객체 관계형 실시간 DBMS, Tachyon의 데이터 타입과 객체지향 질의 설계 및 구현

박유미[0] 배명남 최 완
한국전자통신연구원
{parkym, mnbae, wchoi}@etri.re.kr

A Design and Implementation of Object-Relational Queries and Data types for the Object-Relational Real-time DBMS, Tachyon

Yoo-mi Park[0] Myung-nam Bae and Wan Choi
ETRI

요 약

본 논문은 객체 관계형 실시간 DBMS인 Tachyon이 제공하는 데이터 타입과 객체지향 질의의 설계 및 구현을 기술한다. Tachyon은 고성능 데이터 처리를 지원하는 객체 관계형 메모리상주 데이터베이스 시스템(Main-memory resident ORDBMS) 로 Tachyon의 데이터 타입과 SQL은 ANSI SQL 3 draft를 기반으로 설계 되었으며, 객체 관계형 질의 처리를 위한 OID 데이터 타입을 추가하였다. Tachyon은 클래스 간의 수직적 관계(상속)와 수평적 관계를 정의하고 조작할 수 있는 객체지향 질의를 통하여 융통성 있는 데이터 조작이 가능하며, OID를 기반으로 경로식(path expression)을 이용한 항해 질의(navigational query)를 통하여 죠인(join) 비용을 절감하고, index를 통한 인스턴스 접근 등을 통하여 효율적인 데이터베이스 접근을 시도할 수 있다.

1. 서론

최근 인터넷 등 초고속 정보 통신 서비스의 발달에 따라 빠른 데이터 전송이 정보 처리 업계에 화두가 되었고, 높아지는 데이터 전송속도에 비례하여 데이터 처리양이 많아짐에 따라 보다 신속하고 정확한 데이터 처리가 초고속 정보 통신 사업의 선결과제가 되고 있다. 그러나, 현재 이러한 데이터를 관리하고 있는 기존의 디스크 기반 DBMS로 인터넷 상에서 폭주하는 데이터를 처리하기에는 성능이 떨어지고 DBMS 자체의 규모도 커서 더욱 빠른 데이터 처리를 위한 대안이 요구되고 있는 실정이다. 이러한 현실과 더불어 RAM 가격의 하락에 힘입어 고성능 주기억 장치 상주형 DBMS가 전 세계적으로 시장 점유율을 높여가고 있는 추세이다.

Tachyon은 주기억장치에 상주된 데이터베이스를 관리하는 DBMS로 객체 관계형 모델을 지원하며, 우선순위 기반의 스케쥴링과 soft-deadline 트랜잭션을 처리하는 실시간 DBMS이다. Tachyon은 10여년간의 DREAM-S[1] 개발 경험을 바탕으로 관계형 DBMS의 장점을 살리고 단점을 보완하여 설계 및 구현된 객체 관계형 DBMS로서, 본 논문에서는 Tachyon에서 제공하는 데이터 타입과 객체 지향 SQL 그리고 고성능 데이터 처리를 위한 객체 관계형 질의 처리기의 설계 및 구현에 관하여 기술한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 관련연구에 관하여 기술하고, 제 3장에서는 Tachyon이 제공하는 데이터 타입과 객체 관계형 질의를 기술하고, 제 4장에서는 Tachyon 질의 처리기에 관하여 기술하고, 제 5장에서는 객체 관계형 질의 사용 예를, 그리고 마지막으로 제 6장에서는 결론을 맺는다.

2. 관련연구

embedded DBMS 및 범용 디스크 기반 DBMS 시장의 front-tier로서 전 세계적으로 주기억장치 상주형 DBMS의 개발이 활기를 띠고 있다. 이에 대표적인 DBMS 들로는 미국의 TimesTen과 RAIMA, 영국의 Polyhedra등을 꼽을 수 있다. 본 장에서는 상기 DBMS들과 국산 전전자 교환기용 DBMS인 DREAM-S의 데이터 모델과 질의에 관하여 비교 기술한다.

