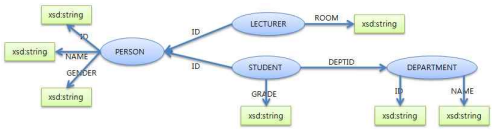

그림 3. 그림 2에 대한 RDF 스키마 그래프
Fig. 3. a RDF schema graph for Fig. 2

그림 3은 그림 2의 데이터베이스 스키마를 기존 연구들의 매핑 규칙을 적용하여 생성한 온톨로지 스키마를 RDF 그래프로 표현한 것이다.

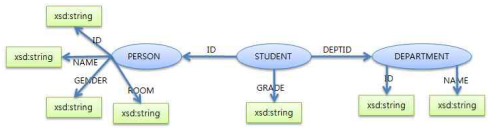

그림 4. LECTURER가 PERSON으로 통합된 RDF 스키마 그래프
Fig. 4. a RDF schema graph in which LECTURER is integrated into PERSON

기존 연구들 중에서 [7]과 [10]은 여러 개의 테이블을 하나의 클래스로 매핑시키는 규칙을 포함하고 있다. 이 규칙을 위한 조건은 각각의 연구마다 차이가 있다. 그림 4는 연구 [7]의 규칙을 적용하여 PERSON 테이블과 LECTURER 테이블을 PERSON 클래스로 매핑시킨 RDF 그래프이다. 따라서 그림 4는 그림 3과 비교하여 LECTURER 클래스가 존재하지 않으며 ROOM 데이터 프로퍼티의 정의역이 LECTURER 클래스가 아닌 PERSON 클래스이다.

기존 연구들은 원칙적으로 하나의 테이블을 하나의 클래스로 매핑시키고자 한다. 하지만 기존 연구들은 하나의 테이블을 두 개의 오브젝트 프로퍼티로 매핑시키는 예외 규칙을 공통적으로 포함한다. 그 규칙은 다음과 같다.

- primary key 이면서 동시에 foreign key인 두 개의 컬럼 c_1과 c_2로 구성되어 있는 하나의 테이블에 대해서, c_1은 테이블 T_1을 참조하고 c_2는 테이블 T_2를 참조한다면, c_1은 하나의 오브젝트 프로퍼티 op_1으로 매핑되며 c_2는 하나의 오브젝트 프로퍼티 op_2로 매핑된다. 이 때, op_1의 정의역은 T_2로부터 유도된 클래스 cls_2이고 op_1의 치역은 T_1으로부터 유도된 cls_1이다. 그리고 op_2의 정의역은 cls_1이고 op_2의 치역은 cls_2이다. 그리고 op_1과 op_2는 각각 서로의 역(inverse)이다.

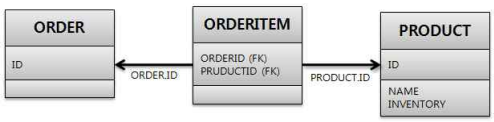

그림 5. Commerce 데이터베이스 스키마 다이어그램 1
Fig. 5. Commerce Database schema diagram 1

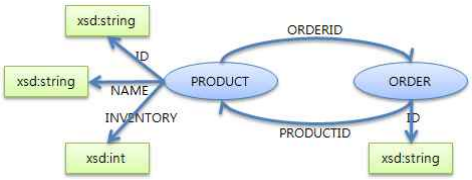

그림 6. 그림 5에 대한 RDF 스키마 그래프
Fig. 6. RDF schema graphs for Fig. 5

그림 6은 그림 5의 데이터베이스 스키마를 예외 규칙을 포함하여 적용한 온톨로지 스키마 RDF 그래프이다. ORDERITEM 테이블은 예외 규칙의 조건을 만족시키는 테이블이다. 따라서 ORDERITEM 테이블은 ORDERITEM 클래스가 아닌 ORDERID 오브젝트 프로퍼티와 PRODUCTID 오브젝트 프로퍼티로 매핑된다.

기존 연구 [8]은 데이터베이스 인스턴스로부터 OWL 개체와 리터럴을 생성하는 규칙을 포함한다. 이 규칙은 하나의 테이블의 하나의 행(row)을 하나의 익명 개체(anonymous individual)로 매핑시키며 그 행의 하나의 컬럼을 위한 데이터 값을 하나의 리터럴로 매핑시킨다. 다음 RDF/XML 코드는 DEPARTMENT 테이블의 ID 컬럼의 값이 'D111' 이고 NAME 컬럼의 값이 'the Department of Economics'인 행을 OWL 개체로 변환한 결과이다.

```
<DEPARTMENT>
  <ID rdf:datatype="&xsd:string">D111</ID>
  <NAME rdf:datatype="&xsd:string">the Department of Economics</NAME>
</DEPARTMENT>
```

2. 기존 매핑 규칙의 문제점

관계형 데이터베이스로부터 OWL 온톨로지를 생성하기 위한 기존 매핑 규칙들은 다음과 같은 문제점을 포함한다.

(ⅰ) 두 개 이상의 데이터베이스 테이블을 통합하여 하나의 클래스를 생성한다.

(ⅱ) 외래키(foreign key)로만 구성된 복합 기본키(composite primary key)를 갖는 테이블은 하나의 클래스로 정의하지 못한다.

(ⅲ) 프로퍼티의 정의역(domain)과 치역(range)이 서로 다른 테이블로부터 유도된 OWL 요소이다.

(ⅳ) 외래키로 구성된 복합 기본키에 종속적인 컬럼은 OWL 요소로 유도하지 못한다.

(ⅴ) 개체 생성 방법이 명확히 기술되어 있지 않거나 익명(anonymous) 개체를 생성한다.

문제점 (ⅰ)은 하나의 개체 추출이 여러 테이블에 분산되어 있는 데이터베이스 레코드 데이터를 요구하기 때문에 데이터베이스 조인 연산을 발생시키는 원인이 된다. 예를 들면, 하나의 PERSON 개체에 대응하는 레코드는 PERSON 테이블과 LECTURER 테이블의 조인 연산에 의해서 얻어진다.

문제점 (ⅱ)와 (ⅲ)은 오브젝트 프로퍼티의 정의역으로 정의된 클래스의 개체 추출은 그 오브젝트 프로퍼티의 치역으로 정의된 클래스의 개체 추출을 내포한다. 즉, 치역으로 정의된 클래스의 개체가 먼저 존재해야한다. 예를 들면, 하나의 ORDER 개체는 적어도 하나 이상의 PRODUCT 개체를 필요로 한다. 즉, 하나의 ORDER 개체 추출은 그림 5의 모든 테이블을 대상으로 하는 SQL 질의를 발생시킨다.

