

웹 기반 데이터베이스 응용의 자동 생성 (Automatic Generation of Web-based Database Applications)

음 두 현⁺ 유 은 자^{**}
(Doohun Eum) (Eunja Yoo)

요약 오늘날 우리는 HTML 폼을 통해 지역 데이터베이스뿐 아니라 원격 데이터베이스도 쉽게 접근할 수 있다. HTML 폼으로 데이터베이스 응용을 구현하는 것이 쉽기는 하나 아직 HTML 폼과 CGI 프로그램은 코딩되어야만 한다. 본 논문에서는 웹 기반 데이터베이스 응용을 자동 생성하는 도구를 소개하는데, 이 도구는 응용 데이터베이스 구축을 위한 SQL 문장들, 사용자 인터페이스로 사용되는 폼들, 그리고 그 폼들을 통해 이루어진 질의를 처리하는 자바 서블릿을 생성한다. 생성될 테이블들과 폼들은 응용을 위한 클래스 다이어그램으로부터 결정된다. 따라서 WebSiteGen으로 명명한 우리의 소프트웨어 도구는 웹 기반 데이터베이스 응용의 구현을 자동화한다.

키워드 : 웹 기반 데이터베이스 응용, 데이터베이스 스키마, HTML 폼, 자바 서블릿, 소프트웨어 도구

Abstract Nowadays, we can easily access a remote database as well as a local database with HTML forms. Although implementing a database application with HTML forms is much simpler than implementing it with a proprietary graphical user interface system, HTML forms and CGI programs still must be coded. We implemented a software tool that automatically generates the SQL statements that create a database for an application, the forms that are used as a user interface, and the Java servlets that retrieve the data requested through the forms. The database tables to be created and the forms to be generated are determined by the class diagram for the application. Our software tool, which we call WebSiteGen, thus simplifies the implementation of a Web-based database application.

Key words : Web-based database application, database schema, HTML forms, Java servlets, software tool

1. 서론

전자 상거래(Electronic Commerce)는 인터넷상에서 이루어지는 활발한 응용 분야이다. 전자상거래 사이트들은 매년 큰 폭으로 증가하여, 책 구입에서 주식 투자에 이르기까지 다양한 형태로 우리들의 일상 생활을 변화시키고 있다. 이러한 전자 상거래 응용의 주된 요소는 웹 기반의 데이터베이스 응용이다.

본 논문에서는 웹 기반 데이터베이스 응용의 구현을 자동화하는 소프트웨어 도구, WebSiteGen을 소개한다. WebSiteGen은 응용 데이터베이스 구축을 위한 SQL

문장들, 사용자 인터페이스로 사용되는 폼들, 그리고 그 폼들을 통해 질의된 정보를 검색하는 자바 서블릿을 자동 생성한다. 생성될 SQL 문장들과 폼들은 데이터베이스 응용을 위한 UML(Unified Modeling Language)[1] 클래스 다이어그램으로부터 결정된다.

WebSiteGen이 생성하는 각 폼은 하나의 주 영역(main area)과 내포된 테이블(nested table)들로 구성된다. 주 영역에는 닷 개체(anchor entity)라 불리는 현재 관심의 대상인 개체에 대한 정보와 닷 개체와 일대일 또는 다대일 관계 타입으로 연관된 개체들에 대한 정보를 표시한다. 이러한 개체를 닷 개체에 대한 첨부 개체(appended entity)라 한다. 내포된 테이블에는 닷 개체와 일대다 또는 다대다 관계 타입으로 연관된 개체들에 대한 정보를 표시한다. 내포된 테이블에 표시되는 개체를 닷 개체에 대한 확장 개체(expanded entity)라 한다. 폼은 여러 개의 내포된 테이블들을 포함할 수 있으나 내포는 한 단계만 허용한다. 즉 내포된 테이블이 다른

· 본 논문은 2001년도 덕성여자대학교 교내연구비 지원으로 수행되었음.

· 본 논문은 KISTEP 2001년도 여자대학교 연구 기반 확충 사업 연구비 지원으로 수행되었음. 과제번호 : 00-N6-05-01-A-03

* 종신회원 : 덕성여자대학교 전산학과 교수

dheum@duksung.ac.kr

** 학생회원 : 덕성여자대학교 전산 및 정보통신 대학원

bitty@namhae.duksung.ac.kr

논문접수 : 2001년 5월 10일

심사완료 : 2002년 3월 7일

테이블을 내포할 수는 없다.

클래스 다이어그램은 응용에 필요한 클래스들과 그들 간의 관계 타입들을 직관적으로 나타낸다. 클래스 다이어그램으로부터 필요한 데이터베이스 구축을 위한 SQL 문장들을 생성해 내는 규칙들은 이미 잘 알려져 있다. 데이터베이스에 저장된 데이터는 주로 폼을 통해 검색되고 갱신된다. 폼은 사용하기 쉬운 인터페이스를 제공하므로 널리 사용되고 있다. 이러한 데이터베이스 및 폼 생성 기능들을 제공하는 잘 알려진 소프트웨어로는 Oracle Design 2000[2]과 Microsoft Access 2000[3]이 있다. 특히 Microsoft Access 2000이 제공하는 폼은 네트워크상에 분산되어 있는 데이터베이스도 접근할 수 있다. 그러나 Oracle Design 2000으로 생성할 수 있는 폼은 한 테이블의 필드들만을 포함한다. 즉 첨부 및 확장 개체를 지원하지 않는다. 또한 Microsoft Access 2000의 폼은 여러 테이블들의 필드들을 포함할 수 있으나 내포된 테이블(확장 개체)을 지원하지 않는다.

WebSiteGen과 같이 웹 응용의 자동 생성을 주 목적으로 하는 상용 제품으로는 CodeCharge[4], 1ClickDBPro[5], ASP.NET[6] 등이 있다. CodeCharge와 1ClickDBPro는 이미 구축된 데이터베이스를 사용하며 테이블에 접근하여 질의를 처리하는 코드는 개발자가 직접 입력해야 한다. 또한 질의를 처리할 폼은 자동 생성되기는 하나 한 폼에는 닷 개체에 대한 정보만을 표시할 수 있다. ASP.NET은 Microsoft.NET 프레임워크 중 하나로, 데이터베이스를 자동 생성할 수 있으며 질의 처리 부분은 선택적으로 자동 생성하거나 직접 입력하는 방식을 사용할 수 있다. 그러나 한 폼에는 닷 개체와 첨부 개체에 대한 정보만 표시할 수 있다.

