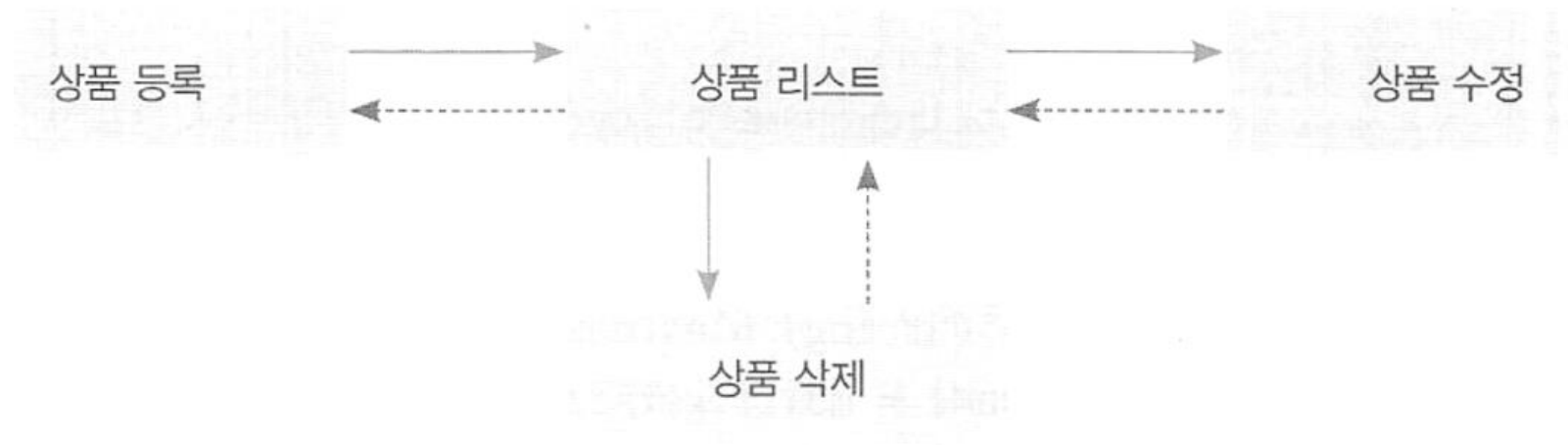


파일 업로드

- 쇼핑몰 이미지 업로드

쇼핑몰 관리자 페이지

❖ 개요



쇼핑몰 관리자 페이지

❖ 상품 리스트

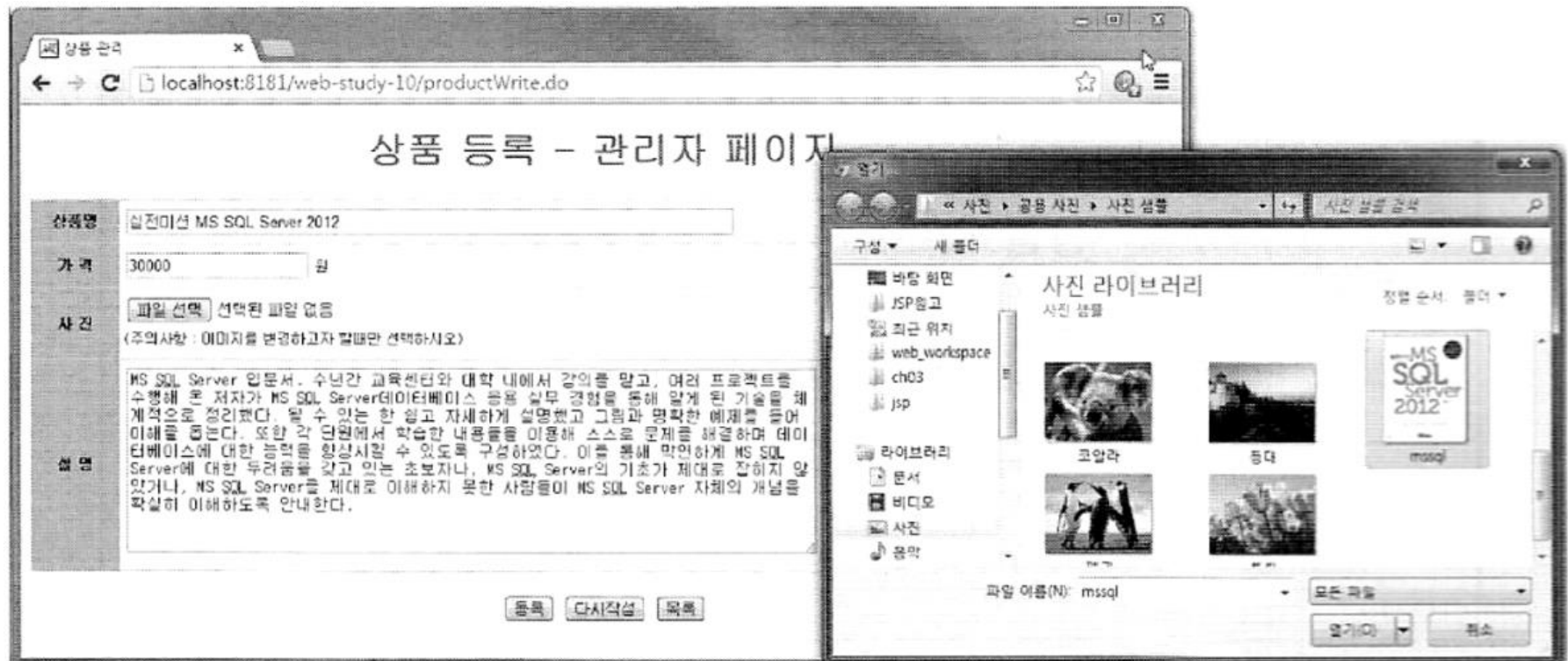
상품 리스트 - 관리자 페이지

상품 등록

번호	제목	가격	수정	삭제
5	jQuery and jQuery Mobile : 이해하기 쉽게 풀어쓰	25000원	상품 수정	상품 삭제
4	Visual C++ MFC 윈도우 프로그래밍	26000원	상품 수정	상품 삭제
3	Dynamic Programming book 시리즈-이 오라클 11g + PL/SQL	25000원	상품 수정	상품 삭제
2	웹포준을 위한 HTML 5	25000원	상품 수정	상품 삭제
1	개념을 콕콕 잡아주는 데이터베이스	27000원	상품 수정	상품 삭제

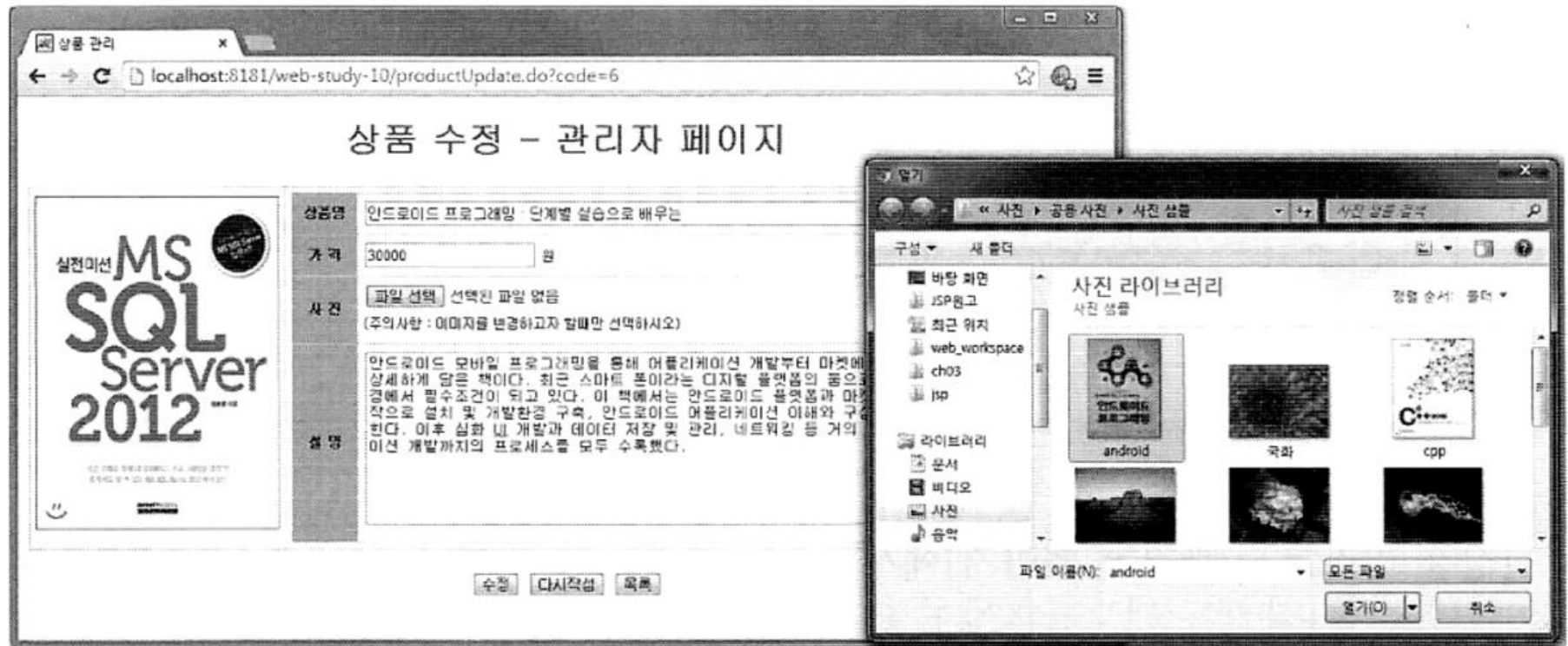
쇼핑몰 관리자 페이지

❖ 상품 등록




쇼핑몰 관리자 페이지

❖ 상품 수정



❖ 상품 삭제

상품 삭제 - 관리자 페이지

	상 품 명	안드로이드 프로그래밍 : 단계별 실습으로 배우는
	가 격	30000 원
	설 명	<p>안드로이드 모바일 프로그래밍을 통해 어플리케이션 개발부터 마켓에 올리는 것까지의 전 과정을 상세하게 담은 책이다. 최근 스마트 폰이라는 디지털 플랫폼의 붐으로 어플리케이션 개발은 IT 환경에서 필수조건이 되고 있다. 이 책에서는 안드로이드 플랫폼과 마켓에 대한 개괄적인 설명을 시작으로 설치 및 개발환경 구축, 안드로이드 어플리케이션 이해와 구성요소 및 툴들을 기본부터 익힌다. 이후 심화 UI 개발과 데이터 저장 및 관리, 네트워킹 등 거의 전 영역에 걸쳐 최종 어플리케이션 개발까지의 프로세스를 모두 수록했다.</p>

[삭제](#) [목록](#)

쇼핑몰 관리자 페이지

❖ jsp 파일 목록

[JSP 페이지(위치 : WebContentWproduct)]