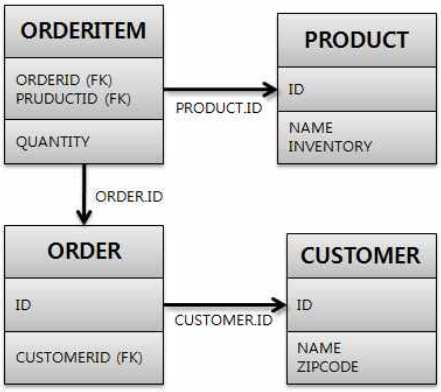

그림 7. Commerce 데이터베이스 스키마 다이어그램 2
Fig. 7. Commerce Database schema diagram 2

문제점 (ⅳ)는 특정 구조의 테이블을 OWL로 변환시키지 못하므로 데이터베이스를 온톨로지로 온전히 래핑할 수 없는 원인이 된다. 예를 들면, 문제점 (ⅳ)에서 기술하는 컬럼은 그림 7의 ORDERITEM 테이블의 QUANTITY 컬럼이다. 기존 규칙들은 그림 5의 ORDERITEM 테이블을 OWL로 매핑시킬 수 있으나 그림 7의 ORDERITEM 테이블은 QUANTITY 컬럼 때문에 OWL로 변환시킬 수 없다.

문제점 (ⅴ)는 익명 개체의 특성에 근거한다. 즉, 익명 개체는 동일한 네임스페이스에서만 유일성을 보장 받을 수 있다는 한계를 가지고 있으며 또한 HasKey 공리(axiom)에 근거한 동일 개체 식별 여부를 판단하기위한 추론을 제공받지 못한다[12].

3. 새로운 매핑 규칙을 위한 요구 사항

기존 매핑 규칙들의 문제점들을 해결하기 위한 규칙은 다음과 같은 요구 사항을 만족시켜야 한다.

(1) 하나의 테이블은 예외 없이 하나의 클래스로 정의할 수 있어야 한다.

(2) 프로퍼티의 정의역과 치역이 동일한 테이블에서 유도된 OWL 요소들로 정의되어야 한다.
(3) 명명된(named) 개체를 생성할 수 있어야 한다.
(4) 명명된 개체의 생성은 데이터베이스 조인 연산을 요구하지 않아야 한다.

III. 본론

이 장은 본 논문에서 제안하는 관계형 데이터베이스로부터 OWL 온톨로지를 생성하는 규칙을 기술한다. 3.1절은 이 규칙의 기본 아이디어를 개략적으로 설명하며 이후의 각각의 절은 개별적인 OWL요소를 생성하기 위한 규칙이다.

1. 본 논문에서 제안하는 규칙의 기본 아이디어

이 절은 본 논문이 제안하는 규칙에 대한 직관적인 이해를 위하여 이 규칙의 기본적인 아이디어를 설명함과 동시에 이 규칙이 기존 매핑 규칙들의 문제점을 개선하기위한 요구 사항을 만족시킨다는 것을 보여준다.

표 2는 본 논문이 제안하는 규칙이 두 모델 사이의 구성요소를 매핑시키는 관계를 보여준다. 표 2의 키(key) 컬럼은 PRIMARY KEY, FOREIGN KEY, UNIQUE 제약 조건에 있는 컬럼을 의미한다. 그림 8은 그림 2의 College 데이터베이스 스키마를 본 논문의 규칙을 적용하여 생성한 OWL 온톨로지 스키마에 대한 RDF 그래프이며 그림 9는 그림 7의 Commerce 데이터베이스 스키마를 변형한 것이다.

표 2. 본 논문의 RDB와 OWL 사이의 매핑
Table 2. mapping between RDB and OWL in this paper

Relational Database		OWL
database schema	a table	a named class
	a key column	a named class, an object property, a data property, an XSD data type
	a non-key column	a data property, an XSD data type
database instance	a row	a named individual
	a key column	a named individual, a literal
	a non-key column	a literal

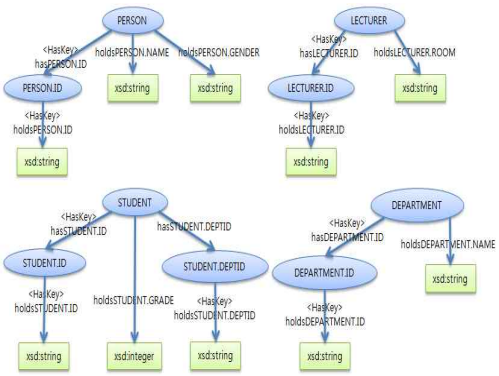

그림 8. 그림 2에 대한 RDF 스키마 그래프
Fig. 8. RDF schema graphs for Fig. 2

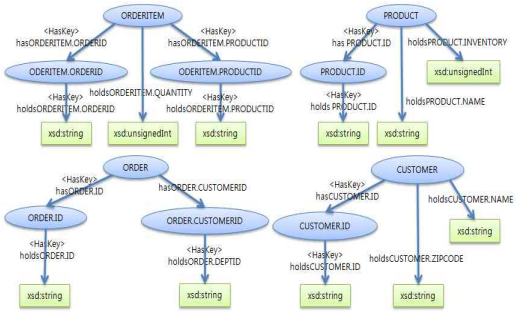

그림 9. 그림 7에 대한 RDF 스키마 그래프
Fig. 9. RDF schema graph for Fig. 7

본 논문의 규칙은 기존 규칙들과 달리 여러 개의 테이블을 통합하여 하나의 클래스를 생성시키거나 테이블로부터 클래스가 아닌 다른 OWL 요소를 생성시키는 예외 규칙 없이 하나의 테이블로부터 그 테이블의 이름과 동일한 하나의 클래스를 생성시킨다. 따라서 본 논문의 규칙은 요구 사항 (1)을 만족시킨다.

본 논문의 규칙은 하나의 컬럼이 PRIMARY KEY 혹은 FOREIGEN KEY 혹은 UNIQUE 제약 조건을 가지고 있다면, 하나의 오브젝트 프로퍼티와 하나의 데이터 프로퍼티를 생성하며, 그 조건을 만족시키지 않는 컬럼은 하나의 데이터 프로퍼티를 생성한다. 본 논문의 규칙에 의해서 생성된 프로퍼티의 정의역과 치역은 그 프로퍼티를 유도한 컬럼이 소속된 테이블로부터 유도된 클래스 혹은 데이터타입이다. 기존 규칙들의 RDF 그래프는 복수개의 테이블이 하나의 RDF 그래프로 표현되는 반면 본 논문의 규칙은 하나의 테이블이 하나의 RDF 그래프로 표현된다. 따라서 본 논문의 규칙은 요구 사항 (2)를 만족시킨다.

