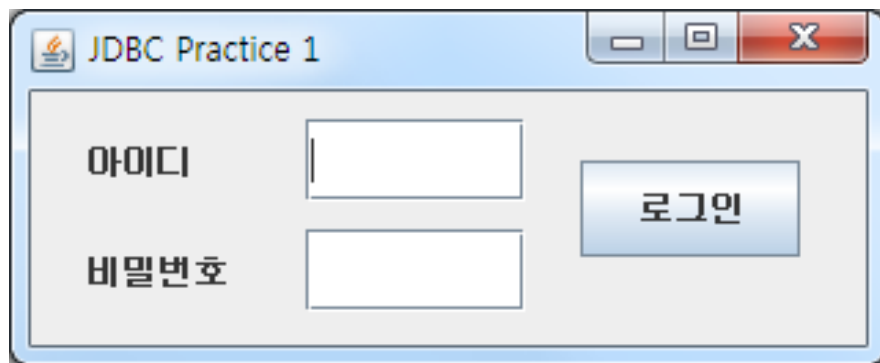


# JDBC와 자바 GUI 실습1

데이터베이스 연구실  
컴퓨터공학과  
한양대학교

# JDBC

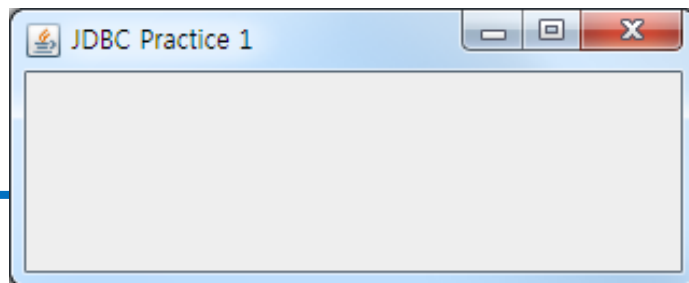
- JDBC를 사용하는 GUI프로그램을 만들어 보자
- JDBC를 활용한 프로그램을 제작함으로써, 데이터베이스 관리 프로그램 자체와 동작환경에 대한 이해를 돕고자 한다
- 로그인 예제
  - 아이디와 비밀번호를 입력을 받아서 데이터베이스에 접속할 수 있도록 만들자



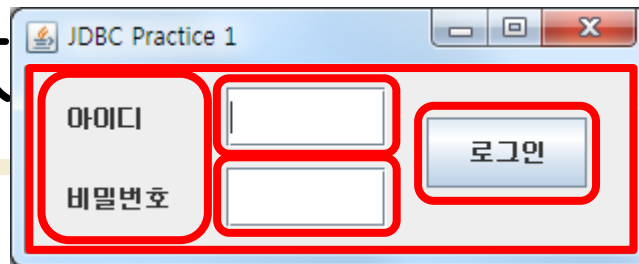
# Window를 하나 생성해보자

- JFrame : 다른 component를 표현할 수 있는 윈도우를 관리하는 클래스

```
public class JDBC_Practice1 {  
    private JFrame frame = new JFrame();  
  
    public JDBC_Practice1() {  
        frame.setTitle("JDBC Practice 1");           // frame 상단의 이름  
        frame.setSize(320, 130);                     // frame의 크기  
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // 종료시 System.exit()을 호출  
        frame.setVisible(true);                      // frame을 화면에 표시  
    }  
  
    public static void main(String[] args) {  
        new JDBC_Practice1();  
    }  
}
```



# Component를 추가해보자



- JPanel
  - Component들을 포함할 수 있는 Container이다.  
Component들의 위치와 모양을 조정하기에 유용하다.
- JLabel
  - Window에 수정할 수 없는 한 라인의 text를 출력
- JTextField
  - 사용자에게 한 라인의 text를 입력 받을 수 있는 도구
- JButton
  - 버튼을 만들어 사용자의 입력을 받을 수 있다.
- JPasswordField
  - JTextField와 같지만 text를 다른 문자로 바꿔 표시한다.

