Regular paper http://dx.doi.org/10.13067/JKIECS.2014.9.9.1005

데이터베이스 SQL 강의 효율 향상을 위한 커리큘럼

최세일*

A Curriculum to Improve the Lecture of Database SQL

Se-ill Choi*

요 약

본 논문은 데이터베이스 SQL 교육 효과를 높이는 방법에 대하여 연구하였다. 실무에서 요구되는 다양한 데이터베이스 응용 사례를 프로그래밍 할 수 있도록 학부 학생들에게 SQL을 교육하기 에는 한정된 교육시간이 부족할 수밖에 없다. 따라서 학부 학생들에게 적은 량의 강의 시간에도 불구하고 실무적 응용이 가능한 수준의 SQL 능력을 배양할 수 있도록 강의 커리큘럼을 개발하는 것이 데이터베이스 교육에서 대단히 중요하다. 본 논문은 다양한 데이터베이스 응용 사례를 유형화하고, 그 유형에 대응하는 SQL 패턴을 제안하였다. SQL 패턴의 수가 몇 개 되지 않음으로 학부 학생들에게 효과적인 SQL 교육이 가능할 것으로 생각된다.

ABSTRACT

This paper researched to improve the educational achievements in Database SQL education. In practice, it is not possible to taste all the cases required at database field within the given class times. Therefore we analyzed out the key issues that students must learn for database education, and provided SQL patterns matching to the issues. To validate the effectiveness of this approach, we programmed practical requirements with those SQL patterns under a hypothetical database.

키워드

Database, SQL, Lecture, Curriculum
데이터베이스, SQL, 강의, 커리큘럼

Ⅰ. 서 론

데이터베이스 교과목은 컴퓨터를 전공하는 학생들이라면 반드시 이수해야 할 핵심 교과목 중의 하나이다. 다른 핵심 교과목들도 모두 마찬가지겠지만 학생들에게 가르쳐야 할 내용은 많고 시간은 정해져 있는 상황에서 어떻게 하면 효과적으로 데이터베이스 지식을 가르쳐야 좋은지 데이터베이스 교육자라면 누구나 한번 고민해 보았을 것이다. 물론 지능 수준이 높고 학업에 대한 의욕이 높은 학생들이라면 많은 내용을 가르친다고 해도 자율 학습을 통하여 스스로 배워 나가겠지만 대부분의 일반수준 학생들은 범위가 많아지면 학습 역효과가 나기 때문에 적절한 규모의 학습량을 교육하는 것이 바람직하다.

본 논문에서는 학생들에게 데이터베이스 언어 인 SQL을 교육할 때, 효과적인 교육 방법에 대하여 연구하였다. 대부분의 데이터베이스 교육자들은 SQL을 교육할 때, 교재에 따라 문법이나 기본 사례를 교육하고 추가적인 내용은 과제를 통하여 습득하도록 유도하는 전형적인 교육방법을 따른다. 그러나 이러한 교육 방법은 학생들의 수준이나 시대의 발전에 따라 같이 변하여야 할 것으로 생각된다. 그렇게 생각하는 이

* 교신저자(corresponding author) : 호남대학교 컴퓨터공학과(sichoi@honam.ac.kr)
접수일자 : 2014. 07. 01 심사(수정)일자 : 2014. 08. 21 게재확정일자 : 2014. 09. 19

유는 기술의 발전과 사회의 진보에 따라 과목마다 가르쳐야 할 내용은 점점 많아지고 있고, 학생 개개인들도 관심을 가지는 분야들이 점점 더 많아지고 있음으로 인해, 학생들이 예전처럼 최선을 다하여 모든 과목을 공부하기가 쉽지 않기 때문이다. 따라서 지적 능력이 뛰어난 우수한 학생들에게도 적정량의 지식을 적절한 방식으로 교육하는 것이 바람직할 것으로 생각되는데, 하물며 보통 지적 능력을 가진 학생들을 교육시키기 위해서는 반드시 가르쳐야 할 지식수준 및 지식의 양을 조절하는 것이 필요할 것으로 생각된다.