TimesTen[2]은 미국 TimesTen사의 관계형 주기억장치 상주 DBMS로 ANSI SQL 92[3] 과 ODBC 2.5 API 를 지원하고 있다. 현재 디스크 기반의 관계형 DBMS의 데이터베이스 캐싱(caching) 기능을 수행하는 front-tier 제품이 디스크 기반 관계형 DBMS상에서 10배 이상의 데이터 처리 성능을 보이고 있다고 발표하고 있다. 영국에서 개발된 Polyhedra[4]는 객체 관계형 주기억장치 상주 DBMS 로서 active query를 지원하며 사용자 인터페이스로 SQL과 ODBC API 및 프로그래밍 API를 지원하고 있다. 미국의 RAIMA[5]

는 관계형 모델과 네트워크 모델을 혼합한 모델을 지원하며, SQL보다는 API에 주력하고 있다. DREAM-S는 국산 전전자 교환기용 데이터를 관리하기 위하여 개발된 주기억장치 상주 DBMS로서 다양한 메모리 접근 방법을 통해 데이터 접근 성능을 높였다. DREAM-S는 SQL을 바탕으로 보다 간략한 문법을 가진 embedded SQL을 제공하고 있다.

3. Tachyon의 데이터 타입과 SQL

Tachyon이 지원하는 데이터 타입과 SQL은 ANSI SQL 3 draft(1993, 9)을 기반으로[3][6-8] 하여 정보통신 응용에서 빈번하게 사용되는 데이터 타입과 SQL을 요약하여 정의되었으며 객체 관계형 질의 처리를 위한 OID 데이터 타입이 추가되었다. 제공되는 데이터 타입은 다음과 같이 4가지 종류가 있다.

- 기본 데이터 타입 : int, smallint, range, double, float, char, varchar(n), char(n), boolean, bit(n)
- 배열형 데이터 타입: int[n], smallint[n], double[n], float[n], boolean[n]
- 복합 데이터 타입: 사용자 정의 데이터 타입
- OID 데이터 타입: oid_ref(1:1), oid_set(1:n)
-

이중 OID 데이터 타입은 인스턴스의 OID를 이용하여 클래스와 클래스간의 관계를 정의하기 위하여 추가되었다. 그림 1의 SQL문은 department 클래스와 employee 클래스를 정의하는 문장으로서 OID 데이터 타입을 이용하여 한 department 에는 1 명 이상의 employee가 그 구성원으로 존재할 수 있고(a), 한 employee는 반드시 한 department 에만 소속될 수 있는 관계(b)를 정의하였다.

```
create class department (
-
    member  oid_set  employee; →(a)
);
create class employee (
  -
    dept oid_ref inverse department.member;→(b)
);
select member->name from department ;
```

그림 1. 클래스 정의 SQL 문

표 1. Tachyon이 제공하는 SQL

SQL statement	SQL command
데이터 정의어	CREATE/DROP CLASS CREATE/DROP TYPE CREATE/DROP INDEX
데이터 조작어	DECLARE CURSOR .. SELECT OPEN CLOSE FETCH "positioned" UPDATE(single-row) "positioned" DELETE(single-row) INSERT "searched" UPDATE(multi-row) "searched" DELETE(multi-row)
트랜잭션 명령어	START TRANSACTION COMMIT ROLLBACK

표 1은 Tachyon이 제공하는 SQL로 크게 데이터 정의어, 데이터 조작어, 트랜잭션 명령어로 구성된다. 이중 실시간(real-time) 데이터 처리를 지원하기 위하여 트랜잭션 명령어 중 ' START TRANSACTION' 사용 시 우선 순위와 마감 시간을 기술할 수 있다. 우선 순위를 지정하는 것은 우선순위 기반 스케줄링에 의해 우선순위가 낮은 트랜잭션이 우선순위가 높은 트랜잭션에 의해 선점 당할 수 있다는 것을 의미한다. 또한 마감 시간을 지정하는 것은 일정 시간 내에 처리되어야 하는 마감 시간을 의미하며, 이 마감 시간이 될 때까지 종료하지 못한 트랜잭션은 철회된다.

4. Tachyon 질의 처리기

Tachyon의 질의 처리기는 다중 쓰레드 기반 질의 처리가 가능하고, 클래스 계층구조에 따른 질의가 가능하며, 상위 클래스에서 정의된 스키마와 주키 인덱스가 하위 클래스로 상속된다. 또한 각 인스턴스의 OID를 사용한 커서 항해로 죠인(join) 대체함으로써 관계형 모델에서 질의 비용이 가장 큰 죠인 비용을 대폭 줄여 질의 처리 성능을 높일 수 있도록 설계되었다.

