관계형 데이터베이스에서 Tachyon 객체-관계 데이터베이스로의 변환 기법

장인기[1] 공회경[2] 이충세[1] 조완섭[3] 최완[4]
충북대학교 전자계산학과[1], 정보산업공학과[2], 경영정보학과[3]
한국전자통신연구원[4]
cig2005@shinbiro.com, khkong@just.chungbuk.ac.kr, csrhee@cbucc.chungbuk.ac.kr,
wscho@trut.chungbuk.ac.kr, wchoi@etri.re.kr

A Transfomation Technique from a Relational Database to the Tachyon Object-Relational Database

In-Ki Jang[1], Hee-Kyung Kong[2], Chung-Se Rhee[1], Wan-Sup Cho[3], and Wan Choi[4]
[1]Dept. of CS, [2]Dept. of II ,[3]Dept. of MIS, Chungbuk National Univ.,
[4]Real-Time DBMS Team ETRI

요 약

전자 상거래 등에서 웹 클라이언트들은 시간이 갈수록 빠른 서비스를 요구하고 있다. 디스크 기반의 관계형 데이터베이스를 그대로 유지하면서도 빠른 응답을 가능하게 하는 방안으로, 메인 메모리 기반 데이터베이스 시스템(Main Memory-Based DBMS)을 Front-End로 사용하는 방법이 제안되고 있다. 본 논문에서는 관계형 데이터베이스 시스템에 객체-관계형 메인 메모리 기반 데이터베이스 시스템인 Tachyon[1]을 Front-End로 운용하여 성능을 개선하는 문제를 다룬다. 특히, 이 경우에 데이터 모델 상의 차이점으로 발생할 수 있는 문제를 정의하고, 그 해결책을 제시한다. 주요 내용으로는 관계 데이터베이스를 Tachyon에 적합한 객체 데이터베이스로 변환하는 기법과, 관계 질의를 객체 질의로의 변환 기법이다. 이러한 변환 기법은 관계 데이터베이스의 Front-End로 객체-관계 데이터베이스를 사용할 수 있도록 하는 연구의 출발점이 될 것이다.

1. 서론

인터넷에서 사용자들은 점점 더 빠른 응답을 요구하고 있고, 웹 서비스의 대부분이 데이터베이스를 근간으로 하기 때문에 데이터베이스의 성능은 웹 서비스를 위한 중요한 요소가 되고 있다. 현재, 데이터베이스의 성능을 향상시키기 위한 다양한 연구가 이루어지고 있으며, 그 중 웹에서의 성능 향상을 위해 메인 메모리 데이터베이스 시스템을 기존의 데이터베이스 시스템과 통합하여 사용하는 방안이 연구되고 있다[2, 3]. 메인 메모리 데이터베이스 시스템을 이용함으로써 기존의 디스크 기반 데이터베이스 시스템에 비해서 성능 향상의 장점을 얻을 수 있다[5]. 메인 메모리에 대한 접근 속도는 디스크에 접근하는 속도보다 빠르기 때문이다.

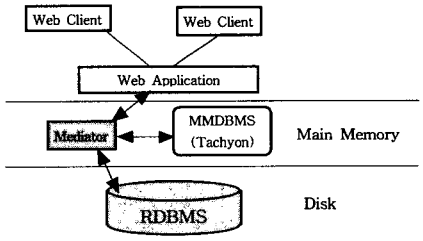
본 논문에서는 관계형 데이터베이스 시스템을 유지한 상태에서 메인 메모리 데이터베이스 시스템을 결합하는 구조를 다룬다. 관계형 스키마에는 객체식별자(OID)[9]나 OID 참조, 역참조(Inverse) 등의 개념이 없고, 객체지향 스키마인 Tachyon에는 외래키가 없으므로 두 모델간의 스키마 변환이 필요하다. 또한 이러한 스키마의 차이로 인해서 질의를 할 경우에도 상호간의 질의를 인식할 수 없는 문제가 발생하게 된다. 이러한 문제점을 해결함으로써, 객체-관계형 모델에 기반한 Tachyon을 관계형 데이터베이스의 Front-End로 사용할 수 있게 된다.

