



2023년 자바 프로그램 사관학교

데이터베이스 프로그래밍

01. 데이터베이스와 MySQL

■ 데이터베이스의 개념

- 데이터베이스Database : 여러 사람이 공유할 목적으로 방대한 데이터를 체계적으로 정리하여 저장한 것으로, 이를 이용하면 데이터를 효율적으로 관리하고 검색할 수 있다.
- 데이터베이스 관리 시스템(DBMS, DataBase Management System) : 데이터베이스를 구성하고 운영하는 소프트웨어 시스템으로 오라클, MS SQL 서버, MySQL 등 데이터베이스 제품을 말한다.

표 10-1 데이터베이스의 종류와 장단점

종류	장점	단점
오라클	<ul style="list-style-type: none">• 사용층이 가장 넓다.• 제품의 우수성이 입증되었다.• 컴퓨터는 물론 대형 서버에도 설치할 수 있다.• 공급업체의 강력한 지원을 받을 수 있다.• 분산 처리를 지원한다.• Express Edition을 제공하여 개발자가 손쉽게 접근할 수 있다.	<ul style="list-style-type: none">• DBMS를 운영하려면 많은 하드웨어 자원이 필요하다.• DBMS 관리가 복잡하다.• 동종 DBMS보다 가격이 비싸다.

01. 데이터베이스와 MySQL

MySQL

- 개발자에게 라이선스가 공개되어 있으며, 가격도 저렴해서 중소 규모의 서비스에 부담 없이 도입할 수 있다.
- 지속적인 성능 향상으로 대형 RDBMS에 사용해도 손색이 없다.
- 오라클이 인수하여 상용 라이선스 및 기술 지원을 받을 수 있게 되었다.

- 기술 지원 및 A/S를 받으려면 상용 라이선스가 필요하다.
- 상용 데이터베이스보다는 성능이 떨어질 것이라는 인식이 팽배하다.
- 대형 데이터베이스 관리 지원은 다소 부족한 편이다.
- 오라클에서 인수한 후 발전이 둔화되었다.
- MariaDB로 이전이 가속화되고 있다.

MS SQL 서버

- 사이베이스의 장점을 계승했다.
- 초기 비용이 비교적 저렴하다.
- 윈도우 서버 환경에 최적화되었다.
- 닷넷(.Net) 개발 플랫폼과 통합되었다.

- 윈도우 서버 운영체제에서만 동작한다.
- 시스템을 확장할 때 라이선스 비용이 상승한다.

IBM DB2

- IBM 제품과 호환성이 뛰어나다.
- 비정형 데이터를 관리할 수 있다.
- PureXML 기술을 사용한다.
- Venom 스토리지 압축 기술을 사용한다.

- 호환성이 제한된다.
- 공급업체의 지원 도구가 부족하다.
- IBM 위주로 시장이 편중되어 있다.

01. 데이터베이스와 MySQL

- | | | |
|--------------|--|----------------------------|
| Apache Derby | <ul style="list-style-type: none">• 100% 순수 자바 기반의 데이터베이스이다.• 애플리케이션 임베디드 데이터베이스로, 별도의 실행 과정 없이 간단하게 사용할 수 있다.• 애플리케이션의 테스트와 배포를 효율적으로 수행할 수 있다. | 일반적인 데이터 서비스 환경에는 적합하지 않다. |
|--------------|--|----------------------------|

[데이터베이스를 도입하면 얻을 수 있는 이점]

- 데이터의 중복을 최소로 줄일 수 있다.
- 데이터 불일치 문제를 해결할 수 있다.
- 데이터를 쉽게 공유할 수 있다.
- 정보 표준화를 이룰 수 있다.
- 데이터에 보안성이 제공된다.
- 데이터의 무결성Integrity이 유지된다.
- 대량의 데이터를 좀 더 빠르게 검색할 수 있다.
- 텍스트를 포함한 다양한 데이터(이미지, 파일 등)를 관리할 수 있다.
- 애플리케이션을 개발하기 쉽다

01. 데이터베이스와 MySQL

■ 데이터베이스의 개념

[데이터베이스를 도입하기 위한 요건]

- DBMS를 구동시킬 수 있는 하드웨어(서버 장비, 하드디스크)가 필요하며, 저장하는 데이터가 증가하면서 지속적으로 추가해야 한다.
- 데이터베이스를 관리하는 전문가인 DBA(DataBase Administrator)가 필요하다.
- 데이터 백업 및 복구와 관련된 전문 기술이 필요하며, 데이터를 백업하는 비용도 추가로 부담해야 한다.

01. 데이터베이스와 MySQL

■ 데이터베이스 테이블

- 테이블Table은 관계형 데이터베이스의 기본 단위로, 데이터베이스는 테이블 간의 관계를 표현한다.

그림 10-1 테이블 구조와 데이터

홍길동, 서울, 1992, 02-123-1234, 남
강동수, 남, 인천, 1993, 032-123-1111
대구, 홍길동, 여, 1991, 010-111-2222
이미녀, 1992, 여, 서울, 02-222-3333

(a) 정리되지 않은 형태

컬럼					컬럼 이름
이름	성별	거주지	출생년도	전화번호	
홍길동	남	서울	1992	02-123-1234	로우
강동수	남	인천	1993	032-123-1111	
홍길동	여	대구	1991	010-111-2222	
이미녀	여	서울	1992	02-222-3333	

(b) 정리된 형태

01. 데이터베이스와 MySQL

■ 데이터베이스 테이블

[테이블 및 테이블을 구성하는 요소]

- **테이블** : 데이터를 공통 속성으로 묶고 분류하여 기록한 형태로, 데이터베이스 관리의 기본이다.
예) 학생 정보 테이블(member)
- **컬럼** : 테이블에서 이름, 성별, 거주지, 출생년도, 전화번호 등 데이터를 구별하는 속성을 말한다. 컬럼Column 또는 필드Field라고 한다.
예) 이름(name), 성별(sex), 거주지(city), 출생년도(birth), 전화번호(tel)
- **로우** : 한 줄 단위의 데이터 집합을 말한다. 로우Row 또는 레코드Record라고 한다.
예) [그림 10-1] (b)의 첫 번째 로우 : 홍길동, 남, 서울, 1992, 02-123-1234

[테이블 구성 요소의 특징]

- 컬럼이나 로우의 위치와 순서는 아무런 의미가 없다.
- 로우는 데이터 하나만 표시할 수 있고, 그룹이나 배열은 허용하지 않는다.
- 각 컬럼은 특정한 형태의 값, 각 로우 데이터는 해당 컬럼에서 요구하는 형태의 값만 포함할 수 있다.

01. 데이터베이스와 MySQL

■ 데이터베이스 키

▶ 키

- 데이터베이스에는 각 데이터를 다른 데이터와 구분할 수 있는 고유 정보가 필요한데, 이를 키(Key)라고 한다.

▶ 주 키

- 테이블 하나에 키가 여러 개일 수 있는데, 그중 절대적으로 구분되는 키를 주 키라고 한다.
- 주 키는 테이블당 하나만 있으며, 각 로우를 구분하는 값이다.

그림 10-2 주 키 지정

주 키					
학번	이름	성별	거주지	출생년도	전화번호
201301	홍길동	남	서울	1992	02-123-1234
201302	강동수	남	인천	1993	032-123-1111
201303	홍길동	여	대구	1991	010-111-2222
201304	이미녀	여	서울	1992	02-222-3333

01. 데이터베이스와 MySQL

■ 데이터베이스 키

▶ 외래 키

- 외래 키는 테이블 간의 관계를 나타내며, 데이터의 일관성을 유지하는 데 사용한다.

그림 10-3 지역 코드 테이블

주 키

지역 코드	지역
1	서울
2	인천
3	부산
4	대구
...	...

그림 10-4 수정된 학생 정보 테이블

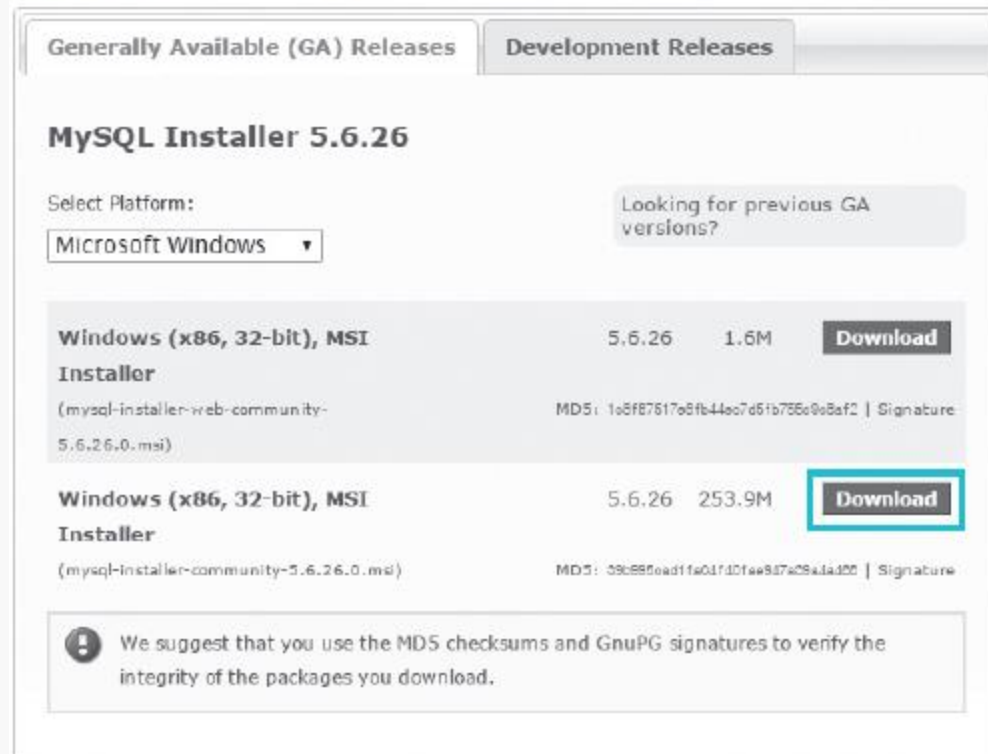
외래 키

학번	이름	성별	거주지	출생년도	전화번호
201301	홍길동	남	1	1992	02-123-1234
201302	강동수	남	2	1993	032-123-1111
201303	홍길동	여	4	1991	010-111-2222
201304	이미녀	여	1	1992	02-222-3333

01. 데이터베이스와 MySQL

- MySQL 설치
- MySQL은 대표적인 공개 데이터베이스 관리 시스템이다.

