KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

JDBC 프로그래밍 - CRUD

[KB] IT's Your Life

KB 국민은행

JDBC 프로그래밍 – CRUD

- **JUnit 라이브러리를 이용해서 다음과 같이 출력되도록 프로그래밍하시오.**
  - 프로젝트명 : jdbc_ex(임의로 변경 가능)
  - 패키지명 : org.scoula.jdbc_ex.test
  - 클래스명 : JUnitCycleTest

```
@BeforeAll
@BeforeEach
test1
@AfterEach
@BeforeEach
test2
@AfterEach
@BeforeEach
test3
@AfterEach
@AfterAll
```

JUnitCycleTest.java

```
package org.scoula.jdbc_ex.test;

import org.junit.jupiter.api.*;

public class JUnitCycleTest {
    @BeforeAll  // 전체 테스트 시작전 1회 실행, static 선언
    static void beforeAll() {
        System.out.println("@BeforeAll");
    }

    @BeforeEach // 테스트 케이스를 시작하기 전마다 실행
    public void beforeEach() {
        System.out.println("@BeforeEach");
    }

    @Test
    public void test1() {
        System.out.println("test1");
    }

    @Test
    public void test2() {
        System.out.println("test2");
    }

    @Test
    public void test3() {
        System.out.println("test3");
    }
    @AfterEach  // 테스트 케이스를 종료하기 전마다 실행
    public void afterEach() {
        System.out.println("@AfterEach");
    }

    @AfterAll   // 전체 테스트를 마지고 종료하기 전 1회. static 선언
    static void afterAll() {
        System.out.println("@AfterAll");
    }

}
```

JDBC 프로그래밍 – CRUD

- Lombok라이브러리를 이용하여 다음을 구현하시오.
  - 패키지명 : org.scoula.jdbc_ex.domain
  - 클래스명 : UserVO
  - 필드 정의
    - String id, String password, String name, String role
  - 기본 생성자와 모든 필드를 매개변수로 받는 생성자를 제공해야 한다.
  - 각 필드에 대한 getter와 setter 메서드를 제공해야 한다.
  - 객체 상태를 쉽게 출력할 수 있도록 toString() 메서드도 제공되어야 한다.
  - Lombok 라이브러리를 사용하여 필요한 메서드를 자동 생성하되, 수동 작성은 하지 않는다.

UserVO.java

```
package org.scoula.jdbc_ex.domain;


import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
public class UserVO {
    private String id;
    private String password;
    private String name;
    private String role;
}
```

JDBC 프로그래밍 – CRUD

- 다음 조건을 따르는 UserDaoImpl.java를 구현하시오.
  - 패키지명 : org.scoula.jdbc_ex.dao
  - UserDao 인터페이스에 각 CRUD기능이 추상메서드로 정의되어 있음.
  - UserDaoImpl.java는 UserDao.java를 구현하여 프로그래밍함.
  - UserDaoImpl.java의 각 메서드는 JDBCUtil.java파일의 getConnection()메서드를 호출하여 db연결을 수행하고 JDBC 3단계부터 구현함.
  - UserDao.java

```
package org.scoula.jdbc_ex.dao;


import org.scoula.jdbc_ex.domain.UserVO;

import java.sql.SQLException;
import java.util.List;
import java.util.Optional;

public interface UserDao {
    // 회원 등록
    int create(UserVO user) throws SQLException;
    // 회원 목록 조회
    List<UserVO> getList() throws SQLException;
    // 회원 정보 조회
    UserVO get(String id) throws SQLException;
    // 회원 수정
    int update(UserVO user) throws SQLException;
    // 회원 삭제
    int delete(String id) throws SQLException;
}
```

JDBCUtil.java

```
package org.scoula.jdbc_ex.common;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.util.Properties;

public class JDBCUtil {
    static Connection conn = null;

    public static Connection getConnection() {

        if(conn != null)
            return conn;

        try {
            Properties properties = new Properties();
            properties.load(JDBCUtil.class.getResourceAsStream("/application.properties"));
            String driver = properties.getProperty("driver");
            String url = properties.getProperty("url");
            String id = properties.getProperty("id");
            String password = properties.getProperty("password");

            Class.forName(driver);
            conn = DriverManager.getConnection(url, id, password);
        } catch (Exception e) {
            e.printStackTrace();
        }

        return conn;
    }
```

```
package org.scoula.jdbc_ex.common;


    public static void close() {
        try {
            if (conn != null) {
                conn.close();
                conn = null;
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

}
```