# Component를 추가해보자 (cont'd)

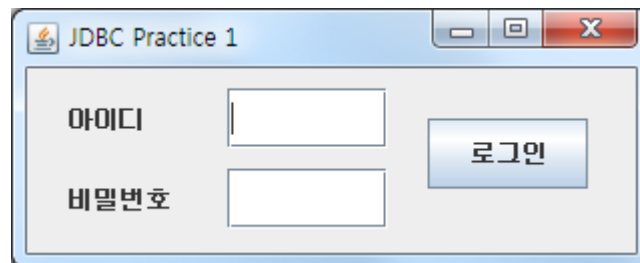
```

private JLabel idLabel = new JLabel("아이디");
private JLabel pwdLabel = new JLabel("비밀번호");
private JTextField idInput = new JTextField();
private JPasswordField pwdInput = new JPasswordField();
private JButton loginButton = new JButton("로그인");

public JDBC_Practice1() {
    panel.setLayout(null);
    // Component들의 위치를 지정
    idLabel.setBounds(20, 10, 60, 30);
    pwdLabel.setBounds(20, 50, 60, 30);
    idInput.setBounds(100, 10, 80, 30);
    pwdInput.setBounds(100, 50, 80, 30);
    loginButton.setBounds(200, 25, 80, 35);
    // Component들을 Panel에 추가
    panel.add(idLabel);
    panel.add(pwdLabel);
    panel.add(idInput);
    panel.add(pwdInput);
    panel.add(loginButton);
    // Panel을 Frame에 추가
    frame.add(panel);

    frame.setTitle("JDBC Practice 1");           // frame 상단의 이름
    frame.setSize(320, 130);                     // frame의 크기
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // 종료시 System.exit()을 호출
    frame.setVisible(true);                      // frame을 화면에 표시
}

```



# Component를 추가해보자 (cont'd)

- Layout
  - Container에서 Component들을 어떻게 배열할 것인지를 정한다.
    - BorderLayout : 동, 서, 남, 북, 가운데로 위치지정
    - FlowLayout : 들어오는 순서대로 위치지정
    - GridLayout : 들어오는 순서대로 행렬모양으로 위치지정
    - null : 개발자가 직접 위치와 크기 지정

# 버튼에 Event를 만들자

- ActionListener : Component에 이벤트가 발생했을 경우 이벤트를 처리하는 핸들러를 호출한다.

```
public class JDBC_Practice1 implements ActionListener {  
  
    public JDBC_Practice1() {  
        panel.setLayout(null);  
        // Component들의 위치를 지정  
        idLabel.setBounds(20, 10, 60, 30);  
        pwdLabel.setBounds(20, 50, 60, 30);  
        idInput.setBounds(100, 10, 80, 30);  
        pwdInput.setBounds(100, 50, 80, 30);  
        loginButton.setBounds(200, 25, 80, 35);  
        // Button에 ActionListener를 연결  
        loginButton.addActionListener(this);  
    }  
  
    public void actionPerformed(ActionEvent e) {  
        if (e.getSource() == loginButton) {  
            // JTextField과 JPasswordField에 입력된 텍스트를 받아온다  
            username = idInput.getText();  
            password = new String(pwdInput.getPassword());  
  
            connectDB();  
        }  
    }  
}
```

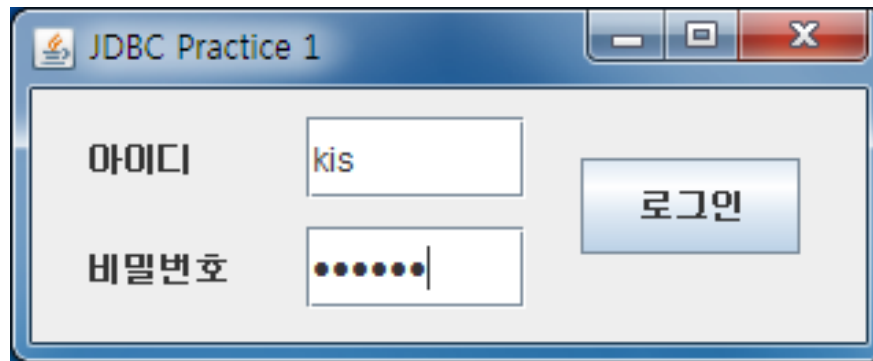
# 버튼에 Event를 만들자 (cont'd)

- addActionListener(ActionListener l);
  - ActionListener인터페이스를 구현한 객체를 이벤트가 발생했을 때 처리할 클래스로 정함
- actionPerformed(ActionEvent e)
  - 이벤트가 발생했을 때 이 함수가 실행된다
- ActionEvent.getSource()
  - 이벤트를 발생시킨 객체를 불러온다
    - (여기서는 loginButton을 눌러서 발생)
- ActionEvent.getCommand()
  - 이벤트를 발생시킨 객체의 text를 불러온다
    - (여기서는 '로그인'버튼을 눌러서 발생)



