경영정보학연구
제12권 제2호
2002년 6월

VRQL : 시각 관계형 데이터베이스 질의어*

이 석 균*

VRQL : A Visual Relational Database Query Language

Lee, Suk Kyoon

In this paper, we propose a visual relational database query language, VRQL, by modifying and extending the recently proposed VOQL*. Like VOQL*, VRQL, based on ven Diagram and graph, naturally reflects the structure of schemas in queries and has recursive formal semantics. However, VRQL has relationally complete expressiveness, while VOQL* is only a conjunctive query language.

In the logical definition part of VRQL, which is the relational version of VOQL*, most features of VOQL* are retained, and the semantics of queries are based on the tuple relational calculus. In the procedural definition part of VRQL, by introducing the concept of VRQL view and set operations, the expressiveness of VRQL is increased to the level equivalent to that of the relational algebra. Due to the introduction of VRQL views, existing queries or temporary queries used in the process of creating queries can be represented with views, so that complex queries may be represented more conveniently. Set operations, used with VRQL views, enable us to represent various queries, beyond the expressiveness of conjunctive query languages.

* 이 연구는 2000년도 단국대학교 대학 연구비의 지원으로 연구되었음.
** 단국대학교 정보컴퓨터학부 컴퓨터과학전공 부교수 Email : sklee@dankook.ac.kr

Ⅰ. 서 론

정보 기술의 급속한 발전과 사회적 요구에 따라 데이터베이스 분야는 최근 많은 발전을 했으며 대부분의 정보 시스템에 데이터베이스가 사용되고 있다. 그러나 데이터베이스를 위한 사용자 인터페이스는 최근 윈도우 기반의 사용 환경에도 불구하고 대개 텍스트 기반의 질의어로 구성되어있거나 사용자 인터페이스를 시각화하였어도 아직 초보적인 수준에 있어 사용에 불편함이 제기되고 있다. 특히 윈도우 기반의 소프트웨어의 발전에 따른 사용 환경의 변화는 텍스트 기반의 질의어보다는 윈도우 환경에 적합한 시각 질의어의 개발을 요구하고 있다.

데이터베이스의 시각 질의어들은 관계형 모델 또는 E-R 데이터 모델에 기반을 두고 있다[Angelaccio, 1990; Cruz, 1992; Czeju, 1987; Mohan, 1993; Vadaparty, 1993; Sockut, 1993; Whang, 1992; Kim, 1999]. 이들 대부분의 시각 질의어들은 Zloof[1977]의 QBE를 확장하거나 그래프에 기초하고 있다. QBE를 확장한 대부분의 시각 질의어의 문제는 단순한 조건의 표현에는 별 문제가 없으나 여러 릴레이션 간의 복잡한 조건의 경우 텍스트에 의존하여 표현함으로 시각 질의의 장점이 훼손되며 각 연산자의 적용범위(scope) 표현의 어려움이 있다. 한편, 그래프 기반의 시각 질의어들은 조건의 표현을 릴레이션과 상수와의 관계 또는 릴레이션과 릴레이션의 관계로 표현하고 있어 투플과 상수 또는 투플과 투플 사이의 조건의 의미를 제대로 전달하지 못하는 문제가 있다. QBE와 그래프 기반의 결합 방식에 기초한 접근 방법도 이러한 문제를 해결하고 있지 못하다.

최근 복잡한 구조의 데이터의 질의를 위해 객체 지향 데이터베이스를 위한 시각 질의어들이 제안되고 있다[Carey, 1996; Chavda, 1997; Gutig, 1994; Kim, 1999; Kim, 2000; Mohan, 1993; 이석균, 2000; 이석균, 2001]. 확장된 QBE에 기초한 객체 지향 질의어는 Mohan[1993]의 VQL, Carey [1996]의 PESTO를 들 수 있다. VQL은 QBE와 같은 템플레이트 기반의 언어로 VQL에서 경로식의 표현은 복수의 템플레이트들과 이들 사이를 연결하는 레이블과 텍스트 변수를 통해 이루어진다. 이를 좀더 발전시킨 것이 PESTO인데, 이는 복합 객체 관계를 복수의 폼과 이를 연결한 선분(edge)을 사용하여 표현한다. PESTO에서 선분으로 연결된 폼 사이의 관계는 단지 클래스간의 관계만을 반영하고 있어 복합 객체들 간의 접근 경로를 의미하는 경로식[Kifer, 1992]의 의미를 제대로 전달하지 못한다[Kim, 2000].

그래프에 기초한 객체지향 질의어로는 VQL, QUIVER[Chavda, 1997], VOQL[Kim, 1999; Kim, 2000], VOQL*[이석균, 2000; 이석균, 2001; Lee, 2001]등이 있는데, VQL은 그래프 기반의 질의어로 부정, 전체정량자, 재귀적 질의문, 스키마 질의 등이 가능하나, 경로식 및 조인 연산 등에 텍스트 변수들이 과다하게 사용되고 있다. 이에 반해 QUIVER는 텍스트를 거의 사용하지 않으며 비교적 편리한 사용자 인터페이스를 제공하나, 형식 의미(formal semantics)가 정의되고 있지 않고 논리곱 언어(conjunctive query language)라는 한계가 있다.

VOQL과 VOQL*는 Harel[1988]의 Higraph에 기초한 시각 질의어로 OOPC(Object-Oriented Predicate Calculus)[Kifer, 1992]에 기초한 형식 의미를 갖는다. VOQL*는 VOQL을 개선한 언어로 시각변수의 개념을 도입하고 경로식을 객체 레벨에서 시각화함으로써 경로식의 표현이 명확하며, 벤다이어그램에 입각한 표현 방식으로 집합 관련 조건 연산의 표현이 자연스럽고 간결하다. VOQL*의 장점은 질의어의 구조가 그 형식 의미를 정의하는 OOPC의 귀납적인 문법 구조를 그대로 반영하므로 질의문의 의미가 명확하다는 점이다. 그러나, VOQL*는 전체정량자, OR, NOT의 연산을 포함하고 있지 못하는 논리곱 언어이어서 질의의 표현력에 한계가 있다.

대부분의 시각 질의어들은 기본적으로 논리곱 언어인데, 이는 2차원 공간에 나열된 조건들을 AND로 연결된 조건으로 해석하는 것이 자연스럽기 때문이다. 물론 일부 언어들은 OR나 NOT을 허용하고 있지만 그 표현 방식은 대개 텍스트 기반의 표현에 의존하고 있으며 전체정량자, OR와 AND, NOT의 적용 범위(scope)를 시각화시키기가 쉽지 않고 시각화한 경우 이들 연산자들이 다중 중첩되었을 때 전체 질의의 표현이 복잡하고 어색할 뿐 아니라 시각 질의어로서의 직관성이 떨어지는 문제가 있다.

본 논문에서는 위의 문제를 해결하고자 하는 시도로 시각질의어에 뷰의 개념과 집합 연산의 도입을 통해 논리곱 언어의 한계를 극복하고자 한다. 이를 위해 VOQL*을 수정 및 확장한 VRQL (Visual Relational Query Language)을 소개한다. VRQL은 VOQL*로부터 객체지향 데이터 모델의 특징인 경로식과 집합관련 비교 연산에 관한 시각적 표현과 이들과 관련된 시맨틱 정의부분이 생략된 관계형 시각 질의어이다. VRQL은 논리적 정의 부분과 절차적 정의 부분으로 구성되는데, 논리적 정의 부분은 VOQL*의 관계형 버전으로 질의문의 의미는 투플 관계해석에 의해 정의되며 논리곱 관계해석과 동등한 표현력을 갖는다. VRQL의 절차적 정의부분에서는 VRQL의 뷰와 집합연산을 도입하여 VRQL의 표현력을 관계적으로 완전(relationally complete)한 수준으로 향상시킨다. VRQL 뷰는 질의 추상화의 표현을 위해, 그리고 질의 연산 결과들 사이에 집합 연산 표현의 편의를 위해 사용된다. 집합 연산은 합집합, 교집합과 차집합으로 통해 VRQL을 논리곱 질의 언어의 한계 이상으로 VRQL의 표현력을 향상시킨다. 본 논문에서는 이렇게 증가된 VRQL의 표현력이 관계대수과 동등함을 증명한다. 본 논문의 내용은 언어의 표현력의 비교 및 논리의 전개의 편리성 때문에 관계형 데이터베이스 질의어를 전제로 설명하였으나 객체 지향 데이터베이스 질의어 즉 VOQL*에도 쉽게 확장 적용할 수 있다.

본 논문의 구성은 다음과 같다. 2장은 VRQL의 기본 정의(논리적 정의)로, 시각적 구성 요소들과 항(term), 논리식(formula)의 개념, 그리고 이들에 기초한 VRQL 질의문의 구조를 설명한다. VRQL 질의문의 의미를 정의하고 VRQL이 논리곱 관계해석의 표현력과 같음을 보인다. 3장은 VRQL의 확장 정의(절차적 정의)로, 시각적 구성 요소로서 뷰의 개념과 집합 연산의 정의를 설명하며 이들의 추가에 따른 VRQL의 표현력의 변화에 대해 설명한다. 4장에서는 결론과 앞으로의 연구 과제에 대해 소개한다.