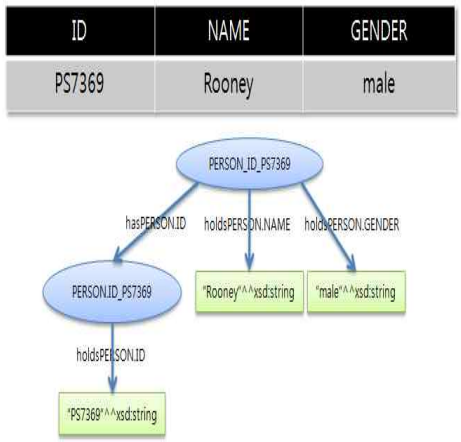

그림 10. 테이블 레코드 예와 RDF 그래프
Fig. 10. An example table record and a RDF graph

그림 10은 College 데이터베이스에서 테이블 STUDENT의 하나의 행으로부터 생성된 하나의 RDF 그래프이다. 개체 STUDENT_ID_PS7369는 하나의 행으로부터 생성된 클래스 STUDENT 타입의 명명된 개체이며 개체 STUDENT.ID_PS7369는 컬럼 STUDENT.ID로부터 생성된 클래스 STUDENT.ID 타입의 명명된 개체이며 개체 STUDENT.DEPTID_D505는 컬럼 STUDENT.DEPTID로부터 생성된 클래스 STUDENT.DEPTID 타입의 명명된 개체이다.

본 논문의 규칙은 데이터베이스 인스턴스의 하나의 행과 PRIMARY KEY 혹은 FOREIGEN KEY 혹은 UNIQUE 제약 조건에 있는 하나의 컬럼의 데이터 값으로부터 하나의 명명된 개체를 생성한다. 본 논문의 규칙은 익명 개체를 생성하지 않는다. 따라서 본 논문의 규칙은 요구 사항 (3)을 만족시킨다.

기존 연구들의 규칙에서 STUDENT 테이블의 하나의 행으로부터 STUDENT 타입의 OWL 개체 생성은 하나의 PERSON 개체와 하나의 DEPARTMENT 개체의 생성을 내포한다. 즉, 세 테이블에 대한 SQL 질의가 발생한다. 반면에 본 논문의 규칙은 그림 10의 RDF 그래프를 얻기 위하여 STUDENT 테이블만을 필요로 한다. 즉, 본 논문의 규칙은 조인 연산을 요구하지 않는다. 따라서 본 논문의 규칙은 요구 사항 (4)를 만족시킨다.

2. 본 논문에서 제안하는 매핑 규칙

다음에 정의된 함수들은 본 논문의 매핑 규칙을 기술하기 위하여 사용된다.

(정의) $PK(T_i)$는 관계형 데이터베이스의 임의의 하나의 테이블 T_i의 PRIMARY KEY 제약 조건에 있는 컬럼들의 집합이다. $FK(T_i)$는 T_i의 FOREIGN KEY 제약 조건에 있는 컬럼들의 집합이다. $UK(T_i)$는 T_i의 UNIQUE 제약 조건에 있는 컬럼들의 집합이다. $SUK(T_i)$는 T_i의 단일 컬럼 UNIQUE 제약 조건에 있는 컬럼들의 집합이다. $MUK(T_i)$는 T_i의 다중 컬럼 UNIQUE 제약 조건에 있는 컬럼들의 집합이다. $UK(T_i)$는 $SUK(T_i) \cup MUK(T_i)$이다. $KEY(T_i)$는 $PK(T_i) \cup FK(T_i) \cup UK(T_i)$이다. $CHK(T_i)$는 T_i의 CHECK 제약 조건에 있는 컬럼들의 집합이다. $CHKIN(T_i)$는 T_i의 열거형의 CHECK 제약 조건에 있는 컬럼들의 집합이고 $CHKIN(T_i) \subset CHK(T_i)$이다. $REF(T_i.C_k)$는 T_i의 임의의 하나의 컬럼 C_k가 참조하는 컬럼이다. $NN(T_i)$는 테이블 T_i의 NOT NULL 제약 조건에 있는 컬럼들의 집합이다.

본 논문에서는 각각의 매핑 규칙으로부터 유도되는 OWL 구문을 OWL Functional Syntax를 사용하여 표현한다.

2.1 Class 엔티티 선언을 위한 규칙

OWL Class 엔티티 선언을 위한 규칙은 다음과 같다.

(규칙 1) 테이블 T_i는 명명된 클래스 cls_{T_i}를 생성한다. 이를 위한 OWL 구문은 다음과 같다.

· Declaration(Class(:cls_{T_i}))

(규칙 2) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$라면, 컬럼 C_k는 명명된 클래스 $cls_{T_iC_k}$를 생성한다. 이를 위한 OWL 구문은 다음과 같다.

· Declaration(Class(:$cls_{T_iC_k}$))

2.2 ObjectProperty 엔티티 선언을 위한 규칙

OWL ObjectProperty 엔티티 선언을 위한 규칙은 다음과 같다.

(규칙 3) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$라면, 컬럼 C_k는 오브젝트 프로퍼티 $op_{T_iC_k}$를 생성한다. 이를 위한 OWL 구문은 다음과 같다.

· Declaration(ObjectProperty(:$op_{T_iC_k}$))

2.3 DataProperty 엔티티 선언을 위한 규칙

OWL DataProperty 엔티티 선언을 위한 규칙은 다음과 같다.

(규칙 4) 테이블 T_i의 컬럼 C_k는 데이터 프로퍼티$dp_{T_iC_k}$를 생성한다. 이를 위한 OWL 구문은 다음과 같다.

· Declaration(DataProperty(:$dp_{T_iC_k}$))

2.4 오브젝트 프로퍼티의 정의역과 치역의 정의를 위한 규칙

OWL 오브젝트 프로퍼티의 정의역과 치역 정의를 위한 규칙은 다음과 같다.

(규칙 5) 오브젝트 프로퍼티 $op_{T_iC_k}$는 클래스 cls_{T_i}를 자신의 정의역으로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· ObjectPropertyDomain(:$op_{T_iC_k}$:cls_{T_i})

(규칙 6) 오브젝트 프로퍼티 $op_{T_iC_k}$는 클래스 $cls_{T_iC_k}$를 자신의 치역으로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· ObjectPropertyRange(:$op_{T_iC_k}$:$cls_{T_iC_k}$)

2.5 데이터 프로퍼티의 정의역과 치역의 정의를 위한 규칙

OWL 데이터 프로퍼티의 정의역 정의를 위한 규칙은 다음과 같다.