파일	설명
productList.jsp	상품 리스트 페이지
productWrite.jsp	상품 등록 페이지
productUpdate.jsp	상품 수정 페이지
productDelete.jsp	상품 삭제 페이지

❖ 유효성 검사 자바스크립트

[자바스크립트(위치 : WebContentWscript)]

파일	설명
product.js	폼에 입력된 정보가 올바른지 판단하는 자바스크립트

쇼핑몰 관리자 페이지

❖ 서블릿 파일 목록

[서블릿 파일(위치 : src\Wcom\Wsaeyan\Wcontroller)]

파일	설명	요청 URL 패턴
ProductListServlet.java	상품 전체 정보를 데이터베이스에서 얻어온다.	productList.do
ProductWriteServlet.java	입력한 상품 정보를 데이터베이스에 추가한다.	productWrite.do
ProductUpdateServlet.java	입력한 정보로 데이터베이스에 상품 정보를 수정한다.	productUpdate.do
ProductDeleteServlet.java	데이터베이스에 상품 정보를 삭제한다.	productDelete.do

❖ VO 클래스

[VO 클래스 (위치: src\Wcom\Wsaeyan\Wdto)]

파일	설명
ProductVO.java	상품 정보를 저장하는 클래스

❖ DAO 클래스

[DAO 클래스 (위치: src\Wcom\Wsaeyan\Wdao)]

파일	설명
ProductDAO.java	데이터베이스 테이블과 연동해서 작업하는 데이터베이스 처리 클래스

쇼핑몰 관리자 페이지

❖ 데이터베이스 준비

- product 테이블 구조

컬럼명	크기	설명
code	number(5)	코드
name	varchar2(100)	이름
price	number(8)	가격
pictureurl	varchar2(50)	사진 경로
description	varchar(1000)	설명

❖ 데이터베이스 준비

- product 테이블 생성

```
create table product(  
    code number(5),  
    name varchar2(100),  
    price number(8),  
    pictureurl varchar(50),  
    description varchar(1000),  
    primary key (code)  
);
```

쇼핑몰 관리자 페이지

❖ 데이터베이스 준비

- product 테이블 데이터

CODE	NAME	PRICE	PICTUREURL	DESCRIPTION
1	개념을 콕콕 잡아주는 데이터베이스	27000	db.jpg	데이터베이스에 관한 모든 것을 쉽고 재미있게 정리한 교재...
2	웹표준을 위한 HTML 5	25000	html5.jpg	HTML5 가이드북. 홈페이지 제작을 위한 필수 선택 HTML 기본 문법...
3	Dynamic Programming book 시리즈-01 오라클 11g + PL/SQL	25000	oracle.jpg	Dynamic 실무 코칭 프로그래밍 Book의 첫번째 책으로, 11g의 새로운 ...
4	Visual C++ MFC 윈도우 프로그래밍	26000	mssql.jpg	Visual C++를 처음 시작하는 독자의 눈높이에 맞춘 Visual C++...
5	jQuery and jQuery Mobile : 이해하기 쉽게 풀어쓰	25000	jquery.jpg	소스 하나로 데스크탑과 모바일 까지 HTML5와 함께 사용한다. 초보자들도 ...

❖ 데이터베이스 준비

- product 테이블을 위한 시퀀스

```
create sequence product_seq start with 1 increment by 1;
```

쇼핑몰 관리자 페이지

❖ 데이터베이스 준비

- 데이터삽입 SQL 스크립트 파일
 - product.sql

```
insert into product values(product_seq.nextval, '개념을 콕콕 잡아주는 데이터베이스', 27000, 'db.jpg', '데이터베이스에 관한 모든 것을 쉽고 재미있게 정리한 교재...');
```

```
insert into product values(product_seq.nextval, '웹표준을 위한 HTML 5', 25000, 'html5.jpg', 'HTML5 가이드북. 홈페이지 제작을 위한 필수 선택 HTML 기본 문법...');
```

```
insert into product values(product_seq.nextval, 'Dynamic Programming book 시리즈-01 오라클 11g + PL/SQL', 25000, 'oracle.jpg', 'Dynamic 실무 코칭 프로그래밍 Book의 첫번째 책으로, 오라클 11g의 새로운 ...');
```

```
insert into product values(product_seq.nextval, 'Visual C++ MFC 윈도우 프로그래밍', 26000, 'mfc.jpg', 'Visual C++를 처음 시작하는 독자의 눈높이에 맞춘 Visual C++...');
```