Ⅱ. VRQL의 기본 정의 - 논리적 정의

VOQL*는 객체 지향 데이터베이스를 위한 시각 질의어로 그래프와 벤 다이어그램에 기초한 문법과 귀납적으로 정의되는 형식 의미를 갖는다[이석균, 2000; Lee, 2001]. VRQL은 이러한 VOQL*을 관계형 데이터베이스 질의어로 수정 및 확장된 시각 질의어로 선언적(논리적) 정의 부분과 절차적 정의 부분으로 구성된다. 2절에서는 VRQL의 선언적 정의 부분을 소개한다.

2.1 VRQL의 기본 문법 구조 및 질의 예제

VRQL의 문법은 벤다이어그램과 그래프에 기초한 시각적 구성 요소와 텍스트에 기초한 텍스트 구성 요소로 정의된다. 시각적 구성 요소에는 블랍(blob), 시각 변수(visual variable), 시각 요소(visual element), 선분(edge)과 스텀프 블랍(stump blob)이 있으며 이들은 질의의 구조를 시각화하는데 사용된다. 텍스트 구성 요소는 블랍, 선분, 스텀프 블랍과 같은 시각적 구성 요소의 의미를 전달하는 레이블과 조건식의 표현에 필요한 비교 연산자(>, >=, =, <, <=)와 상수 값 등으로 구성된다. 시각적 구성 요소들의 표기의 예는 <그림 2-1>에 주어져 있고 그 의미는 다음과 같다.

블 랍: 블랍은 사각형으로 표시되며 릴레이션에 속한 모든 투플들의 집합을 나타내는 시각적 표현이다.

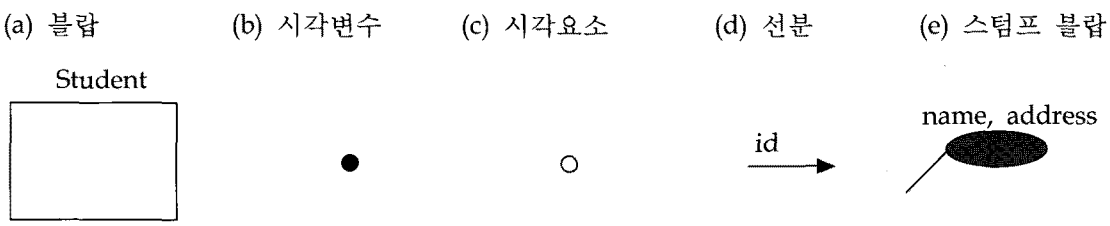
시각변수: 검은 색의 작은 원으로 표시되며 관계 해석의 투플 변수에 대한 시각적 표기이다. 항상 블랍과 함께 사용되며 블랍이 의미하는 릴레이션에 대한 투플 변수의 바인딩을 표현할 때 사용된다.

시각요소: 투명한 작은 원으로 표시되며 투플의 속성 값을 시각화한 것으로 리터럴을 의미한다.[1)]

선 분: 선분은 방향성이 있는 화살표로 시각변수와 시각요소를 연결할 때 사용되며 투플의 속성을 시각적으로 표현한 것이다. 투플을 시각 변수로 나타낼 때 이 투플의 속성은 시각변수로부터의 선분으로 나타낸다. 선분의 위쪽에 레이블은 속성 이름을 의미한다.

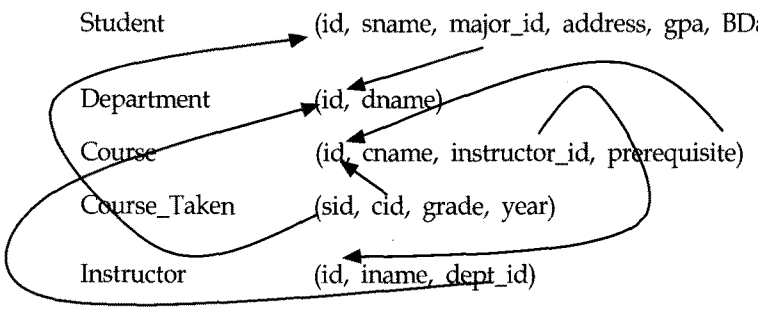
스텀프 블랍: 스텀프 블랍은 시각변수로부터 무방향 선분으로 연결된 타원형으로 표현되며 프로젝트될 속성들의 리스트를 나타낼 때 사용한다. 프로젝트될 속성들의 이름들은 레이블 리스트로 표현된다

시각적 구성 요소들의 예는 <그림 2-1>과 같이 표현된다. <그림 2-1>의 (a), (d), (e)의 레이블들은 각각 릴레이션 이름, 속성 이름과 프로젝트될 속성 이름 리스트를 의미한다.

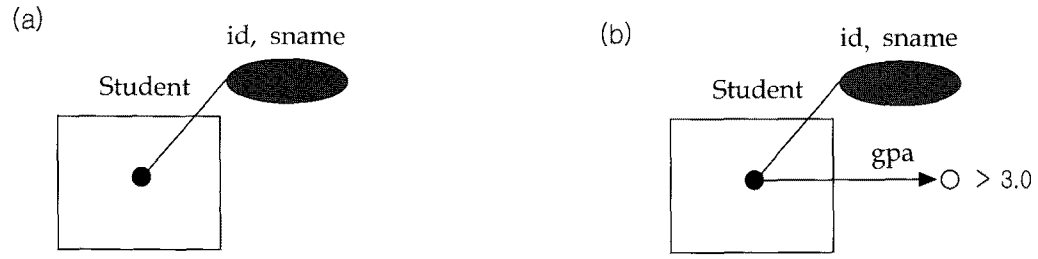


<그림 2-1> 시각적 구성 요소[1)]

다음의 릴레이션 스키마는 본 논문에서 VRQL 질의문의 예제들을 표현할 때 사용한다.



1) VOQL*에서 시각변수의 블랍에의 바인딩은 시각변수를 블랍의 옆 변에 위치시켰으나 VRQL에서는 시각변수를 블랍의 내부의 중심부분에 위치시킨다.



<그림 2-2> VRQL 질의문의 예

위의 스키마에서 기본 키는 밑줄로 외래키는 이탤릭으로 표시하며 외래 키가 나타내는 대상은 화살표로 표시한다. <그림 2-2> (a)는 모든 학생의 학번과 이름을 반환하는 질의문이고, <그림 2-2> (b)는 평점이 3.0이상인 학생의 학번과 이름을 반환하는 VRQL 질의문이다.

<그림 2-2> (a)의 VRQL 질의문에서 시각변수는 블랍 Student 내부에 위치하여 시각변수와 블랍과의 바인딩을 표현하고 있으며 시각변수에 연결된 스텀프 블랍은 프로젝트될 속성들을 나타내고 있다. 이를 관계해석의 질의문으로 표현하면 다음과 같다.

RC 1 {x.id, x.sname | x ∈ Student}

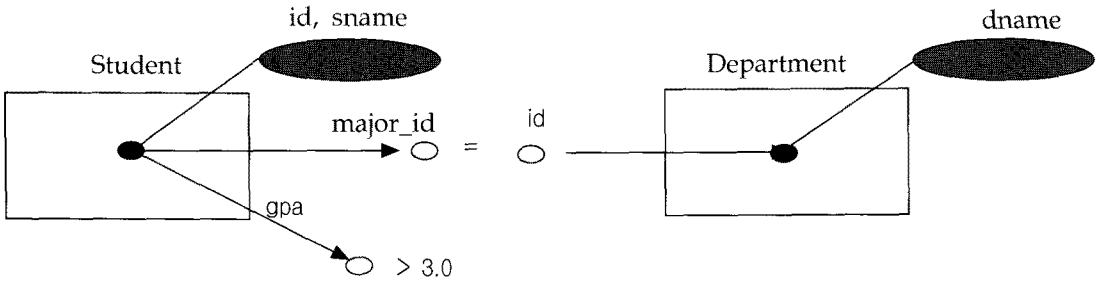
<그림 2-2> (b)는 (a)에 비해 평점에 관한 조건이 추가되어있는데 이는 시각변수로부터 gpa로 레이블되어있는 선분을 통해 시각요소에 연결하여 Student 투플의 gpa속성을 나타낸다. 이때 시각요소는 Student 투플의 gpa 속성을 시각화한 것으로 조건 표현은 시각요소와 상수와의 비교를 통해 나타나있다. <그림 2-2> (b)의 질의문의 의미는 다음의 관계해석의 질의문을 통해 정의된다.

RC 2 {x.id, x.sname | x ∈ Student ∧ x.gpa > 3.0}

RC2의 투플변수 x는 시각변수에 대응하고 x.gpa는 시각변수로부터 선분으로 연결된 시각요소에, 그리고 프로젝션 리스트 x.id, x.sname은 스텀프 블랍과 레이블 리스트에 대응함을 알 수 있다. <그림 2-3>는 평점이 3.0이상인 학생의 학번(id), 이름(sname), 그리고 소속 전공이름(dname)을 반환하는 VRQL 질의문이다.

<그림 2-3>의 질의문은 두 릴레이션에 대한 조인을 표현한 것으로 이에 대한 관계해석의 질의문은 다음과 같다.

