어서와 Java는 처음이지!

제19장 데이터베이스 프로그래밍

학습목차

데이터베이스

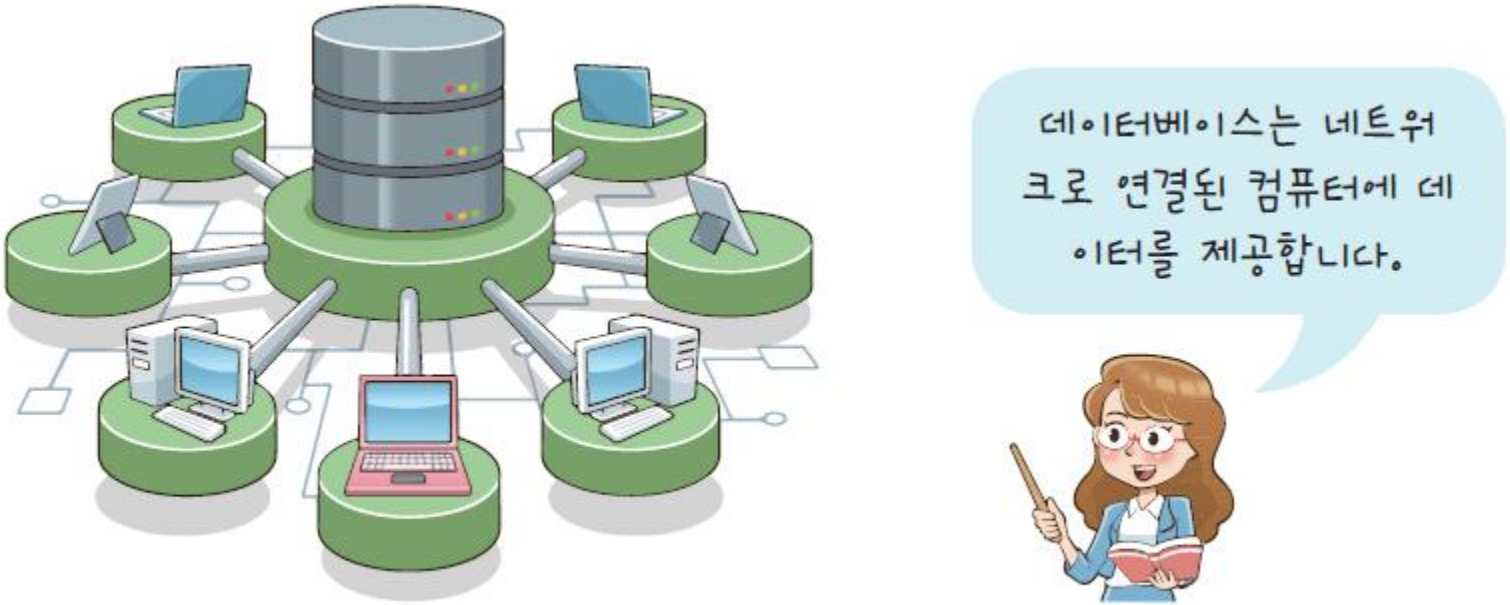


그림 19-1 • 데이터베이스

자바와 데이터베이스

- JDBC(Java Database Connectivity)는 자바 API의 하나로서 데이터베이스에 연결하여서 데이터베이스 안의 데이터에 대하여 검색하고 데이터를 변경할 수 있게 한다.