❖ 데이터베이스 준비

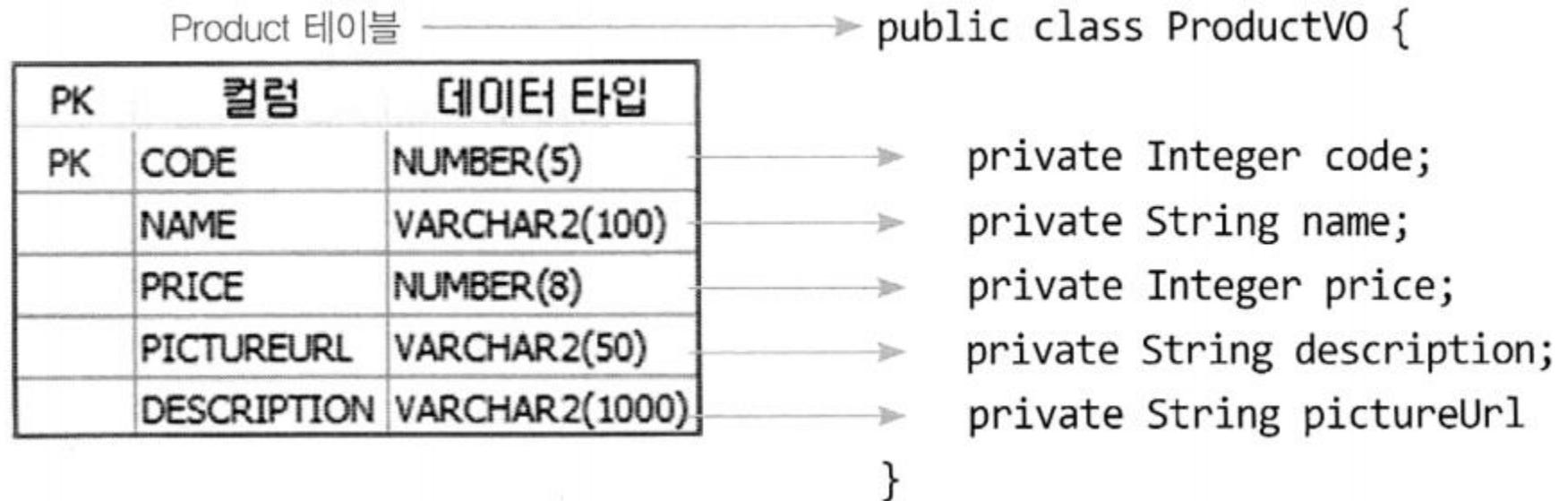
- SQL 스크립트 파일 실행
 - SQL/PLUS에서 @product 실행

```
SQL> @product
```


❖ DBCP 준비

- 톰캣 서버의 `server.xml`
 - `<Context>` 하위 태그로 `<Resource>` 구성

❖ VO 클래스 정의



❖ ProductVO.java

```
package com.saeyan.dto;

public class ProductVO {

    private Integer code;
    private String name;
    private Integer price;
    private String description;
    private String pictureUrl;

    // 생성자
    // Getter/Setter
    // toString()
}
```

❖ ProductDAO 클래스

리턴형	메소드
ProductDAO getInstance()	ProductDAO 객체를 리턴한다.
List<ProductVO> selectAllProducts()	최근 등록한 상품을 먼저 출력한다.
void insertProduct(ProductVO pVo)	전달인자로 받은 VO 객체를 product 테이블에 삽입한다.
int confirmID(String userid)	아이디 중복 확인을 한다. 해당 아이디가 있으면 1을, 없으면 -1을 리턴한다.
int userCheck(String userid, String pwd)	사용자 인증시 사용하는 메소드이다. product 테이블에서 아이디와 암호를 비교해서 해당 아이디가 존재하지 않으면 -1을, 아이디만 일치하고 암호가 다르면 0을, 모두 일치하면 1을 리턴한다.
ProductVO selectProductByCode(String code)	product 테이블에서 상품 코드로 해당 상품을 찾아 상품 정보를 ProductVO 객체로 얻어준다.
void updateProduct(ProductVO pVo)	매개 변수로 받은 VO 객체 내의 코드로 product 테이블에서 검색해서 VO 객체에 저장된 정보로 상품 정보를 수정한다.

❖ Connection 객체 획득/반납 처리 클래스

```
package util;
:

public class DBManager {
    public static Connection getConnection() {
        Connection conn = null;
        try {
            Context initContext = new InitialContext();
            Context envContext = (Context)
                                initContext.lookup("java:/comp/env");
            // jdbc/myoracle이란 이름을 객체를 찾아서 DataSource가 받는다.
            DataSource ds = (DataSource) envContext.lookup("jdbc/myoracle");
            // ds가 생성되었으므로 Connection을 구합니다.
            conn = ds.getConnection();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return conn;
    }
}
```

❖ Connection 객체 획득/반납 처리 클래스

```
// select을 수행한 후 리소스 해제를 위한 메소드
public static void close(Connection conn, Statement stmt, ResultSet rs)
{
    try {
        rs.close();
        stmt.close();
        conn.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}

// insert, update, delete 작업을 수행한 후 리소스 해제를 위한 메소드
public static void close(Connection conn, Statement stmt) {
    try {
        stmt.close();
        conn.close();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
}
```

쇼핑몰 관리자 페이지

❖ 스타일 시트

- WebContent/css/shopping.css

```
#wrap {  
    width: 971px; /* 1024*768로 해상도를 맞추어서 설계 */  
    /* 가운데 정렬을 위한 바깥쪽 여백 설정 */  
    margin: 0;  
    margin-left: auto;  
    margin-right: auto;  
}  
  
h1 {  
    color: green; /* 글 색상 */  
}  
  
img {  
    width: 220px; /* 이미지 너비(가로) */  
    height: 300px; /* 이미지 높이(세로) */  
}
```


쇼핑몰 관리자 페이지

❖ 스타일 시트

- WebContent/css/shopping.css

```
table {  
    width: 100%;  
    border-collapse: collapse;  
    font-size: 12px; /* 글꼴 크기 */  
    line-height: 24px; /* 줄 간격 */  
}  
  
table td,th {  
    border: #d3d3d3 solid 1px; /* 경계선 색상 스타일 굵기 */  
    padding: 5px; /* 안쪽 여백 */  
}  
  
th {  
    background: yellowgreen; /* 배경색 */  
}
```

❖ 스타일 시트

- WebContent/css/shopping.css

```
a {  
    text-decoration: none; /* 링크 밑줄 없애기 */  
    color: black; /* 글 색상 */  
}  
  
a:HOVER {  
    text-decoration: underline; /* 밑줄 */  
    color: green; /* 글 색상 */  
}
```