현재 일반적인 데이터베이스 교육 방법은 교재의 순서에 따라 교재의 내용을 교육하는 방법으로 진행되고 있다. 따라서 현재 시중의 데이터베이스 교재를 분석해 보았는데 대부분의 교재들은 지식의 나열 형식으로 교재가 구성되어 있고[1-3], SQL 분야도 역시 SQL 언어의 수학적 배경과 SQL 문법에 대한 지식 나열 위주로 구성되어 있다. 따라서 이러한 교재로 교육을 하게 되면, SQL에 대한 지식은 충분히 배우겠지만 실무에서 SQL 프로그래밍을 위해서 필요한 업무 콘텍스트(Context)별 SQL 언어의 응용 방법에 대한 이해가 부족하여 실무능력을 배우기 위해서는 추가적으로 많은 훈련 시간이 필요하게 될 것으로 추정된다.

본 논문은 이러한 문제점을 개선하기 위하여 짧은 시간 내에 실무적으로 활용 가능한 SQL 언어를 교육시키기 위한 강의 커리큘럼을 제안하고자 한다. 본 논문의 핵심은 데이터 관리 업무 콘텍스트 유형을 먼저 분류하고, 분류한 콘텍스트 유형에 따라 SQL을 패턴화 한다. 이렇게 가르치고자 하는 기술내용의 범위를 체계화하고, 기술내용 별로 해당 SQL 패턴을 대응시키는 방법을 통하면, 짧은 시간 내에 실무적으로 활용 가능한 SQL 프로그래밍 능력을 교육할 수 있을 것으로 생각된다.

2장에서는 현재 시중에 널리 사용되는 데이터베이스 교재를 검토해보고, 3장에서 교육해야 할 기술내용과 해당 기술 내용별 SQL패턴을 제안한다. 4장에서는 실무에서 요구되는 가상의 요구사항을 활용하여 그 타당성을 검토해보고, 5장에서 결론을 맺는다.

II. 관련 연구

위에서 언급한 것처럼 시중에 판매되는 대부분의 데이터베이스 교재들은 SQL 언어에 대한 수학적 배경에 관한 지식이나, SQL 문법에 대한 지식을 나열하는 방식을 취하고 있다[1-3]. 물론 이러한 교재들이 이론서이고 데이터베이스 개념에 관한 지식을 가르치는데 초점을 두고 있음으로 인해 그럴 수밖에 없는 입장이겠지만, 학생들에게 실무능력을 배양시키는 데에는 한계가 있다. 이러한 현상은 비단 데이터베이스 교과목뿐만 아니라 다른 여타 교과목에서도 같은 상황에 처해 있는 것이 현재 대학 교육의 현실이다.

이러한 연유로 인하여 대학에서는 실무능력을 배양시키기 위하여 프로젝트 형 교과목의 개설, 프로젝트 형 과제의 부과, 산업체 현장 인턴십의 적용 등, 실무와 관련된 교과목을 추가적으로 개설하거나 교과과정 개편을 통하여 학생들에게 추가적인 학습을 요구하고 있다. 그러나 앞에서 언급한 것처럼 학생들에게 많은 양의 학습을 요구하는 것이 학습능력을 증진시키기 보다는 오히려 역효과가 날 수도 있음을 추정해 볼 수도 있다.

따라서 대학에서 교과과정 커리큘럼을 개발할 때는 학생들에게 적정량의 교육 범위 내에서 실무능력을 함께 배양 시킬 수 있는 방법을 고려해야 할 것으로 생각된다.

III. SQL 패턴의 제안

본 장에서는 데이터베이스를 다루는데 필요한 기술내용을 분류하고 그 기술 내용별 SQL 패턴을 제안한다. 그렇게 함으로서 배워야 할 SQL 내용이 많지 않음에도 불구하고 실무적으로 활용 가능한 SQL 지식을 갖추게 될 수 있다.

3.1 데이터베이스를 다루기 위해서 필요한 기술

데이터베이스를 다루기 위해서 SQL 프로그래머가 알아야 할 기술 내용은 크게 다음 4가지로 분류할 수 있다.