RC 3 {x1.id, x1.sname, x2.dname | x1 ∈ Student ∧ x2 ∈ Department ∧ x1.gpa > 3.0 ∧x1.major_id = x2.id}



<그림 2-3> VRQL 질의문의 예

VRQL 질의문에서 두 개 이상의 시각 변수가 사용되는 경우, 이들은 관계해석의 질의문에서 서로 다른 투플 변수로 번역된다.

2.2 VRQL의 형식 정의 : 문법 및 의미

본 절에서는 VRQL의 문법과 의미에 대한 형식적인 정의를 제공하려고 한다. 앞의 설명에서 알 수 있듯이 VRQL의 질의문은 관계해석의 질의문의 논리적 구조와 매우 흡사하다. 이는 VRQL의 문법적 구조가 로직의 구조를 반영하고 있기 때문이다. 즉 VRQL에서 질의문은 벤다이어그램과 그래프 기반의 시각적 표기를 통해 표현되지만 그 문법적 구조는 항(term)과 논리식(formula)과 같은 로직의 문법적인 구조에 기초하고 있다.

2.2.1 VRQL의 항과 의미 정의

VRQL 항(term)은 단순 항(simple term)과 구조적 항(structured term)으로 구분된다. 단순 항은 블랍, 시각요소, 시각변수, 숫자, 텍스트 상수로 구성되며 구조적 항은 시각변수, 시각요소 그리고 레이블된 선분의 결합에 의해 정의된다. 구조적 항은 깊이가 1이하인 트리로 루트는 시각변수, 리프 노드들은 시각요소들로 구성되며 루트와 리프노드들은 레이블된 선분으로 연결된다. <그림 2-2> (a)는 두 개의 단순 항 즉 블랍과 시각변수와 1개의 구조적 항으로, <그림 2-2> (b)는 세 개의 단순 항 즉 블랍, 시각변수, 시각요소와 1개의 구조적 항으로 구성된다. <그림 2-3>의 VRQL 질의문은 8개의 단순 항들과 두 개의 구조적 항들로 구성된다.

다음에서는 VRQL 항에 대한 의미를 정의한다. 관계해석의 질의문의 예에서 볼 수 있었듯이 VRQL 항들은 관계해석의 항들로 번역된다. 번역 알고리즘은 다음과 같다.

VRQL 항의 의미 정의 (번역 알고리즘1)

(a) 블랍은 관계해석의 질의문의 릴레이션으로 번역되며 이때 릴레이션 이름은 블랍의 레이블이다.

(b) 시각변수들은 관계해석의 질의문의 투플 변수들로 번역되며 이때 번역된 이름이 중복되지 않도록 한다.

(c) 시각요소는 선분으로 연결된 시각변수가 투플 변수 x로 번역되고 선분의 레이블 attr은 x.attr로 번역된다.

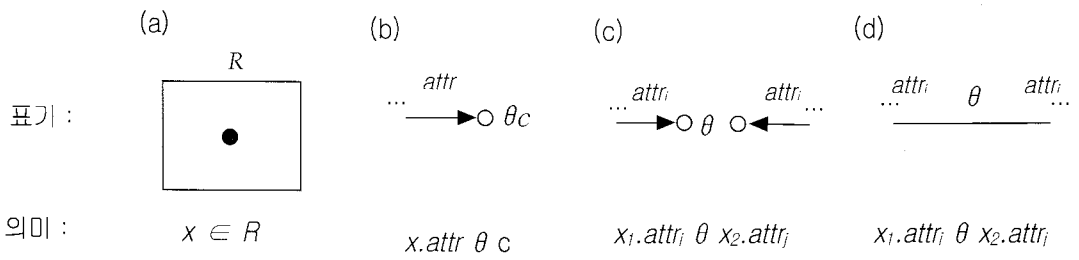
(d) 숫자 및 텍스트 상수는 관계해석의 질의문에서 그대로 숫자와 텍스트 상수로 번역된다.

예를 들어 <그림 2-3>의 VRQL 질의문에서 VRQL 항들을 관계해석의 질의문의 항들로 번역한다면 왼쪽 블랍부터 시작하면 다음과 같다 : Student, x1, x1.major_id, x1.gpa, 3.0, Department, x2, x2.id.

2.2.2 VRQL의 논리식과 의미 정의

VRQL의 논리식은 VRQL의 구조적인 항의 존재로 인해 로직의 논리식과 같은 텍스트 문법적 구조로 정의되지는 않는다. 따라서 본 절에서는 VRQL의 기본 논리식(atomic formula)을 정의하고 이에 기초하여 어떻게 정형 논리식(well-formed formula)이 구성되는지에 관한 규칙을 제공한다.

VRQL의 기본 논리식의 정의는 <그림 2-4>에 주어져있다. 이들 기본 논리식들은 VRQL 항들 사이의 조건을 의미한다. <그림 2-4> (a)는 시각변수가 블랍 R에 바인딩됨을 의미하고 <그림 (b)-(d)>는 항들간의 비교를 나타내는 조건들이다. (b)는 투플의 속성 값과 상수와의 비교이고 (c)는 투플들 간의 속성 값의 비교이다. (d)는 (c)의 단순화된 표기로 동등 비교의 경우에는 비교연산 θ를 생략할 수 있다.



<그림 2-4> 기본 논리식의 표기 및 의미>

기본 논리식들은 VRQL 질의문을 구성하는 기본 조건들이다. 이들 기본 논리식들은 구조적 항들의 시각요소들에 대해 적용되어 전체적으로 복잡한 논리식(조건)를 표현할 때 사용된다. 이와 같이 VRQL 질의문의 구조적 항들에 대해 기본 논리식이 적용되어 생성된 논리식을 정형 논리식(well-formed formula)라고 하는데 정형 논리식은 다음과 같은 조건을 만족해야 한다. 앞으로 의미가 혼동되지 않는 한, VRQL 정형 논리식을 간단히 VRQL 논리식로 표현한다.

VRQL 논리식의 생성 조건

1. 모든 시각변수는 <그림 2-4> (a)와 같이 블랍에 바인딩되어야하며, 시각요소는 반드시 어느 한 시각변수를 루트로 하는 구조적 항의 리프이어야 한다.
2. 모든 시각요소와 상수들은 <그림 2-4>의 (b)-(d) 중의 한 기본 논리식에 참여하여야 한다.

지금까지 VRQL 논리식의 문법에 대해 설명하였다. 위의 조건을 만족하는 VRQL 논리식은 관계해석의 질의문의 논리식로 번역될 수 있다. VRQL 논리식의 의미는 다음과 같이 정의된다.

VRQL 논리식의 의미 정의 (번역 알고리즘2)

1. VRQL 논리식에 속한 모든 기본 논리식들을 기본 논리식의 의미에 따라 로직(또는 관계해석)의 논리식들로 번역한다. 이때 릴레이션 이름, 투플변수 이름과 상수들은 번역 알고리즘1의 결과를 사용한다.
2. 번역된 논리식들의 리스트로부터 투플변수와 릴레이션의 바인딩을 나타내는 모든 논리식들을 리스트의 앞 부분으로 옮긴다.
3. 논리식들을 ∧(AND)로 연결하고 투플변수들에 대해 존재정량자를 논리식의 앞에 추가한다.

위의 번역 알고리즘에 의해 <그림 2-3>의 질의문을 번역하면 다음의 논리식이 된다.

∃x1 ∃x2 [x1 ∈ Student ∧ x2 ∈ Department ∧ x1.gpa > 3.0 ∧ x1.major_id = x2.id]

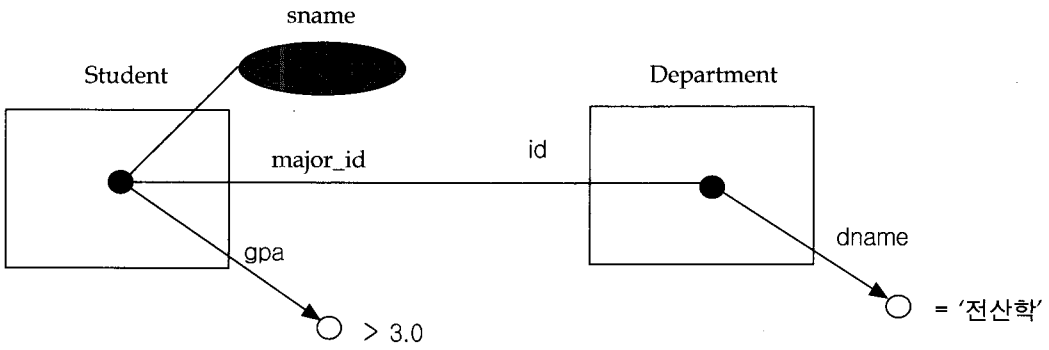
2.2.3 VRQL 질의문과 의미

VRQL 질의문은 VRQL 논리식에 스텀프 블랍을 추가한 것인데, 스텀프 블랍을 통해 프로젝션될 속성들을 지정한다. 다음에서는 VRQL 질의문의 의미를 정의한다.