쇼핑몰 관리자 페이지

❖ 상품 리스트 페이지

- DAO로 상품의 목록 추출
- 표 형태로 화면 출력

상품 리스트 - 관리자 페이지

상품 등록

번호	이름	가격	수정	삭제
5	jQuery and jQuery Mobile : 이해하기 쉽게 풀어쓴	25000원	상품 수정	상품 삭제
4	Visual C++ MFC 윈도우 프로그래밍	26000원	상품 수정	상품 삭제
3	Dynamic Programming book 시리즈-01 오라클 11g + PL/SQL	25000원	상품 수정	상품 삭제
2	웹표준을 위한 HTML 5	25000원	상품 수정	상품 삭제
1	개념을 콕콕 잡아주는 데이터베이스	27000원	상품 수정	상품 삭제

❖ ProductDAO.java

```
package com.saeyan.dao;
:

public class ProductDAO {

    // 싱글톤 패턴 적용
    private ProductDAO() {
    }

    private static ProductDAO instance = new ProductDAO();

    public static ProductDAO getInstance() {
        return instance;
    }
}
```

❖ ProductDAO.java

```
public List<ProductVO> selectAllProducts() {  
    // 최근 등록한 상품 먼저 출력하기  
    String sql = "select * from product order by code desc";  
    List<ProductVO> list = new ArrayList<ProductVO>();  
    Connection conn = null;  
    PreparedStatement pstmt = null;  
    ResultSet rs = null;  
    try {  
        conn = DBManager.getConnection();  
        pstmt = conn.prepareStatement(sql);  
        rs = pstmt.executeQuery();  
        while (rs.next()) { // 이동은 행(로우) 단위로  
            ProductVO pVo = new ProductVO();  
            pVo.setCode(rs.getInt("code"));  
            pVo.setName(rs.getString("name"));  
            pVo.setPrice(rs.getInt("price"));  
            pVo.setPictureUrl(rs.getString("pictureUrl"));  
            pVo.setDescription(rs.getString("description"));  
            list.add(pVo);  
        } // while문 끝  
    }  
}
```

❖ ProductDAO.java

```
        catch (Exception e) {  
            e.printStackTrace();  
        } finally {  
            DBManager.close(conn, pstmt, rs);  
        }  
        return list;  
    } // selectAllProducts() {
```

❖ ProductListServlet.java

```
@WebServlet("/productList.do")
public class ProductListServlet extends HttpServlet {
    :
    protected void doGet(HttpServletRequest request,
                        HttpServletResponse response)
                        throws ServletException, IOException {

        ProductDAO pDao = ProductDAO.getInstance();

        List<ProductVO> productList = pDao.selectAllProducts();

        request.setAttribute("productList", productList);
        RequestDispatcher dispatcher = request
            .getRequestDispatcher("product/productList.jsp");
        dispatcher.forward(request, response);
    }
}
```


❖ product/productList.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
:
<body>
    <div id="wrap" align="center">
        <h1>상품 리스트 - 관리자 페이지</h1>
        <table class="list">
            <tr>
                <td colspan="5" style="border: white; text-align: right"><a
                    href="productWrite.do">상품 등록</a></td>
            </tr>
            <tr>
                <th>번호</th>
                <th>이름</th>
                <th>가격</th>
                <th>수정</th>
                <th>삭제</th>
            </tr>
```

❖ product/productList.jsp

```
<c:forEach var="product" items="${productList}">
  <tr class="record">
    <td>${product.code}</td>
    <td>${product.name}</td>
    <td>${product.price} 원</td>
    <td><a href="productUpdate.do?code=${product.code}">
      상품 수정</a>
    </td>
    <td><a href="productDelete.do?code=${product.code}">
      상품 삭제</a>
    </td>
  </tr>
</c:forEach>
</table>
</div>
</body>
</html>
```

쇼핑몰 관리자 페이지

❖ 상품 리스트 페이지

번호	이름	가격	수정	삭제
5	jQuery and jQuery Mobile : 이해하기 쉽게 풀어쓴	25000원	상품 수정	상품 삭제
4	Visual C++ MFC 윈도우 프로그래밍	26000원	상품 수정	상품 삭제
3	Dynamic Programming book 시리즈-이 오라블 11g + PL/SQL	25000원	상품 수정	상품 삭제
2	웹표준을 위한 HTML 5	25000원	상품 수정	상품 삭제
1	개념을 확실히 잡아주는 데이터베이스	27000원	상품 수정	상품 삭제

쇼핑몰 관리자 페이지

❖ 상품 등록하기

`상품 등록`

상품 관리

localhost:8181/web-study-10/productWrite.do

상품 등록 - 관리자 페이지

상품명	<input type="text"/>
가격	<input type="text"/> 원
사진	<input type="button" value="파일 선택"/> 선택된 파일 없음 (주의사항 : 이미지를 변경하고자 할때만 선택하십시오)
설명	<div></div>

쇼핑몰 관리자 페이지

❖ 상품 등록 서블릿

- doGet
 - 상품 등록 폼 페이지 이동
- doPost
 - 상품 등록 실행

❖ ProductWriteServlet.java

```
@WebServlet("/productWrite.do")
public class ProductWriteServlet extends HttpServlet {
    :
    protected void doGet(HttpServletRequest request,
                        HttpServletResponse response)
                        throws ServletException, IOException {

        RequestDispatcher dispatcher = request
            .getRequestDispatcher("product/productWrite.jsp");
        dispatcher.forward(request, response);