멀티 쓰레드 클라이언트-서버 구조를 지원하는 Tachyon은 그림 2와 같이 네트워크로 연결된 시스템 상의 다중 클라이언트의 데이터베이스 접근 요청을 지원한다.

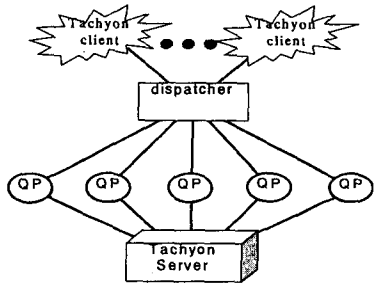

그림 2. 다중 쓰레드 기반 질의 처리기

질의 처리기 수행 절차는 간략하게 정리하여 다음과 같다. 클라이언트로부터 데이터 접근 요청이 오면 Tachyon server가 action 단위로 질의 처리기에게 수행을 요청하게 되고, 요청을 받은 질의 처리기는 질의 구문을 분석하고, 오류가 없을 경우 최적화 정책을 수립한 후 최적화 전략에 따라 질의를 수행하여 결과를 클라이언트에게 전달한다.

5. 객체 관계형 질의 사용 예

본 장에서는 Tachyon에서 제공되는 객체 관계형 질의의 사용 예를 보인다. 그림 3은 이동통신시스템에서 단말기인 Phone과 가입자인 Subscriber와 Cell간의 관계를 모델링한 그림이다. 그림에서 한 Subscriber는 여러 대의 Phone을 소유할 수 있고, 하나의 Cell 내에는 여러 대의 Phone이 위치할 수 있고, 또한 한 대의 Phone은 반드시 한 Subscriber에게 속하고 한 시점에 한 곳의 Cell에만 위치할 수 있는 관계를 보인다. 그림 4는 Phone, Subscriber, Cell을 클래스로 모델링하여 이들의 속성을 정의하고 Tachyon이 제공하는 OID 데이터 타입을 이용하여 클래스간의 관계를 표시한 클래스 스키마이고, 그림 5는 각 클래스들이 가질 수 있는 인스턴스들의 예를 보인 것이다.

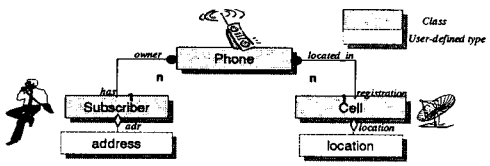

그림 3. 클래스 모델링

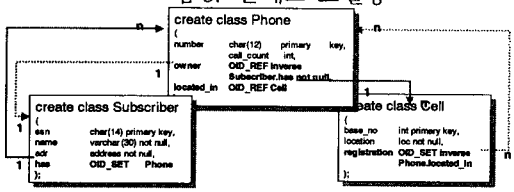

그림 4. 클래스 스키마

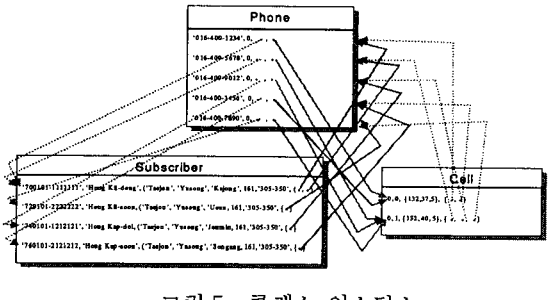

그림 5. 클래스 인스턴스

이와 같은 클래스 관계에서 그림 6 에서와 같이 " base_no가 1 인 Cell 에 위치한 Phone 들의 번호와 그 Phone의 소유자의 주민등록번호와 이름을 출력하라" 는 질의가 요청될 경우, 그림 6의 (a)에서와 같이 관계형 DBMS 에서는 먼저 Cell 과 Phone 을 죠인하여 새로운 릴레이션을 만들고, 다시 이 릴레이션과 Subscriber 릴레이션을 죠인(join)해야 결과를 얻을 수 있는 반면, 그림 5(b)에서와 같이 Tachyon 에서는 물리적 릴레이션의 생성없이 질의 내에서 경로식 (path expression)[7 -8]을 이용하여 Cell 로부터 Phone으로 다시 Phone으로부터 Subscriber로 항해하듯 따라가면 결과를 얻을 수 있다. 경로식을 통하여 각 클래스의 인스턴스가 가지고 있는 관계있는 클래스로의 OID를 따라가면 클래스간 항해 질의가 가능하다. 그 결과 새로운 릴레이션 생성으로 인한 시간적 부담을 줄이고 메모리 사용에 보다 융통성을 가진다.

```
select P.number, S.ssn, S.name
from Cell C, Phone P, Subscriber S
where C.base_no = 1 and C.base_no=P.located_in and
P.owner=S.has;
```