그림 10-5 윈도우 설치 파일 선택

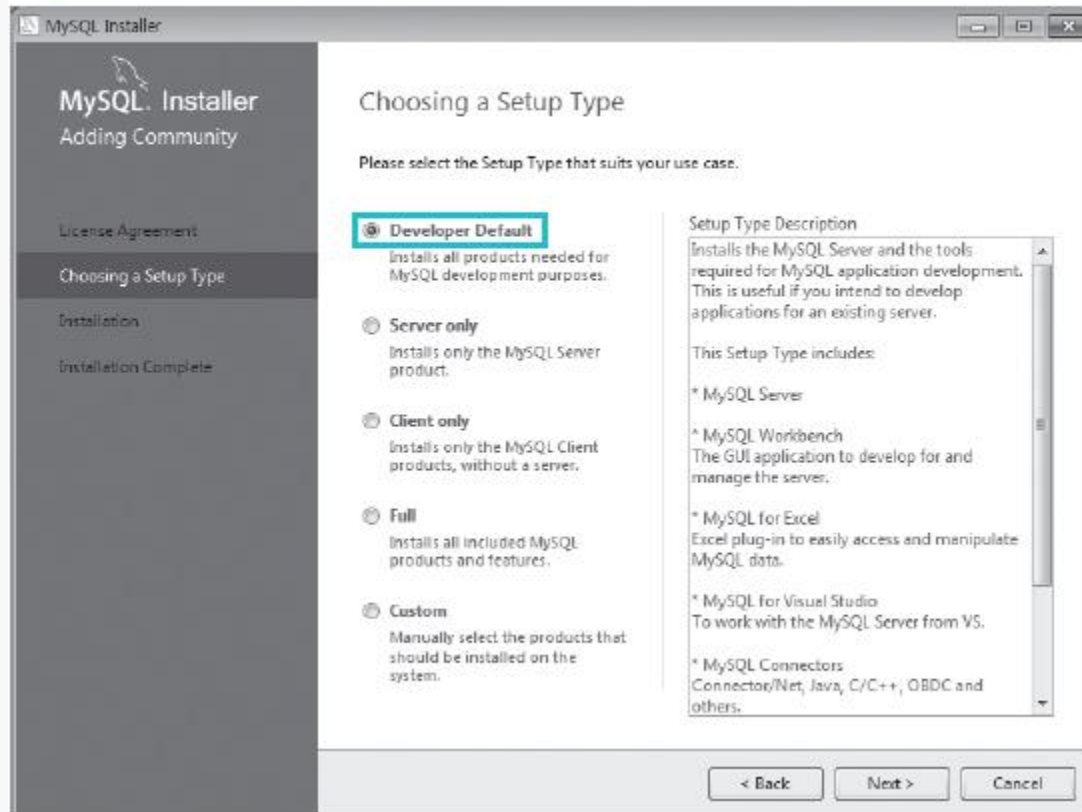


- <http://dev.mysql.com/downloads>에 접속하여 왼쪽의 [MySQL Community Server] 메뉴를 선택

01. 데이터베이스와 MySQL

■ MySQL 설치

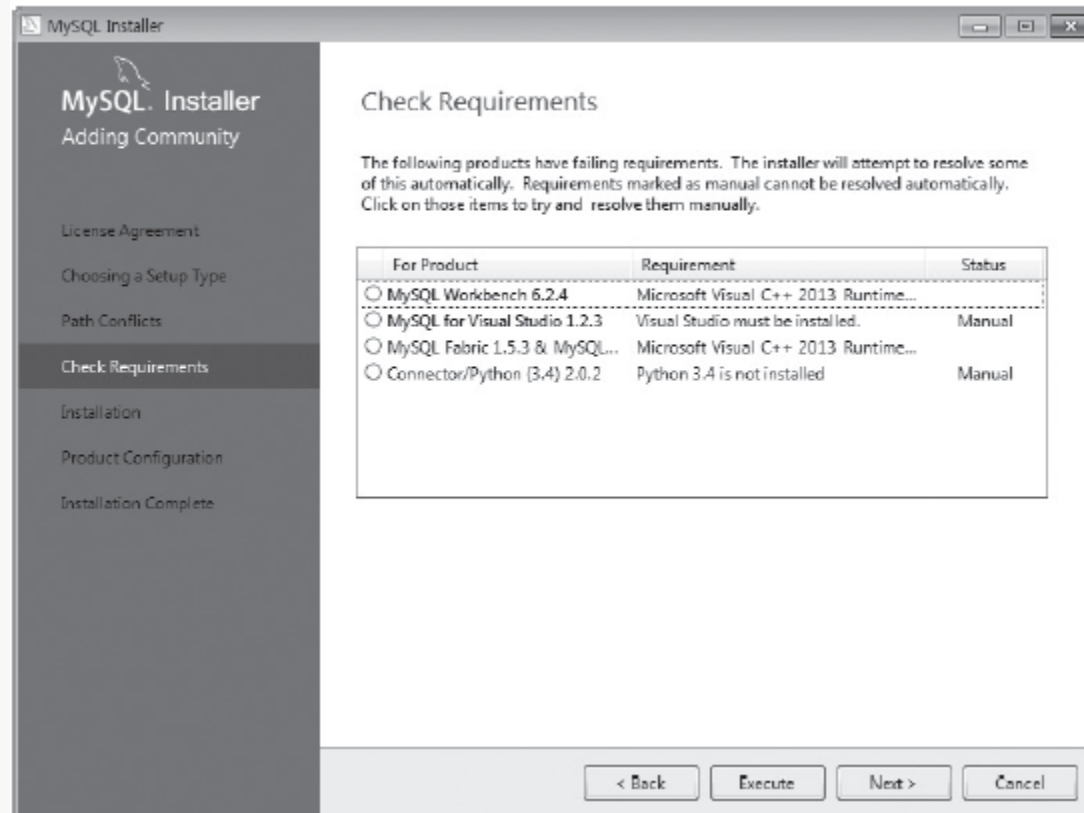
그림 10-6 설치 유형 선택



01. 데이터베이스와 MySQL

■ MySQL 설치

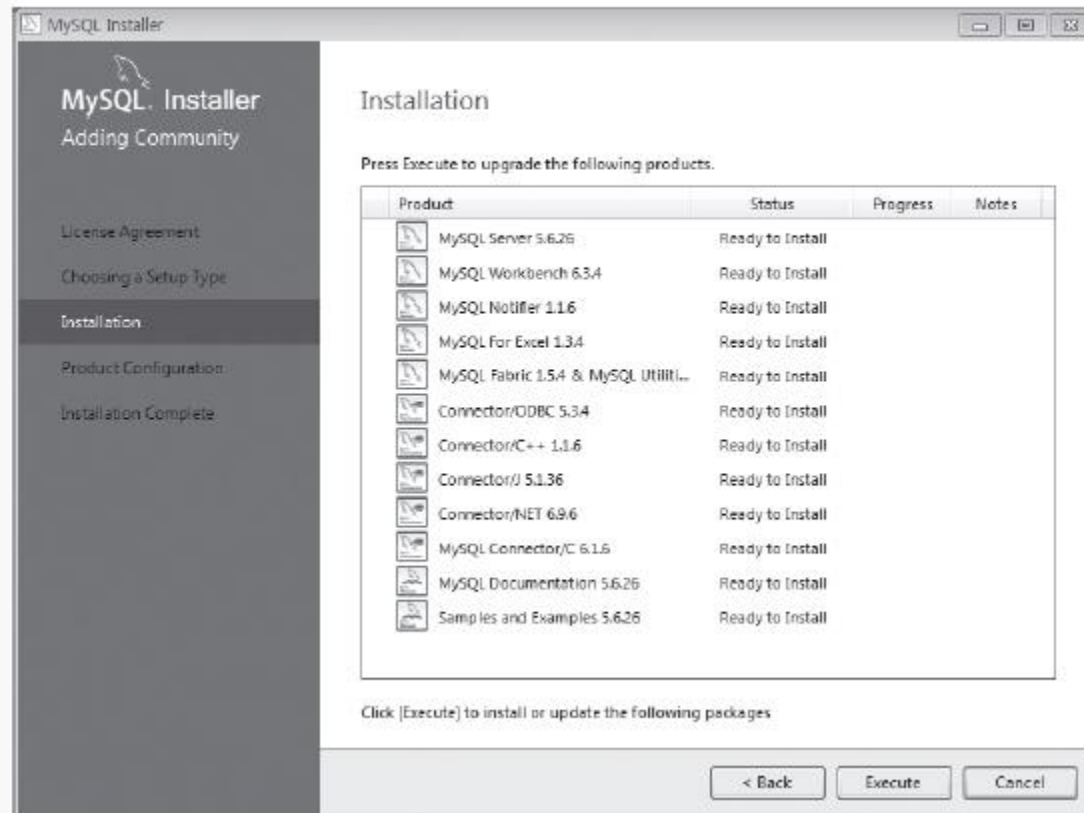
그림 10-7 각종 라이브러리 및 프로그램 설치



01. 데이터베이스와 MySQL

■ MySQL 설치

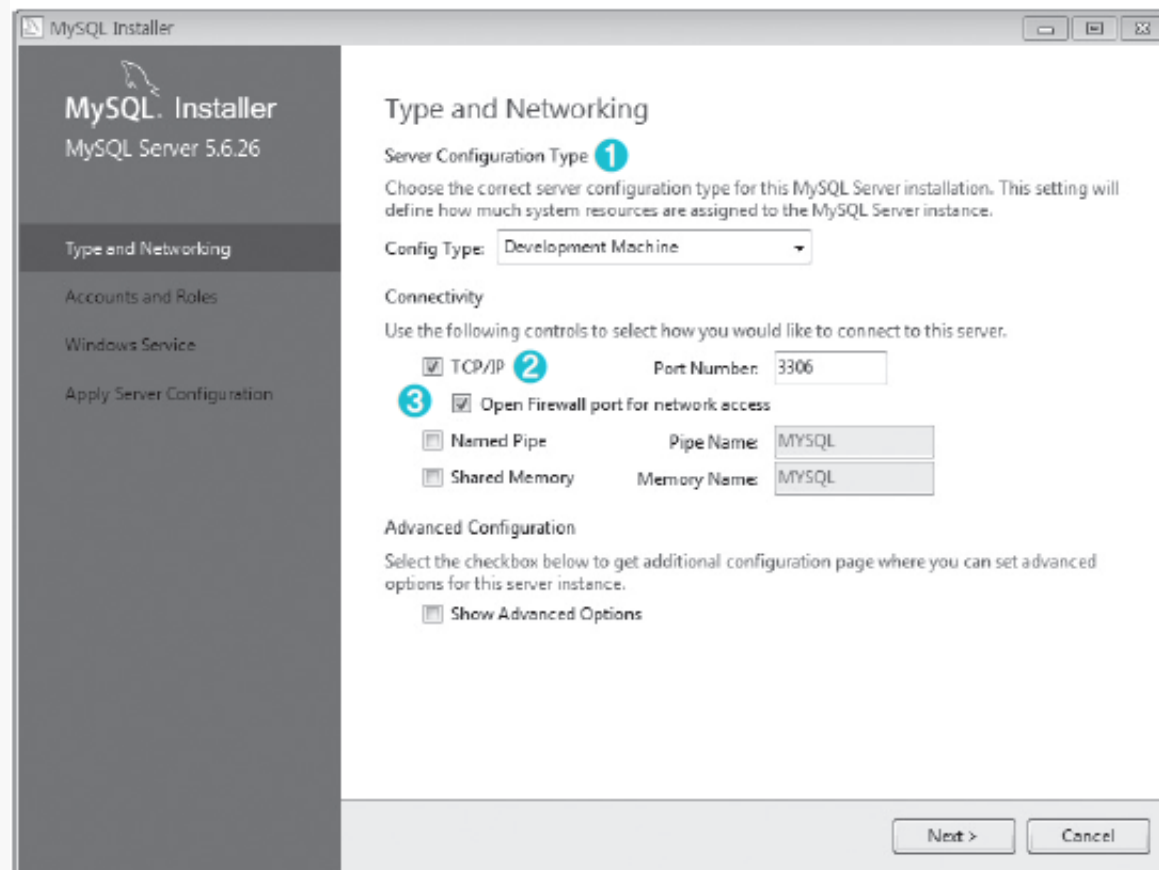
그림 10-8 필요 항목 확인 후 설치



01. 데이터베이스와 MySQL

■ MySQL 커뮤니티 서버의 기본 설정

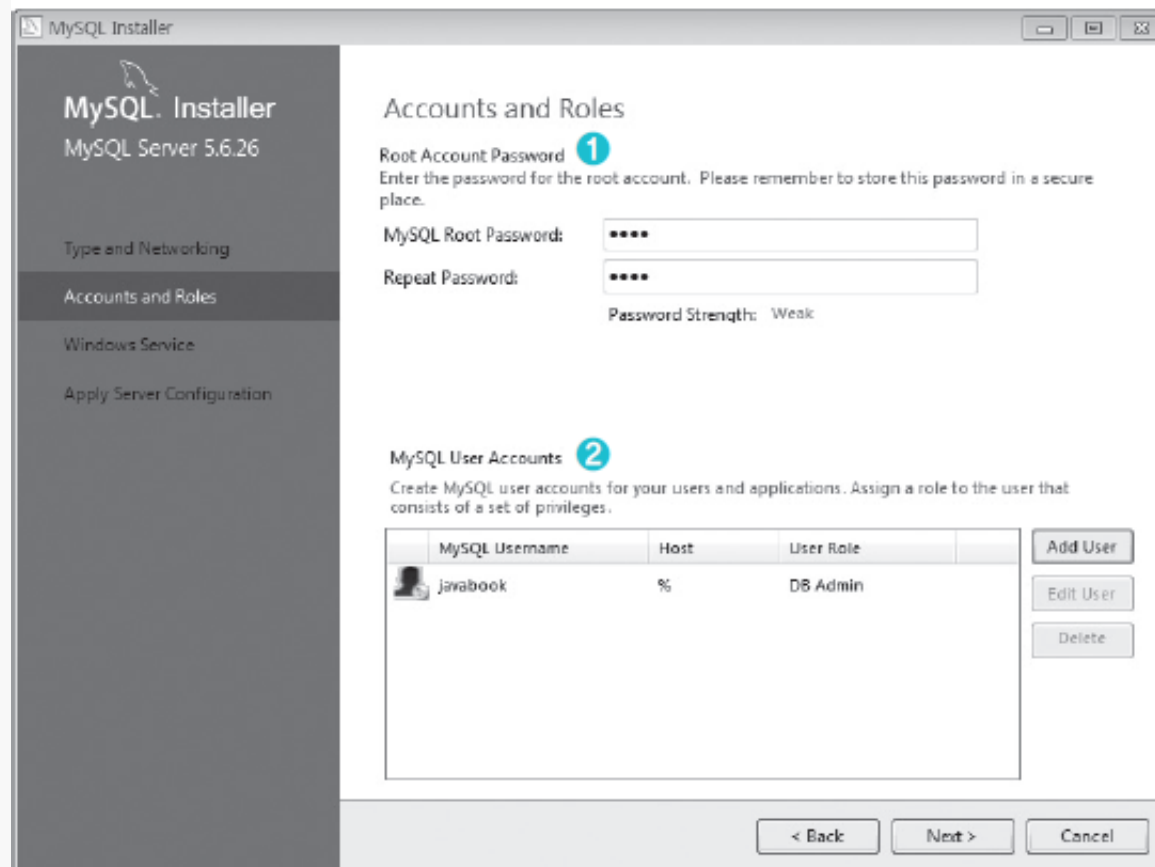
그림 10-9 서버 설정 옵션



01. 데이터베이스와 MySQL

■ MySQL 커뮤니티 서버의 기본 설정

그림 10-10 사용자 계정 설정 옵션



The image shows the 'Accounts and Roles' step of the MySQL Installer for MySQL Server 5.6.26. The left sidebar contains navigation links: 'Type and Networking', 'Accounts and Roles' (selected), 'Windows Service', and 'Apply Server Configuration'. The main area is divided into two sections. The first section, 'Root Account Password' (marked with a blue circle 1), prompts the user to enter a password for the root account, with fields for 'MySQL Root Password' and 'Repeat Password', and a 'Password Strength' indicator showing 'Weak'. The second section, 'MySQL User Accounts' (marked with a blue circle 2), prompts the user to create MySQL user accounts. It features a table with columns 'MySQL Username', 'Host', and 'User Role'. One user is listed: 'javabook' with host '%' and role 'DB Admin'. To the right of the table are buttons for 'Add User', 'Edit User', and 'Delete'. At the bottom of the window are navigation buttons: '< Back', 'Next >', and 'Cancel'.

MySQL Installer
MySQL Server 5.6.26

Type and Networking
Accounts and Roles
Windows Service
Apply Server Configuration

Accounts and Roles

Root Account Password ①
Enter the password for the root account. Please remember to store this password in a secure place.