(규칙 7) $C_k \in KEY(T_i)$인 테이블 T_i의 컬럼 C_k로부터 유도된 데이터 프로퍼티 $dp_{T_iC_k}$는 클래스 $cls_{T_iC_k}$를 자신의 정의역으로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· DataPropertyDomain(:$dp_{T_iC_k}$:$cls_{T_iC_k}$)

(규칙 8) $C_k \notin KEY(T_i)$인 테이블 T_i의 컬럼 C_k로부터 유도된 데이터 프로퍼티 $dp_{T_iC_k}$는 클래스 cls_{T_i}를 자신의 정의역으로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· DataPropertyDomain(:$dp_{T_iC_k}$:cls_{T_i})

OWL 데이터 프로퍼티의 치역 정의를 위한 규칙은 다음과 같다.

(규칙 9) 테이블 T_i의 컬럼 C_k가 $C_k \notin CHK(T_i)$라면, 데이터 프로퍼티 $dp_{T_iC_k}$는 C_k의 데이터 타입에 대응하는 OWL 2 datatype map[12]에 있는 데이터 타입 dt_{T_iC}를 자신의 치역 $dr_{T_iC_k}$로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· DataPropertyRange(:$dp_{T_iC_k}$ dt_{T_iC})

예를 들면, $dp_{T_iC_k}$의 치역 $dr_{T_iC_k}$가 문자열 집합이라면, 이를 위한 OWL 구문은 다음과 같다.

· DataPropertyRange(:$dp_{T_iC_k}$ xsd:string)

표 3은 OWL 2 datatype map이다.

표 3. OWL 2 데이터타입 맵
Table 3. The OWL 2 datatype map

Real Numbers, Decimal Numbers and Integers	owl:real, owl:rational,xsd:decimal, xsd:integer, xsd:nonNegativeInteger, xsd:nonPositiveInteger, xsd:positiveInteger, xsd:negativeInteger, xsd:long, xsd:int, xsd:short, xsd:byte, xsd:unsignedLong, xsd:unsignedInt, xsd:unsignedShort, xsd:unsignedByte
Floating-Point Numbers	xsd:double, xsd:float
Strings	xsd:string, xsd:normalizedString, xsd:token, xsd:language, xsd:Name, xsd:NCName, xsd:NMTOKEN
Boolean Values	xsd:boolean
Binary Data	xsd:hexBinary, xsd:base64Binary
IRIs	xsd:anyURI
Time Instants	xsd:dateTime, xsd:dateTimeStamp
XML Literals	rdf:XMLLiteral

(규칙 10) 테이블 T_i의 컬럼 C_k가 $C_k \in CHKIN(T_i)$라면, 데이터 프로퍼티 $dp_{T_iC_k}$는 C_k를 위한 열거형의 CHECK 제약 조건에 대응하는 DataOneOf 데이터 치역을 자신의 치역 $dr_{T_iC_k}$로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· DataPropertyRange(:$dp_{T_iC_k}$ DataOneOf(lt_1 ... lt_n)) 여기에서, lt_i는 $1 \le i \le n$인 리터럴이다.

예를 들면, $dp_{T_iC_k}$의 치역 $dr_{T_iC_k}$가 문자열 'male'과 'female'을 원소로 하는 집합이라면, 이를 위한 OWL 구문은 다음과 같다.

DataPropertyRange(:$dp_{T_iC_k}$
DataOneOf("male"^^xsd:string "female"^^xsd:string))

(규칙 11) 테이블 T_i의 컬럼 C_k가 $C_k \notin CHKIN(T_i) \wedge C_k \in CHK(T_i)$라면, 데이터 프로퍼티 $dp_{T_iC_k}$는 C_k를 위한 CHECK 제약 조건에 대응하는DatatypeRestriction 데이터 치역을 자신의 치역 $dr_{T_iC_k}$로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· DataPropertyRange(:$dp_{T_iC_k}$ DatatypeRestriction(dt_{T_iC} f_1 lt_1 ... f_n lt_n)) 여기에서, f_i는 constrai ning facet[12], lt_i는 $1 \le i \le n$인 리터럴이다.

예를 들면, $dp_{T_i C_k}$의 치역 $dr_{T_i C_k}$가 1, 2, 3, 4의 정수로 한정된 집합이라면, 이를 위한 OWL 구문은 다음과 같다.

DataPropertyRange(:$dp_{T_i C_k}$
DatatypeRestriction(xsd:integer xsd:minInclusive "1"^^xsd:integer xsd:maxExclusive "5"^^xsd:integer))

2.6 HasKey 공리를 위한 규칙

OWL HasKey 공리를 위한 규칙은 다음과 같다.

(규칙 12) 클래스 cls_{T_i}는 $C_k \in PK(T_i)$인 T_i의 모든 컬럼 C_k들로부터 유도된 오브젝트 프로퍼티 $op_{T_i C_k}$들을 자신의 키(key)로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· HasKey(:cls_{T_i} (:$op_{T_i C_1}$ ... :$op_{T_i C_m}$) ()) 여기에서, $PK(T_i) = \{C_1, ..., C_m\}$이다.

(규칙 13) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$라면, 클래스 $cls_{T_i C_k}$는 데이터 프로퍼티 $dp_{T_i C_k}$를 자신의 키(key)로 정의한다. 이를 위한 OWL 구문은 다음과 같다.

· HasKey(:$cls_{T_i C_k}$ () (:$dp_{T_i C_k}$))

2.7 InverseFunctionalObjectProperty 공리를 위한 규칙

OWL InverseFunctionalObjectProperty 공리를 위한 규칙은 다음과 같다.

(규칙 14) 테이블 T_i의 컬럼 C_k가 $C_k \in SUK(T_i)$라면, 컬럼 C_k로부터 유도된 오브젝트 프로퍼티 $op_{T_i C_k}$는 역함수적 오브젝트 프로퍼티로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· InverseFunctionalObjectProperty(:$op_{T_i C_k}$)

2.8 SubClassOf 공리를 위한 규칙

OWL의 SubClassOf 공리를 위한 규칙은 다음과 같다.

