20장 JDBC

20.1 환경 설정

20.1.1 Oracle SCOTT 계정 활성화

```
C:\TEMP> SQLPLUS SYSTEM/SYS
SQL> @C:\TEMP\SCOTT.SQL
SQL> CONN SCOTT/TIGER
SQL> SELECT * FROM TAB;
SQL> QUIT;
```

[SCOTT.SQL]

```
01    Rem Copyright (c) 1990 by Oracle Corporation
02    Rem NAME
03    REM    UTLSAMPL.SQL
04    Rem  FUNCTION
05    Rem  NOTES
06    Rem  MODIFIED
07    Rem       gdudey       06/28/95 -  Modified for desktop seed database
08    Rem       glumpkin   10/21/92 -  Renamed from SQLBLD.SQL
09    Rem       blinden   07/27/92 -  Added primary and foreign keys to EMP and DEPT
10    Rem       rlim         04/29/91 -             change char to varchar2
11    Rem       mmoore       04/08/91 -             use unlimited tablespace priv
12    Rem       pritto       04/04/91 -             change SYSDATE to 13-JUL-87
13    Rem   Mendels        12/07/90 - bug 30123;add to_date calls so language independent
14    Rem
15    rem
16    rem $Header: utlsampl.sql 7020100.1 94/09/23 22:14:24 cli Generic<base> $ sqlbld.sql
17    rem
18    SET TERMOUT OFF
19    SET ECHO OFF
20
21    rem CONGDON    Invoked in RDBMS at build time.     29-DEC-1988
22    rem OATES:     Created: 16-Feb-83
23
24    GRANT CONNECT,RESOURCE,UNLIMITED TABLESPACE TO SCOTT IDENTIFIED BY TIGER;
25    ALTER USER SCOTT DEFAULT TABLESPACE USERS;
26    ALTER USER SCOTT TEMPORARY TABLESPACE TEMP;
27    CONNECT SCOTT/TIGER
28    DROP TABLE DEPT;
29    CREATE TABLE DEPT
30           (DEPTNO NUMBER(2) CONSTRAINT PK_DEPT PRIMARY KEY,
31              DNAME VARCHAR2(14) ,
32              LOC VARCHAR2(13) ) ;
33    DROP TABLE EMP;
34    CREATE TABLE EMP
35           (EMPNO NUMBER(4) CONSTRAINT PK_EMP PRIMARY KEY,
36              ENAME VARCHAR2(10),
37              JOB VARCHAR2(9),
38              MGR NUMBER(4),
39              HIREDATE DATE,
40              SAL NUMBER(7,2),
41              COMM NUMBER(7,2),
42              DEPTNO NUMBER(2) CONSTRAINT FK_DEPTNO REFERENCES DEPT);
43    INSERT INTO DEPT VALUES
44              (10,'ACCOUNTING','NEW YORK');
45    INSERT INTO DEPT VALUES (20,'RESEARCH','DALLAS');
```

```
INSERT INTO DEPT VALUES
          (30,'SALES','CHICAGO');
INSERT INTO DEPT VALUES
          (40,'OPERATIONS','BOSTON');
INSERT INTO EMP VALUES
(7369,'SMITH','CLERK',7902,to_date('17-12-1980','dd-mm-yyyy'),800,NULL,20);
INSERT INTO EMP VALUES
(7499,'ALLEN','SALESMAN',7698,to_date('20-2-1981','dd-mm-yyyy'),1600,300,30);
INSERT INTO EMP VALUES
(7521,'WARD','SALESMAN',7698,to_date('22-2-1981','dd-mm-yyyy'),1250,500,30);
INSERT INTO EMP VALUES
(7566,'JONES','MANAGER',7839,to_date('2-4-1981','dd-mm-yyyy'),2975,NULL,20);
INSERT INTO EMP VALUES
(7654,'MARTIN','SALESMAN',7698,to_date('28-9-1981','dd-mm-yyyy'),1250,1400,30);
INSERT INTO EMP VALUES
(7698,'BLAKE','MANAGER',7839,to_date('1-5-1981','dd-mm-yyyy'),2850,NULL,30);
INSERT INTO EMP VALUES
(7782,'CLARK','MANAGER',7839,to_date('9-6-1981','dd-mm-yyyy'),2450,NULL,10);
INSERT INTO EMP VALUES
(7788,'SCOTT','ANALYST',7566,to_date('13-7-1987','dd-mm-yyyy'),3000,NULL,20);
INSERT INTO EMP VALUES
(7839,'KING','PRESIDENT',NULL,to_date('17-11-1981','dd-mm-yyyy'),5000,NULL,10);
INSERT INTO EMP VALUES
(7844,'TURNER','SALESMAN',7698,to_date('8-9-1981','dd-mm-yyyy'),1500,0,30);
INSERT INTO EMP VALUES
(7876,'ADAMS','CLERK',7788,to_date('13-7-1987','dd-mm-yyyy'),1100,NULL,20);
INSERT INTO EMP VALUES
(7900,'JAMES','CLERK',7698,to_date('3-12-1981','dd-mm-yyyy'),950,NULL,30);
INSERT INTO EMP VALUES
(7902,'FORD','ANALYST',7566,to_date('3-12-1981','dd-mm-yyyy'),3000,NULL,20);
INSERT INTO EMP VALUES
(7934,'MILLER','CLERK',7782,to_date('23-1-1982','dd-mm-yyyy'),1300,NULL,10);
DROP TABLE BONUS;
CREATE TABLE BONUS
          (
          ENAME VARCHAR2(10)  ,
          JOB VARCHAR2(9)  ,
          SAL NUMBER,
          COMM NUMBER
          ) ;
DROP TABLE SALGRADE;
CREATE TABLE SALGRADE
      ( GRADE NUMBER,
          LOSAL NUMBER,
          HISAL NUMBER );
INSERT INTO SALGRADE VALUES (1,700,1200);
INSERT INTO SALGRADE VALUES (2,1201,1400);
INSERT INTO SALGRADE VALUES (3,1401,2000);
INSERT INTO SALGRADE VALUES (4,2001,3000);
INSERT INTO SALGRADE VALUES (5,3001,9999);
COMMIT;

SET TERMOUT ON
SET ECHO ON
```