    }

    protected void doPost(HttpServletRequest request,
                        HttpServletResponse response)
                        throws ServletException, IOException {
        // 나중에 구현
    }
}
```

❖ product/productWrite.jsp

```
:
<body>
  <div id="wrap" align="center">
    <h1>상품 등록 - 관리자 페이지</h1>
    <form method="post" enctype="multipart/form-data" name="frm">
      <table>
        <tr>
          <th>상 품 명</th>
          <td><input type="text" name="name" size="80"></td>
        </tr>
        <tr>
          <th>가 격</th>
          <td><input type="text" name="price"> 원</td>
        </tr>
        <tr>
          <th>사 진</th>
          <td><input type="file" name="pictureUrl"><br>
            (주의사항 : 이미지를 변경하고자 할때만 선택하시오)</td>
        </tr>
      </table>
    </form>
  </div>
</body>
```


❖ product/productWrite.jsp

```
        <tr>
            <th>설 명</th>
            <td>
                <textarea cols="80" rows="10"
                    name="description"></textarea></td>
        </tr>
    </table>
    <br>
    <input type="submit" value="등록" onclick="return productCheck()">
    <input type="reset" value="다시작성">
    <input type="button" value="목록"
        onclick="location.href='productList.do'">
    </form>
</div>
</body>
</html>
```

❖ ProductDAO.java

```
public void insertProduct(ProductVO pVo) {
    String sql = "insert into product " +
                " values(product_seq.nextval, ?, ?, ?, ?)";
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = DBManager.getConnection();

        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, pVo.getName());
        pstmt.setInt(2, pVo.getPrice());
        pstmt.setString(3, pVo.getPictureUrl());
        pstmt.setString(4, pVo.getDescription());

        pstmt.executeUpdate(); // 실행
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        DBManager.close(conn, pstmt);
    }
}
```

❖ ProductWriteServlet.java

```
protected void doPost(HttpServletRequest request,
                      HttpServletResponse response)
                      throws ServletException, IOException {

    request.setCharacterEncoding("UTF-8");
    ServletContext context = getServletContext();
    String path = context.getRealPath("upload");
    String encType = "UTF-8";
    int sizeLimit = 20 * 1024 * 1024;

    MultipartRequest multi = new MultipartRequest(request, path,
sizeLimit, encType, new DefaultFileRenamePolicy());

    String name = multi.getParameter("name");
    int price = Integer.parseInt(multi.getParameter("price"));
    String description = multi.getParameter("description");
    String pictureUrl = multi.getFilesystemName("pictureUrl");
```

❖ ProductWriteServlet.java

```
ProductVO pVo = new ProductVO();  
pVo.setName(name);  
pVo.setPrice(price);  
pVo.setDescription(description);  
pVo.setPictureUrl(pictureUrl);  
  
ProductDAO pDao = ProductDAO.getInstance();  
pDao.insertProduct(pVo);  
  
response.sendRedirect("productList.do");  
}}
```

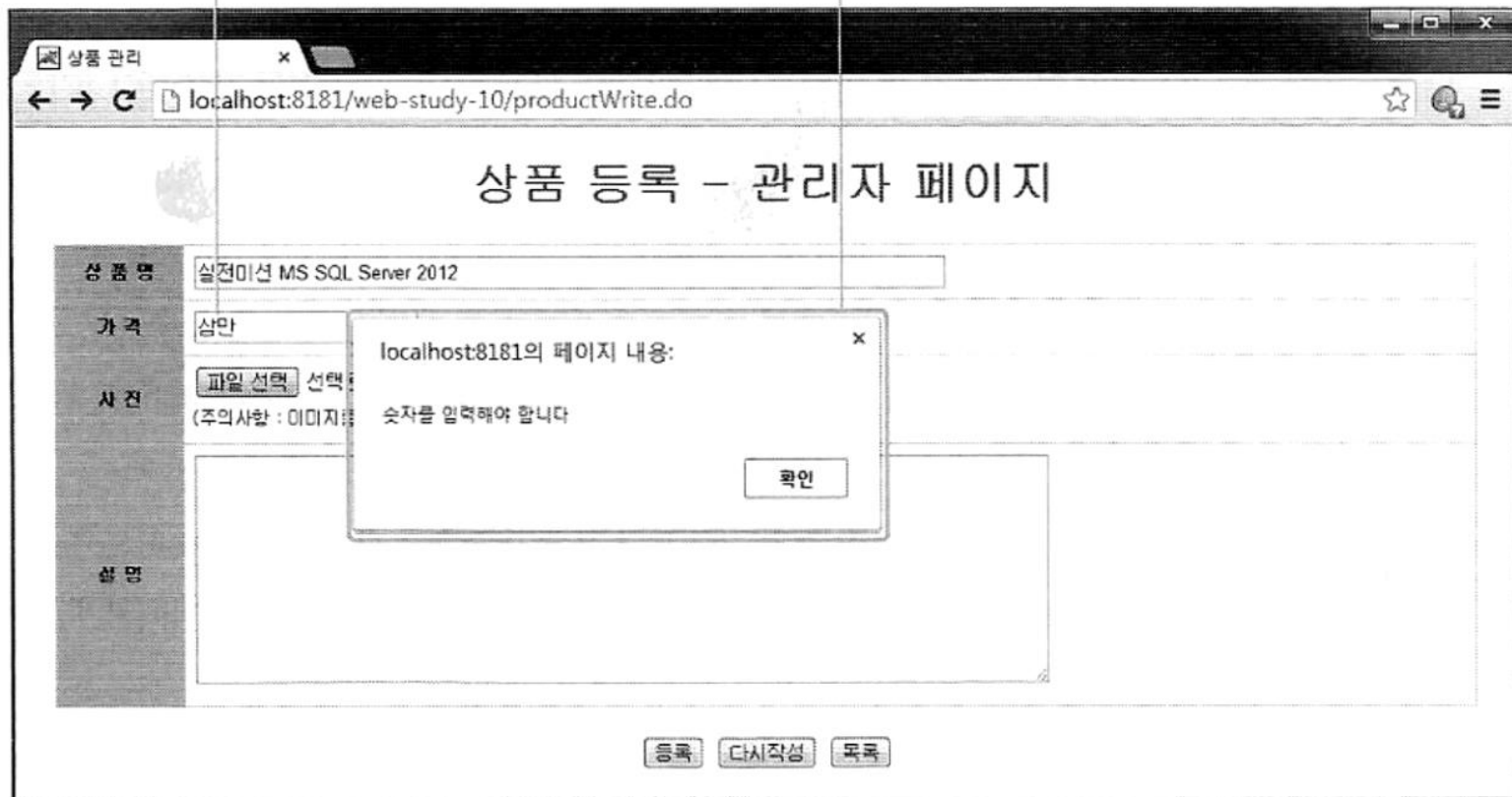
쇼핑몰 관리자 페이지

❖ 폼 데이터 유효성 검사

▼ 상품 등록 화면 에러 메시지

가격은 오라클에서 number로 자료형을 지정하였기 때문에 수치 데이터만 입력 가능하도록 유효성 체크를 해야 함

가격을 한글로 입력하였기에 에러 메시지 출력



쇼핑몰 관리자 페이지

❖ script/product.js