MySQL Root Password:

Repeat Password:

Password Strength: Weak

MySQL User Accounts ②

Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL Username	Host	User Role
 javabook	%	DB Admin

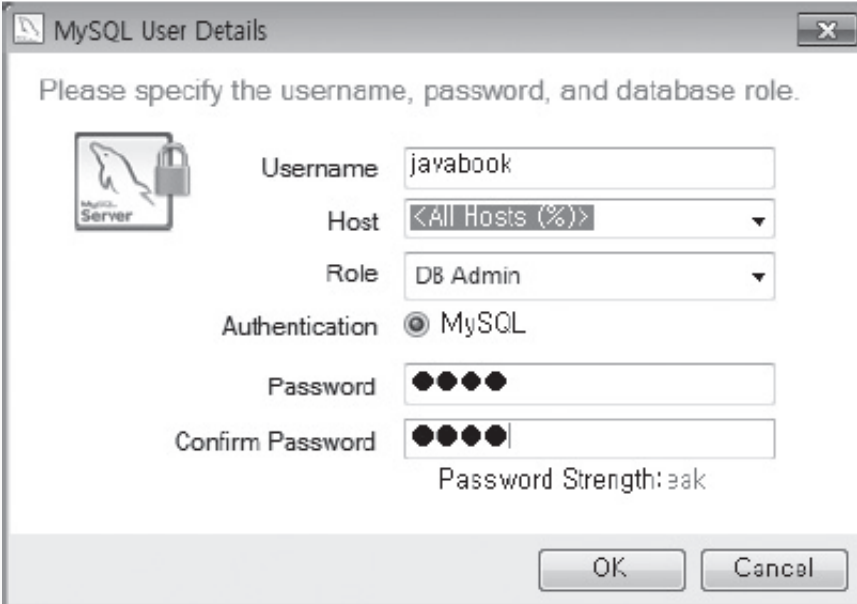
Add User
Edit User
Delete

< Back Next > Cancel

01. 데이터베이스와 MySQL

- MySQL 커뮤니티 서버의 기본 설정


그림 10-11 사용자 추가



The image shows a 'MySQL User Details' dialog box with a title bar containing a MySQL logo and a close button. The main text says 'Please specify the username, password, and database role.' On the left is a MySQL Server icon with a padlock. The form contains the following fields: 'Username' with the text 'javabook'; 'Host' with a dropdown menu showing '<All Hosts (%)>'; 'Role' with a dropdown menu showing 'DB Admin'; 'Authentication' with a radio button selected for 'MySQL'; 'Password' and 'Confirm Password' fields, both containing five black dots; and a 'Password Strength' indicator showing '3ak'. At the bottom are 'OK' and 'Cancel' buttons.

MySQL User Details

Please specify the username, password, and database role.

 Username:

Host:

Role:

Authentication: ☒ MySQL

Password:

Confirm Password:

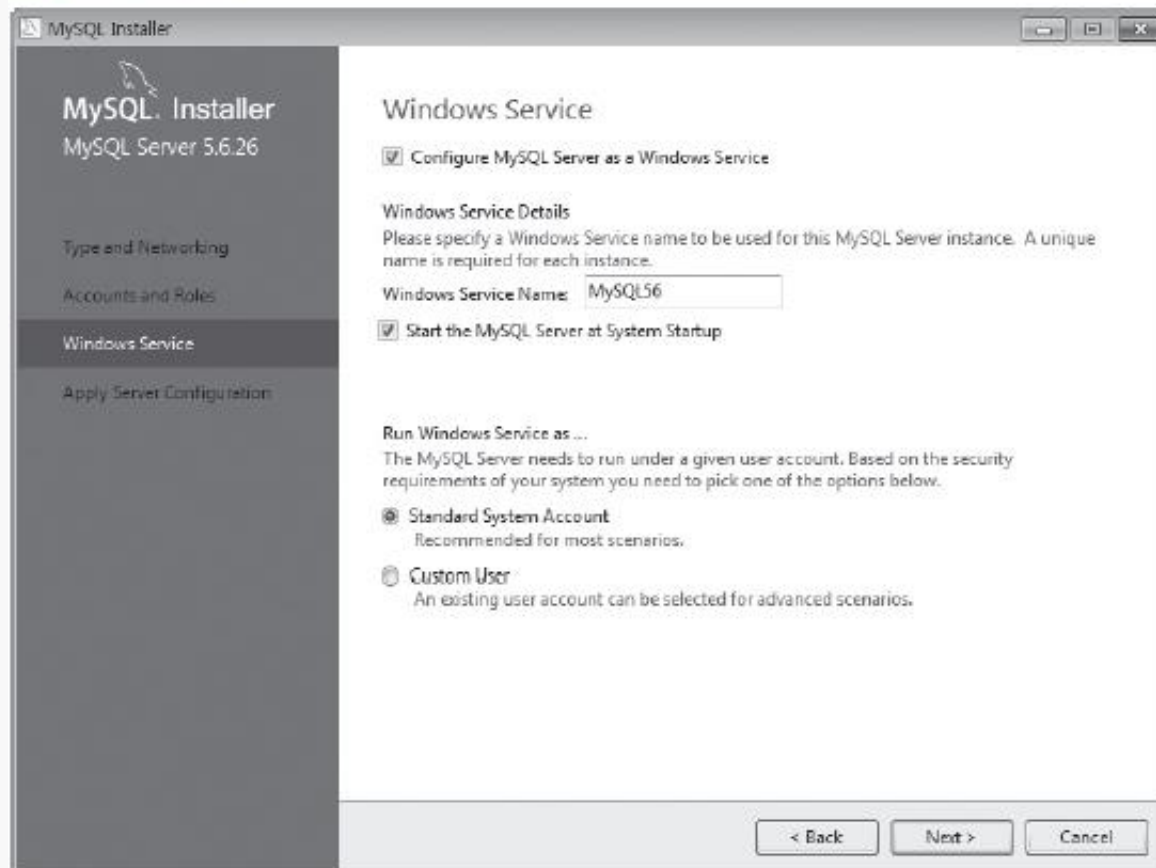
Password Strength: 3ak

OK Cancel

01. 데이터베이스와 MySQL

■ MySQL 커뮤니티 서버의 기본 설정

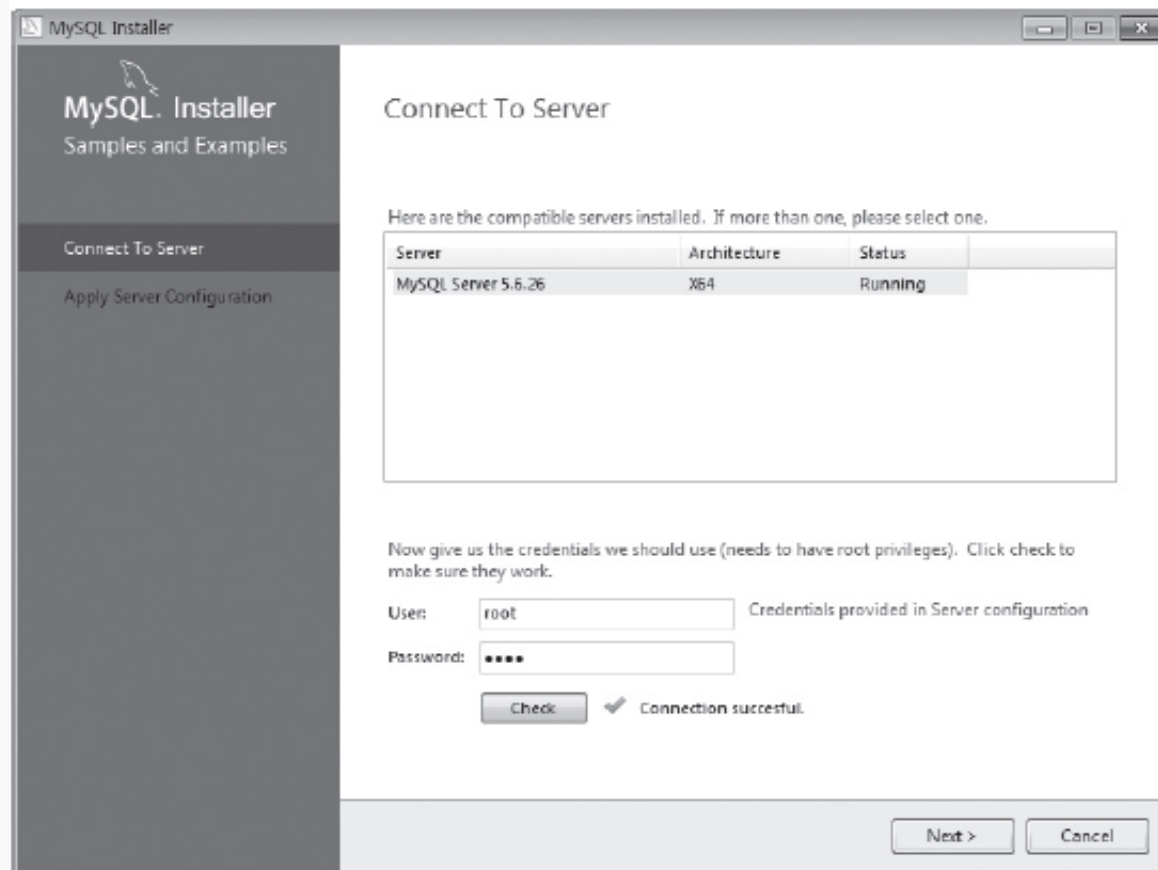
그림 10-12 윈도우 서비스 설정



01. 데이터베이스와 MySQL

- MySQL 커뮤니티 서버의 기본 설정

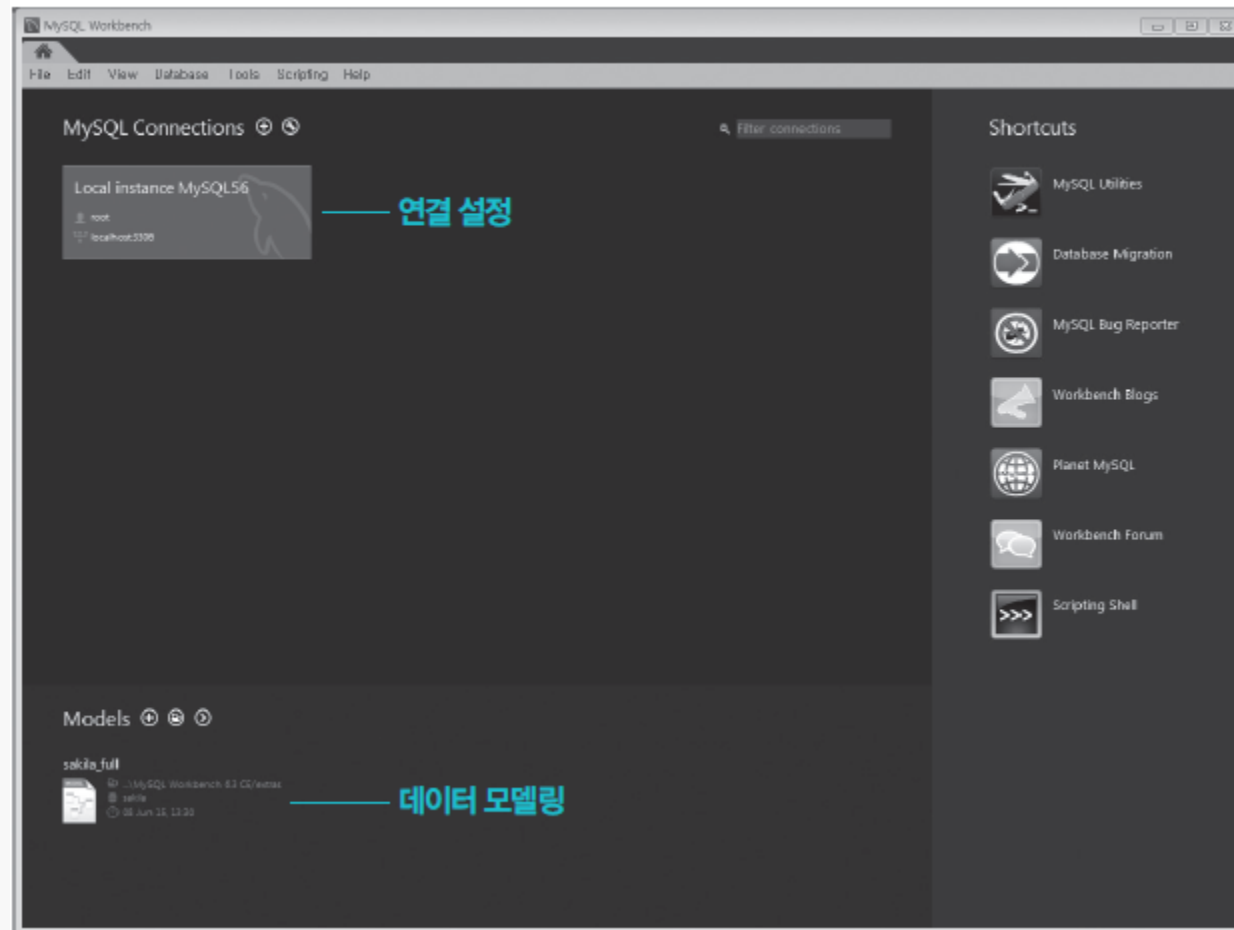
그림 10-13 Samples and Examples 설치



01. 데이터베이스와 MySQL

- MySQL 커뮤니티 서버의 기본 설정

그림 10-14 MySQL Workbench 초기 시작



01. 데이터베이스와 MySQL

■ MySQL Workbench의 연결 설정

그림 10-15 연결 설정

Setup New Connection

Connection Name: Type a name for the connection

Connection Method: Method to use to connect to the RDBMS

Parameters

Hostname: Port: Name or IP address of the server host - and TCP/IP port.

Username: Name of the user to connect with.