본 논문에서는 데이터베이스를 구축하기 위해 필요한 SQL 문장들을 생성하고 그 데이터베이스에 접근할 때 필요한 폼들을 생성하기 위해 지켜야 할 규칙들도 제안한다. 특히 각 폼으로 처리할 수 있는 개체 타입들과 관계 타입들을 결정하는 규칙들과 주어진 폼들로 모든 개체들을 처리할 수 있는 질의 검사하는 규칙들을 제시한다. create table SQL 문장에 외래 키를 통한 무결성 제약조건을 추가하는 방법과 생성된 폼들의 완전성을 검사하는 방법도 논한다. 클래스 다이어그램으로부터 생성된 각 테이블이 모든 가능한 방법으로 조작될 수 있다면 생성된 폼들은 완전하다고 할 수 있다.

2절에서 WebSiteGen의 개요를 설명하고 3절에서는 폼을 자동으로 생성하기 위한 규칙들을 소개한다. 4절에서는 상용 웹 사이트 자동 생성 도구들과 WebSiteGen의 기능을 비교함으로써 WebSiteGen의 장점을 보인다.

5절에서는 WebSiteGen의 대략적인 구현을 소개한다. 마지막 절에서는 본 논문의 결론과 앞으로의 연구 방향을 정리한다.

2. WebSiteGen의 개요

본 절에서는 웹 기반 데이터베이스 응용을 자동으로 생성하는 도구인 WebSiteGen의 개요를 소개한다. 그림 1은 WebSiteGen의 구조이다. WebSiteGen은 데이터베이스 응용을 위한 클래스 다이어그램으로부터, 응용에서 필요한 데이터를 저장하는 관계형 테이블들, 사용자 인터페이스로 사용될 HTML 폼들, 그리고 HTML 폼을 통한 질의를 처리할 자바 서버릿을 자동 생성한다.

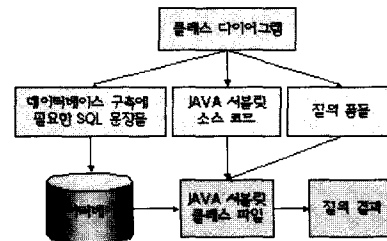


그림 1 WebSiteGen의 구조

WebSiteGen의 기능을 설명하기 위해, 본 논문에서는 간단한 전자 상거래 응용(e-order company)을 예로 들어, 그 클래스 다이어그램을 먼저 보이고 이로부터 생성될 SQL 문장들, 폼들, 그리고 자바 서버릿을 보인다.

2.1 클래스 다이어그램

클래스 다이어그램은 클래스들과 그들간의 관계를 직관적으로 보여준다. 그림 2는 WebSiteGen을 이용해 작성한 e-order company 응용의 클래스 다이어그램이다.

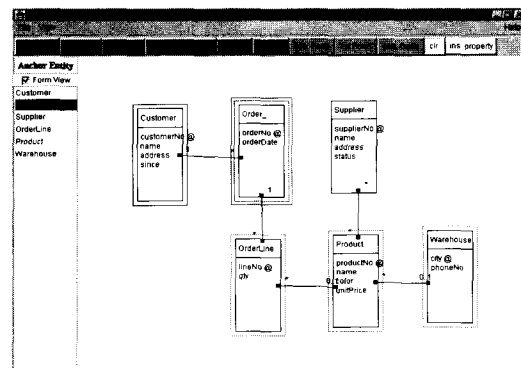


그림 2 e-order company 응용의 클래스 다이어그램

e-order company의 클래스 다이어그램은 다음을 가 정한다.

1. 회사(Company)는 각 고객(Customer)의 이름(name), 주소(address), 그리고 첫 주문 날짜(since)로 고객들을 관리한다. 각 고객에게는 고객 번호(customerNo)가 기본 키로 주어지고 고객은 여러 개의 주문을 요청할 수 있다.

2. 고객이 요청한 각 주문(Order)은 여러 개의 주문항 목(OrderLine)들을 포함한다. 각 주문항목에는 한 제품(Product)에 대한 주문량과 주문한 날짜가 기록된다. 각 주문은 주문 번호(orderNo)를 기본 키로 하고, 주문 날짜(orderDate)를 기록하며, 주문 항목은 항목 번호(lineNo)를 기본 키로 하여 주문량(qty)를 기록한다.

3. 각 제품은 제품 번호(productNo), 이름(name), 색 상(color), 그리고 소매가(unitPrice)의 속성을 가지고, 제품 번호가 기본 키이다.

4. 각 제품은 여러 공급자(Supplier)들에 의해 공급될 수 있다. 각 공급자는 여러 종류의 제품들을 공급할 수 있고 공급자 번호(supplierNo)를 기본 키로 하여, 이름(name), 주소(address), 그리고 공급자 상태(status)들이 기록된다.

5. 회사는 여러 개의 창고(Warehouse)들을 소유하며 각 창고는 위치한 도시(city)에 의해 식별되므로 도시가 기본 키이다. 각 창고에는 비축되어 있는 제품의 수량이 기록된다. 각 창고는 여러 종류의 제품들을 비축할 수 있으나 각 제품은 한 창고에만 비축된다.

2.2 무결성 제약조건을 고려한 테이블의 생성

클래스 다이어그램에 표현된 정보를 이용하여 응용 데이터베이스 구축에 필요한 SQL 문장들을 생성할 수 있다. WebSiteGen은 클래스 다이어그램 내의 각 개체 타입에 대해 하나의 테이블을 생성한다. 또한 클래스 다이어그램 내의 두 개체 타입 A와 B 사이의 관계는 다음과 같이 처리된다.

개체 타입 A와 B의 관계가 일대일 또는 일대다일 경우, 개체 타입 B에 대한 테이블은 개체 타입 A에 대한 테이블의 기본 키를 지정하는 외래 키를 포함하고 관계 타입에 대한 테이블은 생성하지 않는다. 관계가 다대다일 경우는, 관계 타입을 테이블 AB로 구현하고 테이블 AB는 개체 타입 A, B에 대한 테이블들의 기본 키를 지정하는 두 개의 외래 키를 포함한다.