20.1.2. Eclipse에 JDBC 설정

- 오라클용 JDBC Driver 파일(ojdbc7.jar)을 자바 설치 위치(C:\Program Files\Java\jre1.8.0_45\lib\ext)로 복사한다.

- 이클립스에서 오라클 데이터베이스 연동 설정: 이클립스의 Project Explorer 화면에서 JRE System Library에 오른 마우스 클릭하여 추가한다.
- 이클립스 재 구동
- 간단한 자바-오라클 연동 파일을 작성한후 테스트 해본다.
- 만약 데이터베이스 연결 실패 메세지가 나올때에는 C:\oracle\product\10.2.0\db_1\NETWORK\ADMIN 폴더 안에 있는 listener.ora, tnsnames.ora 파일의 Host=컴퓨터 이름, Port=1521 등을 점검한다.

[JDBC_Connect01.java] JDBC 로딩 테스트

```
01  package sec01.exam01_jdbc;
02
03  import java.sql.*;
04
05  public class JDBC_Connect01 {
06
07          public static void main(String[] args) {
08
09                  /** ORACLE JDBC Driver Test *************************/
10                  String driver = "oracle.jdbc.driver.OracleDriver";
11                  /***************************************************/
12
13                  /** My-SQL JDBC Driver Test *************************/
14                  // String driver ="com.mysql.jdbc.Driver";
15                  /***************************************************/
16
17                  try {
18                          Class.forName(driver);
19                          System.out.println("JDBC Driver Loading 성공~!!");
20
21                  } catch (Exception e) {
22                          System.out.println("JDBC Driver Loading 실패~!!");
23                          e.printStackTrace();
24                  }
25          }
26  }
```

[JDBC_Connect02.java] JDBC 접속 테스트

```
01  import java.sql.*;
02
03  public class JDBC_Connect02 {
04
05          public static void main(String[] args) {
06
07                  /** ORACLE JDBC Driver Test ***************************************/
08                  String driver = "oracle.jdbc.driver.OracleDriver";
09                  // String url = "jdbc:oracle:thin:@localhost:1521:orcl";
10                  String url = "jdbc:oracle:thin:@localhost:1521:MYORACLE";
11                  /*****************************************************************/
12
13                  /** My-SQL JDBC Driver ********************************************/
14                  // String driver ="com.mysql.jdbc.Driver";
15                  // String url = "jdbc:mysql://localhost/academy";
16                  /*****************************************************************/
17
18                  Connection con = null;
19
```

```
20                      try {
21
22                              Class.forName(driver);
23
24                              /** ORACLE에서 Connection 객체 **********************************/
25                              con = DriverManager.getConnection(url, "SCOTT", "TIGER");
26                              /***************************************************************/
27
28                              /** My-SQL에서 Connection 객체 **********************************/
29                              // con = DriverManager.getConnection(url, "totoro", "1234" );
30                              /***************************************************************/
31
32                              System.out.println("데이터베이스 연결 성공~!!");
33
34                      } catch (Exception e) {
35                              System.out.println("데이터베이스 연결 실패~!!");
36                              e.printStackTrace();
37                      } finally {
38                              try {
39                                      if (con != null)
40                                              con.close();
41                              } catch (Exception e) {
42                                      System.out.println(e.getMessage());
43                              }
44                      }
45              }
46      }
```