VRQL 질의문의 의미 정의 (번역 알고리즘 3)

VRQL 질의문 expr에 번역 <알고리즘 1>과 2를 적용하여 생성된 논리식을 f라고 할 때,

1. 논리식 f 에 대해 다음과 같이 수정한다 : expr 에서 스텀프 블랍이 연결되어 있는 각 시각변수에 대해 이(시각변수)에 대응하는



<그림 2-5> VRQL 질의문의 예

논리식 f 의 투플변수 xi에 대해 존재정량자를 제거하고 xi ∈ Rk를 논리식의 앞으로 이동하여 나머지 부분과 ∧로 연결한다.

2. 스텀프 블랍이 연결되어있는 시각변수에 대응하는 투플 변수를 xj라 하고 이 스텀프 블랍의 레이블 aj1, ···ajn을 xj.aj1, ···xj.ajn의 속성 리스트로 변환하여 관계해석의 질의문의 프로젝션 영역에 표현한다. 시각변수에 대응하는 투플 변수가 두 개 이상 있을 때는 변환된 속성 리스트를 통합(concatenation)한다.[4]

<그림 2-3>의 질의식에 대한 의미는 RC3을 참조하고 <그림 2-5>는 전산학 전공의 학생 들 중 평점이 3.0 이상인 학생의 이름을 추출하는 VRQL 질의문이다.

<그림 2-5>의 VRQL 질의문의 의미는 다음의 RC4에 주어져있다.

RC4 { x1.sname | x1 ∈ Student ∧ ∃x2 [x2 ∈ Department ∧ x1.gpa > 3.0 ∧ x1.major_id = x2.id ∧ x2.dname = '전산학'] }

2.2.4 VRQL의 논리적 정의와 표현력의 한계

본 절에서는 위에서 설명한 논리적 정의에 기초한 VRQL의 표현력의 범위에 대해 설명한다. VRQL의 논리적 정의에 의해 표현된 질의문을 논리적 VRQL 질의문이라 정의한다. 관계해석의 논리식의 정의에 OR, NOT 그리고 전체정량자을 배제하고 존재정량자와 AND 연산만을 허용하는 관계해석을 논리곱(conjunctive) 관계해석이라 정의하는데, 다음에서는 VRQL의 표현력과 논리곱 관계해석의 표현력이 동등함을 보이고 있다.

정리1) VRQL의 논리적 정의 부분의 표현력은 논리곱 관계 해석의 표현력과 동등하다.

(증명) 위의 정리는 다음의 양 방향의 내용이 참임을 보임으로 증명한다.

4) 두 개 이상의 시각변수가 사용될 경우, 프로젝션 영역에서의 순서는 스텀프 블랍의 생성된 순서에 의해 결정된다고 가정한다. VRQL의 구현 시 보다 나은 방법이 사용될 수 있다.

5) 뷰 생성자는 뷰의 생성 과정을 설명하기 위한 논리적 개념으로 구현 시에는 다른 직관적 표현으로 변경될 수 있다.

(=>) 임의의 논리적 VRQL 질의문은 논리곱 관계해석의 질의문으로 변환 가능하다.

이는 번역 <알고리즘 1, 2>와 3을 통해 증명된다. VRQL의 논리적 정의에 기초한 질의문은 번역 (알고리즘 1, 2)와 3에 의해 관계해석의 질의문으로 생성되기 때문이다.

(<=) 논리곱 관계해석의 임의의 질의문은 논리적 VRQL 질의문으로 변환 가능하다.

이를 위해서는 우선 논리곱 관계해석의 임의의 질의문을 번역 <알고리즘 3>의 결과물 형식으로 변환한다. 이때 변환된 관계해석의 질의문의 형식은 다음의 구조를 갖는다.

{<$x_1, \cdots, x_n$의 속성 리스트로 구성된 프로젝션 리스트> | $x_1 \in R_1 \wedge \cdots \wedge x_n \in R_n \wedge \exists x_{n+1} \cdots \exists x_{n+k}$ [$x_{n+1} \in R_{n+1} \wedge \cdots \wedge x_{n+k} \in R_{n+k} \wedge f(x_1, \cdots, x_n, x_{n+1}, \cdots, x_{n+k})$] }

위에서 $f(x_1, \cdots, x_n, x_{n+1}, \cdots, x_{n+k})$는 각 투플변수 $x_i(1 \le i \le n+k)$의 속성들에 대한 조건들을 AND로 연결한 논리식이다. 논리곱 관계해석의 임의의 질의문은 AND 연산자와 존재정량자의 특성에 의해 항상 위의 구조로 변환 가능하다 [Ullman, 1988].

다음에 위의 형식으로 표현된 관계해석의 질의문에 대한 VRQL의 논리식이 생성될 수 있음을 보인다. 모든 $x_i \in R_i(1 \le i \le n+k)$에 대해 R_i에 대한 블랍 B_i와 x_i에 대한 시각변수 v_i를 생성하고, $f(x_1, \cdots, x_n, x_{n+1}, \cdots, x_{n+k})$ 내의 모든 $x_i.attr_j$에 대해 v_i로부터 선분으로 연결된 시각요소 eij를 생성하고 선분에 $attr_j$으로 레이블을 붙인다. 그리고 $x_i.attr_j$에 관한 조건들을 <그림 2-4>의 VRQL 기본 논리식로 표현한다.

마지막으로 위에서 생성된 VRQL의 논리식에 스텀프 블랍의 생성을 보인다. <$x_1, \cdots, x_n$으로 구성된 프로젝션 리스트>에 표현된 각 $x_l(1 \le l \le n)$에 대해 프로젝트될 속성 리스트를 대응하는 시각변수에 스텀프 블랍을 추가하고 프로젝트될 속성 리스트로 언급된 속성 이름들을 레이블로 포함시킨다.

(증명 끝)

위에서 논리적 정의에 기초한 VRQL은 논리곱 관계해석과 표현력이 동등함을 보였다. 한편, 논리곱 관계 해석은 셀렉션, 프로젝션, 카티전 프로덕트 연산으로 구성된 관계 대수, 즉 SPC 관계대수와 표현력이 동등하다[Abiteboul, 1995; Ullman, 1988]. 따라서 다음의 정리는 도출된다.

정리2) VRQL의 논리적 정의 부분과 논리곱 관계해석, SPC 관계대수의 표현력은 서로 동등하다.

(증명) 정리1과 Abiteboul[1995]의 Theorem 4.4.8 (Equivalence Theorem)로부터 증명된다.

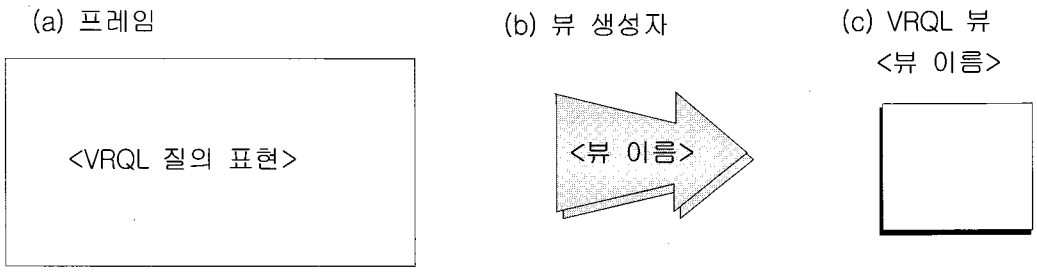
Ⅲ. VRQL의 확장된 정의 : 절차적 정의

본 절에서는 VRQL 뷰와 집합 연산을 도입에 따른 VRQL의 절차적 정의의 소개와 이에 따른 VRQL 질의문의 의미의 정의 및 VRQL의 표현력의 범위에 대해 설명한다.

3.1 VRQL에서의 시각 질의 추상화와 뷰

3.1.1 시각 질의 추상화의 필요성

관계형 데이터베이스의 장점들 중 SQL 질의문을 뷰로 정의하여 이를 릴레이션처럼 다른 SQL 질의문에 사용할 수 있다는 점이다. 그러나 대개의 시각질의어들의 경우, 이러한 관계형 모델의 특성이 시각질의어의 정의에 충분히 활용되고 있지 못했다. 본 절에서는 시각 질의의 결과를 추상화하여 다른 여러 시각 질의문들에 사



<그림 3-1> 뷰 생성 관련 연산자

용될 수 있음을 보이고자 한다. 이때 추상화된 시각 질의를 VRQL 뷰라고 정의한다.

VRQL과 같은 시각 질의어에서는 질의 표현에 있어 각 릴레이션의 관계와 조인 조건을 시각화시켜줌으로서 질의 전체의 논리적 구조를 이해하는데 도움을 준다. 그러나 시각 질의에서도 복잡한 질의문의 경우에는 적절한 추상화가 질의 작성 및 질의의 의미 파악에 도움이 된다. VRQL 뷰는 집합 연산의 표현에 편리하게 사용되는데 이는 3.2절에서 설명한다.

