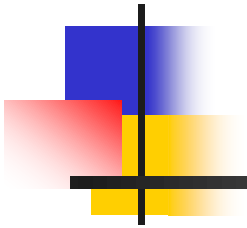
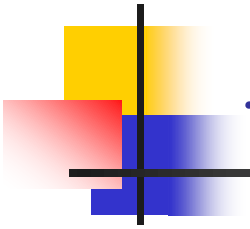


java 16강 - JDBC

양 명 숙
[now4ever7@gmail.com]

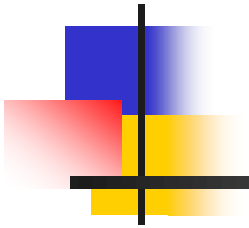


목차

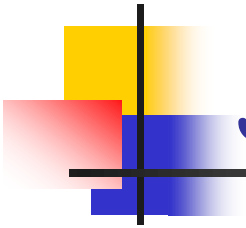
- JDBC

- ResultSet의 커서 이동
- execute() 메서드 이용
- 저장 프로시저 이용

- DAO/DTO



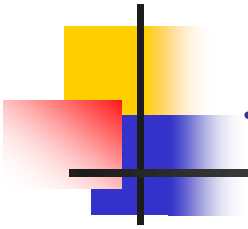
JDBC



JDBC

- JDBC(Java Database Connectivity)
  - **자바** 프로그램과 **데이터베이스를 연결**하는 프로그래밍 방식
  - 자바언어로 데이터베이스에 접근할 때 사용되는 API로 java.sql 패키지를 의미함
  - java 프로그램은 JDBC를 통해 데이터베이스에 연결하여 데이터를 검색하고, 입력, 수정, 삭제할 수 있음
  - 데이터베이스에 접근할 경우 내부적으로 JDBC를 사용함
- 드라이버 설치(ojdbc5.jar, ojdbc6.jar)
  - 자바 프로그램에게, 연결해서 사용할 데이터베이스 프로그램의 사용방법을 알려주는 것
  - C:₩app₩yang₩product₩11.2.0₩dbhome_1₩jdbc₩lib₩ojdbc5.jar 를 복사하여

    c:₩java₩jdk1.8.0_65₩jre₩lib₩ext 에 붙여넣기



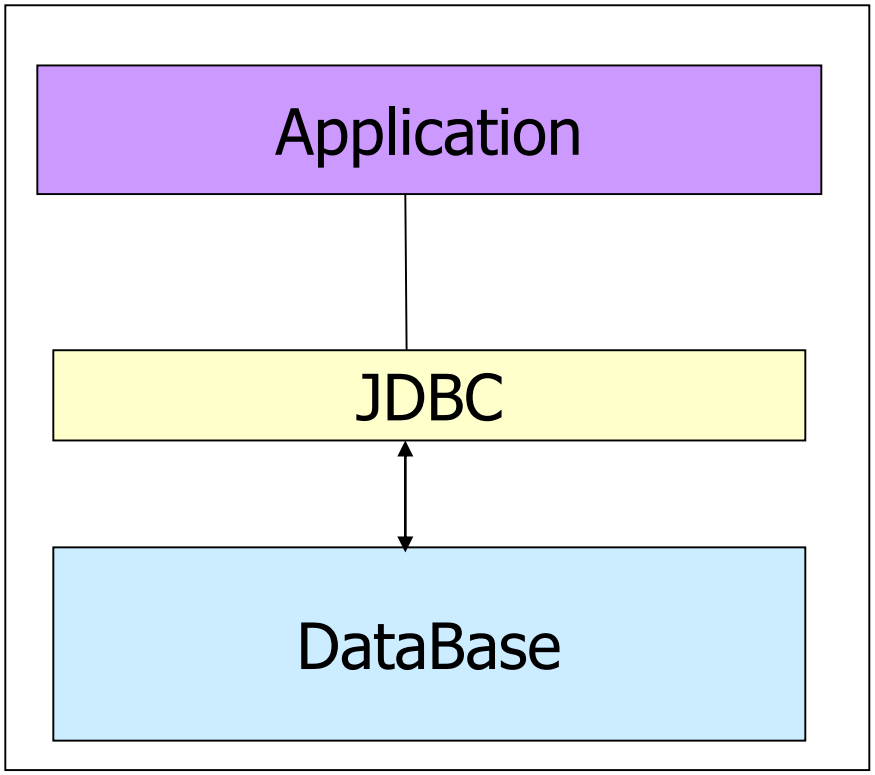
환경 변수 셋팅

- classpath에 추가
  - %classpath%;.;C:₩java₩jdk1.8.0_65₩jre₩lib₩ext₩ojdbc5.jar;
- 이클립스 사용시
  - Build Path - 라이브러리에 ojdbc5.jar 추가