20.2 JDBC 프로그램 작성하기

20.2.1 Statement 활용

■ SQL문을 실행하기 위해서는 Statement 클래스를 이용한다. 주요 메소드는 다음과 같다.

메소드	설명
ResultSet executeQuery(String sql)	주어진 SQL문을 실행하고 결과는 ResultSet 객체에 반환
int executeUpdate(String sql)	INSERT, UPDATE, 또는 DELETE과 같은 SQL문을 실행하고, SQL문 실행으로 영향을 받은 행의 개수나 0을 반환
void close()	Statement 객체의 데이터베이스와 JDBC 리소스를 즉시 반환

[JDBC_Insert.java] Statement 활용

```
01      // 도스 콘솔 창에서 사용자 입력을 받아들이기 위해 BufferedReader
02      import java.io.BufferedReader;
03      import java.io.InputStreamReader;
04      import java.sql.Connection;
05      import java.sql.DriverManager;
06      import java.sql.Statement;
07
08      class JDBC_Insert {
09              public static void main(String[] args) {
10
11                      String driver = "oracle.jdbc.driver.OracleDriver";
12                      String url = "jdbc:oracle:thin:@localhost:1521:xe";
13
14                      Connection con = null;
15                      Statement stmt = null;
16
17                      // ResultSet rs = null;
```

```
18                      String sql;
19
20                      String name, email, tel, no;
21
22                      try {
23                              Class.forName(driver);
24                              con = DriverManager.getConnection(url, "scott", "tiger");
25                              stmt = con.createStatement();
26
27                              // ---JDBC_Insert 추가된 내용-------
28                              // 테이블에 추가할 내용을 도스 콘솔 창에서 사용자의 입력을 받도록 한다.
29                              BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
30
31                              System.out.println(" customer 테이블에 값 입력하기 .....");
32                              System.out.print(" 번호 입력: ");
33                              no = br.readLine();
34                              System.out.print(" 이름 입력: ");
35                              name = br.readLine(); // 테이블에 추가할 name 필드 값을 입력 받음
36                              System.out.print(" 이메일 입력: ");
37                              email = br.readLine(); // 테이블에 추가할 email 필드 값을 입력 받음
38                              System.out.print(" 전화번호 입력: ");
39                              tel = br.readLine(); // 테이블에 추가할 tel 필드 값을 입력 받음
40
41                              // INSERT 쿼리문을 작성
42                              sql = "INSERT into customer(no, name, email, tel) values ";
43                              sql += "(" + no + ",'" + name + "','" + email + "','" + tel + "')";
44
45                              // Statement 객체의 executeUpdate(sql) 메서드를 이용해
46                              int res = stmt.executeUpdate(sql); // 데이터베이스 파일에 새로운 값을 추가
47  시킴
48                              if (res == 1) {
49                                      System.out.println(" Data insert success!! ");
50                              } else {
51                                      System.out.println(" Data insert failed ");
52                              }
53                      } catch (Exception e) {
54                              System.out.println("데이터베이스 연결 실패!");
55                      } finally {
56                              try {
57                                      // if( rs != null ) rs.close();
58                                      if (stmt != null)
59                                              stmt.close();
60                                      if (con != null)
61                                              con.close();
62                              } catch (Exception e) {
63                                      System.out.println(e.getMessage());
64                              }
65                      }
66              }
67  }
```