3.1.2 시각 질의 추상화로서 VRQL 뷰의 정의와 사용

시각질의 추상화는 사각형 형태의 프레임 안에 정의된 VRQL 질의문으로부터 뷰 생성자(질의 추상화 연산자라고도 함)에 의해 VRQL 뷰를 생성함으로 이루어진다. <그림 3-1>에 프레임, 뷰 생성자, VRQL 뷰의 시각적 표현이 주어져 있다. 프레임은 질의를 작성하는 창으로 질의 표현의 기본 단위이다. 뷰 생성자는 프레임에 표현된 질의를 뷰로 표현하는 연산자로 뷰의 이름을 입력받아 뷰를 생성한다[5]. 뷰의 형태는 블랍과는 달리 둥근 모서리의 사각형으로 표현한다. 프레임, 뷰 생성자, VRQL 뷰의 사용 예가 <그림 3-2>에 주어져있다.

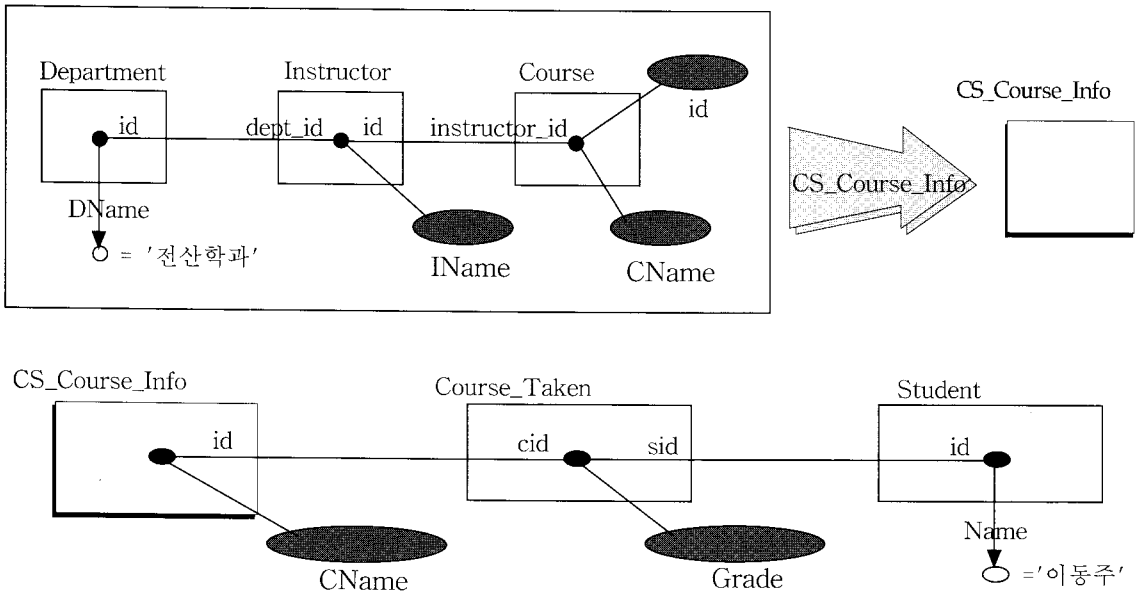
5) 뷰 생성자는 뷰의 생성 과정을 설명하기 위한 논리적 개념으로 구현 시에는 다른 직관적 표현으로 변경될 수 있다.

<그림 3-2>에는 전산학과에 소속된 교수의 이름과 그 교수가 담당한 강의 이름과 과목 번호를 구하는 질의문이 프레임안에 주어져있고 이로부터 뷰를 생성하는 예가 제시되어있다. 생성된 뷰의 이름이 뷰 생성자에 주어지고 이에 따라 생성된 뷰의 이름이 CS_Course_Info로 지정되어있다.

<그림 3-2>에서 생성된 VRQL 뷰의 스키마는 CS_Course_Info(IName, Id, CName)으로 스텀프 블랍들을 통해 정의되며 이는 다른 질의문 내에서 뷰의 사용 시 사용될 수 있다. <그림 3-3>에서는 <그림 3-2>에서 정의된 VRQL 뷰가 사용되는데 이는 이동주가 수강한 전산학 과목의 이름과 학점을 구하는 질의문이다.

<그림 3-3>의 조인조건(id = cid)은 <그림 2-4> (d) 형태의 즉 단순화된 조인이 사용되고 있으며 이를 <그림 2-4> (c)의 형태로 고려할 때 id와 cid에 해당하는 시각요소들이 사용된다. id에 해당되는 시각요소의 존재는 <그림 3-2>의 뷰의 정의식의 (id으로 레이블된) 스텀프 블랍에 기인한다. 한편, <그림 3-3>의 CName으로 레이블된 스텀프 블랍은 <그림 3-2>의 대응하는 스텀프 블랍에 기인한다.

이는 VRQL 뷰가 질의문의 조건 표현 시 사용되는 시각요소 또는 조인조건의 묵시적으로 표현된 시각요소는 VRQL 뷰의 정의 질의문의 스텀프 블랍의 속성 리스트에 속한 속성의 시각적 표현이어야 함을 의미한다. 마찬가지로 질의문의 프로젝션 표현 시 사용되는 VRQL 뷰의 스텀



프 블랍의 속성 리스트는 VRQL 뷰의 정의 질의문의 스텀프 블랍들의 속성 리스트들에 반드시 포함되어야 한다.

3.1.3 VRQL 뷰가 사용된 질의문의 의미

VRQL 뷰가 사용된 VRQL 질의문의 의미는 VRQL 뷰의 확장(expansion)을 통해 정의된다. 이는 SQL 뷰가 SQL 문의 FROM절에 사용될 때 그 SQL 문의 의미는 뷰 확장을 통해 SQL 문의 재 정의가 이루어지는 것처럼, VRQL 질의문에서 VRQL 뷰를 뷰의 정의 질의문으로 치환하여 VRQL 질의문을 재 정의하는 것을 의미한다. VRQL 뷰의 확장 시에는 확장 후 재 정의되는 질의문이 VRQL 문법에 맞아야한다. 다음에는 뷰를 확장할 때 사용되는 규칙을 소개한다.

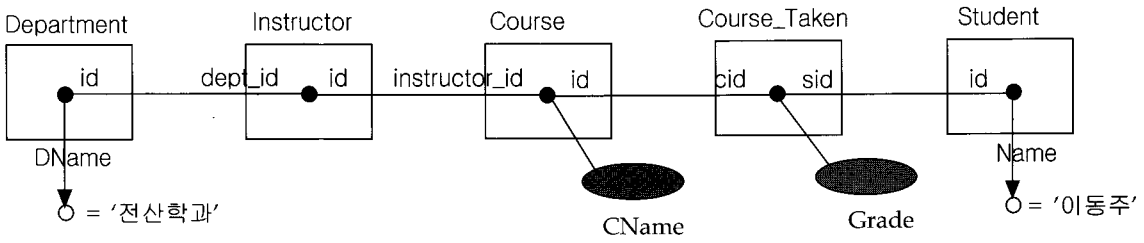
뷰 확장 규칙 (번역 알고리즘4)

뷰를 V, V가 사용된 질의문을 Q, V의 정의 VRQL 질의문을 VDQ, Q에서 V가 확장되었을 때 생성되는 VRQL 질의문을 NQ라 할 때, 다음에 규칙에 따라 뷰 확장이 이루어진다.

a) (뷰의 치환) Q에서 V와 V에 바인딩하는 시각변수를 각각 삭제하고 VDQ를 추가한다.

b) (조건 표현의 조정) Q의 조건 표현에 관련된 V의 속성(명시적 또는 묵시적 시각요소로 표현) attr가 있을 때, a)의 결과식의 VDQ 부분에서 속성 attr를 나타내는 스텀프 블랍(또는 스텀프 블랍의 레이블)을 삭제하고 스텀프 블랍이 연결된 시각변수로부터 (명시적 또는 묵시적 시각요소와 선분을 통해) 속성 attr 를 표시하여 본래 Q에서의 조건을 완성한다.

c) (프로젝션 표현의 조정) b)의 결과식의 VDQ의 남은 스텀프 블랍(또는 스텀프 블랍의 남은 속성 리스트)들 중에서, Q에서 V에 관한 스텀프 블랍들에 대응하는 것들을 제외하고는 다 삭제한다. VDQ의 삭제되지 않은 스텀프 블랍의 레이블된 속성들도 Q에서 V에 대응하는 스텀프 블랍의 속성들이 없은 경우는 모두 삭제한다.



<그림 3-4> 뷰가 확장된 VRQL 질의문

뷰 확장 규칙 b)는 질의문 내에 뷰의 정의 질의문을 치환할 때 본래 질의문의 조건의 의미를 그대로 유지하기 위해 질의문의 나머지 부분과 뷰의 정의 질의문과의 조건 표현을 다시 조정하는 것이고, 규칙 c)는 뷰의 정의 질의문의 프로젝션이 되는 속성들 중에서 불필요한 속성들을 삭제하는 것이다. <그림 3-4>는 뷰 확장 규칙을 통해 <그림 3-3>의 질의문의 뷰가 확장된 것이다.

<그림 3-4>의 질의문에서 블랍 Course의 스텀프 블랍 CName은 <그림 3-3>의 뷰에서 사용된 스텀프 블랍으로 <그림 3-2>의 질의식의 스텀프 블랍 CName이 살아남은 것이다. <그림 3-2>의 스텀프 블랍 IName은 뷰에서 사용되지 않음으로 뷰 확장과정에서 삭제되었다. 이는 뷰 확장 규칙 c)가 반영된 것이다. 한편, <그림 3-2>의 스텀프 블랍 id가 뷰 확장 과정에서 시각 요소 id로 변환되어 <그림 3-4>에서 조인 조건에 참여하고 있다. 이는 뷰 확장 규칙 b)가 사용된 것이다.