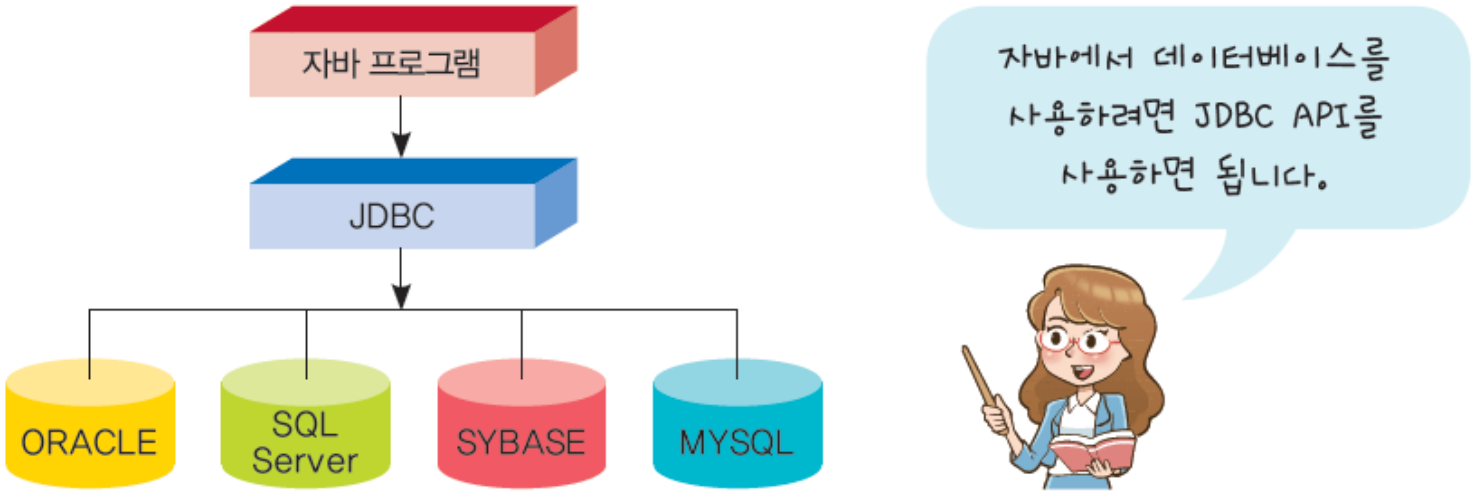


그림 19-2 • 자바와 데이터베이스

데이터베이스 프로그램 개발 절차

① DBMS(DataBase Management System)를 설치

② 자신이 설치한 DBMS에 필요한 JDBC 드라이버를 설치한다.

③ JDBC가 제공하는 기능을 이용하여 데이터베이스 응용 프로그램을 개발한다.

JDBC를 통하여 데이터베이스에 연결되면 그 다음에는 SQL 명령어를 데이터베이스에 전달하면 됩니다.

예제

```
public class host2ip
{
        public static void main ( String[] args ) throws IOException
        {
                String hostname = "www.naver.com";
                try
                {
                        InetAddress address = InetAddress.getByName(hostname);
                        System.out.println("IP 주소: " + address.getHostAddress());
                }
                catch ( UnknownHostException e )
                {
                   System.out.println(hostname + "의 IP 주소를 찾을 수 없습니다.   " );
                }
        }
}
```

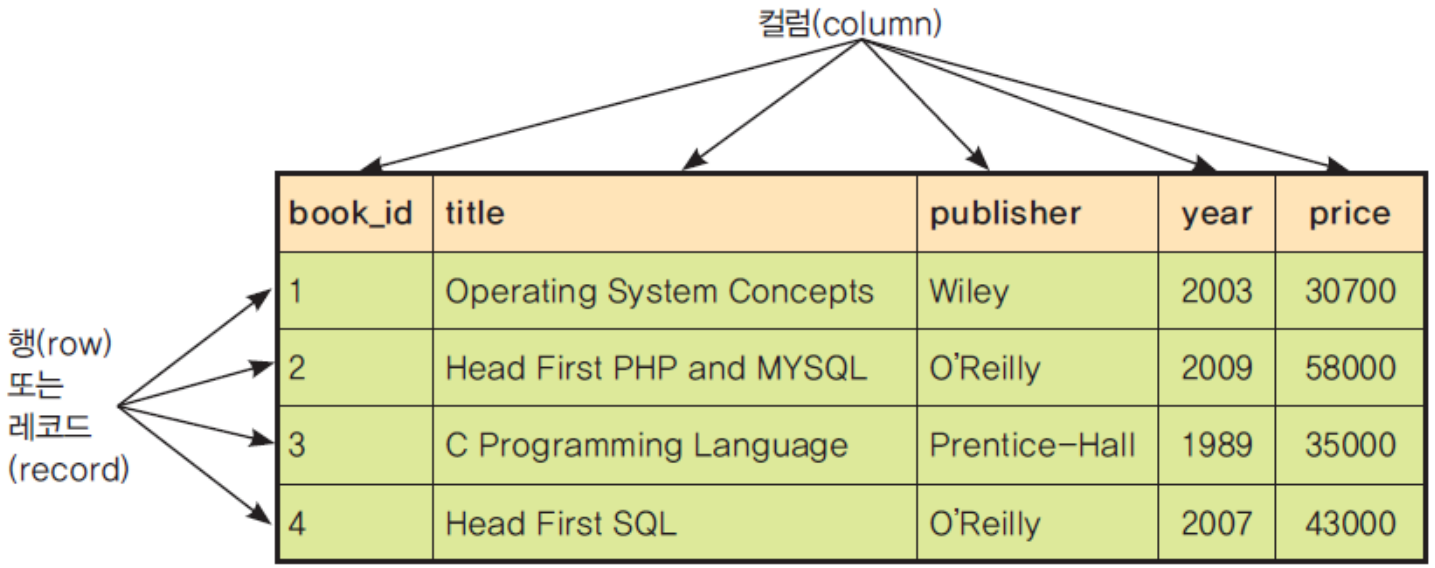

```
IP 주소: 125.209.222.142
```

데이터베이스란?