참조 무결성 규칙은 레코드를 갱신·삭제 후에 부합 되지 않는 외래 키 값이 발생하는 것을 방지할 수 있다. 부합되지 않는 외래 키 값을 방지하기 위해, restricted, cascade, set null의 세 가지 외래 키 규칙이 사용된다

[7]. 무결성 제약조건은 서로 다른 테이블에 존재하는 레코드들 간의 합법적인 관계를 명시하며 관계 타입의 카디널리티와 응용의 의미로부터 유도해 낼 수 있다.

개체 타입 A와 B 사이의 관계가 일대일 또는 일대다 라고 하자. 개체 타입 A쪽의 카디널리티가 1이면, 개체 타입 B에 대한 테이블의 외래 키 속성에 대한 무결성 제약조건으로 cascade나 restricted가 그림 3과 같이 선택되어야 한다.

A쪽의 카디널리티가 0..1이면, B에 대한 테이블의 외래 키 속성에 대한 무결성 제약조건으로는 cascade, set null, 또는 restricted가 선택될 수 있다. 갱신과 삭제 연 산에 대해 서로 다른 무결성 제약조건이 명시될 수 있다. B쪽의 카디널리티 C는 1, *, 또는 1..* 중 어떤 것 이어도 상관없다.

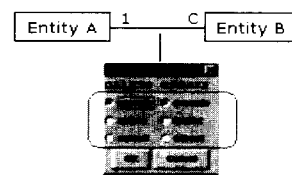


그림 3 일대일 또는 일대다 관계 타입에 대한 무결성 제약조건의 선택

SQL 외래 키 규칙으로는 B쪽의 카디널리티가 1 또는 1..*인 경우의 무결성 제약조건을 개체 타입 A에 대한 테이블에 명시할 수 없다. 그러나 필요하다면 A의 각 인스턴스를 참조하는 B의 인스턴스가 적어도 하나 존재하는지의 여부를 검사하는 트리거를 통해 제약 조건을 구현할 수는 있다.

E-order company 응용에서, WebSiteGen에 의해 생성되는 다음의 SQL 문장은 개체 타입 Order에 대한 테이블 Order를 그림 4와 같이 생성한다. 자동 생성된 create SQL 문장은 선택된 무결성 제약조건을 포함한다.

```
create table Order (
  orderNo    varchar(20) not null,
  orderDate  varchar(20),
  customerNo varchar(20) not null,
  primary key (orderNo),
  foreign key (customerNo)
  references Customer (customerNo)
  on delete cascade
  on update cascade
);
```

그림 4 자동 생성된 create SQL 문장

E-order company 응용에서, 개체 타입 Customer는 개체 타입 Order와 일대다 관계 타입으로 연관된다.

Customer측의 카디널리티가 1이므로 갱신과 삭제 연산에 대한 무결성 제약조건으로 cascade나 restricted가 선택될 수 있으며 이 응용에서는 cascade가 적합하다. 고객 레코드가 삭제될 때, 무결성 제약 조건 "on delete cascade"에 의해 삭제될 고객 레코드의 고객 번호를 가지는 주문 레코드들이 자동적으로 삭제된다. 마찬가지로, Customer 테이블의 고객 레코드의 고객 번호가 새로운 번호로 갱신될 때, "on update cascade"의 무결성 제약조건에 의해 주문 레코드의 고객 번호는 새로운 번호로 자동 갱신된다.

개체 타입 A와 B 사이의 관계 타입이 다대다이면 그림 5와 같이 두 개의 일대다 관계 타입(A와 AB, B와 AB)으로 변환된다. 이때 무결성 제약조건은 앞에 설명한 일대일 또는 일대다 관계 타입의 무결성 제약조건 선택 규칙에 따라 테이블 AB 내의 외래 키들에 명시된다.

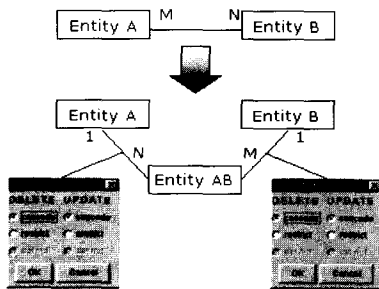


그림 5 다대다 관계 타입을 두 개의 일대다 관계 타입으로 변환

E-order company 응용에서, 개체 타입 Supplier는 개체 타입 Product와 다대다 관계 타입으로 연관되므로 추가적인 SupplierProduct 테이블이 생성된다. 이때 SupplierProduct에 대한 무결성 제약조건은 cascade나 restricted가 선택될 수 있다.

2.3 품의 생성

WebSiteGen이 생성하는 품은 그림 6과 같이 구성된다. 주 영역(main area)에는 현재 관심의 대상인 개체(닻 개체, anchor entity)에 대한 정보와 닻 개체와 직·간접적으로 일대일 또는 다대일 관계 타입으로 연관되는 개체(첨부 개체)들에 대한 정보를 표시한다. 내포된 테이블(nested table)에는 닻 개체와 일대다 또는 다대다의 관계 타입으로 연관되는 개체(확장 개체)들을 표시한다.

그림 7은 e-order company 예제의 주문(Order) 품이며 이 주문 품에서 닻 개체는 한 개의 주문이 된다.

그림 2의 클래스 다이어그램에서, 개체 타입 Order는

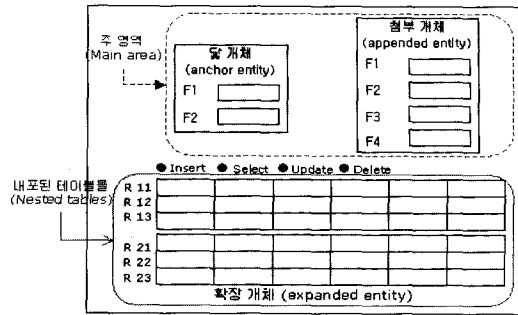


그림 6 품의 구조

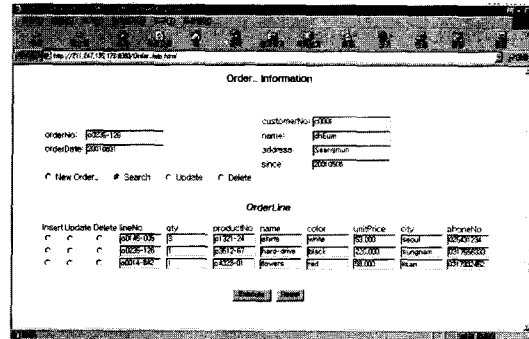


그림 7 e-order company의 주문 품

개체 타입 Customer와 다대일 관계 타입으로 연관되므로 주문과 고객에 관한 정보는 모두 주 영역에 표시될 수 있다. Order는 OrderLine과 일대다 관계 타입으로 연관되므로 여러 개의 주문항목들이 각 주문에 연관된다. 따라서 주문항목들은 내포된 테이블에 표시된다. 또한 하나의 OrderLine은 Product와 다대일 관계 타입으로 연관되므로 각 주문항목에 의해 주문된 제품 정보도 표시될 수 있다. 마지막으로 Product와 Warehouse는 다대일 관계 타입으로 연관되므로 창고에 대한 정보도 제품 정보와 함께 표시될 수 있다.