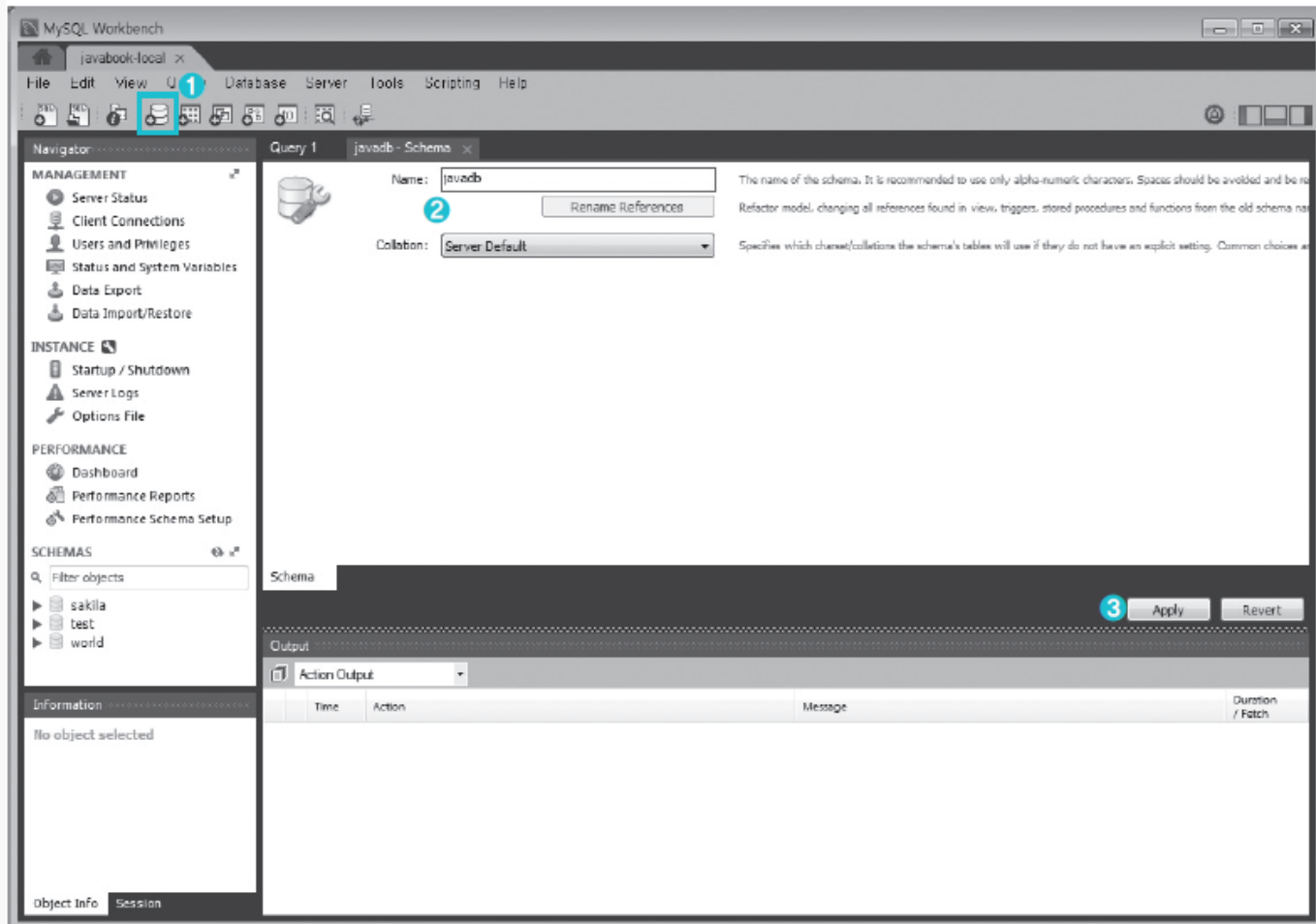
Password: The user's password. Will be requested later if it's not set.

Default Schema: The schema to use as default schema. Leave blank to select it later.

01. 데이터베이스와 MySQL

■ 스키마 생성

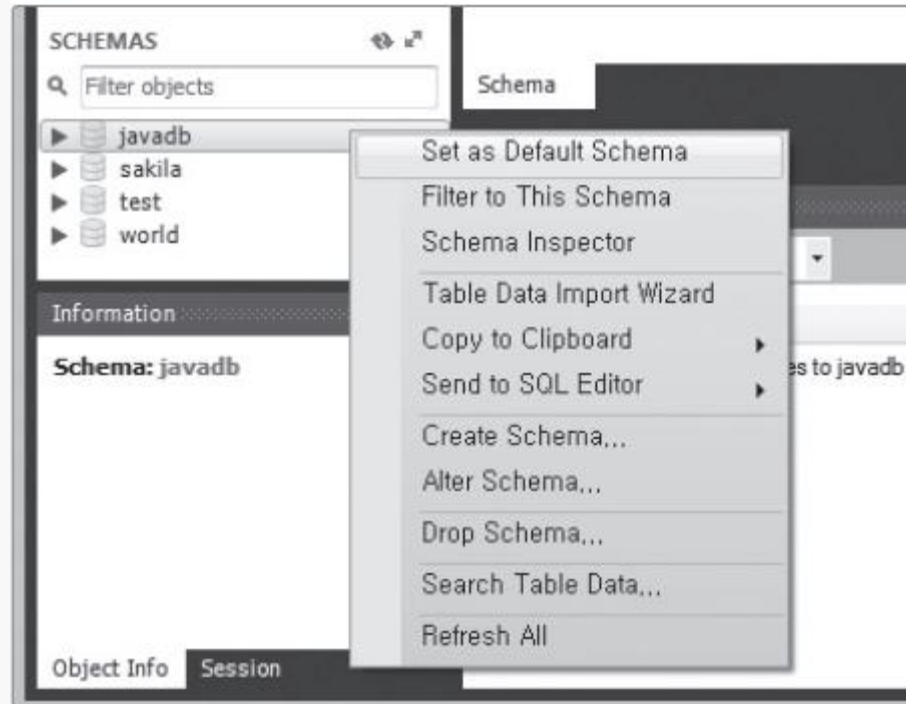
그림 10-16 javadb 스키마 생성



01. 데이터베이스와 MySQL

■ 스키마 생성

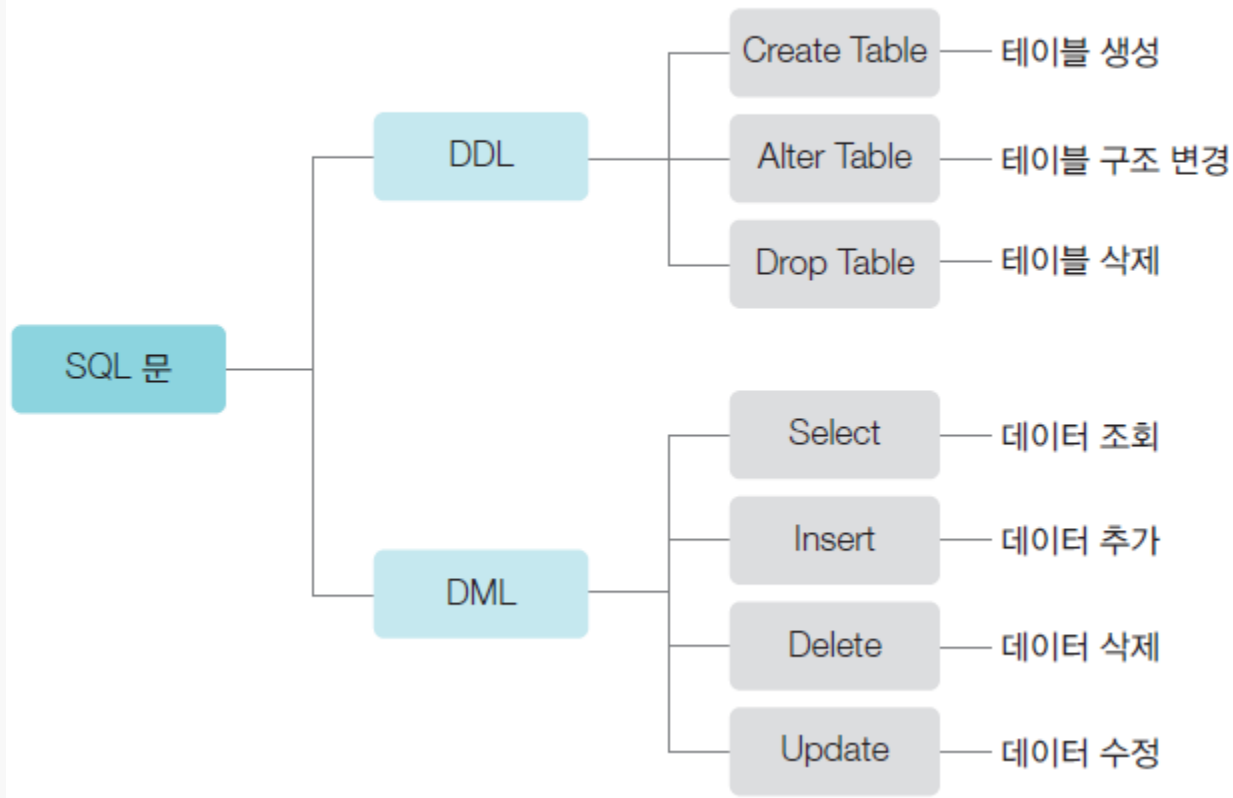
그림 10-17 javadb를 기본 스키마로 설정



01. 데이터베이스와 MySQL

■ 스키마 생성

그림 10-18 SQL 문의 종류



01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 생성 명령어 : CREATE TABLE

- CREATE TABLE은 테이블을 만들 때 사용하는 명령어

```
CREATE TABLE 테이블 이름 (  
    컬럼 이름 데이터형(크기) 옵션,  
    컬럼 이름 데이터형(크기),  
    .....  
)
```


01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 생성 명령어 : CREATE TABLE

그림 10-19 회원 관리를 위한 컬럼을 추가한 테이블 구조

id	username	dept	birth	email

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 생성 명령어 : CREATE TABLE

```
CREATE TABLE member (  
    id INT NOT NULL Primary Key,
```

주 키를 설정하는 부분이다. 중복된 데이터가 있으면 안 되므로 NOT NULL이라는 옵션도 추가한다.

```
    username VARCHAR(20),  
    dept VARCHAR(7),  
    birth DATE,  
    email VARCHAR(40)  
);
```

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 생성 명령어 : CREATE TABLE

그림 10-20 CREATE TABLE 실행 화면

MySQL Workbench interface showing the execution of a CREATE TABLE statement.

SQL Editor:

```
1 CREATE TABLE member (  
2   id INT NOT NULL Primary Key,  
3   username VARCHAR(20),  
4   dept VARCHAR(7),  
5   birth DATE,  
6   email VARCHAR(40)  
7 );  
8  
9 desc member;
```

Result Grid:

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PR		
username	varchar(20)	YES			
dept	varchar(7)	YES			
birth	date	YES			
email	varchar(40)	YES			

Output:

Time	Action	Message	Duration / Fetch
1 17:39:38	desc member	5 row(s) returned	0.015 sec / 0.000 sec

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 구조 변경 명령어 : ALTER TABLE

- ALTER TABLE은 테이블 구조를 변경할 때 사용하는 명령어이다.

ALTER TABLE 테이블 이름 add/modify/drop/enable/disable 컬럼 이름(데이터형)

표 10-2 테이블 구조 변경의 제약 조건

데이터 유무	가능한 작업
자료가 있을 때	<ul style="list-style-type: none">• 컬럼의 데이터 자료형을 변경할 수 있다.• 컬럼의 폭을 줄이거나 늘릴 수 있다.• NULL이나 NOT NULL 속성이 있는 컬럼을 추가할 수 있다.
자료가 없을 때	<ul style="list-style-type: none">• 컬럼의 데이터 자료형을 변경할 수 없다.• 컬럼의 폭을 줄일 수는 없지만 늘릴 수는 있다.• NOT NULL 속성이 있는 필드는 추가할 수 있으나 NULL 속성이 있는 필드는 추가할 수 없다.

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 구조 변경 명령어 : ALTER TABLE

- ALTER TABLE은 테이블 구조를 변경할 때 사용하는 명령어이다.

그림 10-21 전화번호(tel)가 추가된 회원 관리 테이블 구조

id	username	dept	birth	email	tel

```
ALTER TABLE member add(tel varchar(30) NOT NULL);  
ALTER TABLE member modify username varchar(10);  
ALTER TABLE member drop PRIMARY KEY;
```

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 삭제 명령어 : DROP TABLE

- DROP TABLE은 테이블을 완전히 삭제할 때 사용하는 명령어이다.

`DROP TABLE 테이블 이름`

▶ 테이블에 데이터 추가 명령어 : INSERT

- INSERT는 테이블에 새로운 데이터를 추가할 때 사용하는 명령어이다.

`INSERT INTO 테이블 이름(삽입할 컬럼 이름 ……) VALUES(컬럼에 넣을 값 ……)`

```
INSERT INTO member VALUES(201501, '홍길동', '컴퓨터공학과', '1995-08-12', 'hong@hoho.com');
INSERT INTO member VALUES(201502, '아무개', '전자과', '1995-10-03', 'ano@hoho.com');
INSERT INTO member VALUES(201503, '김학생', '컴퓨터공학과', '1996-03-05', 'khs@my.net');
INSERT INTO member(id, dept, username) VALUES(201504, '소프트웨어과', '몰라요');
```

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블 데이터 조회 명령어 : SELECT

- SELECT는 테이블에서 특정 조건에 맞는 데이터를 조회할 때 사용하는 명령어이다

```
SELECT 컬럼 이름 FROM 테이블 이름 WHERE 조건절
```

```
SELECT * FROM member;
```

```
SELECT * FROM member WHERE id = 201503;
```

```
SELECT id, username, birth FROM member WHERE id = 201502 AND username = '아무개';
```

01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블의 데이터 수정 명령어 : UPDATE

- UPDATE는 테이블에서 특정 조건에 해당하는 데이터 내용을 수정할 때 사용하는 명령어이다

```
UPDATE 테이블 이름 SET 컬럼 이름 1 = 수정할 값, 컬럼 이름 2 = 수정할 값, ..... WHERE 조건절
```

```
UPDATE member SET id = 201508, username = '김사랑' WHERE id = 201501;
```


01. 데이터베이스와 MySQL

■ SQL의 기본 명령어

▶ 테이블의 데이터 삭제 명령어 : DELETE

- DELETE는 테이블에서 특정 조건에 맞는 데이터를 삭제할 때 사용하는 명령어이다

UPDATE 테이블 이름 SET 컬럼 이름 1 = 수정할 값, 컬럼 이름 2 = 수정할 값, WHERE 조건절

```
DELETE FROM member WHERE id = 201502;
```

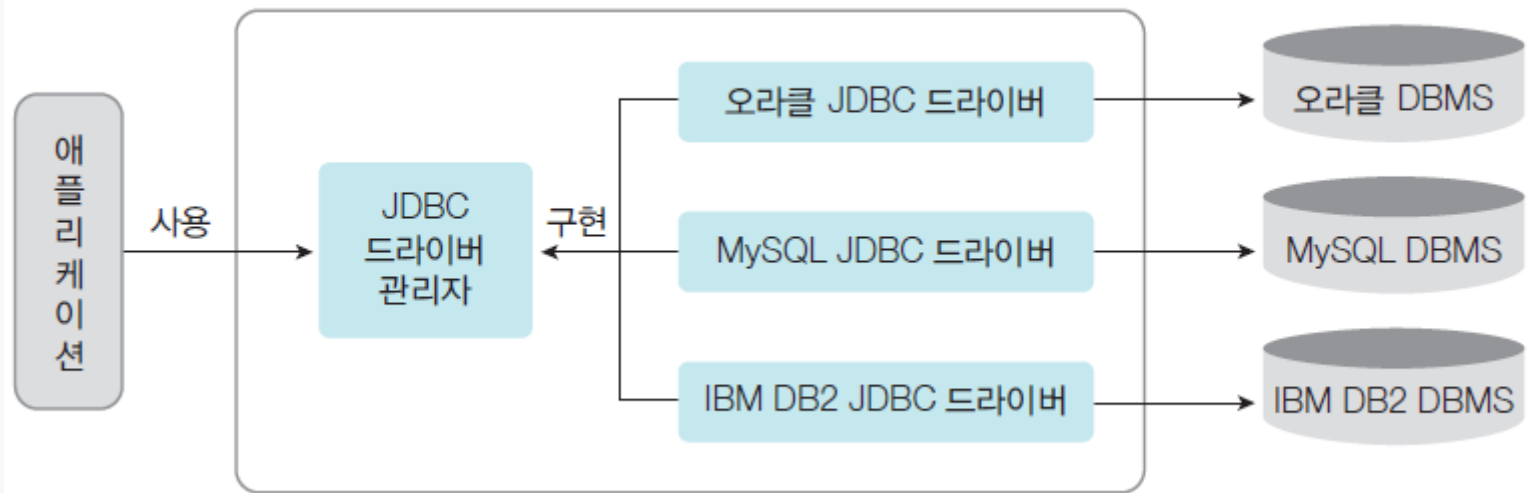
```
DELETE FROM member WHERE id = 201503 AND username = '김학생';
```

02. JDBC의 이해

■ JDBC의 개념

- JDBC는 Java DataBase Connectivity의 약어로, 자바에서 데이터베이스 연동 프로그램을 개발하려고 만든 기술.