- 관계형 데이터베이스(database)는 데이터를 여러 개의 테이블에 나누어서 저장한다.
- 가장 많이 사용되는 DBMS
  - 오라클, 마이크로소프트의 SQL Server, 사이베이스, MySQL

테이블

- 테이블의 하나의 행(row)은 레코드(record)라고 불린다. 이 레코드는 여러 개의 컬럼(column)으로 이루어져 있고, 테이블은 무결성 법칙을 따라서 작성되어야 한다

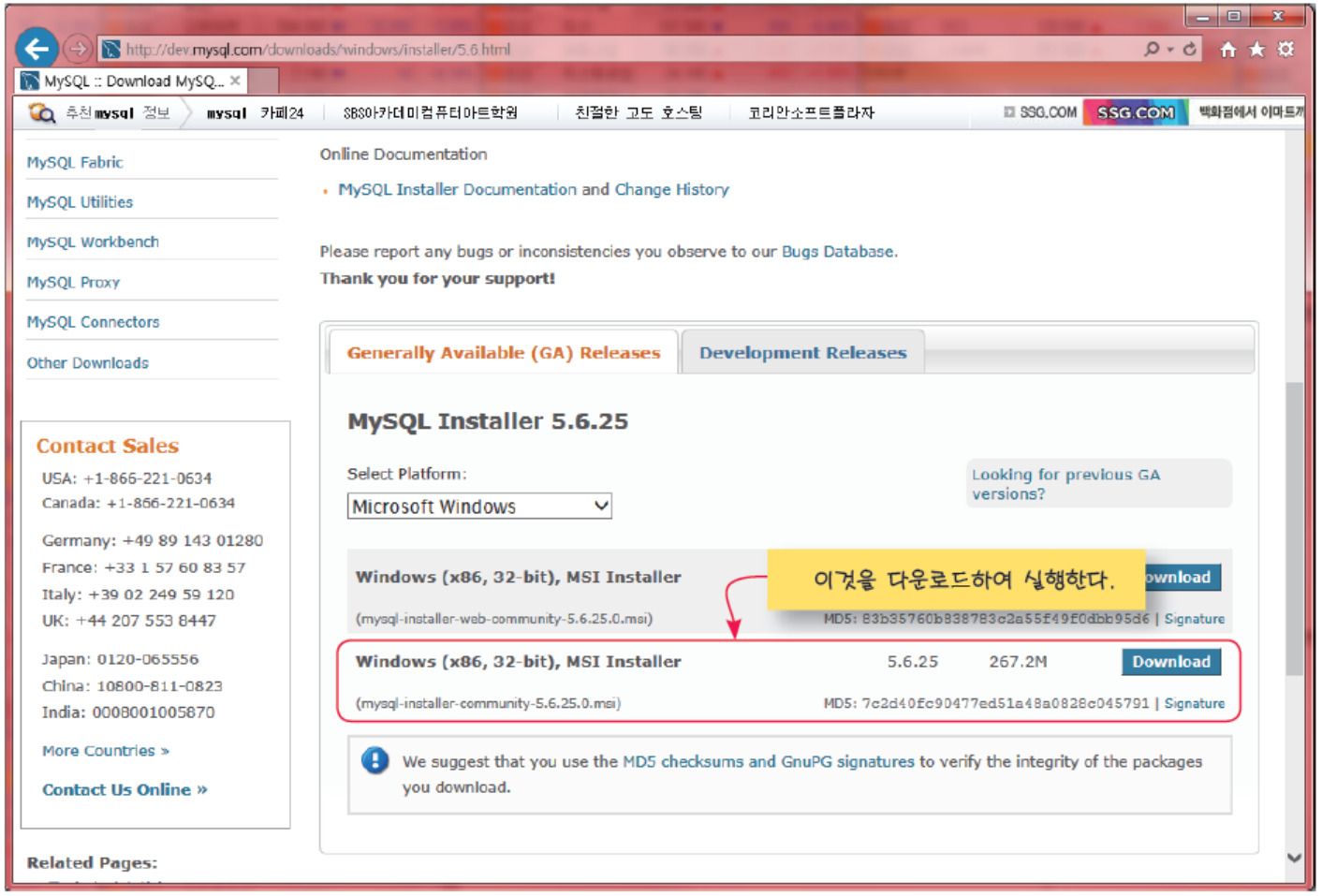


book_id	title	publisher	year	price
1	Operating System Concepts	Wiley	2003	30700
2	Head First PHP and MYSQL	O'Reilly	2009	58000
3	C Programming Language	Prentice-Hall	1989	35000
4	Head First SQL	O'Reilly	2007	43000