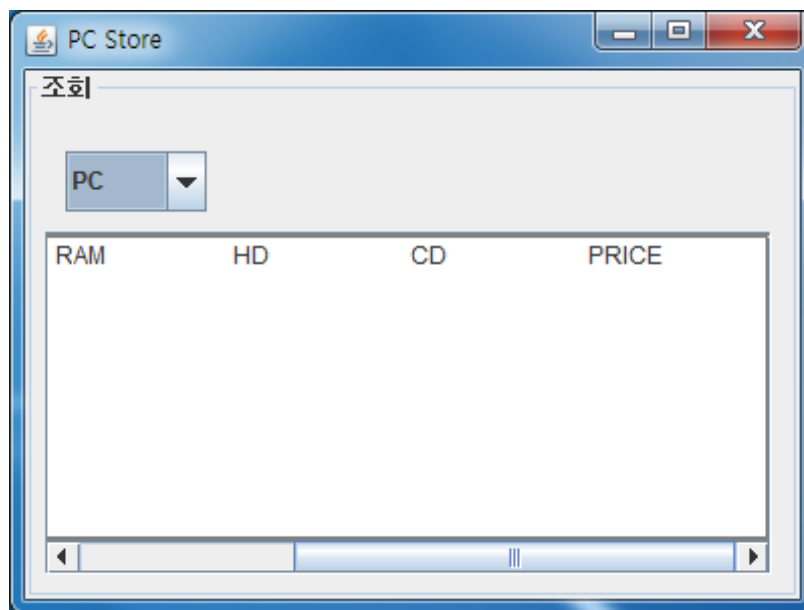
# 실습

- GUI를 활용해 DB 로그인



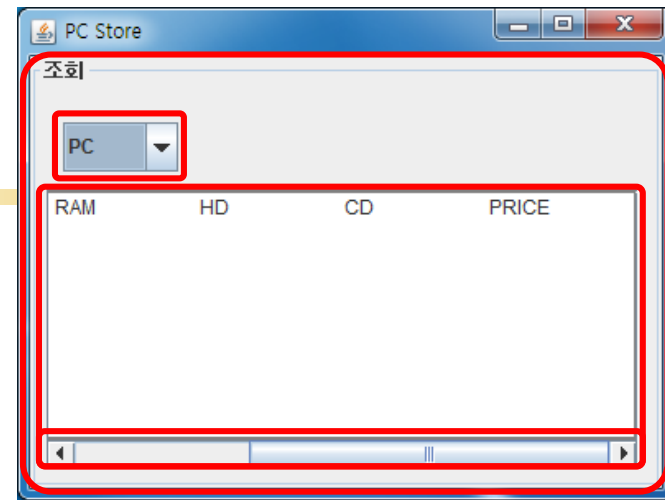
# JDBC

- SQL 질의의 결과를 GUI로 받아보자
- PC가게 예제
  - PC가게의 품목인 PC와 Laptop과 Printer 테이블의 데이터를 출력해본다.



# Component를 추가해보자

- JPanel.setBorder
  - JPanel에 경계를 표시
- JComboBox
  - 선택할 수 있는 값들을 list에 저장하고 그 중 하나의 값을 택함
- JTextArea
  - JTextField와 유사하나, 하나 이상의 라인의 text를 표시
- JScrollPane
  - Component에 스크롤이 가능하도록 만듦



# Component를 추가해보자

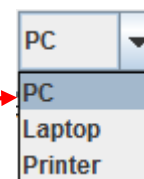
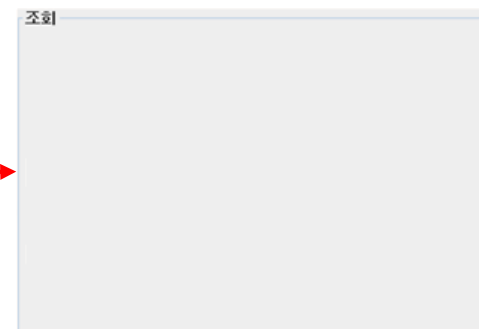
```
private JTextArea check_area = new JTextArea();  
private JComboBox<String> check_box = new JComboBox<String>();
```

```
private void PCstore() {  
    frame.setVisible(false);
```

```
    frame = new JFrame();  
    panel = new JPanel();
```

```
    panel.setFont(new Font("필기체", 1, 12));  
    panel.setBorder(new TitledBorder("조회"));  
    panel.setBounds(380, 80, 490, 280);  
    panel.setLayout(null);
```

```
    check_box.addItem("PC");  
    check_box.addItem("Laptop");  
    check_box.addItem("Printer");
```