JDBC

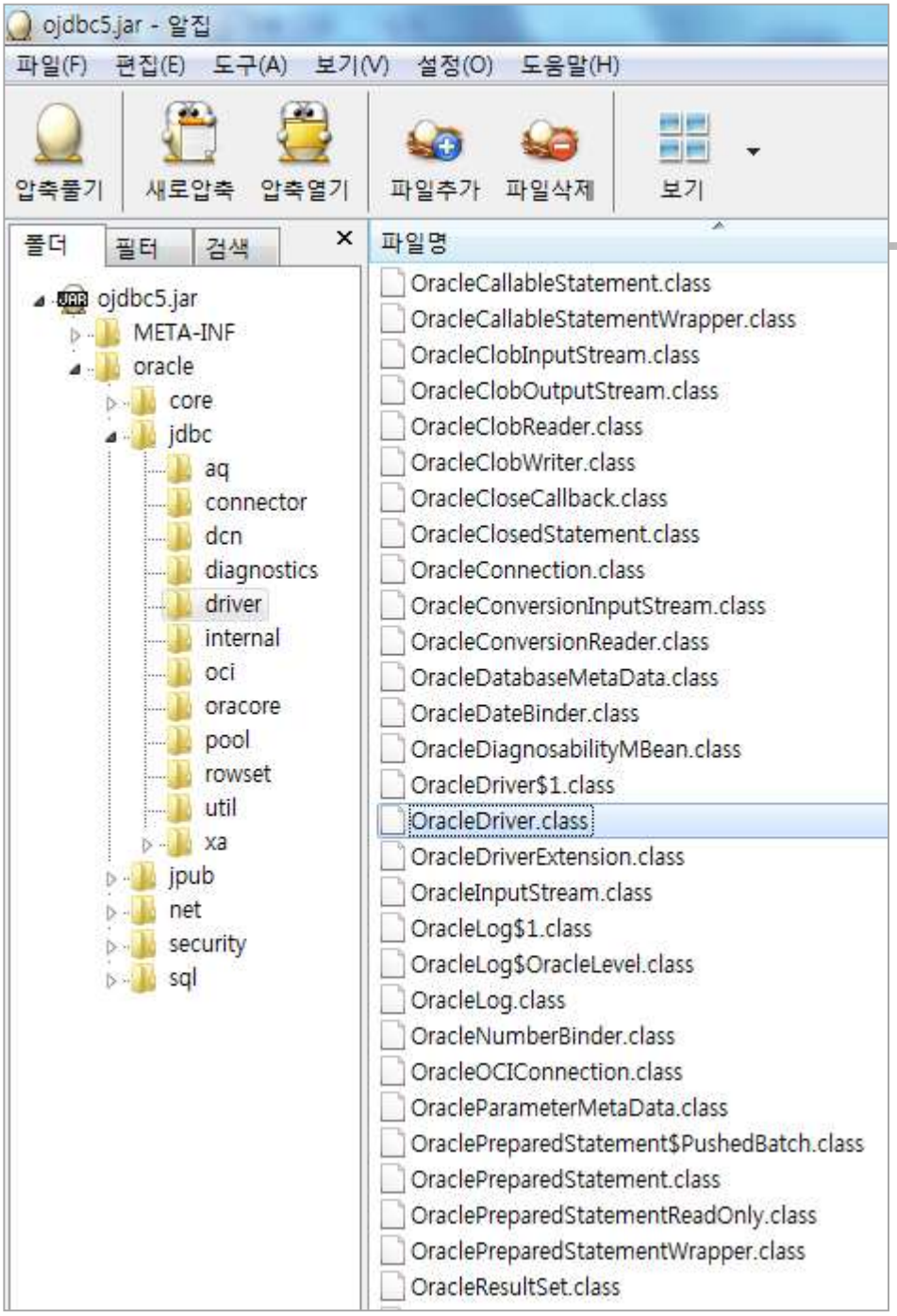
java 프로그램은 **JDBC를 통해 데이터베이스에 연결**하여 데이터를 검색하고, 입력, 수정, 삭제할 수 있음

- JDBC
  - 데이터에 대한 연결성을 제공(연결통로)
  - 필요한 데이터들에 대한 접근을 제공하는 역할



java, jsp

oracle



http://docs.oracle.com/javase/8/docs/api/index.html

ojdbc5.jar

oracle.jdbc.driver.OracleDriver

java.sql 패키지의 인터페이스들을 DBMS 벤더(오라클, ms sql, mysql 등)들이 상속받아 JDBC Driver(클래스 파일들)를 구현.

Interfaces

Array
Blob
CallableStatement
Clob
Connection
DatabaseMetaData
Driver
NClob
ParameterMetaData
PreparedStatement
Ref
ResultSet
ResultSetMetaData
RowId
Savepoint
SQLData
SQLInput
SQLOutput
SQLXML
Statement
Struct
Wrapper

설정정보 확인

- 설정정보 확인
  - C:₩app₩yang₩product₩11.2.0₩dbhome_1₩NETWORK₩ADMIN₩tnsnames.ora, listener.ora => host명과

  - 자바소스에서
    String url = "jdbc:oracle:thin:@192.168.0.105:1521:orcl";

    "jdbc:oracle:thin:@<HOST>:<PORT>:<SID>"

  - <HOST>명을 일치시켜야 함

java.sql 패키지 이용

JDBC 프로그래밍 순서

- url 형식

프로토콜 : 서브프로토콜 : SID

jdbc:oracle:thin:@localhost:1521:orcl
jdbc:mysql://localhost:3306/t_database

- 1. 데이터베이스와 연결하는 드라이버 클래스 찾기(드라이버 로딩)
  - Class.forName("oracle.jdbc.driver.OracleDriver");
- 2. 드라이버 클래스를 통해 데이터베이스 서버와 연결하는 Connection객체 생성
  - String url="jdbc:oracle:thin:@localhost:1521:orcl";
  - String id="hr", pwd="hr123";
  - Connection con = DriverManager.getConnection(url, id, pwd);
- 3. 작업을 처리할 Statement, PreparedStatement, CallableStatement 객체 생성
  - Statement stmt = con.createStatement();
  - 또는 PreparedStatement pstmt = con.prepareStatement("쿼리문")
- 4. Statement/PreparedStatement를 통해 쿼리문 전송
  - (1) insert, delete, update 문인 경우
    - int cnt =stmt.executeUpdate()
  - (2) select 문인 경우
    - ResultSet rs=stmt.executeQuery()
- 5. ResultSet 객체를 통한 Query 결과 처리
- 6. 접속 종료(자원 반납)
  - rs.close(); stmt.close(); con.close();
  - null체크해서 close()해주고, finally 블럭에서 구현

- mysql

Class.forName("com.mysql.jdbc.Driver");
String url="jdbc:mysql://localhost:3306/test_database";

JDBC 프로그래밍 순서

#	NO	NAME	TEL
1	1	홍길동	010-100-2000
2	2	김연아	010-200-5000
3	3	윤아	010-300-7000
4	5	이승기	010-111-2222

- 5. select 문일 경우
  - ResultSet의 논리적 커서를 이동시키면서 각 컬럼의 데이터를 꺼내온다.
  - boolean b=rs.next() : 커서 이동, 커서가 위치한 지점에 레코드가 있으면 true를 리턴, 없으면 false를 리턴한다.
  - **커서는 맨 처음에 첫 번째 행의 직전에 위치**하고 있다가, next()가 호출되면 다음 행으로 이동한다.