그림 10-22 JDBC의 동작 구조

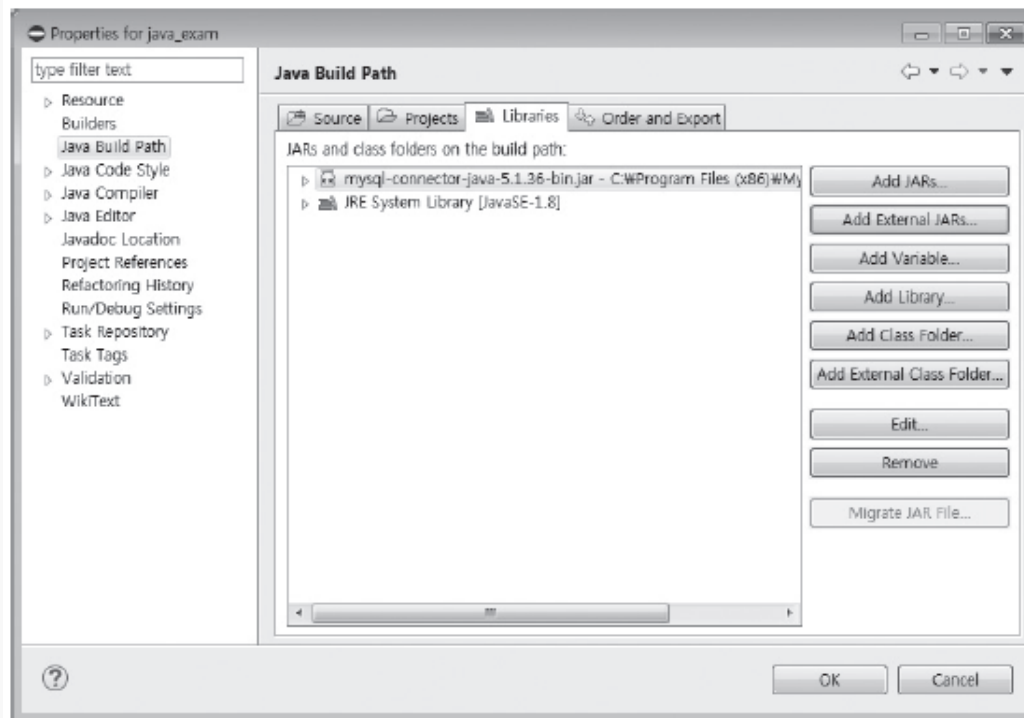


02. JDBC의 이해

■ JDBC의 개념

- JDBC는 Java DataBase Connectivity의 약어로, 자바에서 데이터베이스 연동 프로그램을 개발하려고 만든 기술.
- JDBC는 데이터베이스 연결 및 쿼리에서 표준화된 인터페이스를 정의한다. 데이터베이스 회사에서는 자신들의 데이터베이스에 맞는 JDBC 드라이버(Driver)를 개발하여 배포하기 때문에 개발자들은 데이터베이스 회사와 상관 없이 표준화된 API를 이용하여 프로그램을 개발할 수 있다.

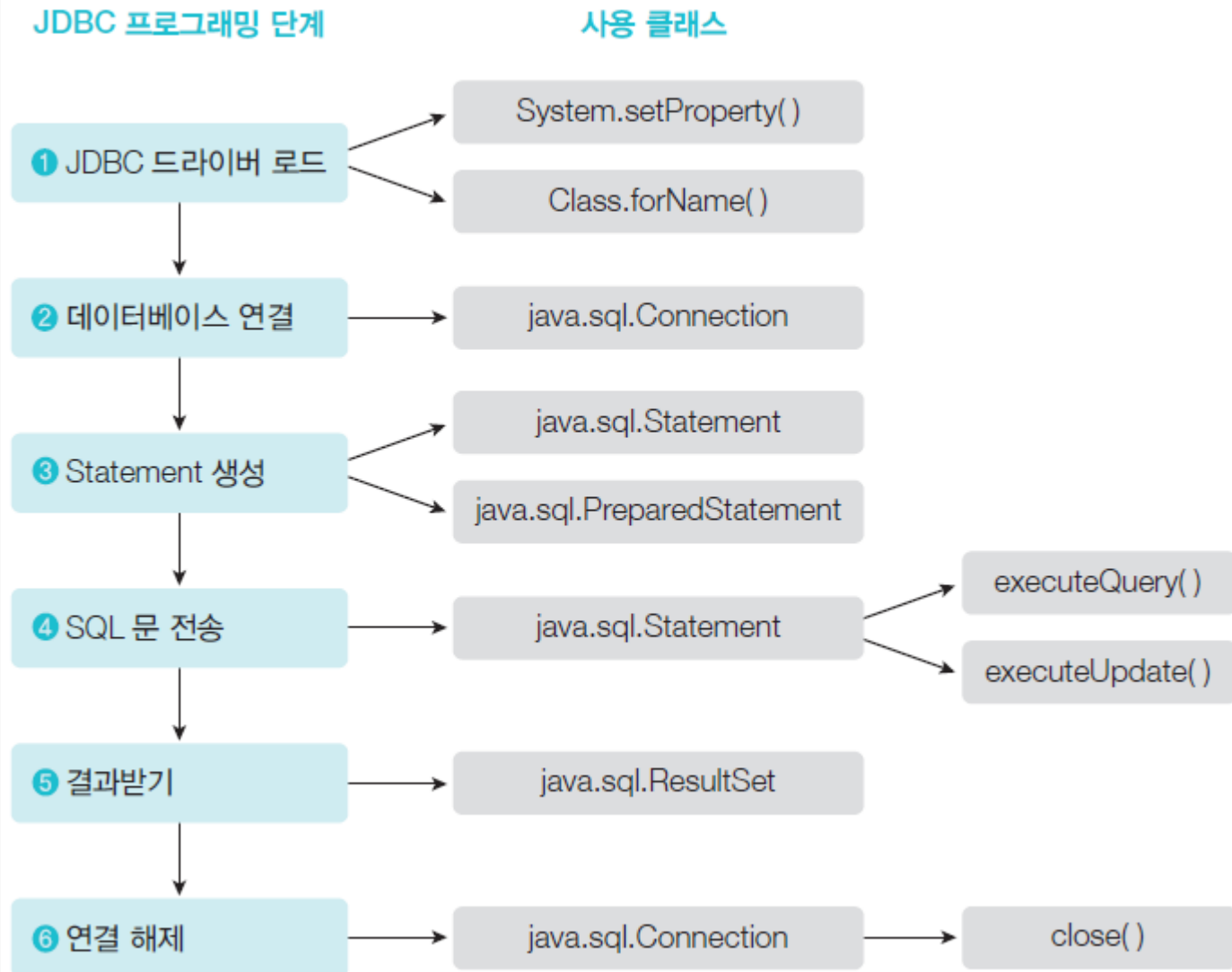
그림 10-23 빌드 경로에 MySQL JDBC 드라이버 추가



02. JDBC의 이해

■ JDBC의 개념

그림 10-24 JDBC 프로그래밍의 단계(JSP)



02. JDBC의 이해

■ JDBC의 설치 및 연결 설정 : 클래스 패스

- 자바 가상머신은 프로그램을 실행할 때 이런 클래스 라이브러리의 경로 정보를 바탕으로 해당 라이브러리를 참조하는데, 이것을 클래스 패스Class Path라고 한다.

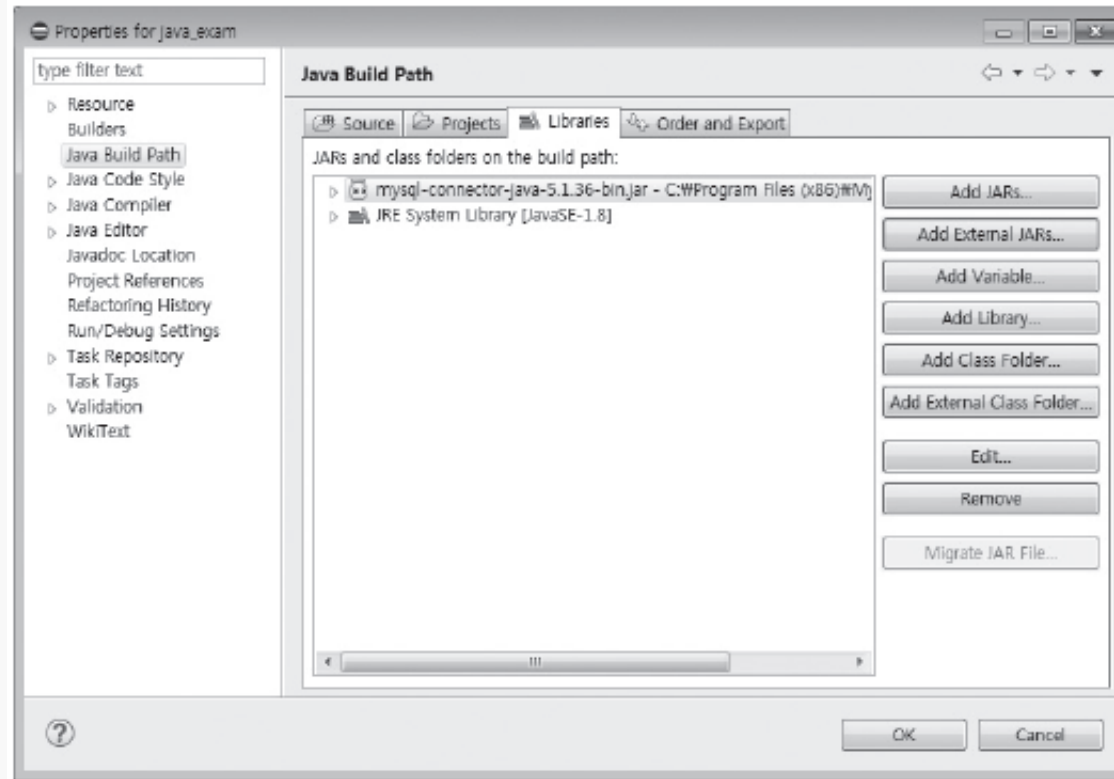
[클래스 패스의 관리법]

1. [java 설치 디렉터리\jre8\lib\ext]에 라이브러리를 복사하는 방법
2. 운영체제의 환경 변수에 CLASSPATH를 설정하는 방법
3. 이클립스 프로젝트 속성에 빌드 경로를 추가하는 방법