(1) 상호 관련된 이웃 데이터의 탐색 방법 : 관계형 데이터베이스는 ER 다이어그램을 이용하여 데이터 문맥 구조를 표현한다. 어떤 데이터들은 이웃하여 연결되어 있고, 어떤 데이터들은 서로 관련이 있기는 하지만 여러 단계의 이웃을 걸쳐 연결이 되어 있다. 관계형 데이터베이스에서 상호 관련된 데이터를 다루

기 위해서는 이러한 데이터 연결 구조를 이해하고 연결 구조를 따라 탐색해 가는 방법을 알아야 한다[4].

(2) 대용량 데이터의 동시 검색을 위한 테이블 핸들링 방법 : 관계형 데이터베이스에서 또 하나의 특징은 관계대수로 설명되는 조건 검색 시, 조건에 맞는 다수의 데이터가 동시에 검색되는 것이다[5]. 이러한 대용량 데이터를 검색하기 위하여 테이블의 합성에 대한 다양한 기술을 충분히 알아야 한다.

(3) 복잡한 검색 논리를 구현하기 위한 언어의 블록 구조화 방법 : SQL 언어도 다른 언어와 마찬가지로 언어적 성격을 띠고 있기 때문에 언어를 구조적으로 잘 사용하기 위해서는 언어를 블록 구조화 하는 방법을 알아야 한다.

(4) 데이터베이스 무결성 보장을 위한 방법 : 데이터베이스 변경을 위한 SQL을 사용 시 반드시 유념해야 할 내용은 데이터베이스 무결성에 관한 사항이다[6]. 물론 데이터베이스 설계 단계에서부터 데이터베이스 무결성을 보장하기 위하여 정규화를 충분히 검토하였다고 하더라도 이것은 SQL 프로그래머가 데이터베이스 무결성을 해치지 않는 조건에서만 무결성이 보장된다. 따라서 실제적으로 무결성에 가장 큰 영향을 주는 것은 SQL 프로그램의 적용 시에 발생하게 됨으로 데이터베이스 변경을 위한 SQL 사용 시에는 반드시 지켜야 할 규칙 중의 하나라고 볼 수 있다.

3.2 데이터구조 네비게이션 능력 배양

학생들이 습득해야 할 중요한 이슈는 데이터베이스 테이블이 서로 관계를 가지고 연결되어 있을 때 이러한 데이터테이블들을 연속하여 탐색해가는 능력을 갖추는 것이다. 이것은 멀티미디어 하이퍼텍스트와 같은 기능과 유사한데[7], 연결된 테이블들을 하나씩 계속 추적해 가는 방법으로 검색을 진행하는 기술이다. 이러한 기술을 습득하기 위해서는 다음 그림 1 패턴을 숙지하면 될 것으로 생각된다.

```
select
from
where predicate(complicated);
```

그림 1. 복합 조건식 패턴
Fig. 1 Complicated logic pattern

3.3 데이터 테이블 합성 능력 배양

데이터 검색에 있어서 또 다른 중요한 이슈는 관계형 대수에 따라 조건에 합당한 대규모 데이터를 동시에 검색하는 것이다. 대규모 데이터를 다루기 위해서는 테이블의 합성과 분할을 수행 할 수 있는 능력을 갖추어야 하는데, 이러한 기능은 SQL의 조인문을 이용하여 실현 가능하다. 조인문은 조인 조건에 따라 여러 가지의 조인문이 필요할 수도 있지만, 실제 실무에서 조인문을 활용하기 위해서는 모든 조인문이 필요한 것은 아니다. 따라서 대용량 데이터 검색을 위하여 필요한 SQL 패턴은 다음 그림 2와 같다.

```
select
from tableA inner join tableB on A.attribute = B.attribute
where predicate(simple);
```