- rs.get~(컬럼인덱스) / rs.get~(컬럼명) 메소드 : 데이터를 꺼내온다.
  - get 뒤에는 컬럼의 데이터 유형에 맞는 자료형을 기재
    - number 인경우 rs.getInt(1);
    - varchar2인경우 rs.getString(2);
    - date인 경우 rs.getDate("regdate");

```
rs.getInt("no");
rs.getString("name");
```

기본 SQL문

```
select [칼럼명 또는 표현식]
from [테이블명, 뷰명]
where 원하는 조건;
```

```
select * from mem where id='hong';
```

```
INSERT INTO table [(column1, column2, ...)]
VALUES (value1, value2, ....)
```

```
INSERT INTO dept2 (dcode, dname, pdept , area)
VALUES (9000, '특판1팀', '영업부', '임시지역');
```

```
insert into mem(no, id, pwd, name, hp)
 values (mem_seq.nextval, 'hong', '1','홍길동', '010-100-2000') ;
```

기본 SQL문

```
UPDATE  table
SET column = value
WHERE 조건;
```

```
UPDATE professor
SET bonus = 100
WHERE position='조교수';
```

```
DELETE FROM  table
WHERE 조건;
```

```
DELETE FROM dept2
WHERE dcode between 9000 and 9100;
```

예제1-insert

```
import java.sql.*;
import java.util.Scanner;
class  InsertTest{
	public static void main(String[] args){
		Scanner sc=new Scanner(System.in);
		System.out.print("이름 : ");
		String name=sc.nextLine();
		System.out.print("전화번호 : ");
		String tel=sc.nextLine();

		Connection conn=null;
		Statement stmt=null;
		try {
			Class.forName("oracle.jdbc.driver.OracleDriver");
			System.out.println("드라이버 로딩 성공!");

			String url="jdbc:oracle:thin:@userpc:1521:orcl8";
			String user="javauser1", pwd="java";

			conn=DriverManager.getConnection(url, user, pwd);
			System.out.println("DB 연결됨!");

			stmt=conn.createStatement();
			String sql="insert into person(no, name, tel) values(person_seq.nextval,"
					+"'"+name+"', '"+tel+"')";
```

예제1-insert

```
                int cnt=stmt.executeUpdate(sql);
                System.out.println(cnt+"개의 레코드가 입력됨!");
        } catch (ClassNotFoundException e) {
                e.printStackTrace();
        } catch (SQLException e) {
                e.printStackTrace();
        }finally{
                try {
                        if(stmt!=null) stmt.close();
                        if(conn!=null)conn.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }

    }//main
}
```

예제2-select

```
드라이버 로딩 성공!
DB 연결됨!
1        홍길동   010-100-2000
2        김연아   010-200-5000
3        윤아     010-300-7000
```

```
class JDBCTest1{
    public static void main(String[] args)    {
        try{
            //1. 드라이버 로딩
            Class.forName("oracle.jdbc.driver.OracleDriver");
            System.out.println("드라이버 로딩 성공!");

            //2. DB와 연결
            //String id="scott", pwd="tiger";
            String url="jdbc:oracle:thin:@userpc:1521:orcl8";
            String user="javauser1", pwd="java";

            Connection con=DriverManager.getConnection(url, user, pwd);
            System.out.println("DB 연결됨!");
            //System.out.println("DB Con: "+con);

            String sql="SELECT no, name, tel FROM person order by no";
            //3. DB에 쿼리문을 전송하기 위한 Statement객체 생성
            Statement stmt=con.createStatement();

            //4. DB에 쿼리문 전송
            ResultSet rs=stmt.executeQuery(sql);
```

예제2-select

```
                    //5. ResultSet의 커서를 이동하면서 데이터 꺼내오기
                    while(rs.next()){
                            int no = rs.getInt("no");  //rs.getInt(1);
                            String name=rs.getString("name");
                            String tel=rs.getString("tel");
                            System.out.println(no+"₩t"+name + "₩t" + tel);
                    }//while

                    //6. DB와 연결된 자원 반납
                    if(rs!=null) rs.close();
                    if(stmt!=null) stmt.close();
                    if(con!=null) con.close();

            }catch (ClassNotFoundException e){
                    e.printStackTrace();
            }catch(SQLException e){
                    e.printStackTrace();
            }
    }
}
```

PreparedStatement

```
class  PreparedStatementTest{
    public static void main(String[] args)    {
        Scanner sc=new Scanner(System.in);
        System.out.print("이름 : ");
        String name=sc.nextLine();
        System.out.print("전화번호 : ");
        String tel=sc.nextLine();

        Connection conn=null;
        PreparedStatement ps=null;
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            System.out.println("드라이버 로딩 성공!");

            String url="jdbc:oracle:thin:@userpc:1521:orcl8";
            String user="javauser1", pwd="java";
            conn=DriverManager.getConnection(url, user, pwd);
            System.out.println("DB 연결됨!");

            String sql="insert into person values(person_seq.nextval,?,?)";
            //? => in parameter
```