JDBC 프로그래밍 – CRUD

UserDaoImpl.java

```
package org.scoula.jdbc_ex.dao;


import org.scoula.jdbc_ex.common.JDBCUtil;
import org.scoula.jdbc_ex.domain.UserVO;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

public class UserDaoImpl implements UserDao {
    Connection conn = JDBCUtil.getConnection();

    // USERS 테이블 관련 SQL 명령어
    private String USER_LIST = "select * from users";
    private String USER_GET = "select * from users where id = ?";
    private String USER_INSERT = "insert into users values(?, ?, ?, ?)";
    private String USER_UPDATE = "update users set name = ?, role = ? where id = ?";
    private String USER_DELETE = "delete from users where id = ?";
```

```
    // 회원 등록
    @Override
    public int create(UserVO user) throws SQLException {
    //구현하는 부분
    }

    // 회원 목록 조회
    @Override
    public List<UserVO> getList() throws SQLException {
     //구현하는 부분
    }

    // 회원 정보 조회
    @Override
    public UserVO get(String id) throws SQLException {
     //구현하는 부분
    }

    // 회원 수정
    @Override
    public int update(UserVO user) throws SQLException {
     //구현하는 부분
    }

    // USERS 테이블 관련 CRUD 메소드
    // 회원 삭제
    @Override
    public int delete(String id) throws SQLException {
      //구현하는 부분
    }
} //class
```

JDBC 프로그래밍 – CRUD

UserDaoImpl.java

```
package org.scoula.jdbc_ex.dao;


import org.scoula.jdbc_ex.common.JDBCUtil;
import org.scoula.jdbc_ex.domain.UserVO;

import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

public class UserDaoImpl implements UserDao {
    Connection conn = JDBCUtil.getConnection();

    // USERS 테이블 관련 SQL 명령어
    private String USER_LIST = "select * from users";
    private String USER_GET = "select * from users where id = ?";
    private String USER_INSERT = "insert into users values(?, ?, ?, ?)";
    private String USER_UPDATE = "update users set name = ?, role = ? where id = ?";
    private String USER_DELETE = "delete from users where id = ?";
```

```
// 회원 등록
  @Override
  public int create(UserVO user) throws SQLException {
      PreparedStatement stmt = conn.prepareStatement(USER_INSERT);
      stmt.setString(1, user.getId());
      stmt.setString(2, user.getPassword());
      stmt.setString(3, user.getName());
      stmt.setString(4, user.getRole());
      int count = stmt.executeUpdate();
      stmt.close();
      return count;
  } //create

  // 회원 목록 조회
  @Override
  public List<UserVO> getList() throws SQLException {
      List<UserVO> userList = new ArrayList<>();
      Connection conn = JDBCUtil.getConnection();
      PreparedStatement stmt = conn.prepareStatement(USER_LIST);
      ResultSet rs = stmt.executeQuery();
      while (rs.next()) {
          UserVO user = new UserVO();
          user.setId(rs.getString("ID"));
          user.setPassword(rs.getString("PASSWORD"));
          user.setName(rs.getString("NAME"));
          user.setRole(rs.getString("ROLE"));
          userList.add(user);
      } //while
      rs.close();
      stmt.close();
      return userList;
  } //getList
```

JDBC 프로그래밍 – CRUD

UserDaoImpl.java

```
// 회원 정보 조회
@Override
public UserVO get(String id) throws SQLException {
    UserVO user = new UserVO();
    PreparedStatement stmt = conn.prepareStatement(USER_GET);
    stmt.setString(1, id);

    ResultSet rs = stmt.executeQuery();
    if (rs.next()) {
        user.setId(rs.getString("ID"));
        user.setPassword(rs.getString("PASSWORD"));
        user.setName(rs.getString("NAME"));
        user.setRole(rs.getString("ROLE"));
    } //if
    rs.close();
   stmt.close();
   return user;
} //get

// 회원 수정
@Override
public int update(UserVO user) throws SQLException {
    PreparedStatement stmt = conn.prepareStatement(USER_UPDATE);
    stmt.setString(1, user.getName());
    stmt.setString(2, user.getRole());
    stmt.setString(3, user.getId());
    int count = stmt.executeUpdate();
    stmt.close();
    return count;
} //update
```

```
    // USERS 테이블 관련 CRUD 메소드
    // 회원 삭제
    @Override
    public int delete(String id) throws SQLException {
      PreparedStatement stmt = conn.prepareStatement(USER_DELETE);
       stmt.setString(1, id);
       int rows = stmt.executeUpdate();
       stmt.close();
       return rows;
    }  //delete
} //class
```