그림 19-4 • 데이터베이스 테이블

MySQL

- MySQL은 www.mysql.com 에서 다운로드



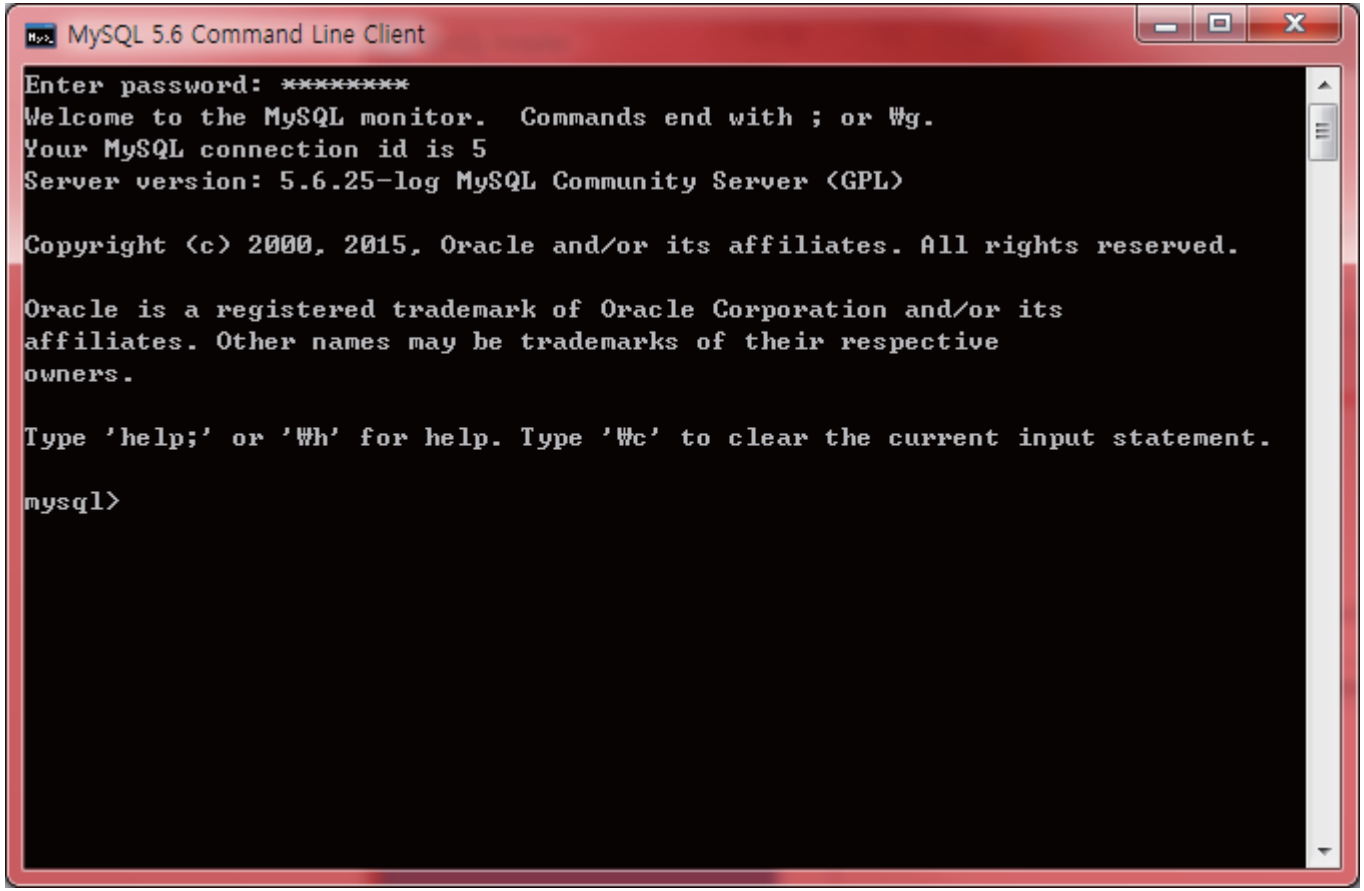
SQL이란?

- 관계형 데이터베이스에서 사용하기 위하여 설계된 언어

구분	명령어	설명
데이터 정의 명령어 (Data Definition Language)	CREATE	사용자가 제공하는 컬럼 이름을 가지고 테이블을 생성한다. 사용자는 컬럼의 데이터 타입도 지정하여야 한다. 데이터 타입은 데이터베이스에 따라 달라진다. CREATE TABLE은 보통 DML보다 적게 사용된다. 왜냐하면 이미 테이블이 만들어져 있는 경우가 많기 때문이다.
	ALTER	테이블에서 컬럼을 추가하거나 삭제한다.
	DROP	데이블의 모든 레코드를 제거하고 테이블의 정의 자체를 데이터베이스로부터 삭제하는 명령어이다.
	USE	어떤 데이터베이스를 사용하는지를 지정
데이터 조작 명령어 (Data Manipulation Language)	SELECT	데이터베이스로부터 데이터를 쿼리하고 출력한다. SELECT 명령어들은 결과 집합에 포함시킬 컬럼을 지정한다. SQL 명령어 중에서 가장 자주 사용된다.
	INSERT	새로운 레코드를 테이블에 추가한다. INSERT는 새롭게 생성된 테이블을 채우거나 새로운 레코드를 이미 존재하는 테이블에 추가할 때 사용된다.
	DELETE	지정된 레코드를 테이블로부터 삭제한다.
	UPDATE	테이블에서 레코드에 존재하는 값을 변경한다.