개체 그룹에 대한 정보를 표시하기 위해, WebSiteGen은 두 종류의 품을 사용한다. 여기서, 개체 그룹이란 한 개의 품으로 표시할 수 있는 개체 타입들의 집합을 말한다. 정보 품(information form)은 앞에서 설명한 것과 같이 주 영역과 내포된 테이블들로 구성된다. 정보 품은 품에 주어진 애트리뷰트 조건을 만족하는 개체들에 대한 질의에 사용된다. 또한 특정 개체와 이 개체에 연관된 개체들의 정보도 표시한다. 반면 리스트 품(list form)은 주 영역이 없이 질의에 의해 검색된 개체들의 리스트만을 보여준다.

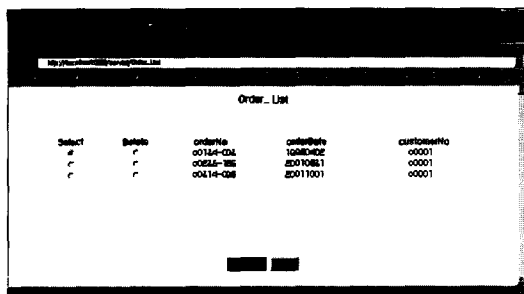


그림 8 주문을 위한 리스트 폼

그림 8의 리스트 폼은 그림 7의 주문 정보 폼에서, customerNo 필드에 c0001을 입력하고 Search를 선택하면 실행되는 질의에 대한 결과 리스트를 보여준다. 검색된 주문 레코드들은 갱신 또는 삭제될 수 있으며 리스트 폼으로부터 선택된 주문에 대한 자세한 정보는 주문 정보 폼에 의해 표시될 수 있다.

2.4 자바 서블릿의 생성

자바 서블릿은 CGI 프로그램과 유사하며 웹 페이지를 동적으로 생성할 수 있다. 그러나 자바 서블릿은 CGI 프로그램과는 달리 서버 측의 확장이므로 서버의

```

import javax.servlet.*;
import javax.servlet.http.*;
import java.sql.*;
....
public class Order_Info extends HttpServlet {
    Connection conn;
    Statement stmt;
    ....
    public Order_Info(String url) {
        // 데이터베이스에 접속한다.
        try { conn = DriverManager.getConnection(url, "admin", "admin"); }
        ....
    }
    public void doPost(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
        // 폼으로부터 데이터를 얻어 벡터 자료구조에 저장한다.
        anchorTableAtt.addElement( req.getParameter("orderNo") );
        eachEntityTable.addElement( req.getParameter("customerNo") );
        ....
        expandTable.addElement(eachEntityTable.clone());
        // 선택된 질의에 따라 함수를 호출한다.
        if (insert) aInsertQuery();
        else if (select) aSelectQuery();
        else if
            ...
        try { display(writer, res); } ....
    }
    public void aInsertQuery() {
        // 저장된 벡터 자료구조로부터 쿼리문장을 만든다.
        query = "insert into Order_ values ( ";
        for(int i=0; i<anchorTableAtt.size()-1; i++) {
            query = query + make((String)anchorTableAtt.elementAt(i)) + ", ";
        }
        query += " ) ";
        try { stmt.executeUpdate(query); } ....
    }
    public void aSelectQuery() {
        ....
    }
    ....
    public void display(PrintWriter writer, HttpServletResponse res) throws SQLException {
        res.setContentType("text/html");
        writer.println("<html>");
        ....
        // 쿼리를 수행해서 얻은 값을 HTML 폼에 보여준다.
        while(rset.next()) {
            ....
            writer.println("<td>" + rset.getString(i) + "</td>");
            ....
        }
        writer.println("</html>");
    }
}

```

그림 9 주문 폼의 서블릿 코드

일부분으로 수행된다. 따라서 자바 서블릿은 CGI 프로그램보다 훨씬 효율적이다. WebSiteGen이 생성하는 자바 서블릿은 Java Database Connectivity(JDBC)를 통해 데이터베이스와 연결된다[8]. 즉 서블릿은 HTML 폼으로부터 데이터를 받아 데이터베이스에 질의를 실행하며 그 결과를 HTML 폼 형식으로 생성한다.

그림 9는 그림 2의 클래스 다이어그램으로부터 자동 생성된 서블릿들 중 주문 정보 폼(그림 7)을 생성하고 이 폼을 통해 이루어진 질의를 처리하는 서블릿 코드이다.

Order_Info 클래스는 서블릿의 속성과 메소드들을 상속받기 위해 javax.servlet 패키지의 HttpServlet 클래스의 하위 클래스로 선언된다. WebSiteGen의 개발 환경(그림 2)에서 사이트 개발자가 클래스 다이어그램을 그린 후 정보 폼 생성 버튼을 누르면, 데이터베이스 서버가 설치된 호스트 이름, DBMS의 ID와 비밀번호를 묻는 다이얼로그 박스가 생성된다. 이때 입력된 값은 Order_Info 클래스의 생성자 메소드의 인수로 제공된다. display 메소드는 웹 브라우저를 통해 보여지는 입력 상자나 버튼 등의 GUI 요소들을 생성한다. HttpServlet 클래스의 doPost 메소드를 재정의(overriding)한 doPost 메소드는 사용자가 폼에 값을 입력함으로써 발생된 질의를 처리하기 위해 aInsertQuery, aSelectQuery 등의 메소드를 호출하고, 이에 따른 결과를 HTTP 폼 형식으로 보여주기 위한 display 메소드를 호출한다. aInsertQuery와 aSelectQuery 등의 메소드들은 주문 정보 폼을 통해 가능한 4가지 질의인 insert, select, delete, update 등의 연산을 수행한다.