(규칙 15) 테이블 T_i의 컬럼 C_k가 $C_k \in FK(T_i)$라면, 클래스 $cls_{T_i C_k}$는 C_k가 참조하는 컬럼 $REF(T_i.C_k)$로부터 유도된 클래스 $cls_{REF(T_i C_k)}$의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:$cls_{T_i C_k}$:$cls_{REF(T_i C_k)}$)

(규칙 16) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$라면, 클래스 $cls_{T_i C_k}$는 하나의 클래스 표현 DataAllValuesFrom(:$dp_{T_i C_k}$ $dr_{T_i C_k}$)의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:$cls_{T_i C_k}$ DataAllValuesFrom(:$dp_{T_i C_k}$ $dr_{T_i C_k}$))

(규칙 17) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$라면, 클래스 $cls_{T_i C_k}$는 하나의 클래스 표현 DataExactCardinality(1 :$dp_{T_i C_k}$ $dr_{T_i C_k}$)의 서브클래스로 정의된다.

· SubClassOf(:$cls_{T_i C_k}$ DataExactCardinality(1 :$dp_{T_i C_k}$ $dr_{T_i C_k}$))

(규칙 18) $i \neq j$인 테이블 T_i, T_j에 대해서, $REF: PK(T_i) \sim PK(T_j)$이면, 즉, 함수 $REF: PK(T_i) \rightarrow PK(T_j)$가 일대일 대응이면, T_i로부터 유도된 클래스 cls_{T_i}는 테이블 T_j로부터 유도된 클래스 cls_{T_j}의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} :cls_{T_j})

(규칙 19) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$라면, 클래스 cls_{T_i}는 하나의 클래스 표현 ObjectAllValuesFrom(:$op_{T_i C_k}$:$cls_{T_i C_k}$)의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} ObjectAllValuesFrom(:$op_{T_i C_k}$:$cls_{T_i C_k}$))

(규칙 20) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i) \wedge C_k \in NN(T_i)$라면, 클래스 cls_{T_i}는 하나의 클래스 표현 ObjectExactCardinality(1 :$op_{T_i C_k}$:$cls_{T_i C_k}$)의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} ObjectExactCardinality(1 :$op_{T_i C_k}$:$cls_{T_i C_k}$))

(규칙 21) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i) \wedge C_k \notin NN(T_i)$라면, 클래스 cls_{T_i}는 하나의 클래스 표현 ObjectMaxCardinality(1 :$op_{T_i C_k}$:$cls_{T_i C_k}$)의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} ObjectMaxCardinality(1 :$op_{T_i C_k}$:$cls_{T_i C_k}$))

(규칙 22) 테이블 T_i의 컬럼 C_k가 $C_k \notin KEY(T_i)$라면, 클래스 cls_{T_i}는 하나의 클래스 표현 DataAllValuesFro m(:$dp_{T_i C_k}$ $dr_{T_i C_k}$)의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} DataAllValuesFrom(:$dp_{T_i C_k}$ $dr_{T_i C_k}$))

(규칙 23) 테이블 T_i의 컬럼 C_k가 $C_k \notin KEY(T_i) \wedge C_k \in NN(T_i)$라면, 클래스 cls_{T_i}는 하나의 클래스 표현 DataExactCardinality(1 :$dp_{T_i C_k}$ $dr_{T_i C_k}$)의 서브클래스로 정의된다. 이를 위한

OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} DataExactCardinality(1 :$dp_{T_i.C_k}$ $dr_{T_i.C_k}$))

(규칙 24) 테이블 T_i의 컬럼 C_k가 $C_k \notin KEY(T_i) \wedge C_k \notin NN(T_i)$라면, 클래스 cls_{T_i}는 하나의 클래스 표현 DataMaxCardin ality(1 :$dp_{T_i.C_k}$ $dr_{T_i.C_k}$)의 서브클래스로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubClassOf(:cls_{T_i} DataMaxCardinality(1 :$dp_{T_i.C_k}$ $dr_{T_i.C_k}$))

2.9 SubObjectPropertyOf 공리 정의를 위한 규칙

OWL SubObjectPropertyOf 공리를 정의하기 위한 규칙은 다음과 같다.

(규칙 25) $i \neq j$인 테이블 T_i, T_j가 규칙 18의 조건을 만족시키고, 테이블 T_i의 컬럼 C_k가 $C_k \in PK(T_i) \wedge C_k \in FK(T_i)$이면, 테이블 T_i의 컬럼 C_k로부터 유도된 오브젝트 프로퍼티 $op_{T_i.C_k}$는 테이블 T_j의 컬럼 $REF(T_i.C_k)$로부터 유도된 오브젝트 프로퍼티 $op_{REF(T_i.C_k)}$의 서브 프로퍼티로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubObjectPropertyOf(:$op_{T_i.C_k}$:$op_{REF(T_i.C_k)}$)

2.10 SubDataPropertyOf 공리 정의를 위한 규칙

OWL SubDataPropertyOf 공리를 정의하기 위한 규칙은 다음과 같다.

(규칙 26) 테이블 T_i의 컬럼 C_k가 $C_k \in FK(T_i)$라면, 컬럼 C_k로부터 유도된 데이터 프로퍼티 $dp_{T_i.C_k}$는 컬럼 C_k가 참조하는 컬럼 $REF(T_i.C_k)$로부터 유도된 데이터 프로퍼티 $dp_{REF(T_i.C_k)}$의 서브 프로퍼티로 정의된다. 이를 위한 OWL 구문은 다음과 같다.

· SubDataPropertyOf(:$dp_{T_i.C_k}$:$dp_{REF(T_i.C_k)}$)

2.11 DisjointClasses 공리 정의를 위한 규칙

OWL DisjointClasses 공리를 정의하기 위한 규칙은 다음과 같다.

(규칙 27) $1 \leq l \leq n$인 클래스 cls_l는 규칙 1과 규칙 2로부터 유도된 임의의 명명된 클래스라고 하자. cls_l가 규칙 2에 의해서 테이블 T_i의 컬럼 C_k로부터 유도된 클래스라면 C_k는 $C_k \in KEY(T_i) \wedge C_k \notin FK(T_i)$이고 cls_l가 규칙 1에 의해서 테이블 T_i로부터 유도된 클래스라면 T_i는 $i \neq j$인 테이블 T_j에 대해서 $REF: PK(T_i) \sim PK(T_j)$인 테이블 T_j가 존재하지 않는다. 이를 위한 OWL 구문은 다음과 같다.

· DisjointClasses(:cls_1 ... :cls_n)

2.12 Named Individual 생성을 위한 규칙

OWL 명명된 개체 생성을 위한 규칙은 다음과 같다.