객체지향 데이터베이스와 관계형 데이터베이스 간의 스키마 사상을 위한 연구가 지속적으로 연구되어 왔지만[4, 5, 6, 7, 8], 기존의 연구에서는 객체지향 데이터베이스에서 관계형 데이터베이스로의 변환에 대한 연구가 주로 이루어져 왔으며, 본 연구에서 다루는 관계 데이터베이스에서 객체-관계 데이터베이스로의 변환에 관한 연구와 메인 메모리 데이터베이스에의 적용에 관한 연구는 미진한 상태이다. 따라서 본 연구는 이질적인 두 데이터베이스 간의 상호 연동에 기여할 수 있을 것이며, Front-End로 메인 메모리 데이터베이스를 이용하는 연구의 기초가 될 것이다.

본 논문에서는 2장에서 이 시스템의 전반적인 소개와 스키마 사상 및 질의 사상의 필요성을 살펴보고, 3장에서 사상 방안을 제시하며, 4장에서 본 연구의 결론을 맺고 향후 과제를 기술한다.

2. 시스템 구조

본 논문에서 사용된 시스템의 구조는 [그림 1]과 같다. [그림 1]에서 중개자(Mediator)[6]는 웹에서 응용을 통해 사용자의 요구가 있을 경우, 이를 판단하고 질의가 적용될 데이터베이스 시스템을 선정하여, 질의 수행을 하는데 필요한 제반 처리를 담당한다. 즉, 웹 응용에게 데이터베이스 시스템에 대한 투명성을 제공하여, 기존의 디스크 기반 데이터베이스 시스템에 비해 보다 향상된 성능을 제공하는 것을 목적으로 한다.



[그림 1] 시스템 구조

[그림 1]의 시스템 구조에서 중개자(Mediator)는 기본적으로 웹에서의 질의를 디스크 기반 데이터베이스 시스템과 메인 메모리 데이터베이스 시스템 중 어느 곳에서 처리하도록 할 것인지 결정한다. 또한, 관계형 모델과 객체-관계형 모델간의 스키마 사상에 대한 알고리즘을 가지고 있어 두 모델의 스키마 변환을 담당한다. 이를 통해 웹에서 들어오는 질의를 파악하고, 처리 수단을 결정하는데 필요한 정보를 얻을 수 있다. 다음으로, 관계형 모델의 스키마 정보와 변환된 객체지향 모델의 스키마 정보를 사용하여 질의를 변환하는데 사용한다. 질의 변환을 위해서는 먼저 각 데이터베이스 시스템의 메타 데이터 정보가 중개자에 있어야 하며, 스키마 변환이 이루어져야 한다.

일반적으로 메인 메모리 데이터베이스에 저장되는 데이터는 주로 핫 데이터(Hot Data)가 된다. 핫 데이터는 사용자에 의해 자주 접근되고,

작은 용량을 가지며, 엄격한 시간적인 제약를 지니는 데이터를 의미한다[5]. 이러한 데이터의 예로는 은행의 계좌 레코드, 전화교환 응용의 라우팅 테이블 정보, 고객의 일반 신상 정보 등을 들 수 있다. 이와는 반대로 콜드 데이터(Cold Data)는 디스크 상에 존재하는 관계형 데이터베이스 시스템에서 다룬다. 관계형 모델에서는 주키와 보조키를 사용하여 릴레이션들 간의 관계를 정의한다. 또한 이를 사용한 상호참조의 가능성이 아주 높으며, 대부분의 사용자의 질의는 이를 사용하여야 한다. 따라서, 특정 릴레이션이 핫 데이터가 되면, 이 릴레이션과 연관된 모든 릴레이션을 핫 데이터로 간주하여 이들 모두를 메인 메모리 데이터베이스에서 관리한다.

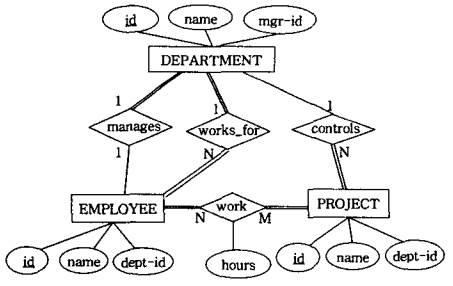
스키마와 질의의 변환 시에 고려되어야 할 점이 데이터의 일관성 유지 방안이다. 변환의 대상이 되는 관계형 데이터베이스 시스템은 오라클, 인포믹스와 같은 일반적인 SQL 92 표준을 따르는 관계형 데이터베이스 시스템이며, 메인 메모리 데이터베이스 시스템은 한국전자통신연구원(ETRI)에서 개발 중인 메인 메모리 상주형 객체-관계형 데이터베이스 시스템인 Tachyon을 가정 한다. 웹에서의 질의는 SQL 92 표준을 따르는 질의로 가정한다.