JDBC 프로그래밍 – CRUD

- 다음과 같이 실행되도록 UserMain.java를 구현하시오.
  - 패키지명 : org.scoula.jdbc_ex
  - 파일명 : UserMain.java
  - DB처리는 UserDaoImpl.java의 각 메서드를 호출하여 사용함.
  - Sql문에 들어갈 데이터는 Scanner를 이용하여 키보드로 입력하여 사용함.

- **데이터 삽입 예제**

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 1
```

```
id, pw, name, role을 순서대로 입력하시오.
id(12글자 이내) >> apple
pw(12글자 이내) >> 1234
name(30글자 이내) >> apple
role(6글자 이내) >> admin
insert 성공!
```

전체 목록 조회 예제

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 2
아이디 : a, 패스워드 : a, 이름 : a, 역할 : a
아이디 : admin, 패스워드 : admin123, 이름 : 관리자, 역할 : ADMIN
아이디 : apple, 패스워드 : 1234, 이름 : apple, 역할 : admin
아이디 : b, 패스워드 : b, 이름 : b, 역할 : b
아이디 : guest, 패스워드 : guest123, 이름 : 방문자, 역할 : USER
아이디 : member, 패스워드 : member123, 이름 : 일반회원, 역할 : USER
아이디 : win, 패스워드 : win, 이름 : win, 역할 : USER
```

JDBC 프로그래밍 – CRUD

Id 검색 예제

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 3
검색하고자하는 id를 입력하시오.
id(12글자 이내) >> apple
아이디 : apple, 패스워드 : 1234, 이름 : apple, 역할 : admin
```

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 3
검색하고자하는 id를 입력하시오.
id(12글자 이내) >> ice
검색 실패, 없는 id입니다.!
```

데이터 갱신 예제

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 4
name, role, id를 순서대로 입력하시오.
name(30글자 이내) >> summer
role(6글자 이내) >> king
id(12글자 이내) >> apple
update 성공!
```

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 4
name, role, id를 순서대로 입력하시오.
name(30글자 이내) >> apple
role(6글자 이내) >> queen
id(12글자 이내) >> ice
update 실패!
```

데이터 삭제 예제

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 5
삭제하고자하는 id를 입력하시오.
id(12글자 이내) >> apple
delete 성공!
```

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 5
삭제하고자하는 id를 입력하시오.
id(12글자 이내) >> ice
delete 실패!
```

```
원하는 작업을 선택하시오.
======================
1. insert
2. selectList
3. selectOne
4. update
5. delete
6. exit
======================
번호 입력>> 6
프로그램을 종료합니다.
```

UserMain.java

```
package org.scoula.jdbc_ex;

import org.scoula.jdbc_ex.dao.UserDao;
import org.scoula.jdbc_ex.dao.UserDaoImpl;
import org.scoula.jdbc_ex.domain.UserVO;
import java.sql.SQLException;
import java.util.List;
import java.util.Scanner;

public class UserMain {
    static UserDao userDao = new UserDaoImpl();
    static final Scanner sc = new Scanner(System.in);

    public static void main(String[] args) throws SQLException {
        while (true) {
            System.out.println("원하는 작업을 선택하시오.");
            System.out.println("====================");
            System.out.println("1. insert");
            System.out.println("2. selectList");
            System.out.println("3. selectOne");
            System.out.println("4. update");
            System.out.println("5. delete");
            System.out.println("6. exit");
            System.out.println("====================");
            System.out.print("번호 입력>> ");

            int choice = sc.nextInt();

            if (choice == 1) {
                insert();
            } else if (choice == 2) {
                selectList();
            } else if (choice == 3) {
                selectOne();
            } else if (choice == 4) {
                update();
            } else if (choice == 5) {
                delete();
            } else if (choice == 6) {
                System.out.println("프로그램을 종료합니다.");
                JDBCUtil.close();
                System.exit(0); //프로그램 종료
            } else {
                System.out.println("선택이 올바르지 않음.");
            }
        }
    }

    private static void printVO(UserVO userVO) {
        System.out.println("아이디 : " + userVO.getId() + ", 패스워드 : " + userVO.getPassword() + ", 이
름 : " + userVO.getName() + ", 역할 : " + userVO.getRole());
    }
```