(규칙 28) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$이면,

(ⅰ) T_i의 튜플 t_i의 C_k에 대한 null 값이 아닌 데이터 값(data value) $t_i[C_k]$는 명명된 개체 $a_{t_i[C_k]}$를 유도한다.
(ⅱ) $a_{t_i[C_k]}$는 자신의 타입을 클래스 $cls_{T_i.C_k}$로 정의한다.
(ⅲ) $a_{t_i[C_k]}$는 데이터 프로퍼티 $dp_{T_i.C_k}$에 의해서 $t_i[C_k]$에 대응하는 리터럴 $t_i[C_k]'$과 연결되도록 정의한다. 이를 위한 OWL 구문은 다음과 같다.

(ⅰ) Declaration(NamedIndividual(:$a_{t_i[C_k]}$))
(ⅱ) ClassAssertion(:$cls_{T_i.C_k}$:$a_{t_i[C_k]}$)
(ⅲ) DataPropertyAssertion(:$dp_{T_i.C_k}$:$a_{t_i[C_k]}$ $t_i[C_k]'$)

(규칙 29) (ⅰ) 테이블 T_i의 튜플 t_i는 명명된 개체 a_{t_i}를 유도한다.

(ⅱ) a_{t_i}는 자신의 타입을 클래스 cls_{T_i}로 정의한다.
(ⅲ) 테이블 T_i의 컬럼 C_k가 $C_k \in KEY(T_i)$이면, a_{t_i}는 오브젝트 프로퍼티 $op_{T_i.C_k}$에 의해서 개체 $a_{t_i[C_k]}$와 연결되도록 정의한다.
(ⅳ) 테이블 T_i의 컬럼 C_k가 $C_k \notin KEY(T_i)$이면, a_{t_i}는 데이터 프로퍼티 $dp_{T_i.C_k}$에 의해서 $t_i[C_k]$에 대응하는 리터럴 $t_i[C_k]'$과 연결되도록 정의한다. 이를 위한 OWL 구문은 다음과 같다.

(ⅰ) Declaration(NamedIndividual(:a_{t_i}))
(ⅱ) ClassAssertion(:cls_{T_i} :a_{t_i})
(ⅲ) ObjectPropertyAssertion(:$op_{T_i.C_k}$:a_{t_i} :$a_{t_i[C_k]}$)
(ⅳ) DataPropertyAssertion(:$dp_{T_i.C_k}$:a_{t_i} $t_i[C_k]'$)

2.13 OWL 엔티티 이름

본 논문의 규칙을 사용해 생성된 OWL 엔티티들의 이름은 다음을 따른다.

(ⅰ) 테이블 T_i로부터 규칙 1에 의해서 유도된 클래스 cls_{T_i}의 이름은 'T_i'이다.
(ⅱ) 테이블 T_i의 컬럼 C_k로부터 규칙 2에 의해서 유도된 클래스 $cls_{T_i.C_k}$의 이름은 '$T_i.C_k$'이다.
(ⅲ) 테이블 T_i의 컬럼 C_k로부터 규칙 3에 의해서 유도된 오브젝트 프로퍼티 $op_{T_i.C_k}$의 이름은 'has$T_i.C_k$'이다.
(ⅳ) 테이블 T_i의 컬럼 C_k로부터 규칙 4에 의해서 유도된

데이터 프로퍼티 $dp_{T_i C_k}$의 이름은 'holds $T_i.C_k$'이다.

(v) 테이블 T_i의 튜플 t_i의 컬럼 C_k의 데이터 값 $t_i[C_k]$로부터 규칙 28에 의해서 유도된 개체 $a_{t_i[C_k]}$의 이름은 '$T_i.C_k$_$t_i[C_k]$'이다.

(vi) $PK(T_i) = \{C_1, \ldots, C_m\}$인 테이블 T_i의 튜플 t_i로부터 규칙 29에 의해서 유도된 개체 a_{t_i}의 이름은 '$T_{i\,k=1}\{\_C_k\_t_i[C_k]\}_{k=m}$'이다.

예를 들면, (i) 그림 7의 테이블 ORDERITEM로부터 유도된 클래스의 이름은 ORDERITEM이다.

(ii) 테이블 ORDERITEM의 컬럼 ORDERID와 컬럼 PRODUCTID는 PRIMARY KEY 컬럼들이므로 이 컬럼들로부터 유도된 클래스의 이름은 각각 ORDER ITEM.ORDERID와 ORDERITEM.PRODUCTID이다.

(iii) 테이블 ORDERITEM의 컬럼 ORDERID와 컬럼 PRODUCTID는 PRIMARY KEY 컬럼들이므로 이 컬럼들로부터 유도된 오브젝트 프로퍼티의 이름은 각각 hasORDERITEM.ORDERID와 hasOR DERITEM.PRODUCTID이다.

(iv) 테이블 ORDERITEM의 컬럼 ORDERID, PRODUCTID, QUANTITY로부터 유도된 데이터 프로퍼티의 이름은 각각 holdsORDERITEM.ORDERI D, holdsO DE RITEM.PR ODUCTID, holdsORDERITE M.Q UANTITY이다.

(v) 테이블 ORDERITEM의 컬럼 ORDERID와 컬럼 PRODUCTID는 PRIMARY KEY 컬럼들이므로 그림 11과 같이 컬럼 ORDERID의 데이터 값이 'OD111'이고 컬럼 PRODUCTID의 데이터 값이 'PD111'라면 이 데이터 값들로부터 유도된 개체의 이름은 각각 ORDERITEM.ORDERID_OD111과 ORDERITEM. PRODUCTID_PD111이다.

(vi) 테이블 ORDERITEM의 컬럼 ORDERID와 컬럼 PRODUCTID는 PRIMARY KEY 컬럼들이므로 그림 11과 같이 컬럼 ORDERID의 데이터 값이 'OD111'이고 컬럼 PRODUCTID의 데이터 값이 'PD111'인 하나의 튜플로부터 유도된 개체의 이름은ORDERITEM_ORDERID_OD111_PRODU CTID_PD111이다.

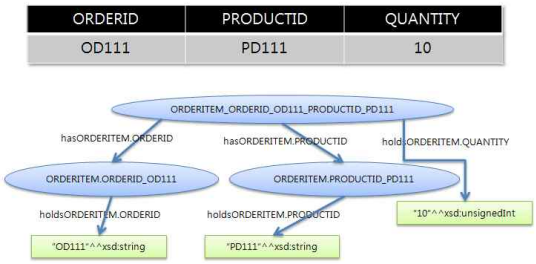

그림 11. 그림 7의 테이블 ORDERITEM의 튜플과 대응하는 RDF 그래프

Fig. 11. a tuple of table ORDERITEM in Fig. 7 and the corresponding RDF graph

3. 온톨로지의 추론

이 절은 본 논문의 규칙을 적용하여 생성한 온톨로지로부터 추론을 통해서 얻어지는 결과를 설명한다.