PreparedStatement

```
                ps=conn.prepareStatement(sql);
                //전처리 시켜놓을 sql문을 PreparedStatement할당 받을 때 매개변수로 전달

                //인 파라미터 값을 세팅
                ps.setString(1, name);//name
                ps.setString(2, tel);//tel

                int n=ps.executeUpdate();
                System.out.println(n+"개 레코드 처리됨");
        }catch (ClassNotFoundException e) {
                e.printStackTrace();
        }catch (SQLException e) {
                e.printStackTrace();
        }finally{
                try {
                        if(ps!=null) ps.close();
                        if(conn!=null) conn.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
    }
}
```

ResultSet의 커서 이동

- ResultSet의 커서를 자유롭게 이동시키기 위한 설정

```
Statement st=con.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
            ResultSet.CONCUR_READ_ONLY);
```

- rs.afterLast() : rs를 마지막 행의 바로 뒤에 위치시킴
  - rs.previous()
- rs.beforeFirst(): 첫번째 행의 직전에 위치
  - rs.next()
- rs.first(): 첫번째 행에 위치
- rs.last(): 마지막행에 위치
- rs.absolute(3): 3번째 행으로 이동
- rs.getRow(): 실제 커서가 위치한 곳의 행의 번호 리턴

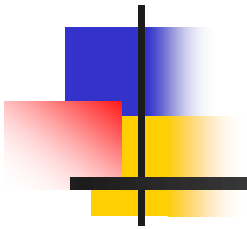
ResultSet의 커서 이동

```
DB연결!
-----------------------------------
 번호    이름     전화
-----------------------------------
7        홍길동   010-777-7777
6        이승기   010-555-7777
5        이승기   010-111-2222
4        김길동   011-333-5555
3        윤아     010-300-7000
2        김연아   010-200-5000
1        홍길동   010-100-2000
-----------------------------------
3: 윤아
```

```
class ReverseSelect{
    public static void main(String[] args){
        Connection conn=null;
        PreparedStatement ps=null;
        ResultSet rs=null;
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            System.out.println("드라이버 로딩 성공!");

            String url="jdbc:oracle:thin:@userpc:1521:orcl8";
            String user="javauser1", pwd="java";

            conn=DriverManager.getConnection(url, user, pwd);
            System.out.println("DB 연결됨!");

            //ResultSet의 커서를 자유 자재로 이동시키려면...
            Statement st=conn.createStatement(
                                ResultSet.TYPE_SCROLL_SENSITIVE,
                                ResultSet.CONCUR_READ_ONLY);

            String sql="select * from person order by no";
            rs=st.executeQuery(sql);
            System.out.println("-----------------------------------");
            System.out.println(" 번호₩t이름₩t전화");
            System.out.println("-----------------------------------");
```

ResultSet의 커서 이동

```
                rs.afterLast();//rs를 마지막 행의 바로 뒤에 위치시킨다.
                while(rs.previous()){
                        String no=rs.getString(1);
                        String name=rs.getString(2);
                        String tel=rs.getString(3);
                        System.out.println(no+"₩t"+name+"₩t"+tel);
                }//while

                System.out.println("--------------------------------");
                rs.absolute(3);
                System.out.println(rs.getInt(1)+": "+rs.getString(2));
        }catch (ClassNotFoundException e) {
                e.printStackTrace();
        }catch (SQLException e) {
                e.printStackTrace();
        }finally{
                try {
                        if(rs!=null) rs.close();
                        if(ps!=null) ps.close();
                        if(conn!=null) conn.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
    }//main
}
```



execute() 메서드 이용

table 생성하기

```
import java.sql.*;
public class CreateTableTest{
	public static void main(String[] args) {
		Connection conn=null;
		Statement stmt=null;
		try{
			//1. 드라이버 로딩
			Class.forName("oracle.jdbc.driver.OracleDriver");
			System.out.println("드라이버 로딩 성공!");

			//2. DB와 연결
			String url="jdbc:oracle:thin:@userpc:1521:orcl8";
			String user="javauser1", pwd="java";

			conn=DriverManager.getConnection(url,user,pwd);
			System.out.println("DB 연결 성공!");

			//3. 쿼리문을 전송하기 위한 Statement객체 얻어오기
			stmt=conn.createStatement();
```

```
            //테이블을 생성하는 sql문 작성
            String sql="create table pd2("
             +"no number primary key,"
             + "pdName varchar2(50) not null,"
             + "price number null,"
             + "regdate  date default sysdate)";

            //4. 쿼리문 실행
            boolean isRs=stmt.execute(sql);
            System.out.println("isRs="+isRs);

            //sequence 생성 sql문
            String sql2="create sequence pd2_seq";
            sql2+=" start with 1 increment by 1 nocache";

            isRs=stmt.execute(sql2);
            System.out.println("isRs="+isRs);
            System.out.println("pd2 테이블 생성 성공!!");
}catch(ClassNotFoundException e){
            System.out.println("드라이버 로딩 실패!");
            e.printStackTrace();
}catch(SQLException e){
            System.out.println("sql error!");
            e.printStackTrace();
}finally{
```

boolean execute(String sql) throws SQLException
-모든 sql문장을 실행시킴
- select문이면 true, select문이 아니면 false를 반환

```
                //5. 자원 반납
                try {
                        if(stmt!=null) stmt.close();
                        if(conn!=null) conn.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
    }

}
```