# Component를 추가해보자

```
check_area.setBorder(new LineBorder(Color.gray, 2));  
check_area.setEditable(false);
```

```
JScrollPane scroll = new JScrollPane();  
scroll.setViewportView(check_area);
```

```
check_box.setBounds(20, 40, 70, 30);  
scroll.setBounds(10, 80, 360, 170);
```

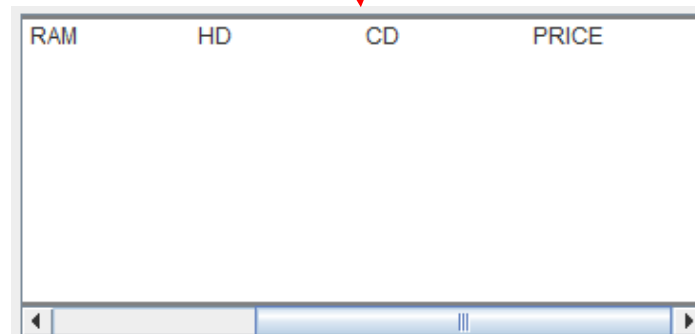
```
check_box.addActionListener(this);
```

```
panel.add(check_box);  
panel.add(scroll);
```

```
frame.add(panel);
```

```
frame.setTitle("PC Store");  
frame.setSize(400, 300);  
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);  
frame.setVisible(true);
```

```
}
```



# ComboBox에 Event를 만들자

- ComboBox를 눌렀을 때 table을 출력한다
  - public void actionPerformed(ActionEvent e) { ... }
  - Private void showTable() throws SQLException { ... }

```
public void actionPerformed(ActionEvent e) {  
    if (e.getSource() == loginButton) {  
        // JTextField와 JPasswordField에 입력된 텍스트를 받아온다  
        username = idInput.getText();  
        password = new String(pwdInput.getPassword());  
  
        connectDB();  
    } else if (e.getSource() == check box) {  
        try {  
            showTable();  
        } catch (SQLException se) {  
            se.printStackTrace();  
        }  
    }  
}
```

# ComboBox에 Event를 만들자

- Cols Table

- 테이블에 저장된 컬럼의 정보를 가지고 있다.

- Table\_name
  - 컬럼이 속한 테이블
- Column\_name
  - 컬럼의 이름
- Data\_type
  - 컬럼의 데이터타입
- Data\_length
  - 컬럼의 데이터크기

※ 모두 대문자로 저장되어 있기 때문에  
값을 명시할 때는 대문자로 해야 한다.

```
SQL> desc cols
```

Name	Null?	Type
TABLE_NAME	NOT NULL	VARCHAR2(30)
COLUMN_NAME	NOT NULL	VARCHAR2(30)
DATA_TYPE		VARCHAR2(106)
DATA_TYPE_MOD		VARCHAR2(3)
DATA_TYPE_OWNER		VARCHAR2(120)
DATA_LENGTH	NOT NULL	NUMBER
DATA_PRECISION		NUMBER
DATA_SCALE		NUMBER
NULLABLE		VARCHAR2(1)
COLUMN_ID		NUMBER
DEFAULT_LENGTH		NUMBER
DATA_DEFAULT		LONG
NUM_DISTINCT		NUMBER
LOW_VALUE		RAW(32)
HIGH_VALUE		RAW(32)
DENSITY		NUMBER
NUM_NULLS		NUMBER
NUM_BUCKETS		NUMBER
LAST_ANALYZED		DATE
SAMPLE_SIZE		NUMBER
CHARACTER_SET_NAME		VARCHAR2(44)
CHAR_COL_DECL_LENGTH		NUMBER
GLOBAL_STATS		VARCHAR2(3)
USER_STATS		VARCHAR2(3)
AUG_COL_LEN		NUMBER
CHAR_LENGTH		NUMBER
CHAR_USED		VARCHAR2(1)
US0_FMT_IMAGE		VARCHAR2(3)
DATA_UPGRADED		VARCHAR2(3)
HISTOGRAM		VARCHAR2(15)