02. JDBC의 이해

- JDBC의 설치 및 연결 설정 : MySQL JDBC 드라이버 설치

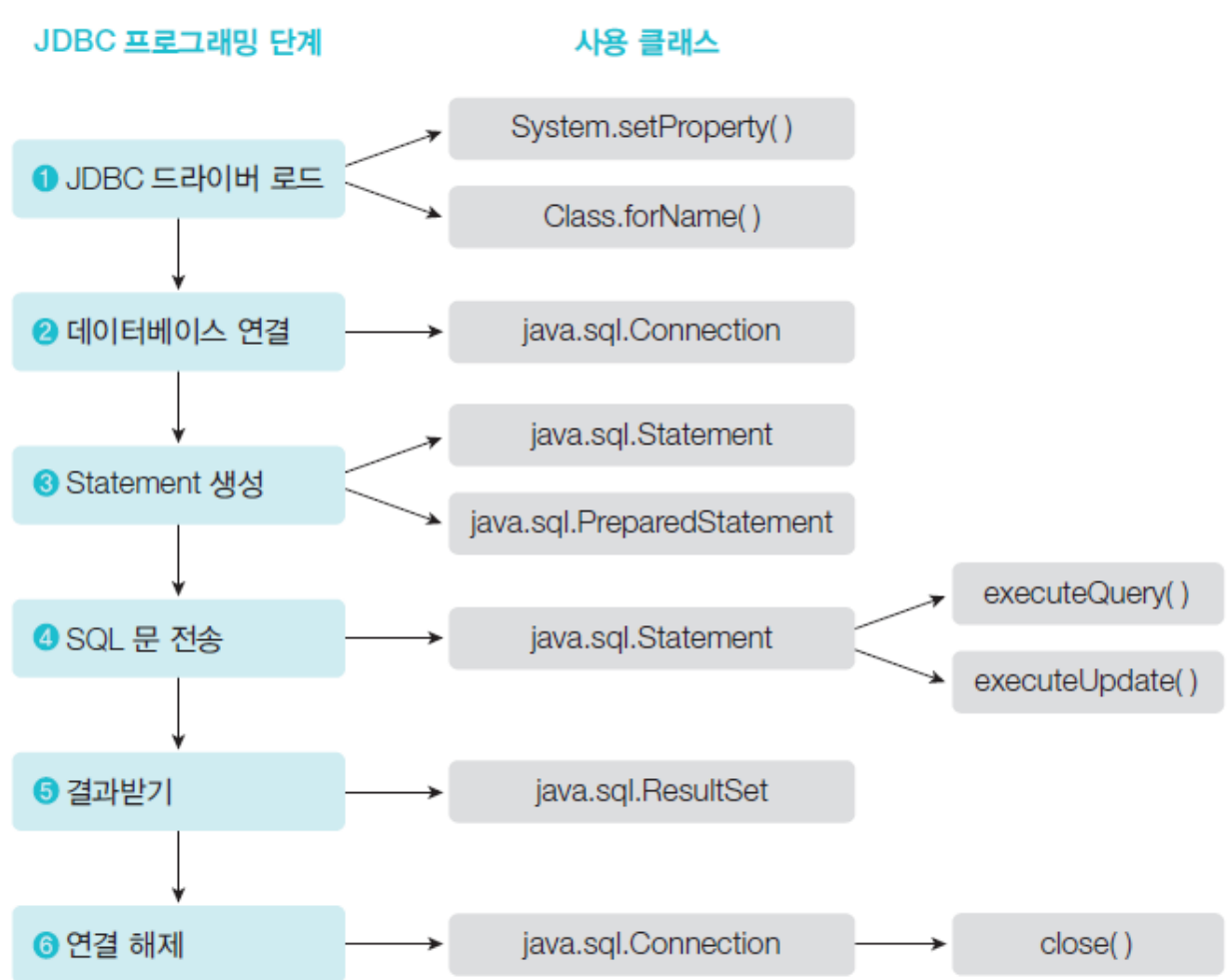
그림 10-23 빌드 경로에 MySQL JDBC 드라이버 추가



02. JDBC의 이해

■ JDBC 프로그램 개발 절차

그림 10-24 JDBC 프로그래밍의 단계(JSP)



02. JDBC의 이해

■ JDBC 프로그램 개발 절차

1단계 : JDBC 드라이버 로드

```
Class.forName("com.mysql.jdbc.Driver");
```

2단계 : 데이터베이스 연결

▶ JDBC URL

```
jdbc:<서브 프로토콜>:<데이터 원본 식별자>
```

```
jdbc:mysql://DB 서버의 IP 주소/스키마:PORT(옵션임)
```

①

②

③

① IP 주소 : MySQL 데이터베이스가 설치된 컴퓨터의 IP 주소 또는 도메인 이름이다.

② 스키마 : 데이터베이스에서 생성한 스키마(데이터베이스) 이름이다.

③ PORT : 기본 설정값인 3306 포트를 사용할 때는 생략할 수 있다.

02. JDBC의 이해

■ JDBC 프로그램 개발 절차

2단계 : 데이터베이스 연결

▶ Connection 클래스 인스턴스 레퍼런스 얻기

```
Connection conn = DriverManager.getConnection(JDBC_URL, "아이디", "비밀번호");
```

①

②

① **JDBC_URL** : 해당 데이터베이스에 맞게 미리 정의한 문자열이다.

② **아이디와 비밀번호** : 시스템에 로그인하는 계정이 아닌 데이터베이스 자체에서 관리하는 계정

02. JDBC의 이해

■ JDBC 프로그램 개발 절차

3단계 : Statement 생성

▶ executeQuery()

- SELECT 문을 수행할 때 사용

```
String sql = "select * from test";  
Statement stmt = conn.createStatement();  
stmt.executeQuery(sql);
```

▶ executeUpdate()

- UPDATE 문, DELETE 문 등을 수행할 때 사용한다.

```
Statement stmt = conn.createStatement();  
stmt.executeUpdate("insert into test values(' "+getUsername()+" ', ' "+getEmail()+" ');
```

02. JDBC의 이해

■ JDBC 프로그램 개발 절차

3단계 : Statement 생성

▶ PreparedStatement

- PreparedStatement는 SQL 문을 미리 만들어 두고 변수를 따로 입력하는 방식이므로, 효율성이 나 유지보수 측면에서 유리하다.

```
PreparedStatement pstmt = conn.prepareStatement("insert into test values(?, ?)");  
pstmt.setString(1, getUsername());  
pstmt.setString(2, getEmail());  
pstmt.executeUpdate();
```

▶ Statement의 close()

```
stmt.close();  
pstmt.close();
```

02. JDBC의 이해

■ JDBC 프로그램 개발 절차

4단계 : SQL 문 전송

- 데이터를 입력·수정·삭제하려고 SQL 문을 만들 때는 PreparedStatement를 사용하여 변수와 적절히 조합.
- 데이터의 입력, 수정, 삭제는 Statement나 PreparedStatement의 executeUpdate() 메서드를 사용

```
int count = pstmt.executeUpdate();
```

```
pstmt.executeUpdate();
```

02. JDBC의 이해

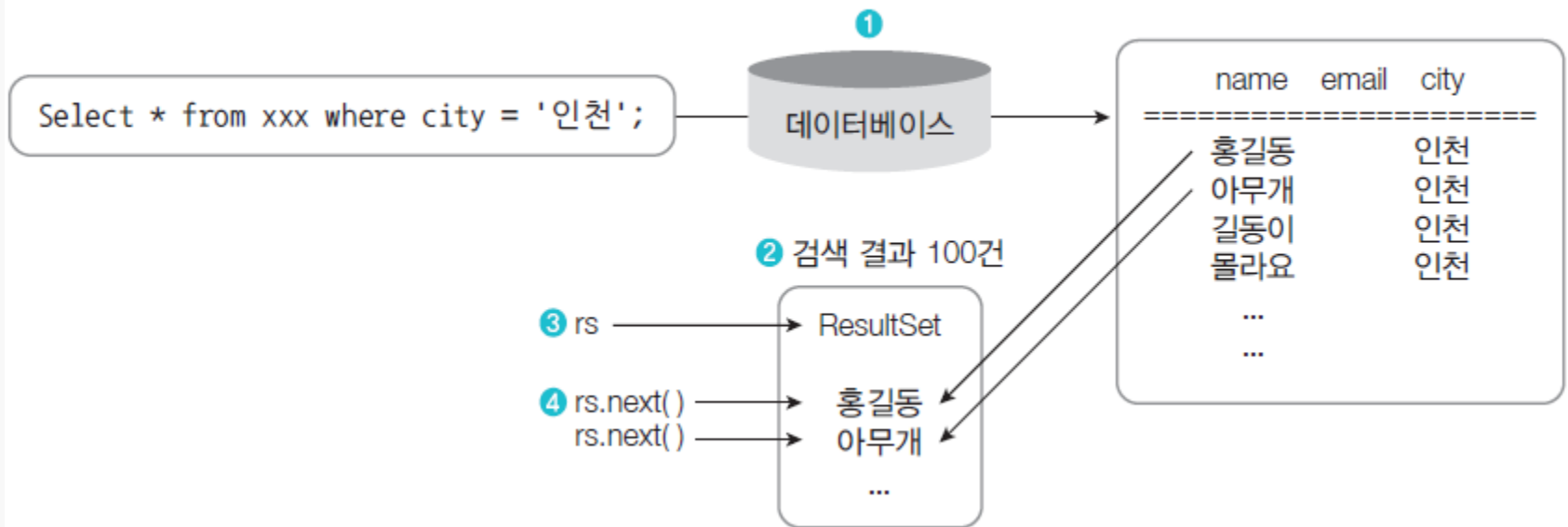
■ JDBC 프로그램 개발 절차

5단계 : 결과받기

- Statement나 PreparedStatement의 executeQuery()를 사용한다.
- 입력, 수정, 삭제와 달리 데이터를 가져올 때는 가져온 결과 데이터를 처리하는 ResultSet 객체가 필요하다.

```
ResultSet rs = pstmt.executeQuery();
```

그림 10-25 SQL 처리 결과와 ResultSet의 관계



02. JDBC의 이해

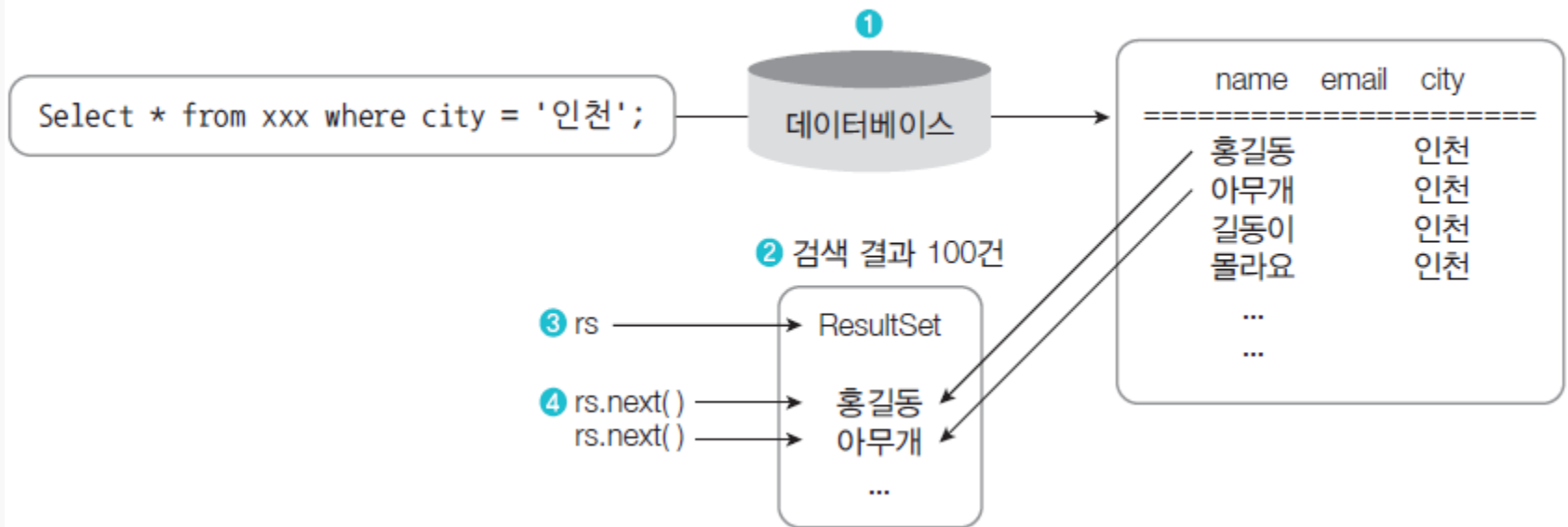
■ JDBC 프로그램 개발 절차

5단계 : 결과받기

- Statement나 PreparedStatement의 executeQuery()를 사용한다.
- 입력, 수정, 삭제와 달리 데이터를 가져올 때는 가져온 결과 데이터를 처리하는 ResultSet 객체가 필요하다.

```
ResultSet rs = pstmt.executeQuery();
```

그림 10-25 SQL 처리 결과와 ResultSet의 관계



02. JDBC의 이해

■ JDBC 프로그램 개발 절차

5단계 : 결과받기

- 로우에서 각 컬럼값을 가져오려면 rs.getXxx() 메서드를 사용

```
ResultSet rs = pstmt.executeQuery();
while(rs.next()) {
    name = rs.getString(1);    // or rs.getString("name");
    age = rs.getInt(2);        // or rs.getInt("email");
}
rs.close();
```

02. JDBC의 이해

■ JDBC 프로그램 개발 절차

6단계 : 연결 해제

데이터베이스 연결은 해당 연결을 이용한 작업이 모두 끝나는 시점에서 `close()` 메서드를 사용하여 해제해야 한다.

```
conn.close();
```

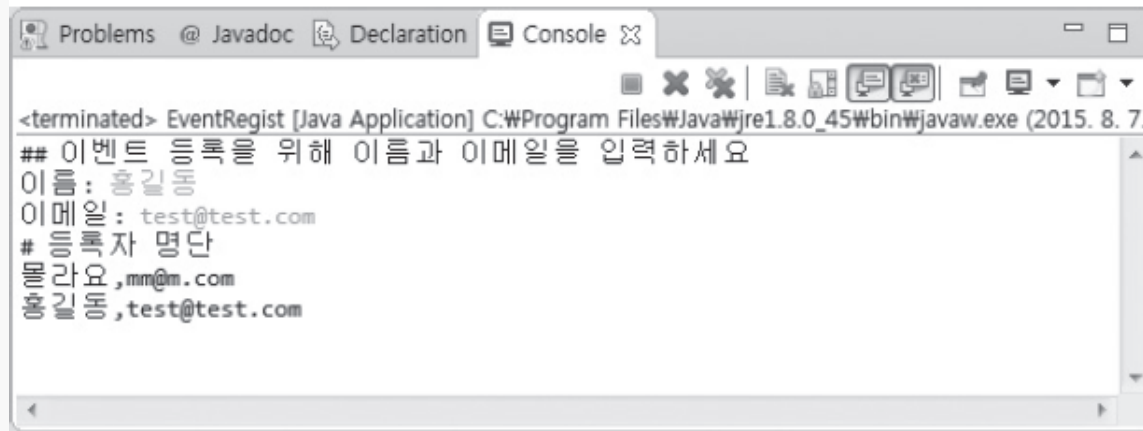

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

예제 개요 및 데이터베이스 테이블 작성

- 콘솔에서 키보드로 이름과 이메일을 입력받아 이벤트에 응모하는 응용 프로그램을 만들어 본다.