3. 관계형 모델에서 객체 관계형 모델로의 사상(Mapping)

본 장에서는 관계형 모델의 스키마와 객체 관계형 모델간의 스키마 사상방법과, 두 모델로의 질의 변환방법에 대해서 논의한다.

3.1 스키마 사상

이 장에서는 관계형 스키마를 객체 관계형 데이터베이스 시스템인 Tachyon의 스키마로 사상하는 방법에 대해서 논의한다. 먼저 전형적인 관계형 스키마로 [그림 2]와 같은 스키마 구조를 가정하자. [그림 2]에서 밑줄이 있는 애트리뷰트는 주키를 의미하며, id로 끝나는 속성명은 외래키를 의미한다.



[그림 2] 예제 관계형 스키마

관계형 모델의 스키마를 객체지향 모델의 스키마로 변환하는 방법은 3단계의 알고리즘으로 구성된다.

가정 : 릴레이션 T는 R(T)로, R(T)와 참조관계가 있는 릴레이션 T′은 R(T′)으로 표시한다. R(T′)의 주키(이하 PK)인 PR(T′)를 참조하는 R(T)의 외래키(이하 FK)는 FR(PR(T′))로 표시한다.

1단계 :

1-1)각 릴레이션은 같은 이름을 갖는 클래스 C(T)와 C(T′)로 사상한다.

1-2) R(T)의 PR(T)는 C(T)의 PR(T)로 변환하며, PK의 속성을 유 지한다.

2단계 : 1 : 1 관계 사상으로, R(T)와 R(T′)이 1 : 1관계라면,

2-1) R(T)의 FR(PR(T′))는 R(T′)의 사상된 C(T′)에 대해 OID_REF로 사상한다.

FR(PR(T′)) OID_REF C(T′)

2-2) C(T′)에는 FR(PR(T′))에 대해서 C(T)_FR(PR(T′))이라는 이름으로 C(T)에 대해서 "OID_REF INVERSE"를 생성한다.

C(T)_FR(PR(T′)) OID_REF INVERSE C(T).FR(PR(T′))

3단계 : 1 : N 관계로, R(T)와 R(T′)가 1 : N관계라면,

3-1) 1 : N의 경우 N 쪽에 FK가 존재하게 된다. FK가 존재하는 릴레이션에 대한 사상을 먼저 하게 된다.

① R(T)에 존재하는 FK인 FR(PR(T))를 2-1)과 같은 방법으로 C(T′)에 사상한다.

FR(PR(T)) OID_REF C(T)

② C(T)에는 C(T′)_FR(PR(T))라는 이름으로 "OID_SET INVERSE"를 생성한다.

C(T′)_F(PR(T)) OID_SET INVERSE C(T′)

[그림 2]에 대해서 사상 알고리즘을 적용하면 다음과 같다.

1단계를 통해서 각 릴레이션에 대한 같은 이름의 클래스인 EMPLOYEE, DEPARTMENT, PROJECT, WORK 를 생성한다. 각 릴레이션의 PK를 사상된 각 클래스의 PK로 설정한다.

2단계의 예로 DEPARTMENT의 mgr-id(EMPLOYEE의 id를 참조)와 EMPLOYEE는 1:1 관계이므로,

DEPARTMENT
mgr-id OID_REF EMPLOYEE(변경 사상)

EMPLOYEE
department_mgr-id OID_REF INVERSE DEPARTMENT.mgr-id
(추가 사상)

3단계에서,
DEPARTMENT와 PROJECT는 1 : N관계이고, PROJECT의 dept-id가 DEPARTMENT의 id를 참조하는 FK이므로,

PROJECT
dept-id OID_REF DEPARTMENT (변경 사상)

DEPARTMENT
mgr-id OID_REF EMPLOYEE
project_dept-id OID_SET INVERSE PROJECT.dept-id
(추가 사상)

사상의 결과로 얻어진 Tachyon 스키마는 관계형 스키마의 사상 알고리즘을 적용하는 중개자를 통해서 이루어지게 된다.

3.2 질의 사상

이 장에서는 관계형 질의를 객체-관계형 모델인 Tachyon에서 인식할 수 있는 질의로 사상하는 방법에 대해서 논의한다.