그림 2. 조인문 패턴
Fig. 2 Join statement pattern

3.4 데이터베이스 언어 구조화 능력 배양

모든 언어가 마찬가지겠지만 SQL도 하나의 언어임으로 언어를 조직적으로 사용하기 위해서는 언어를 조직화 시키는 연습을 해야 한다. 그러나 다른 일반 컴퓨터 언어와 달리 SQL 언어는 구조화를 위한 언어 도구가 충분하지 않고 구조화를 시켜야 할 정도로 프로그램도 길지도 않기 때문에 구조화에 그렇게 신경 쓸 필요 까지는 없다. 하지만 복잡한 논리를 요구하는 검색식의 경우, 블록화 된 SQL을 사용하면 SQL을 이해하는데 훨씬 간편할 수 있다[9]. 다른 여러 가지 블록화 방법이 있을 수 있지만 가장 보편적으로 활용하기 위하여 다음 그림 3과 같은 Nested SQL 패턴을 제안한다.

```
select
from
where predicate(simple) in select
        from
        where predicate(simple) select
                from
                  where predicate(simple);
```

그림 3. 내장 SQL 패턴
Fig. 3 Nested-SQL pattern

3.5 데이터베이스 무결성 요인에 대한 이해 배양

데이터베이스 설계 단계에서 무결성에 대한 이야기를 많이 듣게 되는데 실제 무결성에 대한 경험을 겪지 않은 상태에서 무결성 이야기는 그렇게 피부에 와 닫지 않는다[10]. 이러한 문제는 실제 SQL언어를 배울 때도 언어의 문법을 익히기에 급급하여 SQL 적용 시에 발생할 수 있는 무결성에 대하여 별로 감지하지 못하고 사용하게 된다. 따라서 SQL을 사용하여 데이터베이스를 변경하고자 할 때는 패턴화 하여 변경패턴을 만들어 이해하는 것이 효과적이라고 생각된다.

3.5.1 데이터 추가를 위한 언어 패턴

데이터 추가를 위해서는 해당 테이블뿐 만 아니라 연결된 이웃 테이블에 관련 데이터가 이미 존재하는가 확인 후 데이터를 추가해야 한다. 그렇지 않은 경우 어떠한 연유로 관련되어 있었던 데이터와 새로 추가된 데이터가 중복되어 관련되는 상황이 발생한다. 그림 4는 데이터 추가를 위한 SQL 패턴이다.

```
if conditions for integration <> select
                    from tables
                    where predicate
then insert
      into table-name
      values(,,,);
```

그림 4. 데이터 추가를 위한 무결성 검사 패턴
Fig. 4 Data integration check pattern for data insert

3.5.2 데이터 갱신을 위한 언어 패턴

데이터 갱신을 위해서는 먼저 갱신하고자 하는 데이터가 키인가 아닌가를 그림 5의 무결성 조건 검사를 통하여 같이 확인해야 한다. 만일, 키라고 한다면 기존의 연결 고리가 흐트러지게 되고 결국 데이터의 연결 상태가 문제가 되어 무결성에 위배되는 상황이 발생한다.

```
update
set new data
where predicate;
```

그림 5. 데이터 갱신을 위한 무결성 검사 패턴
Fig. 5 Data integration check pattern for data update

3.5.3 데이터 삭제를 위한 언어 패턴

데이터 삭제를 위해서는 연결된 이웃 데이터가 남아 있으면 안 된다. 이러한 경우, 연결된 데이터를 위하여 현재 데이터를 삭제하지 말고 남겨두어야 한다. 그림 6에서 삭제와 관련된 무결성 조건 검사 패턴을 보여주고 있다.

```
if conditions for integration <> select
                    from tables
                    where predicate
then delete
      from tables
      where predicate
```

그림 6. 데이터 삭제를 위한 무결성 검사 패턴
Fig. 6 Data integration check pattern for data delete

IV. SQL 강좌 사례

본 논문에서는 대학 도서관에서 학생들에게 도서 대출을 하는 데이터베이스를 도서관 자체적으로 구축하였다고 가정하고 그림 7, 일반적 도서관 업무를 처리하고자 할 때, 위에서 언급한 기술 내용을 습득한 학생이 도서관 업무 처리 요구사항을 프로그래밍 가능할 것인가 검증해 본다.