그림 10-26 이벤트 응모 프로그램의 실행 결과



[event_test 테이블 생성 스크립트]

```
CREATE TABLE event (  
    uname varchar(30),  
    email varchar(30)  
);
```

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

프로그램 작성

1. 클래스 선언과 변수 선언부

```
public class EventRegist {  
    Scanner scanner = new Scanner(System.in);  
  
    String jdbcDriver = "com.mysql.jdbc.Driver";  
    String jdbcUrl = "jdbc:mysql://localhost/javadb";  
    Connection conn;  
  
    PreparedStatement pstmt;  
    ResultSet rs;
```

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

프로그램 작성

2. 생성자

```
// 이벤트를 등록하는 생성자 메서드
public EventRegist() {
    System.out.println("## 이벤트를 등록을 위해 이름과 이메일을 입력하세요");
    System.out.print("이름: ");
    String uname = scanner.next();
    System.out.print("이메일: ");
    String email = scanner.next();

    connectDB();
    registUser(uname, email);
    printList();
    closeDB();
}
```

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

프로그램 작성

3. 데이터베이스 연결, 종료 메서드

```
// DB 연결 메서드
public void connectDB() {
    try {
        // 1단계 : JDBC 드라이버 로드
        Class.forName(jdbcDriver);

        // 2단계 : 데이터베이스 연결
        conn = DriverManager.getConnection(jdbcUrl, "javabook", "1234");
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

02. JDBC의 이해

```
// DB 연결 종료 메서드
public void closeDB() {
    try {
        // 6단계 : 연결 해제
        pstmt.close();
        rs.close();
        conn.close();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

프로그램 작성

4. 이벤트 등록 메서드

```
public void registUser(String uname, String email) {  
    String sql = "insert into event values(?, ?)";  
    try {  
        // 3단계 : Statement 생성  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setString(1, uname);  
        pstmt.setString(2, email);  
  
        // 4단계 : SQL 문 전송  
        pstmt.executeUpdate();  
    } catch (Exception e) {  
        e.printStackTrace();  
    }  
}
```

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

프로그램 작성

5. 출력 메서드

```
public void printList() {
    System.out.println("# 등록자 명단");
    String sql = "select * from event";
    try {
        pstmt = conn.prepareStatement(sql);

        // 5단계 : 결과받기
        rs = pstmt.executeQuery();
        while(rs.next()) {
            System.out.println(rs.getString("uname") + ", " + rs.getString(2));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

02. JDBC의 이해

■ JDBC를 활용한 이벤트 응모 프로그램 작성

프로그램 작성

오류 확인할 부분

- jdbcDriver, jdbcUrl 문자열 내용 확인
- MySQL 데이터베이스가 정상적으로 실행되어 동작 중인지 확인
- 스키마, 테이블 이름, 컬럼 이름 등이 모두 프로그램 소스와 일치하는지 확인
- 이클립스 프로젝트의 빌드 경로에 JDBC 드라이버가 추가되었는지 확인
- DB 사용자 계정에 접속 권한이 있는지 확인
- DB 사용자의 아이디와 비밀번호를 맞게 입력했는지 확인

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 프로젝트의 개요

- 이번에 진행할 프로젝트는 쇼핑몰 등에서 사용할 수 있는 상품 관리 프로그램이다. 상품 정보 등록, 등록된 상품 현황 조회, 정보 수정 및 삭제 기능으로 구성한다

표 10-3 상품 관리 프로젝트 클래스

클래스	설명	비고
AppMain	프로그램의 메인 클래스로, 화면 구성과 이벤트 처리를 담당한다.	View, Controller
Product	상품 정보를 표현하는 클래스로, product 테이블과 구조가 동일하다.	Data Object
ProductDAO	데이터베이스와 연동하는 데 필요한 클래스로, 데이터베이스 연결 및 입력, 수정, 삭제 등 실제 처리 기능을 제공한다.	Data Access Object

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 프로젝트의 개요

- 화면 왼쪽에는 상품 정보를 입력하거나 수정할 수 있는 양식이, 화면 오른쪽에는 등록된 상품을 모두 보여 주는 목록을 배치

그림 10-27 프로그램 동작 화면

Product Manager Application V1.0
메시지: 상품정보를 가져왔습니다.

관리번호	상품명	단가	제조사
1	60인치 스마트TV	1500000	삼성전자
2	인공지능 청소기	560000	LG전자
3	블루투스헤드셋	80000	소니

입력 폼:

- 관리번호: 2
- 상품명: 인공지능 청소기
- 단가: 560000
- 제조사: LG전자

버튼: 등록, 조회, 삭제

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 프로젝트의 개요

표 10-4 프로그램 주요 기능

주요 기능	설명	비고
상품 등록	상품 정보 입력 후 [등록] 버튼을 눌러 등록한다.	등록 후에는 오른쪽 목록이 갱신되어 새로 등록한 내용을 확인할 수 있다.
상품 수정	관리번호 콤보박스에서 원하는 상품 번호를 선택하고 [조회] 버튼을 눌러 정보를 로딩한다. 내용을 수정한 후에는 [등록] 버튼을 눌러 업데이트한다.	전체를 선택할 때는 전체 목록을 다시 출력한다.
상품 삭제	관리번호 콤보박스에서 원하는 상품 정보를 불러온 후 [삭제] 버튼을 눌러 삭제한다.	삭제 후 전체 목록을 다시 출력한다.
전체 보기	관리번호 콤보박스에서 전체를 선택하고 [조회] 버튼을 누르면 전체 목록을 다시 출력한다.	프로그램의 시작은 전체 목록 보기에서 시작한다.

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 테이블 생성

- 테이블은 product 이름으로 생성하며, MySQL Workbench를 이용하여 SQL 문을 입력하거나 테이블 생성 메뉴를 이용하여 입력 방식으로 생성할 수 있다.

```
CREATE TABLE product (  
    prcode int(11) NOT NULL AUTO_INCREMENT,  
    prname varchar(45) NOT NULL,  
    price int(11) NOT NULL,  
    manufacture varchar(20) NOT NULL,  
    PRIMARY KEY(prcode)  
) ENGINE = InnoDB AUTO_INCREMENT = 15 DEFAULT CHARSET = utf8;
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 테이블 생성

표 10-5 product 테이블 컬럼의 용도

컬럼	데이터 타입	용도
prcode	int(11)	상품 관리번호[주 키]
prname	varchar(45)	상품 이름
price	int(11)	상품 가격
manufacture	varchar(20)	제조사

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 데이터베이스 연동 클래스 구현 : 상품 정보 클래스 구현

- Product 클래스 구현 : 상품 정보 테이블의 데이터를 표현하는 Data Object 클래스로 테이블 컬럼 구조와 동일하며, 필드와 getter/setter로만 구성

예제 10-1 상품 정보 클래스 구현하기

Product.java

```
01 package javabook.ch10;
02
03 // 상품 정보 테이블 데이터 표현을 위한 클래스
04 public class Product {
05
06     // 컬럼 정보에 따른 필드 선언
07     private int prcode;
08     private String prname;
09     private int price;
10     private String manufacture;
11
12     // getter/setter 메서드
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
13     public int getPrcode() {  
14         return prcode;  
15     }  
16     public void setPrcode(int prcode) {  
17         this.prcode = prcode;  
18     }  
19     public String getPrname() {  
20         return prname;  
21     }  
22     public void setPrname(String prname) {  
23         this.prname = prname;  
24     }  
25     public int getPrice() {  
26         return price;  
27     }  
28     public void setPrice(int price) {  
29         this.price = price;  
30     }
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
31     public String getManufacture() {  
32         return manufacture;  
33     }  
34     public void setManufacture(String manufacture) {  
35         this.manufacture = manufacture;  
36     }  
37 }
```


03. JDBC를 이용한 상품 관리 프로그램 작성

- 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현
 - 상품 정보 데이터베이스와 연동하여 데이터를 처리하는 데 필요한 ProductDAO 클래스를 구현해 본다.

표 10-6 ProductDAO 클래스 주요 메서드

메서드	설명
void connectDB()	DB 연결 메서드
void closeDB()	DB 연결 종료 메서드
ArrayList<Product> getAll()	전체 Product로 구성된 ArrayList를 리턴
Product getProduct(int pcode)	파라미터의 pcode에 해당하는 상품을 리턴
boolean newProduct(Product product)	파라미터의 Product 객체의 내용을 DB에 저장
boolean delProduct(int pcode)	파라미터의 pcode에 해당하는 상품을 삭제
boolean updateProduct(Product product)	파라미터의 Product 객체의 내용으로 업데이트
Vector<String> getItems()	콤보박스용 관리번호 목록을 리턴

03. JDBC를 이용한 상품 관리 프로그램 작성

- 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현
 - 클래스 선언부

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

▶ *getAll() 메서드 구현하기

getAll()은 전체 상품 목록을 가져오는 메서드이다.

[getAll() 메서드의 앞부분]

```
public ArrayList<Product> getAll() {  
    connectDB();  
    sql = "select * from product";  
  
    // 전체 검색 데이터를 전달하는 ArrayList  
    ArrayList<Product> datas = new ArrayList<Product>();  
  
    // 관리번호 콤보박스 데이터를 위한 벡터 초기화  
    items = new Vector<String>();  
    items.add("전체");  
    .....
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

▶ *getAll() 메서드 구현하기

getAll()은 전체 상품 목록을 가져오는 메서드이다.

[while() 부분]

```
while(rs.next()) {  
    Product p = new Product();  
    p.setPrcode(rs.getInt("prcode"));  
    p.setPrname(rs.getString("prname"));  
    p.setPrice(rs.getInt("price"));  
    p.setManufacture(rs.getString("manufacture"));  
    datas.add(p);  
    items.add(String.valueOf(rs.getInt("prcode")));  
}
```

03. JDBC를 이용한 상품 관리 프로그램 작성

- 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

- ▶ *getProduct() 메서드 구현하기

prcode가 WHERE 조건절에 들어간다.

```
sql = "select * from product where prcode = ?";
Product p = null;
try {
    pstmt = conn.prepareStatement(sql);
    pstmt.setInt(1, prcode);
    rs = pstmt.executeQuery();
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

▶ *getProduct() 메서드 구현하기

prcode가 WHERE 조건절에 들어간다.

[rs.next()로 첫 번째 데이터만 처리]

```
rs.next();
p = new Product();
p.setPrcode(rs.getInt("prcode"));
p.setPrname(rs.getString("prname"));
p.setPrice(rs.getInt("price"));
p.setManufacture(rs.getString("manufacture"));
}
```

03. JDBC를 이용한 상품 관리 프로그램 작성

- 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

- ▶ newProduct() 메서드 구현하기

newProduct()는 새로운 상품을 등록하는 메서드이다.

[insert 문 작성을 할 때]

```
sql = "insert into product(prname, price, manufacture) values(?, ?, ?)";
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

▶ delProduct() 메서드 구현하기

delProduct()는 특정 상품을 삭제하는 메서드로, 파라미터로 받은 prcode를 WHERE 조건절에 두어 처리한다.

▶ updateProduct() 메서드 구현하기

updateProduct()는 newProduct()와 구조적으로는 비슷하며, WHERE를 사용하여 특정 데이터만 새로운 내용으로 업데이트한다. SQL 문에서 모든 컬럼을 업데이트하는 형식으로 처리했으며, WHERE 조건절에 prcode가 들어가는 점에 주의한다.

```
sql = "update product set pname = ?, price = ?, manufacture = ? where prcode = ?";
try {
    pstmt = conn.prepareStatement(sql);
    pstmt.setString(1, product.getPname());
    pstmt.setInt(2, product.getPrice());
    pstmt.setString(3, product.getManufacture());
    pstmt.setInt(4, product.getPrcode());
    pstmt.executeUpdate();
}
```


03. JDBC를 이용한 상품 관리 프로그램 작성

- 데이터베이스 연동 클래스 구현 : 상품 정보 데이터베이스 처리 클래스 구현

- ▶ `getItems()` 메서드 구현하기

`getItems()`는 사용자 편의를 위해 관리번호 콤보박스에 들어갈 목록을 제공하는 메서드이다. 벡터 데이터는 `getAll()`에서 추가하며, 여기서는 단순히 `items` 변수를 리턴한다.

03. JDBC를 이용한 상품 관리 프로그램 작성

- 앞에서 살펴본 내용으로 완성한 ProductDAO 클래스의 전체 코드

예제 10-2 상품 정보 데이터베이스 처리 클래스 구현하기

ProductDAO.java

```
01 package javabook.ch10;
02
03 import java.sql.*;
04 import java.util.*;
05
06 public class ProductDAO {
07     String jdbcDriver = "com.mysql.jdbc.Driver";
08     String jdbcUrl = "jdbc:mysql://localhost/javadb";
09     Connection conn;
10
11     PreparedStatement pstmt;
12     ResultSet rs;
13
14     Vector<String> items = null;
15     String sql;
16 }
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
17      // 콤보박스의 상품 관리 번호 목록을 위한 벡터 리턴
18      public Vector<String> getItems() {
19          return items;
20      }
21
22      // 전체 상품 목록을 가져오는 메서드
23      public ArrayList<Product> getAll() {
24          connectDB();
25          sql = "select * from product";
26
27          // 전체 검색 데이터를 전달하는 ArrayList
28          ArrayList<Product> datas = new ArrayList<Product>();
29
30          // 관리번호 콤보박스 데이터를 위한 벡터 초기화
31          items = new Vector<String>();
32          items.add("전체");
33
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
34         try {
35             pstmt = conn.prepareStatement(sql);
36             rs = pstmt.executeQuery();
37
38             // 검색된 데이터 수만큼 루프를 돌며 Product 객체를 만들고 이를 다시 ArrayList에 추가
39             while(rs.next()) {
40                 Product p = new Product();
41                 p.setPrcode(rs.getInt("prcode"));
42                 p.setPrname(rs.getString("prname"));
43                 p.setPrice(rs.getInt("price"));
44                 p.setManufacture(rs.getString("manufacture"));
45                 datas.add(p);
46                 items.add(String.valueOf(rs.getInt("prcode")));
47             }
48         } catch (SQLException e) {
49             e.printStackTrace();
50             return null;
51         } finally {
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
52         closeDB();
53     }
54     return datas;
55 }
56
57 // 선택한 상품 코드에 해당하는 상품 정보를 가져오는 메서드
58 public Product getProduct(int prcode) {
59     connectDB();
60     sql = "select * from product where prcode = ?";
61     Product p = null;
62     try {
63         pstmt = conn.prepareStatement(sql);
64         pstmt.setInt(1, prcode);
65         rs = pstmt.executeQuery();
66         rs.next();
67         p = new Product();
68         p.setPrcode(rs.getInt("prcode"));
69         p.setPrname(rs.getString("prname"));
70         p.setPrice(rs.getInt("price"));
71         p.setManufacture(rs.getString("manufacture"));
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
72         } catch (SQLException e) {
73             e.printStackTrace();
74             return null;
75         } finally {
76             closeDB();
77         }
78         return p;
79     }
80
81     // 새로운 상품을 등록하는 메서드
82     public boolean newProduct(Product product) {
83         connectDB();
84
85         // prcode는 자동 증가 컬럼이므로 직접 입력하지 않음
86         sql = "insert into product(prname, price, manufacture) values(?, ?, ?)";
87         try {
88             pstmt = conn.prepareStatement(sql);
89             pstmt.setString(1, product.getPrname());
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
90         pstmt.setInt(2, product.getPrice());
91         pstmt.setString(3, product.getManufacture());
92         pstmt.executeUpdate();
93     } catch (SQLException e) {
94         e.printStackTrace();
95         return false;
96     } finally {
97         closeDB();
98     }
99     return true;
100 }
101
102 // 선택한 상품을 삭제하는 메서드
103 public boolean delProduct(int prcode) {
104     connectDB();
105     sql = "delete from product where prcode = ?";
106     try {
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
107         pstmt = conn.prepareStatement(sql);
108         pstmt.setInt(1, prcode);
109         pstmt.executeUpdate();
110     } catch (SQLException e) {
111         e.printStackTrace();
112         return false;
113     } finally {
114         closeDB();
115     }
116     return true;
117 }
118 // 수정한 정보로 상품 정보를 업데이트하는 메서드
119 public boolean updateProduct(Product product) {
120     connectDB();
121     sql = "update product set pname = ?, price = ?, manufacture = ? where prcode = ?";
122     try {
123         pstmt = conn.prepareStatement(sql);
124         pstmt.setString(1, product.getPname());
125         pstmt.setInt(2, product.getPrice());
```


03. JDBC를 이용한 상품 관리 프로그램 작성