```
function productCheck() {  
    if (document.frm.name.value.length == 0) {  
        alert("상품명을 써주세요.");  
        frm.name.focus();  
        return false;  
    }  
    if (document.frm.price.value.length == 0) {  
        alert("가격을 써주세요");  
        frm.price.focus();  
        return false;  
    }  
    if (isNaN(document.frm.price.value)) {  
        alert("숫자를 입력해야 합니다");  
        frm.price.focus();  
        return false;  
    }  
    return true;  
}
```

❖ product/productWrite.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Insert title here</title>
    <link rel="stylesheet" type="text/css" href="css/shopping.css">
    <script type="text/javascript" src="script/product.js"></script>
</head>
<body>
    :
```

❖ 상품 수정하기

```
<a href="productUpdate.do?code=${product.code}">상품 수정</a>
```

- code 파라미터의 값을 받아 데이터베이스에서 해당 상품 정보 추출하여 수정 폼 화면으로 이동
- 수정된 내용을 다시 전송하여 데이터베이스 저장

❖ ProudctDAO.java

```
public ProductVO selectProductByCode(String code) {  
    String sql = "select * from product where code=?";  
    ProductVO pVo = null;  
    try {  
        Connection conn = null;  
        PreparedStatement pstmt = null;  
        ResultSet rs = null;  
        try {  
            conn = DBManager.getConnection();  
            pstmt = conn.prepareStatement(sql);  
            pstmt.setString(1, code);  
  
            rs = pstmt.executeQuery();  
            if (rs.next()) {  
                pVo = new ProductVO();  
                pVo.setCode(rs.getInt("code"));  
                pVo.setName(rs.getString("name"));  
                pVo.setPrice(rs.getInt("price"));  
                pVo.setPictureUrl(rs.getString("pictureUrl"));  
                pVo.setDescription(rs.getString("description"));  
            }  
        }  
    }  
}
```

❖ ProudctDAO.java

```
        catch (Exception e) {
            e.printStackTrace();
        } finally {
            DBManager.close(conn, pstmt, rs);
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return pVo;
}
```

❖ ProductUpdateServlet.java

```
@WebServlet("/productUpdate.do")
public class ProductUpdateServlet extends HttpServlet {
    :
    protected void doGet(HttpServletRequest request,
                        HttpServletResponse response)
                        throws ServletException, IOException {

        String code = request.getParameter("code");
        ProductDAO pDao = ProductDAO.getInstance();
        ProductVO pVo = pDao.selectProductByCode(code);
        request.setAttribute("product", pVo);

        RequestDispatcher dispatcher = request
            .getRequestDispatcher("product/productUpdate.jsp");
        dispatcher.forward(request, response);
    }

    protected void doPost(HttpServletRequest request,
                        HttpServletResponse response)
                        throws ServletException, IOException {
    }
}
```

❖ product/productUpdate.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
:
<script type="text/javascript" src="script/product.js"></script>
</head>
<body>
    <div id="wrap" align="center">

        <h1>상품 수정 - 관리자 페이지</h1>
        <form method="post" enctype="multipart/form-data" name="frm">

            <input type="hidden" name="code" value="${product.code}">
            <input type="hidden" name="nonmakeImg"
                value="${product.pictureUrl}">
```

❖ product/productUpdate.jsp

```
<table>
  <tr>
    <td><c:choose>
      <c:when test="${empty product.imageUrl}">
        
      </c:when>
      <c:otherwise>
        
      </c:otherwise>
    </c:choose></td>
    <td>
      <table>
        <tr>
          <th style="width: 80px">상품명</th>
          <td><input type="text" name="name"
            value="${product.name}" size="80"></td>
        </tr>
        <tr>
          <th>가 격</th>
          <td><input type="text" name="price"
            value="${product.price}"> 원</td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```