질의 사상에는 고려해야 할 사항이 있다. 이는 Tachyon에서의 제약으로, 일반적으로 알려진 객체-관계형 모델에서의 제약은 아니다. Tachyon의 질의는 정형화된 단순질의를 실시간으로 처리하도록 고안되었다. 따라서, 새로운 테이블을 생성해야 하는 등의 실행시간 예측 불가능한 질의 규격은 제외되었다. 즉, ORDER BY, GROUP BY, LIKE, HAVING 등을 사용할 수 없다. 두 번째로 비교 연산자로는 =, <>, >, <, >=, <= 등이 사용가능하며, 논리 연산자로는 and 가 가능하다. 세 번째로 값 기반 조인이 허용되지 않는다.

대부분의 웹에서의 질의가 SELECT가 주를 이루고 있기 때문에 본 논문에서 논의하는 질의 사상의 범위는 SELECT만을 다룬다.

각 질의 변환은 다음과 같은 세 단계를 거쳐서 사상된다.

1단계 : WHERE 절의 분석

WHERE 절에서 사용되는 제약 조건의 형식은 다음과 같이 표시한다.[4, 8]

좌측변수 비교연산자 우측변수
$R(T).Attr_1$ OP $R(T').Attr_2$

본 사상 알고리즘에서는 $R(T).Attr_1$과 $R(T').Attr_2$는 PK 또는 FK 속성을 갖는 애트리뷰트로 가정한다.

1-1) 만약 i 번째($i > 1$인 정수) 제약 조건 $Expr_i$ 가 "$R(T).Attr_1$ OP $R(T').Attr_2$" 형태라면,

1-1-1) 중개자 M의 Meta 정보 MMD를 이용하여, $R(T).Attr_1$ 또는 $R(T').Attr_2$ 에 대해, Tachyon 스키마로 변환된 정보를

조사한다.

1-1-2) C(T)와 C(T')와 같은 이름의 노드 N(T)와 N(T')을 생성한다. 만약, 노드가 이미 존재한다면, 그대로 사용한다.

1-1-3) C(T)와 C(T') 사이가 OID_REF와 OID_SET 관계인 경우는 두 노드 중 참조되는 노드의 방향으로 방향성 링크 L(T-T')을 만들고, INVERSE가 존재하면 양방향성 링크를 만든다.

1-1-4) $Attr_1$과 $Attr_2$를 각각 N(T)와 N(T')에 명시한다.

1-2) $Expr_i$가 "R(T).$Attr_i$ OP $V1$"($V1$은 값) 형태라면,

1-2-1) R(T)에 대해 N(T)와 $V1$에 대해서 N($V1$)을 생성한다. 이미 노드가 존재한다면, 그대로 사용한다.

1-2-2) 1-2-1))에서 생성된 N(T)와 N($V1$)에 대해 $V1$으로 향하는 방향성 L(T-V1)을 생성하고 $Attr_1$과 $V1$을 N(T)와 N(V1)에 명시한다.

1-3) 각각의 제약조건에 대해서 1-1)과 1-2) 과정을 수행하여 전체 제약조건 그래프(Constraint Graph; 이하 CG)를 그린다.

1-4) 사상된 질의의 WHERE 절에는 CG의 노드 중에서 들어오는 링크가 없으며, 다른 모든 노드에 도달 가능한 노드의 클래스 이름에서 시작해서 전체 경로를 적어 준다.

2단계 : FROM 절

1-4)에서의 선정된 시작 클래스 명을 적어준다.

3단계 : SELECT 절

3-1) 관계형 질의의 SELECT 절에 명시된 변수(Relational Select Variable; 이하 RSV) 각각에 대해서,

3-1-1) MMD를 이용하여, 2단계에서 결정된 클래스에서 시작하여, RSV까지의 경로(RSV Path; RSVP)를 결정한다.

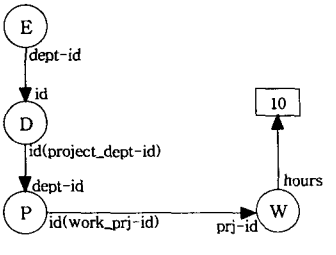
3-1-2) RSV가 FROM 절에 있는 클래스의 속성이라면, 그대로 적어 주고, 그 밖의 경우는 각 RSV를 RSVP로 대체 한다.

위의 세 단계를 이용한 예는 다음과 같다.

```
SELECT E.name,  D.name,  P.name
FROM  employee as E, department as D, project as P, work ad W
WHERE  E.dept-id  =  D.id
   and   D.id  =  P.dept-id
   and   P.id  =  W.prj-id
   and   W.hours = 10;
```

[그림 3] 관계형 질의 예

1단계 적용 결과 제약 조건 그래프는 [그림 4]와 같다.