```
126         pstmt.setString(3, product.getManufacture());
127         pstmt.setInt(4, product.getPrcode());
128         pstmt.executeUpdate();
129     } catch (SQLException e) {
130         e.printStackTrace();
131         return false;
132     } finally {
133         closeDB();
134     }
135     return true;
136 }
137
138 // DB 연결 메서드
139 public void connectDB() {
140     try {
141         Class.forName(jdbcDriver);
142         conn = DriverManager.getConnection(jdbcUrl, "javabook", "1234");
143     } catch (Exception e) {
144         e.printStackTrace();
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
145         }
146     }
147
148     // DB 연결 종료 메서드
149     public void closeDB() {
150         try {
151             pstmt.close();
152             conn.close();
153         } catch (SQLException e) {
154             e.printStackTrace();
155         }
156     }
157 }
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ GUI 처리 클래스 구현 : AppMain 클래스 개요

- GUI는 AppMain 클래스에서 구현한다.

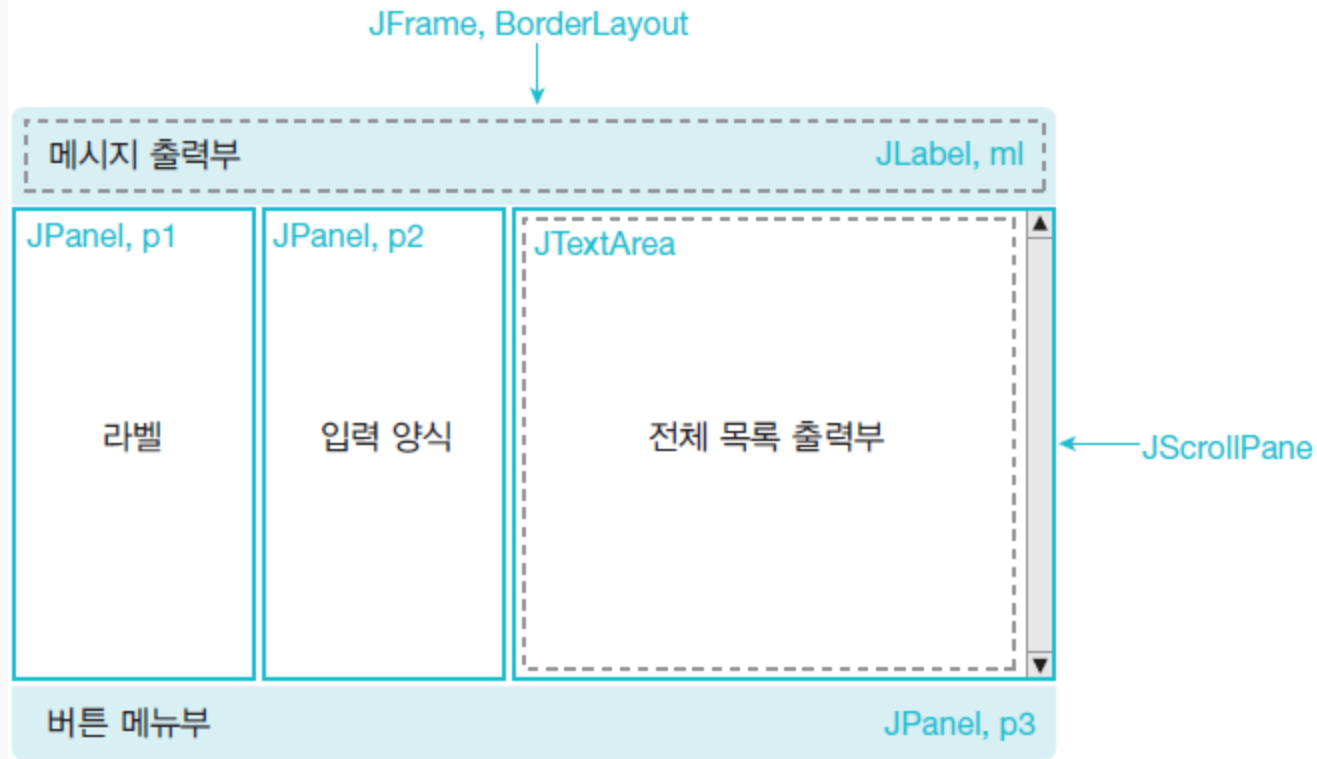
[기본 구성]

- 화면 구성을 위한 컴포넌트 선언 및 초기화
- 생성자에서 화면 크기 등 기본값 설정
- startUI() 메서드에서 화면 구성
- actionPerformed() 메서드에서 이벤트 처리
- refreshData() 메서드에서 화면 데이터 로딩 및 재로딩 처리
- clearField() 메서드에서 입력 양식 초기화

03. JDBC를 이용한 상품 관리 프로그램 작성

- GUI 처리 클래스 구현 : AppMain 클래스 개요

그림 10-28 전체 화면 컴포넌트와 레이아웃



03. JDBC를 이용한 상품 관리 프로그램 작성

- GUI 처리 클래스 구현 : 레이아웃을 이용한 패널 배치

[배치의 최종적인 BorderLayout 코드]

```
add(m1, BorderLayout.PAGE_START);  
add(p1, BorderLayout.LINE_START);  
add(p2, BorderLayout.CENTER);  
add(scroll, BorderLayout.LINE_END);  
add(p3, BorderLayout.PAGE_END);
```

03. JDBC를 이용한 상품 관리 프로그램 작성

- GUI 처리 클래스 구현 : 관리번호를 선택하는 콤보박스 구성

[콤보박스 생성 코드]

```
cb = new JComboBox();  
    ta = new JTextArea(10, 40);  
    JScrollPane scroll = new JScrollPane(ta, JScrollPane.VERTICAL_SCROLLBAR_  
ALWAYS, JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ GUI 처리 클래스 구현 : refreshData() 메서드 구현

- refreshData()는 콤보 박스 데이터 및 전체 목록 데이터를 그때그때 최신 데이터로 업데이트하는 메서드

예제 10-3 refreshData() 메서드 구현하기

```
01      public void refreshData() {
02          ta.setText("");
03          clearField();
04          editmode = false;
05
06          ta.append("관리번호\t상품명\t\t\t단가\t제조사\n");
07          datas = dao.getAll();
08
09          // 데이터를 변경하면 콤보박스 데이터 갱신
10          cb.setModel(new DefaultComboBoxModel(dao.getItems()));
11      }
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
12         if(datas != null) {
13             // ArrayList의 전체 데이터를 형식에 맞춰 출력
14             for(Product p : datas) {
15                 StringBuffer sb = new StringBuffer();
16                 sb.append(p.getPrcode() + "\t");
17                 sb.append(p.getPrname() + "\t\t");
18                 sb.append(p.getPrice() + "\t");
19                 sb.append(p.getManufacture() + "\n");
20                 ta.append(sb.toString());
21             }
22         }
23         else {
24             ta.append("등록된 상품이 없습니다. !!\n상품을 등록해 주세요 !!");
25         }
26     }
```


03. JDBC를 이용한 상품 관리 프로그램 작성

■ 이벤트 처리

- 프로그램 시작
- [등록] 버튼 클릭
- [조회] 버튼 클릭
- [삭제] 버튼 클릭

03. JDBC를 이용한 상품 관리 프로그램 작성

■ 프로그램 시작

- 다른 모든 프로그램과 마찬가지로 시작은 main()에서 처리한다. 먼저 AppMain 객체를 생성하고 startUI() 메서드를 호출하는 것으로 시작 이벤트의 동작이 마무리된다.

■ [등록] 버튼 클릭

- 화면에 데이터를 입력하고 [등록] 버튼을 누르면 발생하는 이벤트로, actionPerformed() 메서드에서는 getSource()가 [등록] 버튼인 b1의 경우에 해당한다.

```
if(obj == b1) {  
    product = new Product();  
    product.setPrname(t1.getText());  
    product.setPrice(Integer.parseInt(t2.getText()));  
    product.setManufacture(t3.getText());  
  
    // 수정일 때  
    if(editmode == true) {  
        product.setPrcode(Integer.parseInt((String)cb.getSelectedItem()));  
    }  
}
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
        if(dao.updateProduct(product)) {
            ml.setText(msg + "상품을 수정했습니다!!");
            clearField();
            editmode = false;
        }
        else
            ml.setText(msg + "상품 수정이 실패했습니다!!");
    }
    // 등록일 때
    else {
        if(dao.newProduct(product)) {
            ml.setText(msg + "상품을 등록했습니다!!");
        }
        else
            ml.setText(msg + "상품 등록이 실패했습니다!!");
    }
    // 화면 데이터 갱신
    refreshData();
}
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ [등록] 버튼 클릭

- 등록·수정 모두 입력 양식에 있는 데이터가 필요하기 때문에 Product 객체를 생성하고 각 필드에 데이터를 입력한다.

```
product.setPrcode(Integer.parseInt((String)cb.getSelectedItem()));
```

03. JDBC를 이용한 상품 관리 프로그램 작성

■ [조회] 버튼 클릭

- [조회] 버튼은 b2이고 콤보박스에 선택된 pcode에 해당하는 데이터를 가져와 양식에 출력한다. 이때 '전체'를 선택하면 화면을 지우고 새로운 데이터를 로딩한다

```
// 조회 버튼일 때
else if(obj == b2) {
    String s = (String)cb.getSelectedItem();
    if(!s.equals("전체")) {
        product = dao.getProduct(Integer.parseInt(s));
        if(product != null) {
            m1.setText(msg + "상품정보를 가져왔습니다!!");
            t1.setText(product.getPrname());
            t2.setText(String.valueOf(product.getPrice()));
            t3.setText(product.getManufacture());
            // cb.setSelectedIndex(anIndex);
            editmode = true;
        }
    }
}
```

03. JDBC를 이용한 상품 관리 프로그램 작성

```
        else {  
            m1.setText(msg + "상품이 검색되지 않았습니다!!");  
        }  
    }  
}
```

- 가격을 양식, 즉 JTextField에 출력할 때는 숫자값을 문자열로 처리해야 하므로 String.valueOf()를 사용한다

```
t2.setText(String.valueOf(product.getPrice()));
```

03. JDBC를 이용한 상품 관리 프로그램 작성


■ [삭제] 버튼 클릭

```
// 삭제 버튼일 때
else if(obj == b3) {
    String s = (String)cb.getSelectedItem();
    if(s.equals("전체")) {
        ml.setText(msg + "전체 삭제는 되지 않습니다!!");
    }
    else {
        if(dao.delProduct(Integer.parseInt(s))) {
            ml.setText(msg + "상품이 삭제되었습니다!!");
        }
        else {
            ml.setText(msg + "상품이 삭제되지 않았습니다!!");
        }
    }
    // 화면 데이터 갱신
    refreshData();
}
```

03. JDBC를 이용한 상품 관리 프로그램 작성

- 프로그램 실행 및 테스트

그림 10-29 최초 실행 화면



Product Manager Application V1.0

##메시지: 프로그램이 시작되었습니다.

관리번호: 전체

상품명:

단가:

제조사:

관리번호	상품명	단가	제조사
------	-----	----	-----

등록 조회 삭제

03. JDBC를 이용한 상품 관리 프로그램 작성

- 프로그램 실행 및 테스트

그림 10-30 등록 후 관리번호 선택

The screenshot shows a window titled "Product Manager Application V1.0" with a message bar that says "## 메시지: 상품정보를 가져왔습니다." Below the message bar, there are four input fields on the left: "관리번호" (Management Number) with a dropdown menu showing "전체" (All), "상품명" (Product Name) with a list showing "전체", "1", "2", "3", "단가" (Unit Price) with a text box showing "3", and "제조사" (Manufacturer) with a text box. On the right, there is a table with four columns: "관리번호", "상품명", "단가", and "제조사". The table contains three rows of data. At the bottom, there are three buttons: "등록" (Register), "조회" (Search), and "삭제" (Delete).

관리번호	상품명	단가	제조사
1	스마트TV	1500000	삼성전자
2	인공지능 청소기	580000	LG전자
3	블루투스 헤드셋	80000	소니

03. JDBC를 이용한 상품 관리 프로그램 작성

- 프로그램 실행 및 테스트

그림 10-31 두 번 상품 조회

Product Manager Application V1.0

메시지: 상품정보를 가져왔습니다

관리번호: 2

상품명: 인공지능 청소기

단가: 580000

제조사: LG전자

관리번호	상품명	단가	제조사
1	스마트TV	1500000	삼성전자
2	인공지능 청소기	580000	LG전자
3	블루투스 헤드셋	80000	소니

등록 조회 삭제