3. 폼의 자동 생성

본 절에서는 클래스 다이어그램으로부터 폼을 생성하기 위한 규칙들을 e-order company 예를 통해 설명한다. 이러한 규칙들을 적용하면 각 폼이 커버할 수 있는 개체 타입들에 대한 그룹을 결정할 수 있다.

3.1 그룹화 문제

다음의 규칙들을 통해 한 폼이 동시에 보여줄 수 있는 개체 타입들을 결정할 수 있다. 개체 타입 A의 한 개체에 대한 정보를 보여주는 폼을 생각하고 개체 타입 A와 개체 타입 B 사이에 관계 타입이 존재한다고 가정하자.

규칙 1

두 개체 타입 A와 B 사이에 일대일 또는 다대일 관계 타입이 존재하면, 타입 B 개체는 타입 A 개체 x에 첨부(append)될 수 있다. 이때 타입 B 개체를 첨부

된 개체(appended entity)라 한다.

규칙 2

두 개체 타입 A와 B 사이에 일대다 또는 다대다 관계 타입이 존재하면, 개체 타입 A의 한 인스턴스에 여러 개의 개체 타입 B의 인스턴스들이 연관될 수 있다. 이때 개체 타입 A의 인스턴스는 이와 연관된 개체 타입 B의 인스턴스들로 확장(expanding)될 수 있고 타입 B 개체를 확장된 개체(expanded entity)라 한다.

규칙 3

첨부 연산은 모든 개체(덧 개체, 첨부된 개체 또는 확장된 개체)에 적용가능하며 한 폼 내에서 첨부 연산의 회수에는 제한이 없다.

규칙 4

확장 연산은 덧 개체 또는 덧 개체에 직·간접적으로 첨부된 개체에 적용할 수 있다. 한 폼 내에는 한번의 확장을 통해 도달할 수 있는 개체들만이 디스플레이될 수 있다.

규칙 5

덧 개체와 덧 개체에 직·간접적으로 첨부된 개체들의 속성 값은 폼의 주 영역에 표시될 수 있다.

규칙 6

확장된 개체와 확장된 개체에 직·간접적으로 첨부된 개체들의 속성 값은 내포된 테이블의 레코드로 표시될 수 있다.

그림 10은 e-order company 데이터베이스 응용의 클래스 다이어그램으로부터 위 규칙들을 적용해 생성된 폼들이 커버하는 개체 그룹들을 보인 것이다.

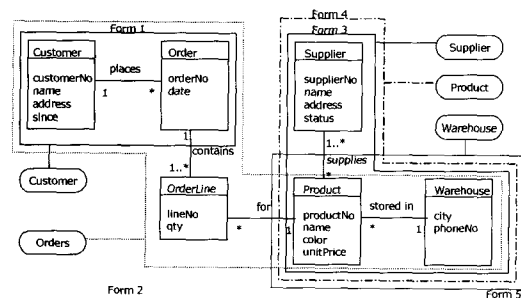


그림 10 각 폼이 커버하는 개체 그룹

Form1의 덧 개체 타입은 Customer이다. Customer는 개체 타입 Order와 일대다 관계 타입으로 연관된다. 따라서 규칙 2에 의해 Order는 확장 개체 타입이 될 수 있다. Order와 OrderLine 사이의 관계 타입은 일대다이

다. 따라서 OrderLine은 이중의 내포된 테이블을 요구하기 때문에 규칙 4에 대해 Form1에 추가될 수 없다. Form2의 닷 개체 타입은 Order이다. Order는 Customer와 다대일 관계 타입으로 연관되므로 규칙 1에 의해 Customer는 Order에 첨부될 수 있다. Order는 OrderLine과 일대다 관계 타입으로 연관되므로 규칙 2에 의해 OrderLine은 Order로부터 확장될 수 있다. 또한 OrderLine은 Product와 다대일 관계 타입으로 연관되므로 규칙 1에 의해 Product는 OrderLine에 첨부될 수 있다. 따라서 그림 7의 주문 폼을 얻게 된다. 그림 2의 클래스 다이어그램에서 선택된 닷 개체에 따른 개체 그룹은 각기 다른 색과 모양의 경계선으로 표시된다.

Form3, Form4, 그리고 Form5도 비슷한 방법으로 얻을 수 있다.

3.2 커버 문제

모든 개체에 대한 정보는 생성된 폼들을 통해 접근될 수 있어야 한다. 즉 모든 개체는 닷 개체가 되든지 아니면 닷 개체로부터 첨부 또는 확장 연산에 의해 도달될 수 있어야 한다. 한 개체가 닷 개체라면 폼의 주 영역을 통해 접근되며 닷 개체로부터 확장된 개체라면 폼에 내포된 테이블의 레코드에 그 정보가 표시된다.

개체 타입 A의 한 인스턴스가 닷 개체이고 개체 타입 B는 개체 타입 A로부터 첨부나 확장 연산을 통해 도달할 수 있다고 가정하자. 그러나 개체 타입 A로부터 개체 타입 B가 도달 가능하다 해도 B의 모든 인스턴스가 A의 인스턴스로부터 도달 가능한 것은 아니다.

개체 타입 A와 B 사이에 가능한 연결 형태는 그림 11과 같이 두 가지이다.

완전 커버(Complete Coverage) : 개체 타입 A의 카디널리티가 최소한 한 개의 인스턴스를 요구하면, 즉 카디널리티가 1, 1..* 등이면, 개체 타입 B의 모든 인스턴스는 타입 A 개체를 닷 개체로 하는 폼에서 접근 가능하다. 이때 개체 타입 B의 카디널리티 C는 무관하다(1, *, 또는 1..*).