# ComboBox에 Event를 만들자

- ComboBox를 눌렀을 때 다음 method를 호출한다.

```
private void showTable() throws SQLException {  
    String specification = "";
```

```
    String sqlStr = "select count(column_name) num from cols where table_name = '"  
        + ((String) check_box.getSelectedItemAt()).toUpperCase() + "'";  
    PreparedStatement stmt = dbTest.prepareStatement(sqlStr);  
    ResultSet rs = stmt.executeQuery();  
  
    rs.next();  
    int number = rs.getInt("num");  
    String[] tables = new String[number];
```

```
    sqlStr = "select column_name from cols where table_name = '"  
        + ((String) check_box.getSelectedItemAt()).toUpperCase() + "'";  
    stmt = dbTest.prepareStatement(sqlStr);  
    rs = stmt.executeQuery();  
  
    for (number = 0; rs.next(); number++) {  
        tables[number] = rs.getString("column_name");  
        specification += tables[number] + '\t';  
    }
```



# ComboBox에 Event를 만들자

- ComboBox를 눌렀을 때 다음 method를 호출한다.

```
for (specification += "\n"; number > 0; number--) {  
    specification += "-----";  
}  
specification += "\n";  
  
sqlStr = "select * from " + (String) check_box.getSelectedItemAt();  
stmt = dbTest.prepareStatement(sqlStr);  
rs = stmt.executeQuery();  
  
for (number = 0; rs.next(); number++) {  
    for (int i = 0; i < tables.length; i++) {  
        specification += rs.getString(tables[i]) + '\t';  
    }  
    specification += "\n";  
}  
  
check_area.setText(specification);  
  
rs.close();  
stmt.close();  
}
```

# MessageBox

- JOptionPane

- *showMessageDialog(Component, Object)*
- *showMessageDialog(Component, Object, String, int)*
- *showMessageDialog(Component, Object, String, int, Icon)*

Component	부모component를 지정하고 dialog가 닫힐 때까지 비활성화, Null 가능
Object	표시할 내용(object이므로 String등 모든 객체 가능)
String	윈도우의 타이틀
int	메시지 dialog의 타입
Icon	메시지에 표시되는 아이콘

# MessageBox

- `JOptionPane.showMessageDialog(Component, Object, String, int, Icon)`

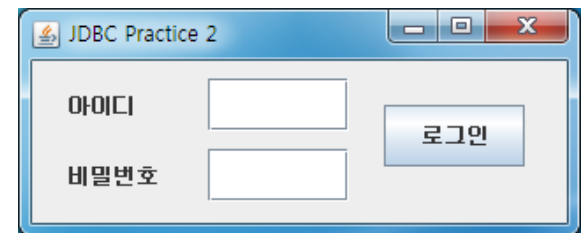
```
JOptionPane.showMessageDialog(null, new JTextArea("메세지\n다이얼로그\n테스트"),  
    "테스트", 2, new ImageIcon("c:\\Users\\kis\\Pictures\\cap.jpg"));
```



# 실습

- RUN SQL Command Line에서 테이블을 하나 생성합니다
- GUI를 활용해 DB 로그인
- 로그인 버튼을 누르면 새로운 창을 띄움

```
create table transaction
(  
    tnumber      integer      not null,  
    tmodel       integer      not null,  
    tcount       integer      not null,  
    tprice       integer      not null,  
    primary key (tnumber)  
);
```