MySQL에서 SQL 실행하기

- MySQL은 다음과 같은 명령어 행 클라이언트를 가지고 있다.



데이터베이스 생성하기


```
DROP DATABASE book_db;

CREATE DATABASE book_db;

USE book_db;


CREATE TABLE books (
    book_id INT NOT NULL auto_increment,
    title VARCHAR(50),
    publisher VARCHAR(30),
    year VARCHAR(10),
    price INT,
    PRIMARY KEY(book_id)
);
```

먼저 같은 이름의 데이터베이스가 이미 있을 수도 있으므로 book_db 데이터베이스를 삭제한다.

book_db라고 하는 데이터베이스를 생성한다.

지금부터 book_db 데이터베이스를 사용한다는 의미이다.

book_db 안에 books라고 하는 테이블을 생성한다. book_id와 title, publisher, year, price 등이 모두 컬럼이 된다. 컬럼 이름 다음에는 컬럼의 타입을 적어준다. 많이 사용되는 컬럼의 타입은 INT, DECIMAL(n, d), CHAR(n), VARCHAR(n), DATE 등이 있다. CHAR(n)은 글자의 개수가 일정한 필드를 나타내고 VARCHAR(n)은 글자의 개수가 가변적일 때 사용한다.

레코드 추가하기


```
mysql> USE book_db;

INSERT INTO books (title, publisher, year, price)
   VALUES('Operating System Concepts', 'Wiley', '2003', 30700);

INSERT INTO books (title, publisher, year, price)
   VALUES('Head First PHP and MYSQL', 'OReilly', '2009', 58000);

INSERT INTO books (title, publisher, year, price)
   VALUES('C Programming Language ', 'Prentice-Hall', '1989', 35000);

INSERT INTO books (title, publisher, year, price)
   VALUES('Head First SQL', 'OReilly', '2007', 43000);
```

레코드 검색하기


```
mysql> USE book_db;
mysql> SELECT title, publisher, price FROM books;
+---------------------------+--------------+-------+
| title                     | publisher    | price |
+---------------------------+--------------+-------+
| Operating System Concepts | Wiley        | 30700 |
| Head First PHP and MYSQL  | OReilly      | 58000 |
| C Programming Language    | Prentice-Hall | 35000 |
| Head First SQL            | OReilly      | 43000 |
+---------------------------+--------------+-------+
```

검색시 조건 지정


```
mysql> SELECT * FROM books WHERE title LIKE 'Head First%';
+---------+--------------------------+-----------+------+-------+
| book_id | title                    | publisher | year | price |
+---------+--------------------------+-----------+------+-------+
|       2 | Head First PHP and MYSQL | OReilly   | 2009 | 58000 |
|       4 | Head First SQL           | OReilly   | 2007 | 43000 |
+---------+--------------------------+-----------+------+-------+
```

정렬하려면


```
mysql> SELECT * FROM books WHERE price > 30000 and price < 50000;
+---------+--------------------------+--------------+------+-------+
| book_id | title                    | publisher    | year | price |
+---------+--------------------------+--------------+------+-------+
|       1 | Operating System Concepts | Wiley        | 2003 | 30700 |
|       3 | C Programming Language   | Prentice-Hall | 1989 | 35000 |
|       4 | Head First SQL           | OReilly      | 2007 | 43000 |
+---------+--------------------------+--------------+------+-------+
```

레코드 수정하기


```
mysql> USE book_db;

mysql> UPDATE books SET price = 30000 WHERE year LIKE '19%';
Query OK, 1 row affected (0.02 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> SELECT * FROM books;
+---------+--------------------------+---------------+------+-------+
| book_id | title                    | publisher     | year | price |
+---------+--------------------------+---------------+------+-------+
|       1 | Operating System Concepts | Wiley         | 2003 | 30700 |
|       2 | Head First PHP and MYSQL  | OReilly       | 2009 | 58000 |
|       3 | C Programming Language    | Prentice-Hall | 1989 | 30000 |
|       4 | Head First SQL            | OReilly       | 2007 | 43000 |
+---------+--------------------------+---------------+------+-------+
```