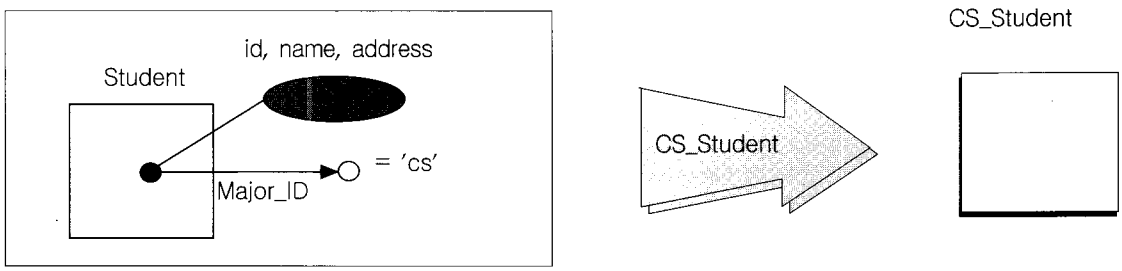
뷰가 사용된 VRQL 질의문의 의미는 우선 뷰를 뷰의 확장 규칙대로 확장한 VRQL 질의문의 의미로 정의되며, VRQL 뷰의 도입은 VRQL의 표현력을 증대시키지는 않음을 알 수 있다. <그림 3-3>의 VRQL 질의문의 의미는 다음과 같다.

{x3.CName, x4.Grade | x3 ∈ Course ∧ x4 ∈ Course_Taken ∧

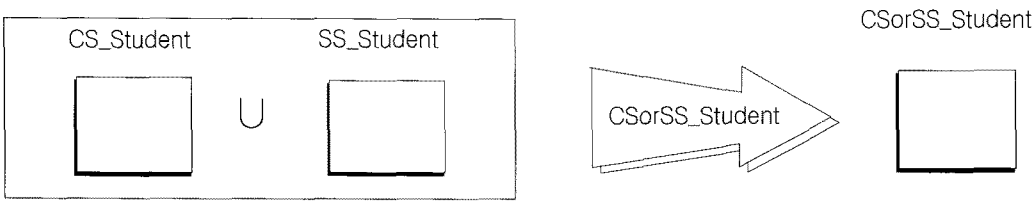
∃x1 ∃x2 ∃x5 [x1 ∈ Department ∧ x1.DName = '전산학과' ∧

x2 ∈ Instructor ∧ x1.id = x2.dept_id ∧ x2.id = x3.instructor_id ∧

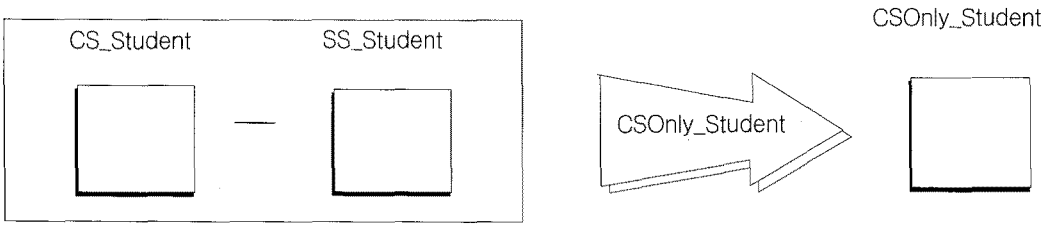
x5 ∈ Student ∧ x4.sid = x5.id ∧ x5.Name = '이동주'] }



<그림 3-5> 전산학 전공 학생 정보를 위한 VRQL 뷰



<그림 3-6> 합집합 연산을 뷰로 추상화한 예



<그림 3-7> 차집합 연산을 뷰로 추상화한 예

3.2 VRQL에서의 집합 연산

본 절에서는 VRQL 질의문의 표현력을 증대시키기 위해 위해 집합 연산을 도입한다. VRQL은 벤다이어그램에 기초한 표기로 인해 집합 연산의 표현이 자연스럽게 정의된다.

3.2.1 VRQL 집합 연산의 표기 및 예

VRQL의 집합 연산에는 합집합(∪), 교집합(∩), 차집합(—) 연산을 포함하는데, 이들 연산은 이진연산으로 두 개의 블랍 또는 뷰에 적용하여 뷰를 생성한다. 이때 집합 연산에 인수로 주어지는 블랍 또는 뷰들은 관계대수의 합집합 가능(Union-compatible)의 조건을 만족해야 한다.

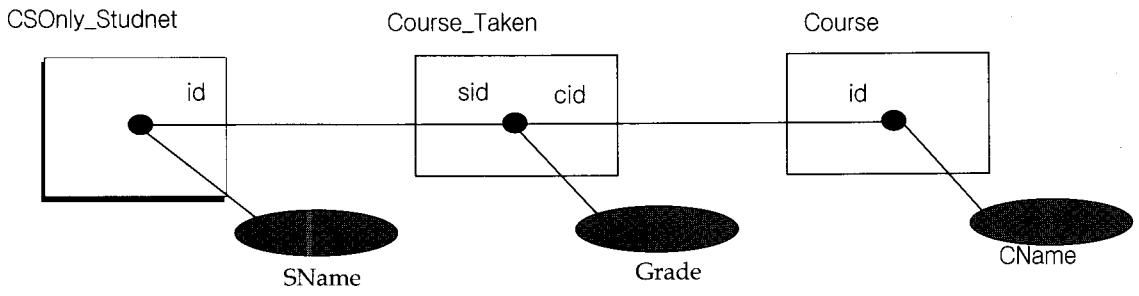
<그림 3-5>에서는 전산학 전공의 Major_ID가 'cs'라 할 때 전산학 전공 학생의 학번, 이름과 주소를 구하는 질의문과 이로부터 뷰 SS_Student를 생성한다. 유사한 방법으로 통계학 전공 학생의 학번, 이름과 주소를 구하는 뷰 SS_Student를 생성했다고 가정하자.

<그림 3-6>은 합집합 연산을 통해 전산학 또는 통계학 전공 학생들의 이름과 주소를 구하는 연산이다. <그림 3-7>은 차집합 연산을 통해 통계학을 전공하지 않는 전산학 전공의 학생들의 이름과 주소를 구하는 연산이다. 교집합 연산도 <그림 3-6>과 <그림 3-7>과 유사하게 보일 수 있다.

한편, 언어 정의의 편의를 위해 집합 연산의 사용 시, 대상 블랍 또는 뷰에 대해 조건이나 프로젝션의 지정은 하지 않는다고 가정한다.

3.2.2 집합 연산이 사용된 VRQL 질의문의 의미

집합 연산의 결과로 생성된 뷰도 <그림 3-8>과 같이 VRQL 질의문에 사용될 수 있다. 그러나 기존의 뷰 확장 규칙은 집합 연산이 고려되지 않았기 때문에 <그림 3-8>과 같은 질의문의 의미를 정의할 수 없다. 따라서 본 절에서는 뷰의 정의 질의문에 집합 연산이 사용된 뷰가 질의문에 사용되고 있을 때 적용하는 집합연산 분배규칙을 설명한다.



<그림 3-8> VRQL 뷰를 사용한 질의문의 예

집합연산 분배규칙 (번역알고리즘5)

VRQL 질의문 Q에 뷰 정의 질의문이 V1 * V2으로 정의된 뷰 V가 포함된다고 가정하자. 단, * 는 ∪, ∩, — 중 하나의 연산자를 의미한다. 질의문 Q에서 V를 V1으로 뷰 확장한 VRQL 질의문을 Q1, 질의문 Q에서 V를 V2로 뷰 확장한 VRQL 질의문을 Q2라고 하면, 질의문 Q는 Q1 * Q2로 변환될 수있다. 이때 뷰 V는 질의문 Q에서 집합연산 *에 대해 분해된다고 한다.

집합연산 분배규칙을 <그림 3-8>에 적용하면 다음과 같다.

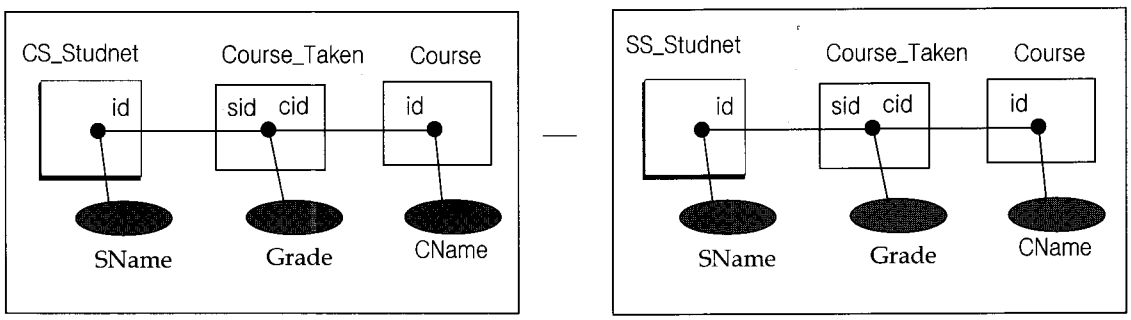
<그림 3-10>의 노드 N4와 노드 N5는 노드 N2와 노드 N3에 각각 뷰 확장 규칙이 적용된 VRQL 질의문을 의미한다. 위의 예에서 알 수 있듯이 집합연산 분배법칙은 트리를 확장해나가나 뷰 확장 규칙은 단지 리프노드의 내용만을 변경한다.