UserMain.java

```
public static void insert() throws SQLException {
		//구현
}

public static void selectList() throws SQLException {
		//구현
}

public static void selectOne() throws SQLException {
		//구현
}

public static void update() throws SQLException {
		//구현
}

public static void delete() throws SQLException {
		//구현
}
```

UserMain.java

```
package org.scoula.jdbc_ex;

import org.scoula.jdbc_ex.dao.UserDao;
import org.scoula.jdbc_ex.dao.UserDaoImpl;
import org.scoula.jdbc_ex.domain.UserVO;
import java.sql.SQLException;
import java.util.List;
import java.util.Scanner;

public class UserMain {
    static UserDao userDao = new UserDaoImpl();
    static final Scanner sc = new Scanner(System.in);

    public static void main(String[] args) throws SQLException {
        while (true) {
            System.out.println("원하는 작업을 선택하시오.");
            System.out.println("====================");
            System.out.println("1. insert");
            System.out.println("2. selectList");
            System.out.println("3. selectOne");
            System.out.println("4. update");
            System.out.println("5. delete");
            System.out.println("6. exit");
            System.out.println("====================");
            System.out.print("번호 입력>> ");

            int choice = sc.nextInt();

            if (choice == 1) {
                insert();
            } else if (choice == 2) {
                selectList();
            } else if (choice == 3) {
                selectOne();
            } else if (choice == 4) {
                update();
            } else if (choice == 5) {
                delete();
            } else if (choice == 6) {
                System.out.println("프로그램을 종료합니다.");
                JDBCUtil.close();
                System.exit(0); //프로그램 종료
            } else {
                System.out.println("선택이 올바르지 않음.");
            }
        }
    }

    private static void printVO(UserVO userVO) {
        System.out.println("아이디 : " + userVO.getId() + ", 패스워드 : " + userVO.getPassword() + ", 이
름 : " + userVO.getName() + ", 역할 : " + userVO.getRole());
    }
```

UserMain.java

```
public static void insert() throws SQLException {
    UserVO user = new UserVO();

    System.out.println("id, pw, name, role을 순서대로 입력하시오.");
    System.out.print("id(12글자 이내) >> ");
    user.setId(sc.next());
    System.out.print("pw(12글자 이내) >> ");
    user.setPassword(sc.next());
    System.out.print("name(30글자 이내) >> ");
    user.setName(sc.next());
    System.out.print("role(6글자 이내) >> ");
    user.setRole(sc.next());

    int result = userDao.create(user);

    if (result == 1) {
        System.out.println("insert 성공!");
    }
}
```

```
public static void selectList() throws SQLException {
    List<UserVO> list = userDao.getList();

    if (list.size() == 0) {
        System.out.println("검색 실패, 데이터없음.");
    }else {
        for (UserVO userVO : list) {
            printVO(userVO);
        }
    }
}

public static void selectOne() throws SQLException {
    System.out.println("검색하고자하는 id를 입력하시오.");
    System.out.print("id(12글자 이내) >> ");
    String id = sc.next();

    UserVO userVO = userDao.get(id);

    if (userVO.getId() == null) {
        System.out.println("검색 실패, 없는 id입니다.!");
    }else {
        printVO(userVO);
    }
}
```

UserMain.java

```
public static void update() throws SQLException {

        //가방 만들고
        UserVO user = new UserVO();

        //입력받아 가방에 넣고
        System.out.println("name, role, id를 순서대로 입력하시오.");
        System.out.print("name(30글자 이내) >> ");
        user.setName(sc.next());
        System.out.print("role(6글자 이내) >> ");
        user.setRole(sc.next());
        System.out.print("id(12글자 이내) >> ");
        user.setId(sc.next());

        //dao의 update메서드 호출하면서 가방 전달하고
        int result = userDao.update(user);

        //결과 처리
        if (result == 1) {
            System.out.println("update 성공!");
        }else {
            System.out.println("update 실패!");
        }
    }
```

```
public static void delete() throws SQLException {
        System.out.println("삭제하고자하는 id를 입력하시오.");
        System.out.print("id(12글자 이내) >> ");
        String id = sc.next();

        int result = userDao.delete(id);

        //결과 처리
        if (result == 1) {
            System.out.println("delete 성공!");
        }else {
            System.out.println("delete 실패!");
        }
    }
}
```