레코드 삭제하기

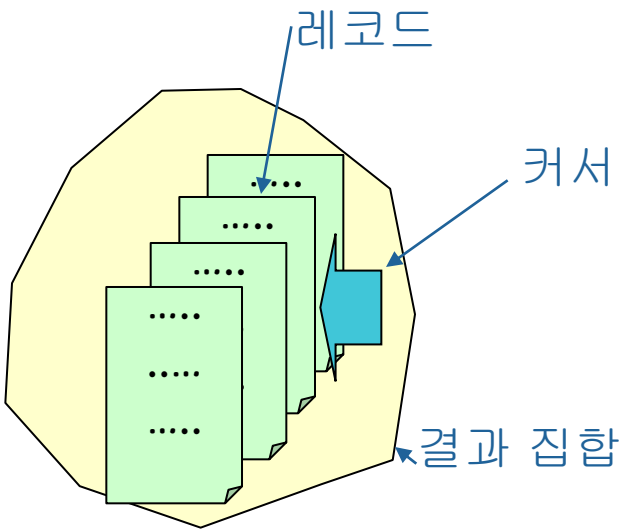

```
mysql> USE book_db;

mysql> DELETE FROM books WHERE year LIKE '19%';
Query OK, 1 row affected (0.03 sec)

mysql> select * from books;
+---------+--------------------------+---------------+------+-------+
| book_id | title                    | publisher     | year | price |
+---------+--------------------------+---------------+------+-------+
|       1 | Operating System Concepts | Wiley         | 2003 | 30700 |
|       2 | Head First PHP and MYSQL  | OReilly       | 2009 | 58000 |
|       4 | Head First SQL            | OReilly       | 2007 | 43000 |
+---------+--------------------------+---------------+------+-------+
```

결과 집합과 커서

- 쿼리의 조건을 만족하는 레코드들의 집합이 결과 집합(result set)이다.
- 커서(cursor)는 결과 집합의 레코드들을 포함하고 있는 파일에 대한 포인터



JDBC 드라이버 설치

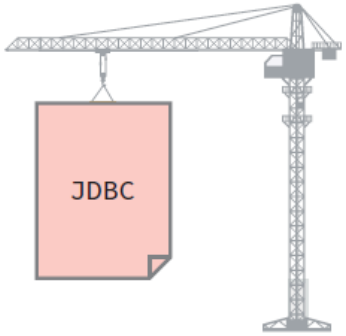
1. http://dev.mysql.com/downloads/connector/j/로부터 드라이버를 다운로드받아서 압축을 푼다.

2. 다음은 자바 가상 기계가 이 드라이버 파일을 찾을 수 있도록 하여야 한다. 클래스 경로를 나타내는 환경 변수인 CLASSPATH를 변경 또는 압축된 아카이브 파일을 *jre*/lib/ext 디렉토리에 복사


```
C> copy  mysql-connector-java-5.1.35-bin.jar  C:\
                              Program Files\java\jre1.8.0_45\lib\ext
```

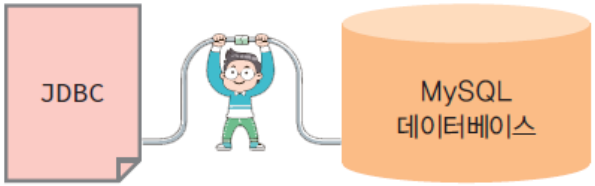
❶ JDBC 드라이버를 적재

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



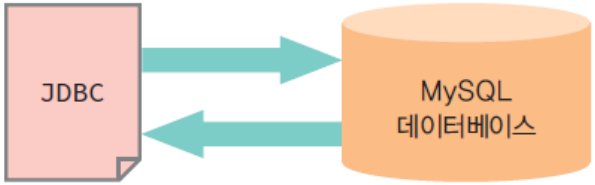
❷ 데이터베이스 연결

```
Connection con =
DriverManager.getConnection(url, user, pw);
```



❸ SQL 문장 작성 및 전송

```
Statement stmt = con.createStatement();
ResultSet rs =
stmt.executeQuery("SELECT * FROM books");
```



❹ 결과 집합 사용 후 연결 해제

```
while (rs.next()) {
int number = rs.getInt("book_id");
String name = rs.getString("title");
}
```

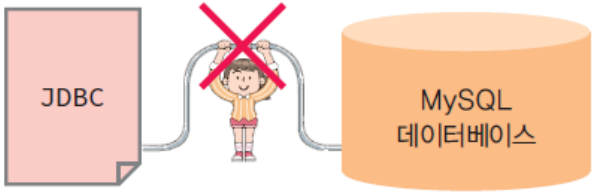


그림 19-5 • JDBC를 이용한 데이터베이스 사용 절차

드라이버 클래스 적재

```
try {
    Class.forName("com.mysql.jdbc.Driver");
}
catch (ClassNotFoundException e) {
    System.out.println("드라이버를 찾을 수 업습니다“);
}
```

지정된 이름의 클래스를
찾아서 메모리로 적재한다.

데이터베이스 연결

```
String url = “jdbc:mysql://localhost/book_db”;
String user = “root”;
String password = “password”;
con = DriverManager.getConnection(url, user, password);
```

사용자 아이디와 패스워드를 사용하여 데이터베이스에 연결.