집합연산으로 확장된 VRQL 질의문의 의미는 절차적인 특성을 반영하여 관계대수에 기초하여 정의한다. 이에 대한 자세한 내용은 다음의 번역 <알고리즘 6>을 참조하시오.

<그림 3-9>는 <그림 3-8>에 변환 규칙이 적용된 결과로 <그림 3-8>의 뷰 CSOnly_Student가 뷰 CS_Student와 뷰 SS_Student로 대치되어 각각 두 프레임에 표현되어있고 두 프레임은 차집합 연산(-)으로 연결되어있다.

VRQL 질의문의 정규화, 정규형과 VRQL 질의 트리

집합연산이 사용된 VRQL 질의문에 집합연산 분배규칙과 뷰 확장 규칙을 반복 적용하게 되면 뷰 정의 질의문에 집합연산이 사용된 모든 뷰들이 분해된다. 이때 결과로 반환되는 VRQL 질의문은 논리적 VRQL 질의문들과 집합연산들로



<그림 3-9> 집합연산 분배규칙을 적용한 질의문의 예

구성된다. 이와 같이 VRQL 질의문이 뷰 확장 규칙과 집합연산 분배규칙을 통해 뷰의 확장과 뷰의 분해가 이루어져 논리적 VRQL 질의문과 집합연산들로 구성된 형태를 VRQL 질의문의 정규형이라고 하고 이 과정을 VRQL 질의문의 정규화라고 한다.

정규화과정은 내부 노드는 집합연산자를 나타내고 리프 노드는 VRQL 질의문을 나타내는 이진 트리의 생성 과정으로 이해할 수 있으며, 이때 정규형은 모든 내부 노드들은 집합연산자를 나타내고 모든 리프노드들은 논리적 VRQL 질의문을 나타내는 트리가 된다. 이러한 이진 트리를 VRQL 질의 트리라고 한다.

VRQL 질의 트리에서 뷰 확장 규칙과 집합연산 분배규칙은 항상 트리의 리프 노드에 적용되는데, 뷰 확장 규칙은 적용 대상의 리프노드의 VRQL 질의문을 뷰 확장을 통해 생성되는 VRQL 질의문으로 변경한다. 집합연산 분배규칙은 적용대상의 리프노드를, 집합연산자를 루트로 하고 집합연산 분배규칙의 결과로 생성되는 두 VRQL 질의문들을 각각 리프노드로 표현하는 이진트리로 대치하는 것을 의미한다. VRQL 질의 트리를 통한 정규화 과정의 예가 <그림 3-10>에 주어져있다. 노드 N1이 <그림 3-8>의 질의문을, 노드 N2와 N3은 각각 <그림 3-9>의 왼쪽 프레임의 질의문과 오른쪽 프레임의 질의문을 나타낸다고 하자.

확장된 VRQL 질의문의 의미 정의(번역알고리즘 6)

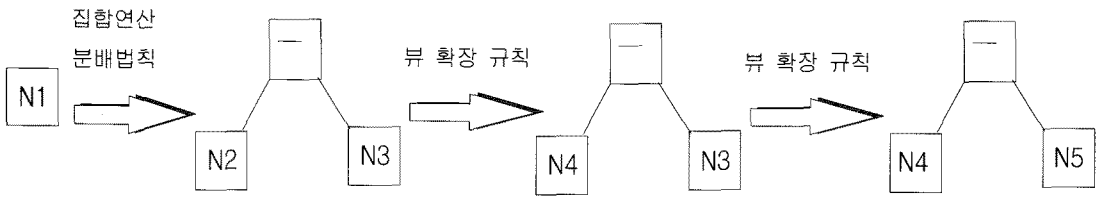
(a) (VRQL 질의문의 정규화) VRQL 질의문에 뷰 확장 규칙과 집합 분배규칙을 통해 VRQL 질의문의 정규형에 대응하는 VRQL 질의 트리 VTree를 생성한다.

(b) (a)에서 생성된 VTree의 각 리프노드의 내용, 즉 논리적 VRQL 질의문 vrqi에 대해 번역 <알고리즘 1, 2, 3>을 적용하여 논리곱 관계해석의 질의문 rcqi을 생성하여 리프노드의 내용으로 대치한다. 이 작업을 모든 리프노드에 대해 반복한다.

(c) (b)에서 변경된 VTree의 각 리프노드의 내용, 즉 논리곱 관계해석의 질의문들을 관계대수의 질의문으로 변경한다. (이 단계에서 사용될 알고리즘은 본 논문에서는 소개하지않고자 한다. 이를 위해서는 [Abiteboul, 1995; Ullman, 1988]을 참고하시오.)

(d) VTree의 모든 노드의 내용(집합연산자 또는 관계대수의 질의문)을 순차접근(inorder traversal) 방식을 통해 방문하면서 노드의 내용을 출력하여 집합연산을 포함한 관계대수의 질의문을 생성한다.

위에서 설명한 번역 <알고리즘 6>에서는 VRQL 질의문을 관계대수의 질의문으로 변환하였다. 이는 단지 다음 절에서 설명할 정리 3의 내용



<그림 3-10> VRQL 질의문의 정규화 과정

의 증명의 편의를 위한 것으로 VRQL 질의문의 의미는 집합연산을 AND, OR, NOT으로 번역하여 관계해석으로 정의하여도 관계없다.

3.3 VRQL의 표현력

논리적 정의에 기초한 VRQL의 표현력은 논리곱 관계 해석의 표현력과 동등함을 2.2.4절에서 설명하였다. 본 절에서는 절차적 정의 즉 뷰와 집합 연산의 도입을 통한 확장된 VRQL의 표현력이 관계 대수 표현력과 같음을 보인다.

정리 3) VRQL 뷰와 집합연산을 통해 확장된 VRQL의 표현력은 관계대수의 표현력과 동등하다.

(증명) (=>) 번역 <알고리즘 6>에 의해 간단히 증명된다. 즉, 확장된 VRQL의 임의의 질의문은 번역 알고리즘 6을 통해 관계대수의 질의문으로 변환이 가능하다.

(<=) 관계대수의 임의의 질의문은 VRQL 질의문으로 변환 가능함을 보인다. 이는 관계연산자의 수에 대한 간단한 귀납법(induction)으로 증명 가능하다. 관계대수의 관계적으로 완전(re lationally complete)한 연산자 집합은 실렉션, 프로젝션, 카티전 프로덕트, 합집합, 차집합 연산으로 구성되므로 이들 각각의 연산자가 사용되는 경우를 분석해야한다.

Basis case: 관계연산자가 사용되지 않는 경우, 즉 임의의 릴레이션 R으로 구성된 경우이다. 이에 대한 VRQL의 질의문은 릴레이션 R에 대응하는 블랍 만을 생성하면 된다.

Induction case: N 미만의 관계대수 연산자가 사용되는 관계대수 질의문이 VRQL 질의문으로 변환가능하다고 가정하고 N 개의 관계대수 연산자가 사용되는 경우를 고려하자. 이때 다음의 다섯 경우만을 고려하면 충분하다.

1) E = E1∪E2. 가정에 의해 E1과 E2가 각각 VRQL 질의문으로 표현된다고 하고 이들을 Q1과 Q2로 나타낸다고 하자. Q1과 Q2로 표현된 VRQL 질의문들에 대해 각각 뷰를 생성하고 이들을 V1과 V2라고 하면 E는 VRQL 질의문 V1 ∪ V2로 표현된다.
2) E = E1-E2. 1)의 경우와 유사한 방법으로 V1 - V2로 표현된다.
3) E = πa1, a2, ···, ak(E1). E1에 대한 VRQL 질의문을 Q1이라 할 때, Q1에 대한 뷰를 생성하고 이를 V1이라 하자. 그러면 E는 V1에 시각변수와 스텀프 블랍을 추가하고 스텀프 블랍에 속성 리스트 a1, a2, ··· ak를 레이블로 표현한 VRQL 질의문으로 변환된다.
4) E=E1×E2. E1과 E2에 대응하는 VRQL 질의식을 Q1과 Q2라고 할 때, 프레임안에 Q1과 Q2를 아무런 조건 지정없이 표현하면 이 식이 E를 나타내는 VRQL 질의문이 된다.
5) E = σcond(E1). *cond*는 단순 실렉션 조건으로 attri θ attrj 또는 attri θ c의 형태를 갖는다고 가정하자(단, attri, attrj 는 속성이름, c는 상수, θ는 비교연산자이다). E1에 대응하는 VRQL 질의식을 Q1, 그리고 이를 뷰로 표현한 VRQL 질의문을 V1이라 할 때, E는 V1에 시각변수를 추가하고 이에 대해 cond에 해당하는 VRQL 조건을 표현한다. 그리고 추가된 시각변수에 스텀프 블랍을 추가하여 모든 속성들을 레이블로 표현한다. 이렇게 생성된 VRQL 질의문은 E에 대응하는 VRQL 질의문이 된다. 단순 실렉션 조건이 아닌 경우, 즉 조건에

NOT이 포함된 경우에는 NOT이 포함되지 않은 조건으로 변환하고 OR 및 AND로 중첩된 조건인 경우에는 이를 일련의 단순 실렉션 조건을 가진 관계대수의 질의문으로 분해하여 처리하면 된다. 자세한 내용은 Ullman[1988]의 Lemma 3.4와 Lemma 3.5를 참조하시오.