입력

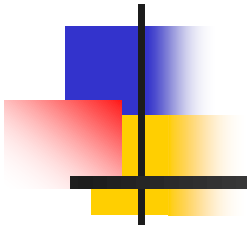
```
import java.sql.*;
import java.util.*;
public class InsertTest {
	public static void main(String[] args) {
		System.out.println("[상품 입력]");
		Scanner sc=new Scanner(System.in);
		System.out.print("상품명 : ");
		String pdName=sc.nextLine();
		System.out.print("가격 : ");
		int price=sc.nextInt();

		Connection conn=null;
		Statement stmt=null;
		try{
			Class.forName("oracle.jdbc.driver.OracleDriver");
			String url="jdbc:oracle:thin:@userpc:1521:orcl8";
			String user="javauser1", pwd="java";
			conn=DriverManager.getConnection(url,user,pwd);

			stmt=conn.createStatement();
			String sql="insert into pd(no, pdname, price)"
					+ " values(pd_seq.nextval,"
					+"'"+pdName+"', "+price+")";
			System.out.println(sql);

			boolean isRs=stmt.execute(sql);
```

```
                if(!isRs){ //select문이 아니면
                        int cnt=stmt.getUpdateCount();
                        //DML문에 의해 처리된 행의 갯수 반환

                        String res=(cnt>0)?"상품등록 성공":"상품등록 실패";
                        System.out.println(res);
                }
        }catch(ClassNotFoundException e){
                System.out.println("드라이버 로딩 실패!");
                e.printStackTrace();
        }catch(SQLException e){
                System.out.println("sql error!");
                e.printStackTrace();
        }finally{
                try {
                        if(stmt!=null) stmt.close();
                        if(conn!=null) conn.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
    }
}
```

조회

```
import java.sql.*;
public class SelectTest {
	public static void main(String[] args){
		String url="jdbc:oracle:thin:@userpc:1521:orcl8";
		String user="javauser1", pwd="java";

		Connection conn=null;
		Statement stmt=null;
		ResultSet rs=null;
		try {
			Class.forName("oracle.jdbc.driver.OracleDriver");
			conn = DriverManager.getConnection(url,user,pwd);
			stmt=conn.createStatement();

			String sql="select * from pd order by no desc";

			boolean isRs=stmt.execute(sql);
			if(isRs){//select문
				rs=stmt.getResultSet();
				while(rs.next()){
					int no=rs.getInt("no");
					String pdName=rs.getString("pdname");
					int price=rs.getInt("price");
					Date regdate=rs.getDate("regdate");
				System.out.println(no+"₩t"+pdName+"₩t"+price+"₩t"+regdate);
				}//while
			}//if
```

```
		} catch (ClassNotFoundException e) {
			e.printStackTrace();
		} catch (SQLException e) {
			e.printStackTrace();
		}finally{
			try {
				if(rs!=null) rs.close();
				if(stmt!=null) stmt.close();
				if(conn!=null)conn.close();
			} catch (SQLException e) {
				e.printStackTrace();
			}
		}

	}//main
}
```



저장 프로시저 이용

저장 프로시저

```
create or replace procedure personInsert
(p_name in varchar2,
p_tel in varchar2)
is
begin
      insert into person
      values(person_seq.nextval, p_name,p_tel);

      commit;
  exception when others then
       raise_application_error(-20001, 'person 테이블에 insert 실패!');
       rollback;
end;
```

저장 프로시저

```
create or replace procedure personList
(personCursor out SYS_REFCURSOR)
is
begin
    OPEN personCursor For
    select no,name,tel from person
    order by no desc;

 exception when others then
       raise_application_error(-20002, 'person 테이블 조회 실패!');
end;
```

저장 프로시저 이용

```
create or replace procedure personInsert
(p_name in varchar2,
p_tel in varchar2)
is
begin
  insert into person
  values(person_seq.nextval, p_name,p_tel);

  commit;
end;
```

```
class CallableStatementTest2{
    public static void main(String[] args){
        Scanner sc=new Scanner(System.in);
        System.out.print("이름 : ");
        String name=sc.nextLine();
        System.out.print("전화번호 : ");
        String tel=sc.nextLine();

        String url="jdbc:oracle:thin:@userpc:1521:orcl8";
        String user="javauser1", pwd="java";
        Connection conn=null;
        CallableStatement ctmt=null;
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            conn = DriverManager.getConnection(url,user, pwd);
            System.out.println("DB연결됨!");

            String sql="{call personInsert(?,?)}";
            ctmt=conn.prepareCall(sql);
            ctmt.setString(1, name); //name
            ctmt.setString(2, tel); //tel

            ctmt.execute();
            System.out.println("저장 프로시저 실행 완료!");
```

저장 프로시저 이용하는 경우 처리

```
            } catch (ClassNotFoundException e) {
                        e.printStackTrace();
            }catch (SQLException e) {
                        e.printStackTrace();
            }finally{
                        try {
                                    if(ctmt!=null) ctmt.close();
                                    if(conn!=null) conn.close();
                        } catch (SQLException e) {
                                    e.printStackTrace();
                        }
            }
      }//main
}
```

```
public class CallTest_Out {
    public static void main(String[] args){
        String url="jdbc:oracle:thin:@userpc:1521:orcl8";
        String user="hr", pwd="hr123";

        Connection conn=null;
        CallableStatement ctmt=null;
        ResultSet rs=null;
        try {
                Class.forName("oracle.jdbc.driver.OracleDriver");
                conn = DriverManager.getConnection(url,user, pwd);
                System.out.println("DB연결됨!");

                String sql="{call infoProf_proc(?,?,?)}";
                CallableStatement cs=conn.prepareCall(sql);
                cs.setInt(1, 1002);
                cs.registerOutParameter(2, oracle.jdbc.OracleTypes.VARCHAR);
                cs.registerOutParameter(3, oracle.jdbc.OracleTypes.NUMBER);

                cs.execute();

                String s1 =cs.getString(2);
                int n1 = cs.getInt(3);
                System.out.println("s1="+s1+", n1="+n1);
        } catch (ClassNotFoundException e) {
                e.printStackTrace();
```

```
create or replace procedure infoProf_proc
(v_profno   in professor.profno%type,
v_name  out professor.name%type,
v_pay out professor.pay%type)
is
begin
    select name, pay into v_name, v_pay
    from professor
    where profno=v_profno;
end;
```

```
s1=박승곤, n1=380
```