데이터베이스 연결 예제

```
import java.sql.*;
public class ConnectDatabase {
	public static Connection makeConnection()
	{
		String url = "jdbc:mysql://localhost/book_db";
		String id = "root";
		String password = "password";
		Connection con = null;
		try {
			Class.forName("com.mysql.jdbc.Driver");
			System.out.println("드라이버 적재 성공");
			con = DriverManager.getConnection(url, id, password);
			System.out.println("데이터베이스 연결 성공");
		} catch (ClassNotFoundException e) {
			System.out.println("드라이버를 찾을 수 없습니다.");
		} catch (SQLException e) {
			System.out.println("연결에 실패하였습니다.");
		}
		return con;
	}
```

데이터베이스 연결 예제

```
public static void main(String arg[]) throws SQLException {
        Connection con = makeConnection();
    }
```

드라이버 적재 성공
데이터베이스 연결 성공

SQL 문장 수행

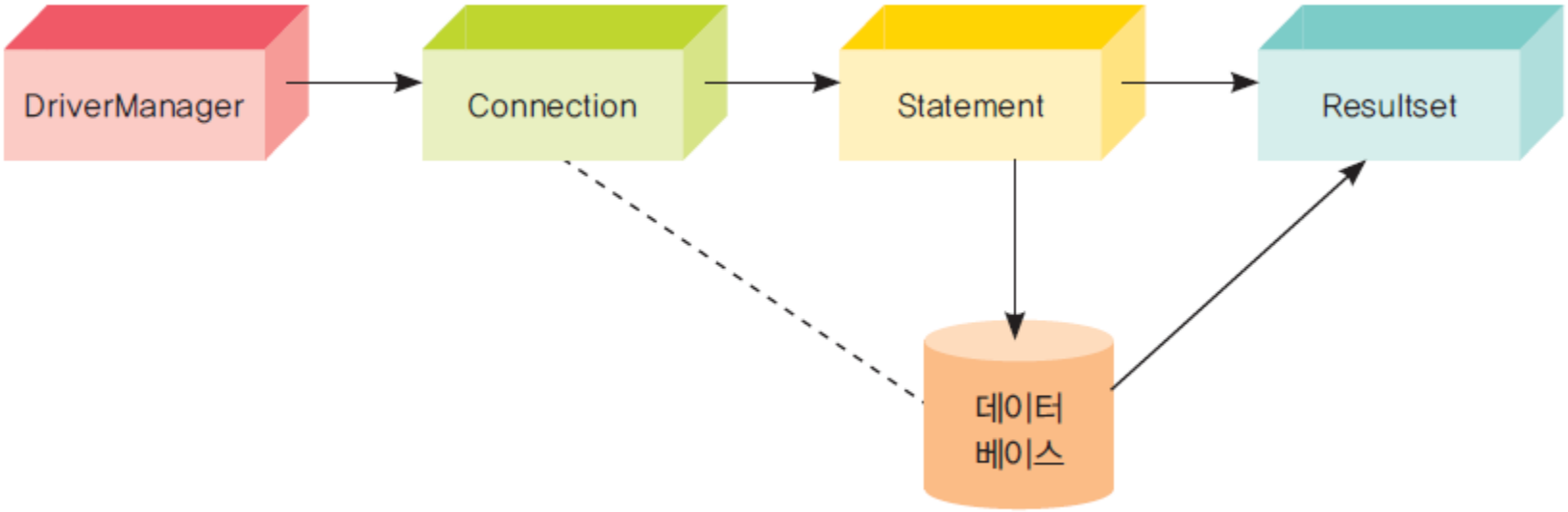


그림 19-6 • Connection, Statem.ent, ResultSet 인터페이스의 역할

SQL 문장 수행

```
Statement s = con.createStatement();                          // 문장 객체 생성
String select = "SELECT * FROM books ORDER BY book_id";       // SQL 문장 생성
ResultSet rows = s.executeQuery(select);                      // SQL 문장 실행
```

SQL문장을 실행하고 결과 집합을 반환한다.

데이터베이스 연결 예제

```
import java.sql.*;

public class SQLSelectTest {
    public static Connection makeConnection()
    {
      ...
      // 앞의 코드와 동일
    }
    public static void main(String arg[]) throws SQLException {
        Connection con = makeConnection();
        Statement stmt = con.createStatement();
        ResultSet rs = stmt.executeQuery("SELECT * FROM books");
        while (rs.next()) {
            int id = rs.getInt("book_id");
            String title = rs.getString("title");
            System.out.println(id + " " + title);
        }
    }
}
```

결과 집합에서 다음 레코드로 이동한다.

현재 레코드에서 필드의 값을 가져온다.