Ⅳ. 결 론

본 논문에서는 벤다이어그램과 그래프에 기초한 시각 관계형 질의어 VRQL을 제안하였다. QBE 즉 도메인 관계 해석에 기초한 대개의 다른 관계형 시각 질의어들과는 달리 VRQL의 논리적 정의 부분은 투플 관계 해석에 기초하고 있고 귀납적인 형식 의미 정의를 가지고 있다. 벤다이어 그램에 기초한 시각적 표현은 릴레이션을 투플들의 집합으로 자연스럽게 표현할 수 있도록 하고 조인 조건을 포함한 다양한 조건들의 직관적인 시각 표현이 가능하게 해 준다.

대개의 기존 시각 질의어들이 논리곱 연산의 표현은 자연스럽게 이루어졌으나 NOT이나 OR와 같은 연산들을 표현하지 못하였거나 또는 그 표현이 어색하였다. 반면에 VRQL에서는 벤다이어그램에 기초한 시각적 표현과 투플 관계 해석에 기초한 의미 정의로 말미암아 집합 표현이 매우 편리하다. 따라서 VRQL의 절차적 정의 부분에서 합집합, 교집합, 차집합과 같은 집합 연산을 도입하였다. 이러한 VRQL에서는 집합 연산 도입으로 인해 논리합(disjunction)과 부정(negation) 연산을 합집합과 차집합 연산을 통해 표현이 가능하게 되었다.

한편, 질의 과정의 중간 결과나 기존 질의의 내용을 뷰를 통해 추상화하여 질의문의 생성시 편의를 도모하였다. 뷰와 집합 연산의 도입은 VRQL에서 질의문의 표현의 편리성은 물론 그 표현력의 범위도 확대시켰다. 합집합과 차집합을 통해 확대된 VRQL의 표현력은 논리곱 질의언어의 수준을 넘어, 관계적으로 완전한 수준으로 증대시켰다. 이의 증명을 위해 본문에서 관계대수의 표현력과 동등함을 보였다.

본 논문에서 제안한 VRQL은 그 표기가 직관적일 뿐 아니라 관계적으로 완전한 수준의 표현력을 지닌 시각 질의어이나 아직은 이론적 수준에서 정의된 언어로 아직 미비한 점들이 많다. 현재의 VRQL에는 집계함수(aggregation function)와 그룹화 연산(group-by)들이 지원되고 있지 못하다. 이들에 대한 연구와 VRQL의 구현을 위해 윈도우 환경에서 사용자의 직관에 더욱 부응하는 시각적 표기의 개발이 진행중이다.

〈참 고 문 헌〉

[1] 이석균, "VOQL*: 귀납적으로 정의된 형식 시맨틱을 지닌 시각 객체 질의어," 한국정보과학회 논문지 : 데이터베이스, 27권, 2호, 2000년 6월, pp. 151-164.

[2] 이석균, 나연묵, 서용무, "시각적 객체지향 데이터베이스 질의어의 설계 및 질의처리기의 구현," 경영정보학연구, 11권 2호, 2001년 6월, pp. 121-139.

[3] Abiteboul, S., Hull, R., and Vianu, V., Foundations of Databases, Addison Wesley Publishing Co. 1995.

[4] Angelaccio, M., Catarci, T., and Santucci, G., "QBD*; A Graphical Query Language With Recursion," IEEE Trans. on Software Engineering, Vol. 16, No. 10, October 1990, pp. 1150-1163.

[5] Bertino, E. et al., "Object-Oriented Query Language: The Notion and the Issues," IEEE Trans. on Knowledge and Data Engineering, Vol. 1, No. 3, June 1992, pp. 223-237.

[6] Carey, M., Haas, L., Maganty, V., and Will iams, J., "PESTO: An Integrated Query /Browser for Object Databases," in Proc. Intl, Conf, on Very Large Data Bases, 1996, pp. 203-214.

[7] Chavda, M., and Wood, P., "Towards an ODMG-Compliant Visual Object Query Language," In Proc. Intl. Conf. on Very Large Data Bases, Athens, Greece, 1997, pp. 456-465.

[8] Cruz, I., Mendelzon, A., and Wood, P., "Graphical Query Language supporting Recursion," In Proc. ACM SIGMOD Intl. Conf. on Management of Data, 1987, pp. 323-330.

[9] Czejdo, B., Elmasri, R., and Rusinkiewicz., M., "A Graphical Data Manipulation Language for an Extended Entitiy- Relationship Model," IEEE Computer, Vol. 23 Mar. 1990, pp. 26-36.

[10] Frohn, J., Lausen, G., and Uphoff, H., "Access to Objects by Path Expressions and Rules," In Proc. the 20th VLDB Conference, 1994, pp. 273-294.

[11] Guting, R., "GraphDB: Modeling and Quer ying Graphs in Databases," In Proc. the 20th Intl Conf. on Very Large Data Bases, 1994, pp. 297-308.

[12] Gyssens, M., Paredaens, J., Van den Bussche, J., and Van Gucht, D., "AGraph- Oriented Object Database Model," IEEE Trans. on Knowledge and Data Engineering, Vol. 6, No. 4, August 1994, pp. 572-586.

[13] Harel, D., "On Visual Formalisms," In Comm. of the ACM, Vol. 31, No. 5, 1988, pp. 514-530.

[14] Kifer, M., Kim, W., and Sagiv, Y., "Querying Object-Oriented Databases," In Proc. ACM SIGMOD Intl. Conf. on Management of Data, San Diego, CA, 1992, pp. 393-402.

[15] Kim, Jeonghee, Han, Tae-sook, Lee, Suk Kyoon, "Visualization of Path Expressions in a Visual ObjectOriented Database Query Language," In Proc. Intl Conf. on Database Systems for Advanced Applications, Taiwan, 1999, pp. 99-108.

[16] Kim, Jeonghee, Han, Tae-sook, Lee, Suk Kyoon, "VOQL : A Visual Object-Oriented Database Query Language For Visualizing Path Expressions," International Journal of Computer Systems, Science and Engineering, Vol. 15, No. 4, July 2000, pp. 215-232.

[17] Lee, Suk Kyoon, Whang, Kyu-Yong, "VOQL*: A Visual Object Query Language With Inductively Defined Formal Semantics," Journal of Visual Languages and Computing, Vol. 12, No. 4, Academic Press, August 2001, pp. 413-433.

[18] Mohan, L. and Kashyap, R. L., "A Visual Query Language for Graphical Interaction With Schema-Intensive Databases," IEEE Trans. on Knowledge and Data Engineering, Vol. 5, No. 5, 1993, pp. 843-858.

[19] Sockut, G.H., Burns, L. M., Malhotra, A., and Whang, K-Y., "GRAQULA: A Graphical Query Language for Entity-Relationship or Relational Databases," Data and Knowledge Engineering, Vol. 11, 1993, pp. 171-202.

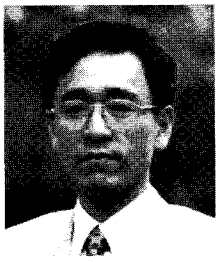
[20] Whang, K. et al., "Two-dimensional Specification of Universal Quantification in a Graphical Database Query Language," IEEE Trans. on Software Engineering, Vol. 18, No. 3, March 1992., pp. 216-224

[21] Vadaparty, K., Aslandogan, Y. A., and Ozsoyoglu, G., "Towards a Unified Visual Database Access," In Proc. ACM SIGMOD Intl. Conf. on Management of Data, 1993, pp. 357-366.

[22] Ullman, J., Principles of Database and Knowledge-Base Systems, Volume 1, Computer Science Press, San Francisco 1988.

[23] Zloof, M., "Query By Example," IBM Systems Journal, Vol. 16, 1997, pp. 324-3437.

◆ 저자소개 ◆



이석균 (Lee, Suk Kyoon)
서울대학교 경제학과에서 학사, University of Iowa에서 전산학 석사 및 박사를 취득하였다. IEEE 8th International Conference on DataEngineering에서 최우수 논문상 수상을 수상하였으며 세종대학교 정보처리학과에서 전임 강사로 근무하였다. 현재 단국대학교 자연과학부 부교수이다. 주요 관심 분야는 데이타베이스에서 불완전 정보관리, 데이타베이스 시각 질의어 설계, 다중처리기에서의 실시간 스케줄링, 데이터웨어하우스, 데이터 마이닝, XML 등이다.

◈ 이 논문은 2002년 4월 8일 접수하여 1차 수정을 거쳐 2002년 5월 21일 게재확정되었습니다.