부분 커버(Partial Coverage) : 개체 타입 A의 카디

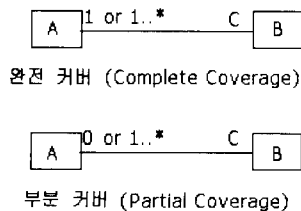


그림 11 개체들 간의 가능한 연결 형태

널리티가 0을 허용하면, 즉 카디널리티가 0.1, * 등이면, 개체 타입 B의 일부 인스턴스들은 A의 어떤 인스턴스와도 연관되지 않을 수 있다. 즉 이러한 개체들은 타입 A 개체를 닷 개체로 하는 폼에 의해 표시될 수 없다. 따라서 개체 타입 B는 B의 모든 개체들을 표시할 수 있는 다른 폼에 의해 커버되어야 한다.

3.3 폼의 완전성

각 폼에 필요한 연산들을 정의하기 위해, 각 폼에 하나의 use case를 연계시킨다. use case는 actor라 불리는 사용자 입장에서 본 기능의 단위를 나타낸다. use case 다이어그램은 use case, actor들과 함께 그들 간의 관계를 보여준다[9].

클래스 다이어그램으로부터 생성된 폼들을 통해 모든 개체들과 관계들을 모든 가능한 방법으로 조작할 수 있다면 그 폼들은 완전하다고 할 수 있다. 즉 클래스 다이어그램 내의 개체 타입과 관계 타입들로부터 생성된 관계형 테이블들은 개체들과 관계들에 관한 완전한 정보를 저장할 수 있으므로 이 테이블들이 모든 가능한 방법으로 조작될 수 있다면 생성된 폼들이 완전하다고 할 수 있다. 한 테이블에 대해 가능한 연산은 검색, 삽입, 갱신, 그리고 삭제 연산이 있다. 따라서 생성된 폼들이 연산들을 모든 테이블에 대해 수행할 수 있다면 폼들은 완전하다. 이때 한 테이블에 대한 연산들은 서로 다른 폼들에 의해 지원될 수도 있다.

3.3.1 주문 정보에 관한 Use Case

주문 정보에 관한 use case 다이어그램은 그림 12와 같다. 테이블 Order에 대해 검색, 삽입, 갱신, 삭제 연산이 가능하고 테이블 OrderLine에 대해서는 삽입, 갱신, 삭제 연산이 이 use case와 연계된 폼으로 가능함을 알 수 있다. 그러나 Customer, Product, Warehouse에 관한 정보는 조작될 수 없고 OrderLine과 연관된 정보로만 보여지게 된다.

그림 8의 리스트 폼은 질의에 의해 검색된 Order 레코드들을 표시한다. 이 폼으로 각 Order 레코드들 삭제

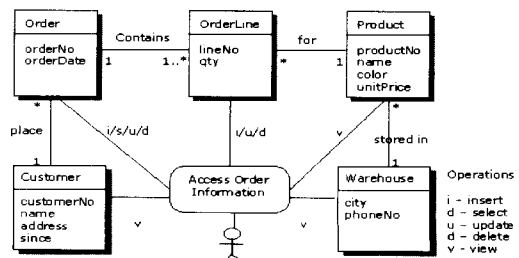


그림 12 주문 정보에 관한 use case 다이어그램

할 수 있으며 선택된 Order 레코드는 Order 정보 폼에 의해 표시된다.

3.3.2 제품 정보에 관한 Use Case

제품 정보에 관한 use case 다이어그램은 그림 13과 같다. 이 use case의 actor는 Product 테이블에 대해 검색, 삽입, 갱신, 삭제 연산을 수행 할 수 있다. 또한 개체 타입 SupplierProduct로부터 생성된 테이블에 삽입, 갱신, 삭제 연산을 수행할 수 있다. 현재 선택된 제품을 공급하는 Supplier에 관한 정보와 그 제품이 비치된 Warehouse에 관한 정보가 표시된다.

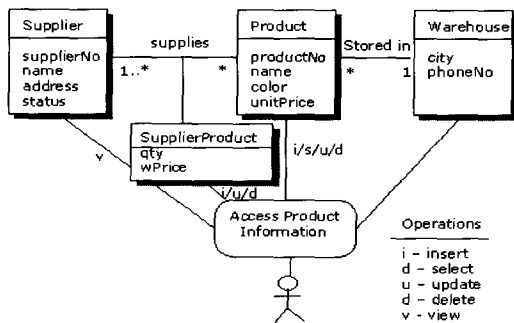


그림 13 제품 정보에 관한 use case 다이어그램

제품에 관한 정보 폼은 그림 14와 같다. 테이블 Product에 대해서 검색, 삽입, 갱신, 삭제 연산이 가능하다. update를 선택한 후, 제품의 정보를 입력하고 실행 버튼을 누르면 제품의 데이터가 갱신된다. 이때 Product와 일대다의 관계 타입을 가진 OrderLine의 제약 사항인 cascade에 따라 OrderLine의 외래 키가 자동으로 갱신된다. supplierNo, productNo, qty, unitPrice 필드를 가지는 관계형 테이블 SupplierProduct가 생성되고 이 테이블에 삽입, 삭제, 갱신 연산이 허용되어야 한다.

Customer, Supplier, Warehouse 정보에 관한 use case들과 폼들도 이와 유사하다.

Use case로 지원되는 연산들을 점검함으로써 e-order company 응용을 위해 생성된 모든 테이블에 대해 검색, 삽입, 갱신, 삭제 연산이 적용될 수 있음을 보일 수 있다. Customer 레코드는 Customer 폼을 통해 삽입, 검색, 수정, 삭제 연산이 적용될 수 있다. 비슷한 방법으로 Order, Product, Supplier, Warehouse 레코드들도 Order, Product, Supplier, Warehouse 폼으로 처리될 수 있다. 각 OrderLine은 Order 폼의 내포된 테이블에 의해 처리될 수 있다. 삽입, 삭제, 갱신 연산은 Supplier 또는 Product 폼과 함께 SupplierProduct 테이블에서 처리될 수 있다.

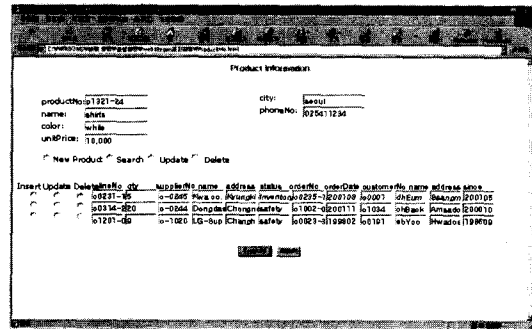


그림 14 제품 정보 폼

4. 상용 웹 사이트 생성기와의 기능 비교

본 절에서는 WebSiteGen과 상용 웹 사이트 생성기들을 비교한다. 표 1은 WebSiteGen과 상용 웹사이트 생성 도구인 1ClickDBPro, CodeCharge, ASP.NET의 기능을 비교한 것이다.