20.2.2 PreparedStatement 활용

[JDBC_Insert01.java] PreparedStatement 활용

```
01  package sec02.exam02_prestatement;
02  import java.io.BufferedReader; // 도스 콘솔 창에서 사용자 입력을 받아들이기 위해 BufferedReader
03  import java.io.InputStreamReader;
04  import java.sql.Connection;
05  import java.sql.DriverManager;
06  import java.sql.PreparedStatement;
```

```
07  import java.sql.ResultSet;
08
09  class JDBC_Insert01 {
10            public static void main(String[] args) {
11
12                        String driver = "oracle.jdbc.driver.OracleDriver";
13                        String url = "jdbc:oracle:thin:@localhost:1521:MYORACLE";
14
15                        Connection con = null;
16                        PreparedStatement pstmt = null;
17
18                        //ResultSet rs = null;
19                        String sql;
20
21                        String name, email, tel, no;
22
23                        try {
24                                  Class.forName(driver);
25                                  con = DriverManager.getConnection(url, "SCOTT", "TIGER");
26
27                                  // ---JDBC_Insert 추가된 내용-------
28                                  // 테이블에 추가할 내용을 도스 콘솔 창에서 사용자의 입력을 받도록 한다.
29                                  BufferedReader br = new BufferedReader(new InputStreamReader(
30                                                      System.in));
31
32                                  System.out.println(" customer 테이블에 값 입력하기 .....");
33                                  System.out.print(" 번호 입력: ");
34                                  no = br.readLine();
35                                  System.out.print(" 이름 입력: ");
36                                  name = br.readLine(); // 테이블에 추가할 name 필드 값을 입력 받음
37                                  System.out.print(" 이메일 입력: ");
38                                  email = br.readLine(); // 테이블에 추가할 email 필드 값을 입력 받음
39                                  System.out.print(" 전화번호 입력: ");
40                                  tel = br.readLine(); // 테이블에 추가할 tel 필드 값을 입력 받음
41
42                                  int ino = Integer.parseInt(no);
43
44                                  // INSERT 쿼리문을 작성
45                                  sql = "INSERT into customer (no, name, email, tel) values (?, ?, ?, ?)";
46
47                                  pstmt = con.prepareStatement(sql);
48                                  pstmt.setInt(1, ino);
49                                  pstmt.setString(2, name);
50                                  pstmt.setString(3, email);
51                                  pstmt.setString(4, tel);
52                                  int result=pstmt.executeUpdate();
53                                  if(result == 1){
54                                            System.out.println("데이터 입력 성공");
55                                  }else{
56                                            System.out.println("데이터 입력 실패");
57                                  }
58
59                        } catch (Exception e) {
60                                  System.out.println("데이터베이스 연결 실패!");
61                        } finally {
62                                  try {
63                        //                  if (rs != null)
64                        //                            rs.close();
65                                  if (pstmt != null)
66                                            pstmt.close();
67                                  if (con != null)
68                                            con.close();
69                                  } catch (Exception e) {
70                                            System.out.println(e.getMessage());
71                                  }
```

```
72                      }
73              }
74      }
```

[꿀팁] 저장 프로시저 실행 방법

- Stored Procedure(저장 프로시저): 데이터베이스 내에 프로시저를 선언하여 클라이언트가 필요할 대마다 호출하여 사용하는 프로시저이다. 이것은 클라이언트에서 SQL 문을 실행하는 것과 달리 데이타베이스쪽에서 프로시저로 존재하는 것이기 때문에, 클라이언트에서 저장된 프로시저를 실행만 해주면 그 프로시저 내용이 바로 처리되므로 실행 속도 또한 더 빠르며, 부하가 적다는 장점이 있다.

- CallableStatement 객체: Stored Procedure(저장된 프로시저)를 호출하기 위해 존재하는 객체로 PreparedStatement 객체를 상속받아 사용한다.
  - setXXX() 메소드: PreparedStatement 객체를 사용할 때와 똑같이 사용 가능 하다.
  - registerOutParameter() 메소드: 프로시저에서 넘어오는 값을 반환받기 위해서는 꼭 사용 해야 한다. 이 메소드는 프로시저로부터 넘어오는 값의 타입을 지정해주는 역활을 한다.
  - execute() 메소드: CallableStatement 객체를 실행시킨다.

```
// CallableStatement 객체 활용
// ....
conn = ds.getConnection();

CallableStatement cs = conn.prepareCall("{call procedure_name(?,?,?)}");
cs.setInt(1,1);
cs.setString(2,"홍길동");
cs.registerOutParameter(3,java.sql.Types.VARCHAR);
cs.execute();

out.println("<h3>"+cs.getString(3)+"</h3>");
cs.close();
```