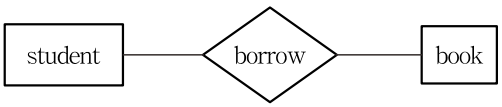

그림 7. 도서관 데이터베이스 ER 다이어그램
Fig. 7 ER diagram for the library database

4.1 특정한 학생이 빌려본 도서의 목록을 검색

관계형 데이터베이스에 주로 발생하는 트랜잭션으로 특정한 학생, 예를 들면 “홍길동”이 대출해간 도서를 모두 검색하는 트랜잭션을 프로그램 하고자 할 때, 이러한 검색은 3개의 서로 연결된 테이블을 모두 검색해야 한다. 그러나 이러한 경우, 시간이 많이 소요되는 조인문까지 사용할 필요는 없고, 단순히 3개의 테이블을 탐색하는 형식으로 프로그래밍하는 것이 효과적이다. 3개의 테이블을 탐색하는 패턴을 활용하여

프로그래밍하면 다음 그림 8과 같다.

```
select book.title
from student, borrow, book
where student.name="Hong Gildong"
        and student.number=borrow.st_no
        and borrow.book_no = book.number;
```

그림 8. 복합 조건식 패턴 사례
Fig. 8 A complicated logic pattern example

4.2 다수의 도서를 대출해간 학생들의 명단

예를 들어 "데이터베이스"라는 제목을 가진 도서가 도서관에 20권 이상 비치되어 있다고 가정하고, 그 도서들을 빌려간 학생들을 모두 검색하고자 할 때, 각각의 도서를 검색하고 그 도서를 빌려간 학생을 한 명씩, 한 명씩 검색하는 것 보다는 테이블들을 통합하여 검색하는 것이 더 효과적이다. 이와 같이 조건에 검색되는 데이터가 복수 개 존재할 때, 검색하는 방법으로 활용할 수 있는 문법이 조인문이다. 조인문을 사용하는 방법은 다음 그림 9와 같다.

```
select student.name
from (student inner join borrow
      on student.number=borrow.st_no)
      inner join book
      on borrow.book_no=book.number
where book.title="database";
```

그림 9. 조인문 패턴 사례
Fig. 9 A join statement pattern example

4.3 복잡한 검색조건의 프로그래밍 방법

예를 들어 "철학"책을 빌린 학생들이 주로 빌리는 책들은 무슨 책인가와 같이 복잡한 검색을 수행하고자 할 때, 주어진 조건 사이에 선후 관계가 존재하기도 하고, 조건 자체도 2개 이상으로 복잡함으로 인해 복잡한 조건을 하나의 문장으로 나열하기가 쉽지 않다. 이러한 경우 복잡한 조건을 기술하는 방법으로 블록화 된 프로그래밍이 필요한데, 이러한 조건을 만족 시키기 위하여 필요한 것이 바로 Nested SQL이다. Nested SQL을 사용하여 프로그래밍하면 다음 그림 10과 같다.

```
select book.title
from (student inner join borrow
        on student.number=borrow.st_no)
        inner join book
        on borrow.book_no=book.number
where student.name in select student.name
from (student inner join borrow
        on student.number=borrow.st_no)
        inner join book
        on borrow.book_no=book.number
where book.title="philosophy";
```

그림 10. 내장 SQL 패턴 사례
Fig. 10 A nested-SQL pattern example

4.4 데이터의 삭제, 갱신, 추가 프로그래밍

특정한 학생의 졸업, 혹은 특정한 서적의 분실 등으로 인하여 학생 데이터를 삭제하거나 혹은 특정한 서적 데이터를 삭제하는 경우, 이러한 데이터와 관련된 대출 데이터는 믿을 수 없는 상태가 된다. 또 신입생 데이터를 추가하거나 아니면 신규도서의 입력을 하는 과정에서 데이터를 삽입할 때 잘못된 키 정보를 삽입하게 되면, 신규 데이터 임에도 불구하고 이미 대출된 적이 있었던 것으로 데이터 오류가 발생할 수도 있다. 이와 같이 데이터의 삭제, 갱신, 변경 시에는 그림 11과 같이 반드시 연결된 데이터에 대한 확인이 선행되어야 한다[8].

```
if null <> select name
                from student
                where number="3003"
then insert
      into student
      values ("3003", "Hong Gildong",
      "010-0120-2345");
```