그림 12는 그림 2의 College 데이터베이스의 PERSON, STUDENT 그리고 DEPARTMENT 테이블에 대한 샘플 데이터이다.

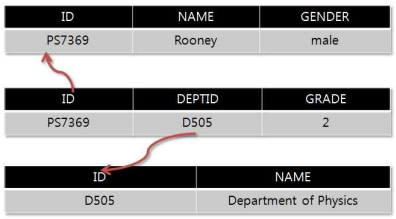

그림 12. 테이블 레코드 예

Fig. 12. Sample table records

그림 13의 STUDENT_ID_PS7369 개체와 DEPARTMENT_ID_D505 개체는 본 논문의 규칙에 의해서 그림 12의 STUDENT와 DEPARTMENT 테이블 데이터로부터 유도된다.

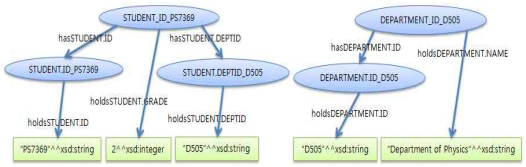

그림 13. 그림 12의 STUDENT와 DEPARTMENT에 대한 RDF 그래프

Fig. 13. RDF Graphs for STUDENT and DEPARTMENT in Fig. 12.

이 두 개체는 추론에 의해서 다음의 assertion들이 추가된다.
ObjectPropertyAssertion(:hasSTUDENT.DEPTID
:STUDENT_ID_PS7369 :DEPARTMENT.ID_D505)
ObjectPropertyAssertion(:hasDEPARTMENT.ID
:DEPARTMENT_ID_D505 :STUDENT.DEPTID_D505)

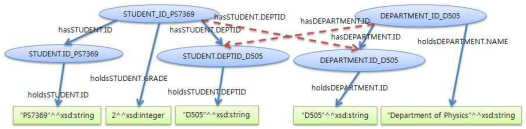

그림 14. 그림 13에 대한 추론 후의 RDF 그래프
Fig. 14. a RDF Graph after reasoning for Fig. 13

그림 14의 점선은 추론된(inferred) assertion들에 대응한다. 이러한 assertion들이 추가되는 이유는 개체 STUDENT.DEPTID_D505와 개체 DEPARTMENT.ID_D505가 동일한 개체로 추론되기 때문이다. 그 근거는 그림 15와 같이 STUDENT.DEPTID 클래스가 DEPARTMENT.ID 클래스의 서브 클래스이며, holdsSTUDENT.DEPTID 데이터 프로퍼티가 holdsDEPARTMENT.ID 데이터 프로퍼티의 서브 프로퍼티이며, holdsSTUDENT.DEPTID 데이터 프로퍼티가 STUDENT.DEPTID 클래스의 key 프로퍼티이며, holdsDEPARTMENT.ID 데이터 프로퍼티가 DEPARTMENT.ID 클래스의 key 프로퍼티일 때, STUDENT.DEPTID 타입의 하나의 명명된 개체가 holdsSTUDENT.DEPTID 데이터 프로퍼티를 통해서 갖는 데이터 값이 DEPARTMENT.ID 타입의 하나의 명명된 개체가 holdsDEPARTMENT.ID 데이터 프로퍼티를 통해서 갖는 데이터 값과 동일하면, 추론기는 이 두 명명된 개체를 동일한 개체로 식별하기 때문이다.

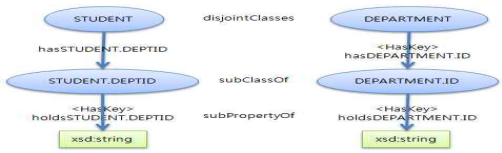

그림 15. STUDENT.DEPTID와 DEPARTMENT.ID의 관계
Fig. 15. Relationship between STUDEN.DEPTID and DEPARTMENT.ID

따라서 본 논문의 규칙에 의해 생성된 온톨로지는 테이블 사이의 참조 관계를 규칙 2에 의해서 생성된 클래스들에 대한 subClassOf, subPropertyOf 그리고 HasKey 공리를 통해서 대응시킨다.

그림 16의 PERSON_ID_PS7369 개체와 STUDENT_ID_PS7369 개체는 본 논문의 규칙에 의해서 그림 12의 PERSON와 STUDENT 테이블 데이터로부터 유도된다.

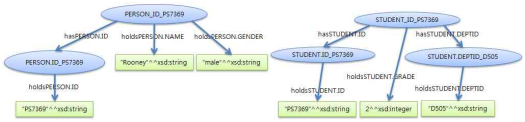

그림 16. 그림 12의 PERSON과 STUDENT에 대한 RDF 그래프
Fig. 16. RDF Graphs for PERSON and STUDENT in Fig. 12.

이 두 개체는 추론에 의해서 다음의 assertion들이 추가된다.
ObjectPropertyAssertion(:hasPERSON.ID
:PERSON_ID_PS7369 :STUDENT.ID_PS7369)
ObjectPropertyAssertion(:hasSTUDENT.ID
:PERSON_ID_PS7369 :STUDENT.ID_PS7369)
ObjectPropertyAssertion(:hasSTUDENT.DEPTID
:PERSON_ID_PS7369 :STUDENT.DEPTID_D505)
ObjectPropertyAssertion(:hasPERSON.ID
:STUDENT_ID_PS7369 :PERSON.ID_PS7369)
ObjectPropertyAssertion(:hasSTUDENT.ID
:STUDENT_ID_PS7369 :PERSON.ID_PS7369)