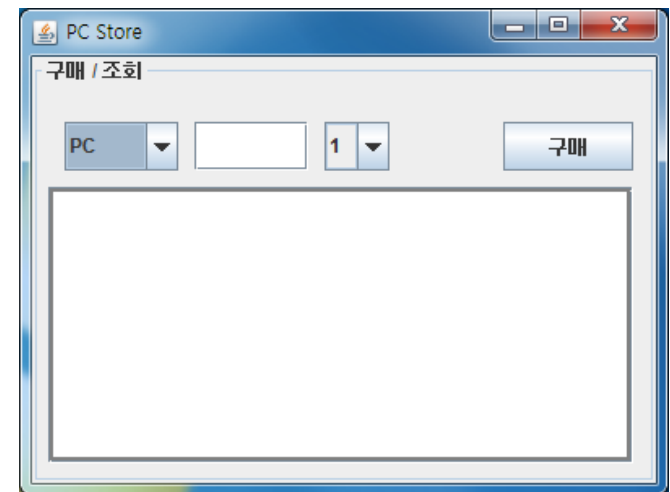


JDBC Practice 2

아이디

비밀번호

로그인



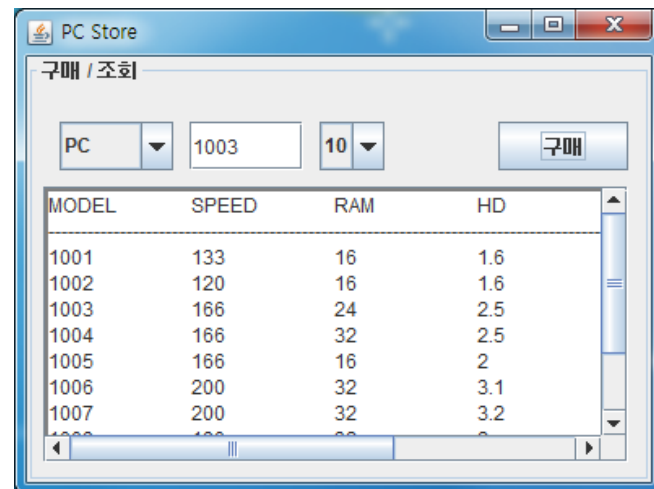
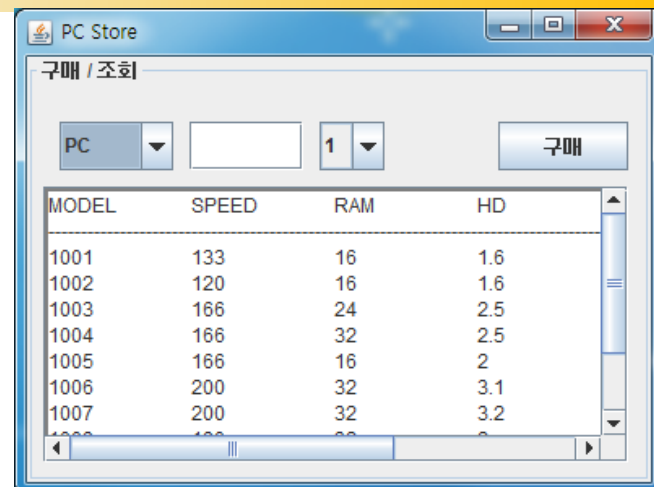
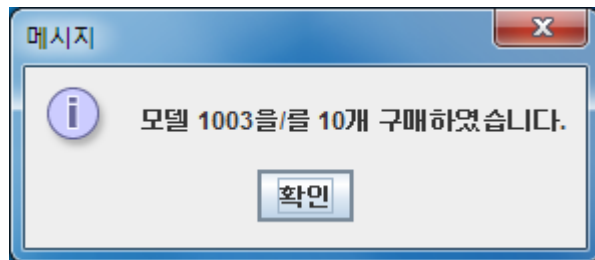
PC Store

구매 / 조회

PC  1

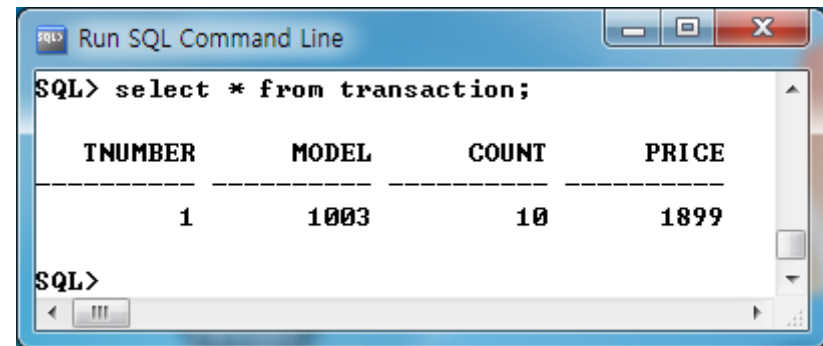
# 실습

- 품목 ComboBox를 선택시 목록을 표시
- 해당 모델번호를 입력, 수량 선택 후 구매버튼을 누르면 구매 확인창을 띄워준다.



# 실습

- Transaction 테이블에 insert 된 것을 확인해보자



\* (추가) 모델번호가 해당 품목에서 없을 때 예외처리