(a) 관계형 DBMS에서의 질의

```
select registration ->number, registration ->owner->ssn,
registration->owner->name
from Cell
where base_no = 1;
```

(b) Tachyon 에서의 질의

그림 6. 죠인 질의 비교

이는 관계형 모델에서 가장 큰 비용이 소요되며, 가장 빈번하게 사용되는 명령어 중 하나인 죠인 명령의 비용을 줄일 수 있으며, 경로식을 통하여 질의도 간단해 지는 장점이 있다. 또한 이러한 클래스간의 관계는 하위 클래스에도 상속되어 상위 클래스가 가지는 관계를 하위 클래스도 가지게 된다. 상위 클래스에 대한 질의는 SQL 문법에 따라 자신의 클래스에 대해서만 질의가 가능하기도 하고, 자신을 포함한 하위 클래스까지도 질의의 범위에 속할 수 있도록 SQL을 정의하였다.

6. 결론

엔진은 관계형 모델을 지원하고 사용자 인터페이스에서 객체 지향 특성을 지원하는 일부 객체 관계형 주기억장치 상주 DBMS와는 달리 Tachyon은 엔진이 객체 관계형 모델에 적합하도록 개발되었기 때문에, 객체 관계형 질의를 효율적으로 처리할 수 있다. 또한 데이터베이스가 주기억장치에 상주하므로 디스크 기반 DBMS에 비해 디스크 I/O를 줄일 수 있으며, 데이터 접근 알고리즘들이 주기억장치 접근을 기본으로 고안되었기 때문에 데이터 접근 성능을 높일 수 있다. 또한 Tachyon은 상속 관계에 있는 클래스에 대한 질의 및 1:1, 1:n 또는 m:n관계의 두 클래스에 대한 질의를 효율적으로 수행할 수 있는 SQL을 지원하고, OID를 이용한 커서 항해로 데이터 검색의 성능을 높일 수 있다.

예상되는 주요 활용 분야로는 실시간 데이터 처리와 고신뢰성이 요구되는 초고속 통신시스템 분야, 전자상거래/웹응용의 검색 효율을 증대시키기 위한 front-end DBMS, 대형 시소러스 구축과 같이 복잡한 질의와 데이터 모델링이 요구되는 응용 등을 꼽을 수 있다.

앞으로 Tachyon은 사용자 정의 메소드(method)와 같은 보다 진보된 객체 지향 질의와 다중 언어 지원을 위해 개방형 접근 API를 제공할 예정이며, 사용자의 편의를 위하여 객체지향 스키마를 모델링 할 수 있는 도구를 지원할 예정이다.

7. 참고 문헌

[1]. Yong-ik Yoon, et.al., "Scalable Distributed Real -time Database Management for Switching System," ISS 97, pp539-545, Sep. 1997.

[2]. TimesTen Performance Software, *In-Memory Data Management Technical White Paper*, White paper, TimesTen Performance Software.

[3]. C.J. Date and Hugh Darwen, " A Guide to THE SQL STANDARD," 4th Ed., Addison Wesley, 1997.

[4]. Polyhedra Plc., *Bloor Research Overview of Polyhedra* , White Paper, Polyhedra Plc.

[5]. Centura Software Co., *Raima Database Manager++ In Real-time and Embedded Systems* , White Paper, Centura Software Co..

[6]. Michael Stonebreaker and Paul Brown , " Object-Relational DBMSs : Tracking the Next Great Wave, " San Francisco: Morgan Kaufmann Publishers, 1999.

[7]. Cynthia Maro Saracco , " Universal Database Management: A Guide to Object/Relational ," San Francisco: Morgan Kaufmann Publishers, 1998.

[8]. Clement T. Yu; Weiyi Meng. , " Principles of Database Query Processing for Advanced Applications ," San Francisco: Morgan Kaufmann Publishers, 1998.