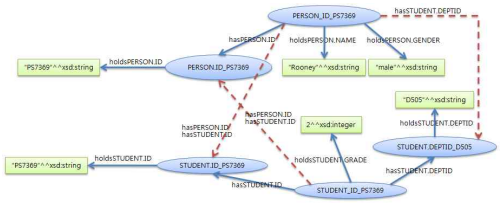

그림 17. 그림 16에 대한 추론 후의 RDF 그래프
Fig. 17. a RDF Graph after reasoning for Fig. 16

그림 17의 점선은 추론된(inferred) assertion들에 대응한다. 이러한 assertion들이 추가되는 이유는 개체 PERSON _ID_PS7369와 개체 STUDENT_ID_PS7369가 동일한 개체로 추론되기 때문이다. 그 근거는 그림 18과 같이 STUDENT 클래스가 PERSON 클래스의 서브 클래스이며, hasSTUDENT.ID 오브젝트 프로퍼티가 hasPERSON.ID 오브젝트 프로퍼티의 서브 프로퍼티이며, hasSTUDENT.ID 오브젝트 프로퍼티가 STUDENT 클래스의 key 프로퍼티이며, hasPERSON.ID 오브젝트 프로퍼티가 PERSON 클래스의 key 프로퍼티일 때, STUDENT 타입의 하나의 명명된 개체가 hasSTUDENT.ID 오브젝트 프로퍼티를 통해서 갖는 하나의 STUDENT.ID 타입의 명명된 개체가 PERSON 타입의 하나의 명명된 개체가 hasPERSON.ID 오브젝트 프로퍼티를 통해서 갖는 하나의 PERSON.ID 타입의 명명된 개체와 동일하면, 추론기는 이 두 명명된 개체를

동일한 개체로 식별하기 때문이다.

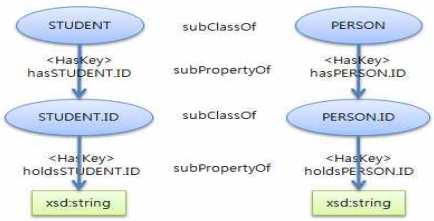

그림 18. STUDENT와 PERSON의 관계
Fig. 18. Relationship between STUDENT and PERSON

따라서 본 논문의 규칙에 의해 생성된 온톨로지는 테이블 사이의 상속을 규칙 1에 의해서 생성된 클래스들에 대한 subClassOf, subPropertyOf 그리고 HasKey 공리를 통해서 대응시킨다.

본 논문에서 제안하는 규칙은 이 규칙을 적용하여 생성한 온톨로지가 무결성 제약 조건을 포함한 관계형 데이터베이스 스키마의 시맨틱과 데이터베이스 인스턴스의 데이터를 모두 포함할 수 있도록 설계되었다. 또한 기존 연구들과 비교하여 개체 추출이 복수개의 테이블을 요구하지 않음에도 불구하고 추론을 통하여 테이블 사이의 참조와 상속을 표현할 수 있다.

IV. 결론

본 논문은 기존 관계형 데이터베이스를 데이터베이스 스키마 변형과 데이터 마이그레이션과 같은 별도의 과정을 거치지 않고도 시맨틱 웹 환경에서 OWL 2 DL 웹 온톨로지로 사용할 수 있는 매핑 규칙을 제안하였다. 이 규칙은 특정 구조의 테이블이 존재하는 데이터베이스로부터는 OWL 온톨로지를 생성할 수 없거나 개체 추출을 위하여 높은 비용의 데이터베이스 조인 연산이 불필요하게 수반되는 기존 매핑 규칙들의 문제점을 개선하였다. 본 연구를 기반으로 하는 향후 연구는 본 논문에서 제안한 규칙을 적용하여 생성한 관계형 데이터베이스 기반의 웹 온톨로지를 시맨틱 웹 환경에서 사용하기 위한 매핑 툴을 개발하고자 한다.

참고문헌

[1] T. Berners-Lee, "Weaving the Web," HarperBusiness, 2000.

[2] C. Blakeley, "RDF Views of SQL Data (Declarative SQL Schema to RDF Mapping)," OpenLink Software, 2007.

[3] C. Bizer, and R. Cyganiak, "D2RQ – Lessons Learned," Position paper for the W3C Workshop on RDF Access to Relational Databases, Cambridge, USA, 2526 October 2007.

[4] S. Auer, S. Dietzold, J. Lehmann, S. Hellmann, and D. Aumueller, "Triplify - Light-weight linked data publication from relational databases," Proceedings of the 18th International Conference on World Wide Web (WWW) 2009.

[5] G. Būmans, and K. Čerāns, "RDB2OWL: a Practical Approach for Transforming RDB Data into RDF/OWL," Proceedings of the 6th International Conference on Semantic Systems, Graz, Austria, 2010.

[6] N. Cullot, R. Ghawi, and K. Yétongno, "DB2OWL: A Tool for Automatic Database-to-Ontology Mapping," In Proceedings of the 15th Italian Symposium on Advanced Database Systems (SEBD 2007), Torre Canne di Fasano (BR), Italy, pp. 491-494, June 2007.

[7] M. Li, X. Du, and S. Wang, "Learning Ontology from Relational Database," Proceedings of the 4th International Conference on Machine Learning and Cybernetics. Vol. 6 pp. 3410-3415, 2005.

[8] S. Sane, and A. Shirke, "Generating OWL Ontologies from Relational Databases for the Semantic Web," International Conference on Advances in Computing, Communications and Control, pp. 157-162, 2009.

[9] Z. Xu, S. Zhang, and Y. Dong, "Mapping between relational database schema and OWL ontology for deep annotation," Proc. of IEEE/WIC/ACM Int. Conf. on Web Intelligence, , pp. 548-552, Hong Kong, China, 2006.

[10] S. Zhou, G. Meng, H. Ling, H. Zhang, "Tool for Translating Relational Databases Schema into Ontology for Semantic Web," 2010 Second International Workshop on Education Technology and Computer Science, etcs, vol. 1, pp. 198-201, 2010.

[11] P.V.Biron, and A. Malhotra, "XML Schema Part 2: Datatypes Second Edition," W3C Recommendation 28 October 2004, http://www.w3.org/TR/xmlschema-2/

[12] B. Motik, P.F.Patel-Schneider, and B. Parsia, "OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax," W3C Recommendation 27 October 2009, http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/

저 자 소 개

최 지 웅

2001 : 숭실대학교 컴퓨터학부 학사
2003 : 숭실대학교 컴퓨터학과 석사
2007 - 2008 : 고등기술연구원 연구원
2003 - 현재 : 숭실대학교 컴퓨터학과 박사과정
관심분야 : 시맨틱 웹, BI, 보안
Email : iamjwchoi@gmail.com

김 명 호

1989 : 숭실대학교 컴퓨터학부 학사
1991 : 포항공과대학교 전자계산학과 공학석사
1995 : 포항공과대학교 전자계산학과 공학박사
1995 : 한국전자통신연구소 선임연구원
1998, 2006 : 미국 테네시주립대 교환교수
1995-현재 : 숭실대학교 컴퓨터학부 교수
관심분야 : 분산/병렬 컴퓨팅, 그리드, 웹서비스, BI, 보안
Email : kmh@su.ac.kr