표 1 WebSiteGen과 상용 웹 사이트 생성기들의 기능 비교

웹 사이트 생성기	WebSiteGen	1ClickDBPro	CodeCharge	ASP.NET
한 폼에서 표현 가능한 개체 타입	몇 개체 첨부 개체 확장 개체	몇 개체	몇 개체	몇 개체 첨부 개체
폼 생성	자동	반자동	반자동	반자동
테이블 생성	자동	자동	생성 못함	자동
질의 생성	Servlet 자동 생성	ASP 수동 또는 자동 생성	ASP, JSP, PHP 수동 생성	ASP 자동 생성
개발 환경	시각적 (클래스 다이어그램)	비 시각적	시각적	시각적

1ClickDBPro는 몇 개체만을 표시하는 폼을 반자동으로 생성한다. 즉 개발자가 지정하는 한 개의 테이블 내에 국한된 정보만 표시할 수 있다. 클래스 다이어그램을 사용하는 WebSiteGen과는 달리 입력 폼을 사용하여 데이터베이스의 스키마를 생성한다. 또한 폼을 통한 질의를 처리할 수 있는 ASP 코드를 자동으로 생성하거나, 직접 SQL 질의문을 개발자가 입력하여 수동으로 생성하는 두 가지 방식을 지원한다. 그러나 1ClickDBPro는 WebSiteGen과는 달리, 몇 개체와 다대일이나 일대일의 관계 타입으로 연관되는 첨부 개체와 몇 개체와 일대다의 관계 타입으로 연관되는 확장 개체들을 하나의 폼으로 함께 표시할 수 없다. 또한 1ClickDBPro의

개발 환경은 WebSiteGen에 비해 시각적이지 못하다.

CodeCharge는 IClickDBPro와 마찬가지로 닷 개체만을 표현하는 폼을 반자동으로 생성하며 이미 구축된 데이터베이스만을 사용해야 한다. 또한 폼을 통한 질의 처리는, 개발자가 필요한 SQL 질의문을 직접 입력해주어야 ASP, JSP 또는 PHP 형식으로 생성한다. Code Charge는 웹 사이트를 구성하는 페이지들의 상호 관계와 작용을 다이어그램으로 보여주므로 시각적인 개발 환경을 제공한다.

ASP.NET은 Microsoft.NET 프레임워크의 일부분으로서 Visual Studio.NET 도구와 MS SQL Server 7.0 데이터베이스를 이용하여 웹 사이트를 자동으로 생성할 수 있다. ASP.NET이 한 폼에서 보여줄 수 있는 개체는 닷 개체와 첨부 개체이다. 폼을 통한 질의를 처리할 ASP 코드를 자동 생성한다. 또한 개발 툴들을 이용하여 폼의 GUI를 생성할 수 있고 개체간의 관계를 다이어그램으로 도식화하여 데이터베이스 스키마를 생성하는 시각적 개발 환경을 지원한다.

표 1에서 보듯이, WebSiteGen이 생성하는 폼은 닷 개체 타입과 일대일, 다대일, 일대다 타입으로 연관되는 개체 타입의 개체들을 닷 개체와 함께 한 폼에 표시해 줄 수 있다는 장점이 있다. 또한 WebSiteGen은 설계 초기에 작성되는 클래스 다이어그램으로부터 필요한 모든 컴포넌트들을 자동 생성하는 시각적인 프로그래밍 환경을 제공할 뿐만 아니라, 웹 사이트를 수정하고자 할 때엔 클래스 다이어그램만 간단히 변경함으로써 수정 사항을 쉽게 반영할 수 있다.

5. WebSiteGen의 구현

WebSiteGen의 프로토타입은 Java로 구현되었다. WebSiteGen은 데이터베이스 구축을 위한 SQL 문장들, HTML 폼들, 그리고 자바 서블릿을 생성한다. 자바 서블릿은 Java Web Server(JWS)에 의해 실행되고 HTML 폼들을 동적으로 생성한다.

그림 15는 WebSiteGen의 구현에 사용된 클래스들의 대략적인 계층도와 함께 클래스 다이어그램으로부터 자동 생성된 SQL 문장, 폼, 서블릿이 클라이언트가 폼을 통해 질의를 요청할 때 어떻게 상호 작용하게 되는지의 과정을 보인 것이다.

그림의 과정 1부터 6까지는 개발자가 웹 사이트를 구축하는 과정이다. Entity 클래스는 클래스 다이어그램에 표현된 개체 타입들을 생성하는 클래스이다. Infor Relationship 클래스는 생성된 개체 타입의 다른 개체

타입과의 관계 타입을 정의한다. 그림 2의 WebSiteGen의 개발 환경에서 코드 생성 버튼을 누르면 완성된 클래스 다이어그램으로부터 데이터베이스를 구축하기 위한 create SQL 문장이 생성된다. create SQL 문장들은 SqlGen 클래스가 InfoSql과 AbInforSql 클래스를 이용하여 확장자가 '.sql'인 파일에 저장한다. InfoSql과 AbInfoSql 클래스는 각 개체들의 속성 값들을 이용해 create table 문장을 생성하는 클래스들이다. WebSiteGen의 개발 환경에서 폼 생성 버튼을 누르면, HTML 폼들을 생성하고 이 폼들을 통해 데이터베이스에 SQL 질의문을 수행할 수 있는 서블릿 코드가 생성된다. 폼은 FormGen 클래스, 서블릿은 ServletGen 클래스에 의해 생성된다. 생성된 폼, create table SQL문, 서블릿 코드들은 서버에 저장된다.

과정 7부터 과정 10까지는 구축된 웹 사이트를 클라이언트 측에서 폼을 통해 질의를 요청해서 결과를 얻을 때까지의 과정을 보인 것이다. 사용자가 웹 브라우저를 통해, 서버에 저장된 HTML 파일을 요청하여 폼이 표시되면 사용자는 이 폼에 데이터를 입력하여 질의를 요청한다. 이는 서버 측의 서블릿을 수행시키게 되며 서블릿 코드는 SQL 질의문을 이용해 데이터베이스에 접근한다. 질의문의 결과는 HTML 폼 형태로 클라이언트 측의 웹 브라우저에 의해 표시된다.