실행 결과

드라이버 적재 성공
데이터베이스 연결 성공
1 Operating System Concepts
2 Head First PHP and MYSQL
3 C Programming Language
4 Head First SQL

레코드 수정, 삭제

```
import java.sql.*;

public class SQLInsertTest {
	public static Connection makeConnection() {
		...// 전과 동일
	}

	public static void main(String arg[]) {
		addBook("Artificial Intellegence", "Addison Wesley", "2002", 35000);
	}
```

레코드 수정, 삭제

```
private static void addBook(String title, String publisher, String year,
        int price) {
    Connection con = makeConnection();
    try {
        Statement stmt = con.createStatement();
        String s = "INSERT INTO books (title, publisher, year, price) VALUES ";
        s += "('" + title + "','" + publisher + "','" + year + "','"
                + price + "')";
        System.out.println(s);
        int i = stmt.executeUpdate(s);
        if (i == 1)
            System.out.println("레코드 추가 성공");
        else
            System.out.println("레코드 추가 실패");
    } catch (SQLException e) {
        System.out.println(e.getMessage());
        System.exit(0);
    }
  }
}
```

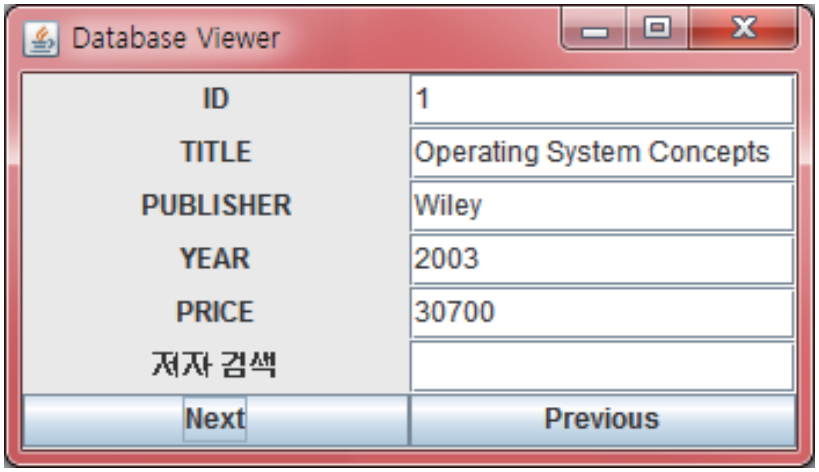
레코드를 수정할때 사용.

실행 결과

드라이버 적재 성공
데이터베이스 연결 성공
INSERT INTO books (title, publisher, year, price) VALUES
('Artificial Intellegence','Addison Wesley','2002','35000')
레코드 추가 성공

LAB: 데이터베이스 레코드 뷰어 작성

- 다음과 같이 그래픽 사용자 인터페이스를 이용하여 데이터베이스 테이블의 내용을 화면에 표시하는 프로그램을 작성하여 보자.



SOLUTION

```
class MyFrame extends JFrame {
        JTextField id, title, p, year, price, author;
        JButton previousButton, nextButton, InsertButton, deleteButton,
                                searchButton;
        ResultSet rs;
        Statement stmt;
        public MyFrame() throws SQLException {
                super("Database Viewer");
                Connection con = makeConnection();
                stmt = con.createStatement();
                rs = stmt.executeQuery("SELECT * FROM books");
                setLayout(new GridLayout(0, 2));
                add(new JLabel("ID", JLabel.CENTER));
                add(id = new JTextField());
                add(new JLabel("TITLE", JLabel.CENTER));
                add(title = new JTextField());
                add(new JLabel("PUBLISHER", JLabel.CENTER));
                add(p = new JTextField());
                add(new JLabel("YEAR", JLabel.CENTER));
                add(year = new JTextField());
                add(new JLabel("PRICE", JLabel.CENTER));
                add(price = new JTextField());
                add(new JLabel("저자 검색", JLabel.CENTER));
                add(author = new JTextField());
```

SOLUTION

```
previousButton = new JButton("Previous");
previousButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent event) {
                try {
                        rs.previous();
                        id.setText("" + rs.getInt("book_id"));
                        title.setText("" + rs.getString("title"));
                        p.setText("" + rs.getString("publisher"));
                        year.setText("" + rs.getString("year"));
                        price.setText("" + rs.getInt("price"));
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
});
```

SOLUTION

```
nextButton = new JButton("Next");
nextButton.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent event) {
                try {
                        rs.next();
                        id.setText("" + rs.getInt("book_id"));
                        title.setText("" + rs.getString("title"));
                        p.setText("" + rs.getString("publisher"));
                        year.setText("" + rs.getString("year"));
                        price.setText("" + rs.getInt("price"));
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
});
add(nextButton);
add(previousButton);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setSize(350, 200);
// pack();
setVisible(true);
}
```

SOLUTION

```
        public static Connection makeConnection() {
                String url = "jdbc:mysql://localhost/book_db";
                String id = "root";
                String password = "password";
                Connection con = null;
                try {
                        Class.forName("com.mysql.jdbc.Driver");
                        System.out.println("드라이버 적재 성공");
                        con = DriverManager.getConnection(url, id, password);
                        System.out.println("데이터베이스 연결 성공");
                } catch (ClassNotFoundException e) {
                        System.out.println("드라이버를 찾을 수 없습니다.");
                } catch (SQLException e) {
                        System.out.println("연결에 실패하였습니다.");
                }
                return con;
        }
}
public class SQLSelectTest {
        public static void main(String[] args) throws SQLException {
                new MyFrame();
        }
}
```

Q & A