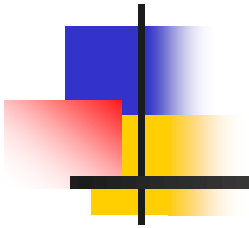
```
        }catch (SQLException e) {
                e.printStackTrace();
        }finally{
                try {
                        if(rs!=null) rs.close();
                        if(ctmt!=null) ctmt.close();
                        if(conn!=null) conn.close();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
    }
}
```

```
create or replace procedure personList
(personCursor out SYS_REFCURSOR)
is
begin
    OPEN personCursor For
    select no,name,tel from person
    order by no desc;
end;
```

```
public class CallableStatementTest3{
    public static void main(String[] args){
        String url="jdbc:oracle:thin:@userpc:1521:orcl8";
        String user="javauser1", pwd="java";

        Connection conn=null;
        CallableStatement ctmt=null;
        ResultSet rs=null;
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            conn = DriverManager.getConnection(url,user, pwd);
            System.out.println("DB연결됨!");

            String sql="{call personList(?)}";
            CallableStatement cs=conn.prepareCall(sql);
            cs.registerOutParameter(1, oracle.jdbc.OracleTypes.CURSOR);    //커서타입으로 지정
            cs.execute();

            rs=(ResultSet)cs.getObject(1);
            while(rs.next()){
                int no=rs.getInt("no");
                String name=rs.getString("name");
                String tel=rs.getString("tel");
                System.out.println(no+"₩t"+name+"₩t"+tel);
            }
        }
```

```
4    손연재    010-300-4000
3    김연아    010-200-3000
2    홍길동    010-100-2000
```

```
		catch (ClassNotFoundException e) {
			e.printStackTrace();
		}catch (SQLException e) {
			e.printStackTrace();
		}finally{
			try {
				if(rs!=null) rs.close();
				if(ctmt!=null) ctmt.close();
				if(conn!=null) conn.close();
			} catch (SQLException e) {
				e.printStackTrace();
			}
		}
	}
}
```



Person 테이블 예제

예제

```
---------------Main Menu-----------
1.등록  2.검색  3.삭제  4.출력  5.종료
-------------------------------------
번호를 입력하세요
1
이름을 입력하세요
홍길동
전화번호를 입력하세요
010-700-9999
입력 성공!


---------------Main Menu-----------
1.등록  2.검색  3.삭제  4.출력  5.종료
-------------------------------------
번호를 입력하세요
2
검색할 사람 이름 :
홍길동

------------사원 목록-------------
번호        이름        전화번호
-----------------------------------
1           홍길동      010-100-2000
83          홍길동      010-700-9999
7           홍길동      010-777-7777
21          홍길동      011-999-7878
```

```
---------------Main Menu-----------
1.등록  2.검색  3.삭제  4.출력  5.종료
-------------------------------------
번호를 입력하세요
4

------------사원 목록-------------
번호        이름        전화번호
-----------------------------------
1           홍길동      010-100-2000
2           김연아      010-200-5000
3           윤아        010-300-7000
5           이승기      010-111-2222
6           이승기      010-555-7777
7           홍길동      010-777-7777
21          홍길동      011-999-7878
22          김연아      010-300-5588
42          이승기      010-100-3000
43          이승기2     010-100-4000
45          이승기      010-555-7799
63          홍길순      010-6000
82          김재신      010-100-3000
83          홍길동      010-700-9999

---------------Main Menu-----------
1.등록  2.검색  3.삭제  4.출력  5.종료
-------------------------------------
번호를 입력하세요
3
삭제할 번호 입력!
43
삭제 성공!
```

DBManager

```
/** 드라이버 로딩과 DB 연결을 하는 클래스 */
public class DBManager {
	static{
		try{
			Class.forName("oracle.jdbc.driver.OracleDriver");
			System.out.println("드라이버 로딩 성공!");
		}catch(ClassNotFoundException e){
			System.out.println("드라이버 로딩 실패!");
			e.printStackTrace();
		}
	}//static 초기화 블럭

	public static Connection getConnection() throws SQLException{
		String url="jdbc:oracle:thin:@userpc:1521:orcl8";
		String user="javauser1";
		String pwd="java";

		Connection conn=DriverManager.getConnection(url,user,pwd);
		System.out.println("DB연결, conn="+conn);
		return conn;
	}
```

```
    public static void dbClose(ResultSet rs, PreparedStatement ps,Connection conn) throws SQLException
    {
            if (rs != null)rs.close();
            if (ps != null)ps.close();
            if (conn != null)conn.close();
            System.out.println("DB close!");
    }
    public static void dbClose(ResultSet rs, Statement stmt, Connection conn) throws SQLException {
            if (rs != null)rs.close();
            if (stmt != null)stmt.close();
            if (conn != null)conn.close();
            System.out.println("DB close!");
    }

    public static void dbClose(Statement stmt, Connection conn) throws SQLException {
            if (stmt != null)stmt.close();
            if (conn != null)conn.close();
            System.out.println("DB close!");
    }

    public static void dbClose(PreparedStatement ps,Connection conn) throws SQLException {
            if (ps != null)             ps.close();
            if (conn != null)conn.close();
            System.out.println("DB close!");
    }
}
```