그림 11. 데이터 추가를 위한 무결성 검사 패턴
Fig. 11 A Data integration check pattern for data insert example

V. 결 론

본 논문에서는 데이터베이스를 공부하는 학생들에

게 효과적으로 데이터베이스 SQL 강의를 수행하기 위하여 필요한 커리큘럼에 대하여 알아보았다. 주어진 학사일정 시간 내에 실무적 능력을 갖춘 SQL 프로그래머를 양성하기 위하여 실무적 상황에 알맞도록 SQL을 패턴화하여 제안하였다. 이러한 SQL 패턴을 배웠을 때 실제로 실무에서 응용 가능한지 확인하기 위하여 도서관 업무라는 가상의 실무조건을 설정하고 제안한 SQL 패턴을 이용하여 도서관 업무를 처리하여 보았다. 본 연구를 통하여 제안한 SQL 커리큘럼이 단지 몇 개의 SQL 패턴을 이용하여 다양한 실무 프로그래밍이 가능함을 보임으로서, 제안한 커리큘럼의 효용성을 확인하였다.

따라서 제안한 커리큘럼을 이용하여 학생들에게 데이터베이스 SQL 강의를 수행할 경우, 학습량이 대폭 축소됨에 따라 데이터베이스 교육 효과를 크게 개선시킬 수 있을 것으로 예상된다. 그러나 현재 제안한 커리큘럼이 실제 학생들의 강의에 직접 사용되고 있음에도 불구하고, 이 커리큘럼이 예전의 커리큘럼이라든가 여타 다른 커리큘럼에 비하여 얼마나 더 효율이 좋은가는 검토해본 적이 없다. 따라서 본 연구를 좀 더 현실화시키기 위해서는 제안한 커리큘럼을 실제 학생들에게 적용시킨 사례의 효율 분석할 필요가 남아있다.

References

[1] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database System Concepts 6th Edition.* New York : McGraw-Hills, 2010.

[2] R. Ramakrishnan and J. Gehrke, *Database Management Systems 3rd Edition.* New York : McGraw-Hills, 2004.

[3] S. Lee, *Database Concepts.* Seoul : Jungik Publishing Co., 2010.

[4] S. Choi, "A Curriculum Design for the Data Modeling in Database Subject," *Proc. of Korean Institue of Smart Media,* vol. 3, no. 1, Daegu, Korea, Apr. 2014, pp. 230-233.

[5] C. Kim "A Range Query Method using Index in Large-scale Database Systems," *J. of the Korea Institute of Electronic Communication Science,* vol. 7, no. 5, 2012, pp. 1095-1101.

[6] S. Choi, "An Approach to Teach the Data Table Normalization," *Proc. of the Korea Institute of Electrononic Communication Sciences,* vol. 8 no. 1, Busan, Korea, June 2014, pp. 432-436.

[7] G. Gwak, "A Feature Analysis for a Transition to NoSQL Database," Master's Thesis, *Hanbat University,* 2013.

[8] W. Hur, "A Study on the Design of Scalable NoSQL for System Integration Service," Master's Thesis, *Seoul City University,* 2012.

[9] J. Rodzvilla, "A Review of Learning SQL," *J. of Web Librarianship,* vol. 4, no. 4, 2010, pp. 465-466.

[10] G. Park, "A Design and Implementation of the SQL Performance Analyzer to Improve Database Performances," Master's Thesis, *Joongang University,* 2010.

저자 소개

최세일(Se-ill Choi)

1984년 한양대학교 전자공학과 졸업(공학사)

1989년 플로리다공과대학교 대학원 전산학과 졸업(공학석사)

2002년 (호)모나쉬대학교 대학원 전산학과 졸업(공학박사)

1993년~현재 호남대학교 컴퓨터공학과 교수

1990년~1993 삼성전자 컴퓨터부문 선임연구원

1984년~1989 LG전자 컴퓨터사업부 사원

※ 관심분야 : 소프트웨어공학, 데이터베이스, 전자상거래