20.2.3 Timestamp 활용

[JDBC_Select02.java] Timestamp 활용

```
01    package sec02.exam02_timestamp;
02    import java.io.BufferedReader;  // 도스 콘솔 창에서 사용자 입력을 받아들이기 위해 BufferedReader
03    import java.io.InputStreamReader;
04    import java.sql.Connection;
05    import java.sql.DriverManager;
06    import java.sql.PreparedStatement;
07    import java.sql.ResultSet;
08    import java.sql.Timestamp;
09
10    class  JDBC_Insert02{
11    public static void main(String[] args) {
12
13      String driver = "oracle.jdbc.driver.OracleDriver";
14      String url = "jdbc:oracle:thin:@localhost:1521:MYORACLE";
15
```

```
16     Connection con = null;
17     PreparedStatement pstmt =  null;
18 
19    // ResultSet  rs   = null;
20     String sql;
21 
22      String name, email, tel, no, address;
23 
24        try{
25         Class.forName(driver);
26         con = DriverManager.getConnection(url, "SCOTT", "TIGER" );
27 
28         //---JDBC_Insert 추가된 내용-------
29         // 테이블에 추가할 내용을 도스 콘솔 창에서 사용자의 입력을 받도록 한다.
30         BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
31 
32         System.out.println(" customer 테이블에 값 입력하기 .....");
33         System.out.print(" 번호 입력: ");
34         no = br.readLine().trim();
35         System.out.print(" 이름 입력: ");
36         name = br.readLine().trim();            //테이블에 추가할 name 필드 값을 입력 받음
37         System.out.print(" 이메일 입력: ");
38         email = br.readLine().trim();             //테이블에 추가할 email 필드 값을 입력 받음
39         System.out.print(" 전화번호 입력: ");
40         tel = br.readLine().trim();               //테이블에 추가할 tel 필드 값을 입력 받음
41         System.out.println("주소를 입력 하세요?");
42         address = br.readLine().trim();
43               int ino = Integer.parseInt(no);
44 
45               Timestamp ts = new Timestamp(System.currentTimeMillis());
46 
47         // INSERT 쿼리문을 작성
48               sql = "INSERT into customer (no, name, email, tel, address, reg_date) values (?, ?, ?, ?,?,?)";
49 
50               pstmt = con.prepareStatement( sql );
51               pstmt.setInt(1, ino);
52               pstmt.setString(2, name);
53               pstmt.setString(3, email);
54               pstmt.setString(4, tel);
55               pstmt.setString(5, address);
56               pstmt.setTimestamp(6, ts);
57               int result=pstmt.executeUpdate();
58                       if(result == 1){
59                                 System.out.println(" Data insert success!! ");
60                       }else{
61                                 System.out.println(" Data insert failed ");
62                       }
63       } catch(Exception e){
64         System.out.println("데이터베이스 연결 실패!");
65       } finally{
66                       try{
67    //                           if( rs != null )   rs.close();
68                                 if( pstmt != null ) pstmt.close();
69                                 if( con != null )  con.close();
70                        } catch(Exception e){
71                                 System.out.println( e.getMessage());
72                       }
73       }
74     }
75    }
```