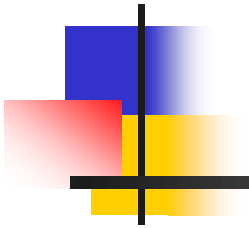
PersonConsole

```
public class PersonConsole {
	static Scanner sc=new Scanner(System.in);
	public static void main(String[] args) {
		while(true){
			mainMenu();
			int type=0;
			type=sc.nextInt(); //메뉴번호 입력

			switch(type){
			case 1: //등록
				try {
					register();
				} catch (SQLException e) {
					e.printStackTrace();
				}
				break;
			case 4: //출력
```

```
                                try {
                                        printAll();
                                } catch (SQLException e) {
                                        e.printStackTrace();
                                }
                                break;
                        case 5: //종료
                                System.out.println("프로그램을 종료합니다!!");
                                System.exit(0);
                        default:
                                System.out.println("잘못 선택!");
                        }//switch
                }//while
        }
        public static void mainMenu(){
                System.out.println("₩n---------------Main Menu-----------");
                System.out.println("1.등록  2.검색  3.삭제  4.출력  5.종료");
                System.out.println("-------------------------------------");
                System.out.println("번호를 입력하세요");
        }
        public static void register() throws SQLException{
                System.out.println("이름을 입력하세요");
                sc.nextLine();
                String name=sc.nextLine();
                System.out.println("전화번호를 입력하세요");
                String tel=sc.nextLine();
                int cnt = insertPerson(name, tel); //저장
                String str = cnt>0?"입력 성공!₩n":"입력 실패!₩n";
```

```
        System.out.println(str);
}//register
public static int insertPerson(String name,String tel) throws SQLException{
        Connection conn=null;
        PreparedStatement ps=null;
        int n=0;
        try{
                conn= DBManager.getConnection();
                String sql="insert into person values(person_seq.nextval,?,?)";
                ps=conn.prepareStatement(sql);
                ps.setString(1, name);
                ps.setString(2, tel);

                n=ps.executeUpdate();
        }finally{
                DBManager.dbClose(ps, conn);
        }
        return n;
}//insertPerson

public static void printTitle()
{
        System.out.println("₩n------------사원 목록-------------");
        System.out.println("번호₩t이름₩t전화번호");
        System.out.println("-----------------------------------");
}
```

```
    public static void printAll() throws SQLException{
        Connection conn=null;
        PreparedStatement ps=null;
        ResultSet rs=null;
        try{
                conn=DBManager.getConnection();

                String sql="select * from Person order by no";
                ps=conn.prepareStatement(sql);
                rs=ps.executeQuery();
                printTitle();
                while(rs.next()){
                        int no=rs.getInt("no");
                        String name=rs.getString("name");
                        String tel=rs.getString("tel");
                        System.out.println(no+"₩t"+name+"₩t" + tel+"₩t");
                }//while
        }finally{
                DBManager.dbClose(rs, ps, conn);
        }
    }

}
```



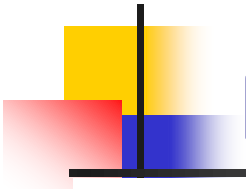
DAO/DTO 이용

DAO(Data Access Object)

- DAO(Data Access Object)
  - 데이터베이스 작업을 전담하는 객체
  - 데이터베이스와 연계하여 처리할 프로그램을 정규화해 둔 클래스
  - 데이터베이스에 입력, 수정, 삭제, 검색 등의 작업을 미리 캡슐화해 두어 사용하기 편하도록 만든 형태
  - CRUD
    - C : create, insert
    - R : read, select
    - U : update
    - D : delete

• 빈즈 규약(캡슐화된 객체)
- 멤버변수는 private으로
- 멤버변수에 대한 접근은 getter/setter로

- DTO(Data Transfer Object), VO(Value Object), Bean
  - 객체를 표현한 한 단위
  - 데이터를 전달하는 단위
  - 데이터를 하나의 객체로 관리할 목적으로 만들어 둔 클래스의 객체
  - 거의 데이터베이스의 table과 동일한 필드를 갖는다 .



PersonDTO

```
public class PersonDTO{
	private int no;
	private String name;
	private String tel;

	public PersonDTO() {
		super();
	}
	public PersonDTO(int no, String name, String tel) {
		super();
		this.no = no;
		this.name = name;
		this.tel = tel;
	}
	public int getNo() {
		return no;
	}
	public void setNo(int no) {
		this.no = no;
	}
```

PersonDTO

```
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getTel() {
		return tel;
	}
	public void setTel(String tel) {
		this.tel = tel;
	}

	@Override
	public String toString() {
		return "PersonDTO [no=" + no + ", name=" + name + ", tel=" + tel + "]";
	}
}
```