❖ product/productUpdate.jsp

```
        <tr>
            <th>사 진</th>
            <td><input type="file" name="pictureUrl"><br>
(주의사항 : 이미지를 변경하고자 할때만 선택하시오)</td>
        </tr>
        <tr>
            <th>설 명</th>
            <td><textarea cols="90" rows="10"
name="description">${product.description}</textarea>
            </td>
        </tr>
    </table>
</td>
</tr>
</table>
<br>
<input type="submit" value="수정" onclick="return productCheck()">
<input type="reset" value="다시작성">
<input type="button" value="목록"
    onclick="location.href='productList.do'">
</form>
</div>
</body>
</html>
```

❖ ProudctDAO.java - 상품 정보 수정

```
public void updateProduct(ProductVO pVo) {  
    String sql = "update product set "  
        " name=?, price=?, pictureurl=?, description=? where code=?";  
    Connection conn = null;  
    PreparedStatement pstmt = null;  
    try {  
        conn = DBManager.getConnection();  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setString(1, pVo.getName());  
        pstmt.setInt(2, pVo.getPrice());  
        pstmt.setString(3, pVo.getPictureUrl());  
        pstmt.setString(4, pVo.getDescription());  
        pstmt.setInt(5, pVo.getCode());  
  
        pstmt.executeUpdate();// 쿼리문 실행한다.  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        DBManager.close(conn, pstmt);  
    }  
}
```

❖ ProductUpdateServlet.java - 상품 정보 수정

```
@WebServlet("/productUpdate.do")
public class ProductUpdateServlet extends HttpServlet {
    :
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {

        request.setCharacterEncoding("UTF-8");
        ServletContext context = getServletContext();
        String path = context.getRealPath("upload");
        String encType = "UTF-8";
        int sizeLimit = 20 * 1024 * 1024;

        MultipartRequest multi = new MultipartRequest(request, path,
            sizeLimit, encType, new DefaultFileRenamePolicy());

        String code = multi.getParameter("code");
        String name = multi.getParameter("name");
        int price = Integer.parseInt(multi.getParameter("price"));
        String description = multi.getParameter("description");
        String pictureUrl = multi.getFilesystemName("pictureUrl");
```


❖ ProductUpdateServlet.java - 상품 정보 수정

```
        if (pictureUrl == null) {
            pictureUrl = multi.getParameter("nonmakeImg");
        }

        ProductVO pVo = new ProductVO();
        pVo.setCode(Integer.parseInt(code));
        pVo.setName(name);
        pVo.setPrice(price);
        pVo.setDescription(description);
        pVo.setPictureUrl(pictureUrl);

        ProductDAO pDao = ProductDAO.getInstance();
        pDao.updateProduct(pVo);

        response.sendRedirect("productList.do");
    }
}
```

쇼핑몰 관리자 페이지

❖ 상품 삭제하기

`상품삭제`

- GET 요청시 삭제할 상품정보 출력
- POST 요청시 실제 삭제



❖ ProductDeleteServlet.java

```
@WebServlet("/productDelete.do")
public class ProductDeleteServlet extends HttpServlet {
    :
    protected void doGet(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {

        String code = request.getParameter("code");
        ProductDAO pDao = ProductDAO.getInstance();
        ProductVO pVo = pDao.selectProductByCode(code);
        request.setAttribute("product", pVo);

        RequestDispatcher dispatcher = request
            .getRequestDispatcher("product/productDelete.jsp");
        dispatcher.forward(request, response);
    }

    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {
    }
}
```

❖ product/productDelete.jsp

```
:  
<body>  
  <div id="wrap" align="center">  
    <h1>상품 삭제 - 관리자 페이지</h1>  
    <form action="productDelete.do" method="post">  
      <table>  
        <tr>  
          <td>  
            <c:choose>  
              <c:when test="${empty product.pictureUrl}">  
                  
              </c:when>  
              <c:otherwise>  
                  
              </c:otherwise>  
            </c:choose>  
          </td>  
        </tr>  
      </table>  
    </div>  
  </body>
```

❖ product/productDelete.jsp

```
<td>
  <table>
    <tr>
      <th style="width: 80px">상 품 명</th>
      <td>${product.name}</td>
    </tr>
    <tr>
      <th>가 격</th>
      <td>${product.price}원</td>
    </tr>
    <tr>
      <th>설 명</th>
      <td><div style="height: 220px; width: 100%">
        ${product.description}</div></td>
    </tr>
  </table>
</td>
</tr>
</table>
```

❖ product/productDelete.jsp

```
        <br>
        <input type="hidden" name="code" value="${product.code}">
        <input type="submit" value="삭제">
        <input type="button" value="목록"
            onclick="location.href='productList.do'">
    </form>
</div>
</body>
</html>
```

❖ ProductDeleteServlet.java

```
@WebServlet("/productDelete.do")
public class ProductDeleteServlet extends HttpServlet {
    :
    protected void doPost(HttpServletRequest request,
                          HttpServletResponse response)
        throws ServletException, IOException {

        String code = request.getParameter("code");
        ProductDAO pDao = ProductDAO.getInstance();
        pDao.deleteProduct(code);

        response.sendRedirect("productList.do");
    }
}
```

❖ ProductDAO.java

```
public void deleteProduct(String code) {  
    String sql = "delete product where code=?";  
    Connection conn = null;  
    PreparedStatement pstmt = null;  
    try {  
        conn = DBManager.getConnection();  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setString(1, code);  
  
        pstmt.executeUpdate();// 쿼리문 실행  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        DBManager.close(conn, pstmt);  
    }  
}
```