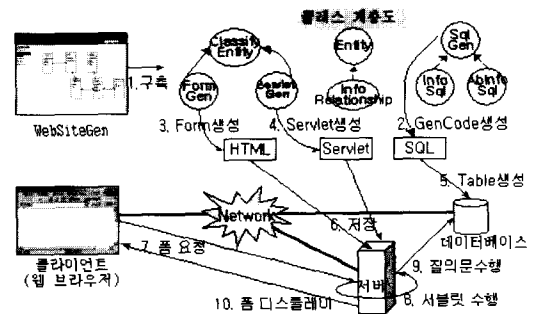


그림 15 폼, 질의문, 서블릿의 자동 생성과 수행 과정

E-order company 응용인 경우, WebSiteGen의 개발 환경에서 작성한 클래스 다이어그램으로부터 생성된 create table SQL 문장들은 무결성 제약조건을 고려해서 Customer, Order_, OrderLine, Product, Supplier, Warehouse, 그리고 SupplierProduct의 테이블들을 생성한다. 3절에서 설명한 폼 생성 규칙에 따라, 각 폼이 커버할 개체 그룹들도 그림 2와 같이 클래스 다이어그램에 표시된다. Order_ 폼의 닷 개체 타입인 Order_를

화면 왼쪽의 리스트에서 선택하고 Form View 메뉴를 선택하면 닷 개체 타입인 Order_는 적색의 이중 실선으로 둘러싸이고, 이 개체 타입을 중심으로 연관된 개체 타입들인 Customer, OrderLine, Product, 그리고 Warehouse 중 확장 개체 타입인 Customer는 청색의 실선으로, 첨부 개체 타입인 OrderLine, Product, Warehouse는 모두 회색의 실선에 의해 둘러싸인다. Customer 폼, Product 폼, Supplier 폼, Warehouse 폼을 위한 개체 그룹들도 모두 닷 개체 타입으로 선택되면 이와 같은 방식으로 표시된다.

6. 결론 및 향후 연구

클래스 다이어그램으로부터 웹 기반의 데이터베이스 응용을 자동으로 생성하는 방법을 제안하고 이 작업을 자동 수행하는 프로토타입, WebSiteGen을 구현하였다. WebSiteGen은 응용 데이터베이스 구축을 위한 SQL 문장들을 생성하고 사용자 인터페이스로 사용되는 폼들을 생성하며 데이터베이스와 폼들을 연결하는 자바 서블릿을 생성한다. 따라서 웹 기반 데이터베이스 응용의 구현을 자동화한다. 표 1과 같이 상용 제품들과의 기능 비교를 통해 WebSiteGen의 장점을 설명하였다.

또한 본 논문에서는 클래스 다이어그램으로부터 폼들을 생성하기 위한 규칙들을 제시하였다. 각 폼에 의해 수행될 연산들은 use case를 사용하여 정의하였다. 생성될 폼들에 의해 커버될 개체 타입들과 관계 타입들에 대해 설명하였으며 외래 키에 무결성 제약조건을 명시하는 문제를 다루었다. 생성된 폼들의 완전성을 검사하는 방법도 제시하였다.

WebSiteGen은 웹 기반 데이터베이스 응용을 자동 생성할 수 있는 소프트웨어 도구의 근간이 될 수 있다. 향후, ASP, PHP, 그리고 JSP와 같은 서버 측의 컴퓨팅 환경을 위해 동적으로 HTML 페이지를 생성할 수 있도록 할 것이다.

감사의 글

WebSiteGen의 구현에 도움을 준 덕성여대 전산학과 오하영, 김연희 학생들에게 감사의 마음을 전합니다.

참고 문헌

- [1] Grady Booch, James Rumbaugh, and Ivar Jacobson, The Unified Modeling Language User Guide, Addison-Wesley, 1998.
- [2] Michael W. Stowe, Oracle Developer/2000 Handbook, 2nd Ed., PH, 1999.
- [3] Ken Getz and Mike Gilbert, Access 2000 Developer's Handbook, Vol. 1, Sybex, 1999.
- [4] <http://support.codecharge.com/files/CodeChargeManual.pdf>
- [5] <http://1clickdb.com/1clickdbpro/launch.html>
- [6] <http://www.microsoft.com/korea/msdn/library/Dnaspp/html/aspnetmigriissues.asp>
- [4] C.J. Date, An Introduction to Database Systems, 6th Ed., Addison-Wesley, 1995.
- [5] Rational Group, UML Notation Guide, Version 1.1, 1997.
- [6] Ramez Elmasri and Shamkanth B. Navathe, Fundamentals of Database Systems, 2nd Ed., Benjamin/Cummings, 1994.
- [7] Angel R. Puerta, Henrik Eriksson, John H. Gennar, and Mark A. Musen, Model-Based Automated Generation of User Interfaces, In Proceedings of the 12th National Conference on Artificial Intelligence, Vol. 1, 1994, 471-477.
- [8] Michel E. Adiba and Christine Collect, Management of Complex Objects as Dynamic Forms, In Proceedings of the 14th International Conference on Very Large Data Bases, 1988, 134-147.
- [9] Raymond J. A. Buhr, Use Case Maps as Architectural Entities for Complex Systems, IEEE Transactions on Software Engineering, Vol. 24, No.12, 1998, 1131-1155.



음 두 현

1984년 2월 서강대학교 전자공학과(학사). 1987년 10월 오레곤 주립대학교 컴퓨터 공학과(석사). 1990년 10월 오레곤 주립대학교 컴퓨터 공학과(박사). 1991년 1월 ~ 1992년 2월 전자통신연구원 인공지능연구실. 1992년 3월 ~ 현재 덕성여자대학교 전산학과 부교수. 1999년 9월 ~ 2000년 8월 오레곤 주립대학교 전산학과 객원교수. 관심분야는 객체지향 시스템, 컴포넌트 기반 시스템



유 은 자

2000년 2월 덕성여자대학교 전산학과(학사). 현재 덕성여자대학교 전산·정보통신 대학원 재학중. 관심분야는 객체지향 시스템, 컴포넌트 기반 시스템