PersonDAO

```
public class PersonDAO {
	public int insertPerson(PersonDTO dto) throws SQLException{
		Connection conn=null;
		PreparedStatement ps=null;
		int rowCnt=0;
		try{
			conn = DBManager.getConnection();
			String sql="insert into person values(person_seq.nextval,?,?)";
			ps=conn.prepareStatement(sql);
			ps.setString(1, dto.getName());
			ps.setString(2, dto.getTel());
			rowCnt=ps.executeUpdate();
			System.out.println("입력 처리, rowCnt="+rowCnt);
		}finally{
			DBManager.dbClose(ps, conn);
		}
		return rowCnt;
	}//insertPerson
	public int deletePerson(int no) throws SQLException{
		Connection conn=null;
		PreparedStatement ps=null;
```

```
        int rowCnt=0;
        try{
                conn = DBManager.getConnection();
                String sql="delete from Person where no=?";
                ps=conn.prepareStatement(sql);
                ps.setInt(1, no);
                rowCnt= ps.executeUpdate();
                System.out.println("삭제 처리, rowCnt="+rowCnt);
        }finally{
                DBManager.dbClose(ps, conn);
        }
        return rowCnt;
}
public int updatePerson(String name, String tel, int no) throws SQLException{
        Connection conn=null;
        PreparedStatement ps=null;
        int rowCnt=0;
        try{
                conn = DBManager.getConnection();
                String sql="update Person set name=?, tel=? WHERE no=?";
                ps=conn.prepareStatement(sql);
                ps.setString(1,name);
                ps.setString(2,tel);
                ps.setInt(3, no);
                rowCnt= ps.executeUpdate();
                System.out.println("수정 처리, rowCnt="+rowCnt);
        }finally{
                DBManager.dbClose(ps, conn);
        }
```

```
        return rowCnt;
}
public List<PersonDTO> selectAll() throws SQLException{
        Connection conn=null;
        PreparedStatement ps=null;
        ResultSet rs=null;
        List<PersonDTO> list=new ArrayList<PersonDTO>();
        try{
                conn = DBManager.getConnection();
                String sql="select * from Person order by no";
                ps=conn.prepareStatement(sql);
                rs=ps.executeQuery();
                while(rs.next()){
                        int no=rs.getInt("no");
                        String name=rs.getString("name");
                        String tel=rs.getString("tel");
                        PersonDTO dto=new PersonDTO(no,name,tel);
                        list.add(dto);
                }//while
                System.out.println("전체 조회, list.size()="+list.size());
        }finally{
                DBManager.dbClose(rs, ps, conn);
        }
        return list;
}//selectAll
public List<PersonDTO> selectByName(String name) throws SQLException{
        Connection conn=null;
        PreparedStatement ps=null;
```

```
        ResultSet rs=null;
        List<PersonDTO> list = new ArrayList<PersonDTO>();
        try{
                conn = DBManager.getConnection();
                String sql="select * from Person where name=?";
                ps=conn.prepareStatement(sql);
                ps.setString(1, name);
                rs=ps.executeQuery();
                while(rs.next()){
                        int no=rs.getInt("no");
                        String tel=rs.getString("tel");

                        PersonDTO dto=new PersonDTO(no,name,tel);
                        list.add(dto);
                }//while
                System.out.println("이름으로  조회, list.size()="+list.size());
        }finally{
                DBManager.dbClose(rs, ps, conn);
        }
        return list;
    }//selectByName

}
```

PersonManager

```
public class PersonManager {
	public static final int FIND=1;
	public static final int ALL=2;
	private PersonDAO personDao;
	private Scanner sc=new Scanner(System.in);

	public PersonManager(){
		personDao=new PersonDAO();
	}
	/**메인 메뉴를 보여주는 메서드*/
	public void mainMenu(){
		System.out.println("₩n----------------Main Menu-----------");
		System.out.println("1.등록  2.검색  3.삭제  4.출력  5.종료");
		System.out.println("-----------------------------------");
		System.out.println("번호를 입력하세요");
	}
	public void register(){
		System.out.println("이름을 입력하세요");
		String name=sc.nextLine();
		System.out.println("전화번호를 입력하세요");
		String tel=sc.nextLine();
		PersonDTO dto =new PersonDTO();
		dto.setName(name);
```

```
		dto.setTel(tel);
		try {
				int cnt = personDao.insertPerson(dto);
				String msg = cnt>0?"입력 성공!":"입력 실패";
				System.out.println(msg);
		} catch (SQLException e) {
				e.printStackTrace();
		}
}//register
public void removePerson(){
		System.out.println("삭제할 번호를 입력하세요");
		String sNo=sc.nextLine();
		int no=0;
		if(sNo!=null && !sNo.isEmpty()){
				no=Integer.parseInt(sNo);
		}else{
				System.out.println("번호를 입력하셔야 합니다!!");
				return;
		}
		try {
				int cnt = personDao.deletePerson(no);
				String msg = cnt>0?"삭제 성공!":"삭제 실패";
				System.out.println(msg);
		} catch (SQLException e) {
				e.printStackTrace();
		}
}
public void printTitle(){
```

```
        System.out.println("₩n------------사원 목록------------");
        System.out.println("번호₩t이름₩t전화번호");
        System.out.println("-------------------------------");
}

public void printAll(int n){
        List<PersonDTO> alist=null;
        if(n==ALL){//모두 보기
                try {
                        alist=personDao.selectAll();
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }else if(n==FIND){//이름으로 검색
                System.out.println("검색할 사람 이름은?");
                String name=sc.nextLine();
                try {
                        alist=personDao.selectByName(name);
                } catch (SQLException e) {
                        e.printStackTrace();
                }
        }
        if(alist==null||alist.isEmpty()){
                System.out.println("등록된 사원이 없습니다");
                return;
        }
```

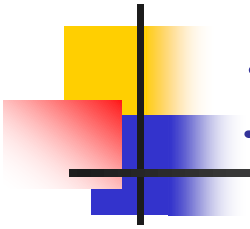
```
        printTitle();
        for(int i=0;i<alist.size();i++){
                PersonDTO dto = alist.get(i);
                int no = dto.getNo();
                String name =dto.getName();
                String tel =dto.getTel();
                System.out.println(no+"₩t"+name+"₩t" + tel+"₩t");
        }//for
    }
}
```

PersonConsole

```
public class PersonConsole {
    public static void main(String[] args) {
        PersonManager manager=new PersonManager();

        Scanner sc=new Scanner(System.in);
        while(true){
            manager.mainMenu();
            int no=sc.nextInt(); //메뉴번호 입력

            switch(no){
            case 1: //등록
                manager.register();
                break;
            case 2: //검색
                manager.printAll(PersonManager.FIND);
                break;
```

PersonConsole

```
                    case 3: //삭제
                            manager.removePerson();
                            break;
                    case 4: //출력
                            manager.printAll(PersonManager.ALL);
                            break;
                    case 5: //종료
                            System.out.println("프로그램을 종료합니다.");
                            System.exit(0);
                    default:
                            System.out.println("잘못 선택함!");
                            continue;
                    }//switch
            }//while
    }

}
```



과제

- 1. PersonDAO 클래스에
  - updatePerson() : 수정
  - selectByNo() :번호로 조회하는 메서드 추가
- 2. PersonConsole2에서 각 메서드 호출
- 3. book 테이블을 이용하여
  - Insert, update, delete, select(전체 데이터 조회)
  - Select by no(번호로 조회) 처리하기