20.2.4 Swing 활용

[InsertMember.java]

```
01    package sec02.exam03_swing;
02
03    import java.awt.*;
04    import java.awt.event.*;
05    import javax.swing.*;
06    import java.sql.*;
07
08    public class InsertMember extends JFrame implements ActionListener {
09              Connection con;
10              PreparedStatement pstmt;
11              Timestamp reg_date;
12
13              private String names[] = { "아이디", "비밀번호", "비밀번호 재입력", "이름" };
14              private JLabel labels[];
15              private JTextField fields[];
16              private JButton register, cancle, reWrite;
17              private JPanel panelCenter, panelSouth;
18              private int size = 4;
19
20              public InsertMember() {
21                        super("회원등록");
22
23                        labels = new JLabel[size];
24                        fields = new JTextField[size];
25
26                        for (int i = 0; i < labels.length; i++)
27                                  labels[i] = new JLabel(names[i]);
28
29                        for (int i = 0; i < fields.length; i++)
30                                  fields[i] = new JTextField();
31
32                        panelCenter = new JPanel();
33                        panelCenter.setLayout(new GridLayout(size, 2));
34                        for (int i = 0; i < size; i++) {
35                                  panelCenter.add(labels[i]);
36                                  panelCenter.add(fields[i]);
37                        }
38
39                        register = new JButton("회원등록");
40                        cancle = new JButton("종료");
41                        reWrite = new JButton("다시작성");
42
43                        panelSouth = new JPanel();
44                        panelSouth.add(register);
45                        panelSouth.add(cancle);
46                        panelSouth.add(reWrite);
47
48                        add(panelCenter, "Center");
49                        add(panelSouth, "South");
50                        setBounds(300, 300, 300, 250);
51                        setVisible(true);
52                        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
53
54                        register.addActionListener(this);
55                        cancle.addActionListener(this);
56                        reWrite.addActionListener(this);
57              }// 생성자 end
58
59              // Database 연결 부분
60              public void dbConnect() {
```

```
61
62                      try {
63                              Class.forName("oracle.jdbc.driver.OracleDriver");
64                      } catch (ClassNotFoundException cnfe) {
65                              cnfe.printStackTrace();
66                              System.out.println("드라이버 로딩에 실패");
67                      }
68
69                      try {
70                              String url = "jdbc:oracle:thin:@localhost:1521:MYORACLE";
71                              String userId = "SCOTT";
72                              String userPass = "TIGER";
73
74                              con = DriverManager.getConnection(url, userId, userPass);
75                      } catch (SQLException e) {
76                              e.printStackTrace();
77                              System.out.println("커넥션 설정에 실패");
78                      }
79              }
80
81              // Event 처리 부분
82              public void actionPerformed(ActionEvent ae) {
83
84                      if (ae.getSource() == register) { // 회원 등록
85                              dbConnect();
86                              insertMember();
87                              clearFields();
88                      } else if (ae.getSource() == cancle) { // 종료
89                              System.exit(0);
90                      } else if (ae.getSource() == reWrite) { // 다시작성
91                              clearFields();
92                      }
93              }
94
95              // 회원 가입 처리 부분
96              public void insertMember() {
97                      Timestamp reg_date = new Timestamp(System.currentTimeMillis());
98                      String data[] = getFieldValues();
99
100                     // if(data[0].equals("") || data[1].equals("") || data[2].equals("") ||
101                     // data[3].equals("")){
102                     if    (fields[0].getText().equals("")    ||    fields[1].getText().equals("")    ||
103 fields[2].getText().equals("")
104                                     || fields[3].getText().equals("")) {
105                             JOptionPane.showMessageDialog(this, "모든 정보를 입력 하세요!");
106                     } else if (fields[1].getText().equals(fields[2].getText())) {
107                             // }else if(data[1].equals(data[2])){ //비밀 번호가 일치하면 query문 실행
108
109                             String sql = "insert into mem02 values(?,?,?,?)";
110
111                             try {
112                                     pstmt = con.prepareStatement(sql);
113                                     // pstmt.setString(1,data[0]); //아이디
114                                     // pstmt.setString(2,data[1]); //비밀번호
115                                     // pstmt.setString(3,data[3]); //이름
116
117                                     pstmt.setString(1, fields[0].getText()); // 아이디
118                                     pstmt.setString(2, fields[1].getText()); // 비밀번호
119                                     pstmt.setString(3, fields[3].getText()); // 이름
120
121                                     pstmt.setTimestamp(4, reg_date); // 회원가입 날짜
122                                     int result = pstmt.executeUpdate();
123
124                                     if (result == 1) {
125                                             JOptionPane.showMessageDialog(this, "회원 가입 완료");
```

```
126                                                 // this.dispose();
127                                                 MemberManagement management = new MemberManagement();
128
129                                         } else {
130                                                 JOptionPane.showMessageDialog(this, "회원 가입 실패");
131                                         }
132
133                                 } catch (SQLException e) {
134                                         e.printStackTrace();
135                                         System.out.println("새로운 레코드 추가에 실패");
136                                 }
137
138                         } else { // 비밀 번호가 일치하지 않으면 메시지 박스
139                                 JOptionPane.showMessageDialog(this, "비밀번호가 일치하지 않습니다.");
140                         }
141                 }
142
143                 // 다시 작성 처리 부분
144                 public void clearFields() {
145                         for (int i = 0; i < size; i++) {
146                                 fields[i].setText("");
147                         }
148                 }
149
150                 // 입력한 회원 정보값을 구하는 부분
151                 public String[] getFieldValues() {
152                         String values[] = new String[size];
153
154                         for (int i = 0; i < size; i++)
155                                 values[i] = fields[i].getText();
156
157                         return values;
158                 }
159
160                 public static void main(String[] args) {
161                         InsertMember insert = new InsertMember();
162                 }
163         }
```