[그림 4] [그림 3]질의의 WHERE절에 대한 제약조건 그래프

2 단계를 적용하면, [그림 4]의 그래프에서 처음 시작되는 노드인 Employee와 마지막 노드인 Work만 나타난다. 사각형으로 표시된 값 1004는 클래스를 나타내지 않고 있으므로 제외된다.

3 단계를 적용하면, SELECT 절에는 시작노드인 E 부터 시작되어, E.name은 그대로 E.name으로, D.name은 E.dept-id ->name으로, P.name은 E.dept-id ->project_dept-id ->name으로 SELECT가 가능하다.

위의 결과로 사상되어 나타는 질의는 [그림 5]와 같다. Select 절에서의 -> 표시에 의한 경로는 중개자가 관리하는 스키마 변환 정보를 이용하여 처리하게 된다.

```
SELECT  E.name,  E.dept-id ->name,
        E.dept-id -> project_dept-id ->name
FROM    Employee as E
WHERE  E.dept-id -> project_dept-id -> work_prj-id -> hours = 10;
```

[그림 5] 사상된 질의

사용자들은 항상 [그림 3]과 같은 순서로 제약조건을 나타내지는 않는다. 하지만 1단계의 과정은 제약 조건이 어떤 순서로 되어 있더라고 결과적으로는 같은 그래프는 나타내게 된다.

이러한 질의 변환은 중개자에서 관리하는 메타데이터 정보가 없으면 불가능하다. 1단계의 제약조건 그래프에서 각 비교연산자를 기준으로 좌측변수와 우측변수가 서로 참조하고 있는지 여부는 앞에서 설명한 스키마 사상의 결과를 중개자가 관리하면서 질의 사상 시에 적용하게 된다. 3단계의 SELECT 절에 대한 변환 역시 중개자가 가지고 있는 메타데이터 정보를 이용해서 가능하게 된다.

4. 결론 및 향후 과제

본 논문에서는 디스크 기반 관계형 데이터베이스 시스템에 메인 메모리 상주형 객체-관계 데이터베이스 시스템을 Front-End로 사용하기 위해 스키마와 질의의 사상, 그리고 데이터의 일관성 유지 방안에 대해서 설명하였다. 또한 이러한 두 모델의 사상을 위해서 메타 데이터를 관리하고 각 사상을 실제 수행하는 중개자의 역할과 기능을 기술하였다. 이를 통해, 이질적인 두 데이터베이스 간의 상호 연동이 가능하며, Tachyon을 핫 데이터를 유지하는 Front-End로 사용함으로써 보다 빠른 응답 시간을 제공할 수 있다.

향후에는 질의 변환시에 발생하는 많은 경우가 좀더 세부적으로 연구될 것이며, 특히, 데이터의 변경 및 두 데이터베이스 간의 트랜잭션 조정에 대한 연구와 중개자에 대한 세부 연구가 계속될 것이다.

5. 참고문헌

[1] Tachyon, White Paper, "http://namoo.etri.re.kr"

[2] TimesTen, White Paper, "http://www.timesten.com"

[3] 공회경, 이현우, 정문권, 장인기, 조완섭, 박유미, "전자상거래용 웹 데이터베이스에서 메모리 상주 DBMS를 이용한 성능 향상 방안", 한국정보기술응용학회 춘계학술대회, 건양대학교, p. 341-345, 2000

[4] Clement Y., Zhang Y., Meng W., W. Kim, Gaoming W., Tracy P., and Son D., "Translation of Object-Oriented Queries to Relational Queries", In Proc. Int'l Conf. on Data Engineering, pp. 90-97, 1995

[5] Hector. G., "Main Memory Database Systems: An Overview", IEEE Transactions on Knowledge and Data Engineering, Vol. 4, No. 6, Dec. 1992

[6] Leonid. S., Antonija. M., Slobodanka. D., Dejan. M., "Bridging objects and relations: a mediator for an OO front-end to RDBMSs", Information and Software Technology 41, pp. 57-66, 1999

[7] Meng, W., Kamada, A., Yu-Hsi Chang, "Transformation of relational schemas to object-oriented schemas", In Proc. Int'l Conf. on Computer Software and Applications, pp. 356-361, 1995

[8] Meng, W., Yu, C., Kim, W., Wang, G., Pham, T., and Dao, S., "Construction of a Relational Front-end for Object-Oriented Database Systems", In Proc. Int'l Conf. on Data Engineering, pp. 476-483, 1993

[9] W. Kim, *Introduction to Object-Oriented Databases*, The MIT Press, 1990.