

13장. Java – 오라클 DB 연동



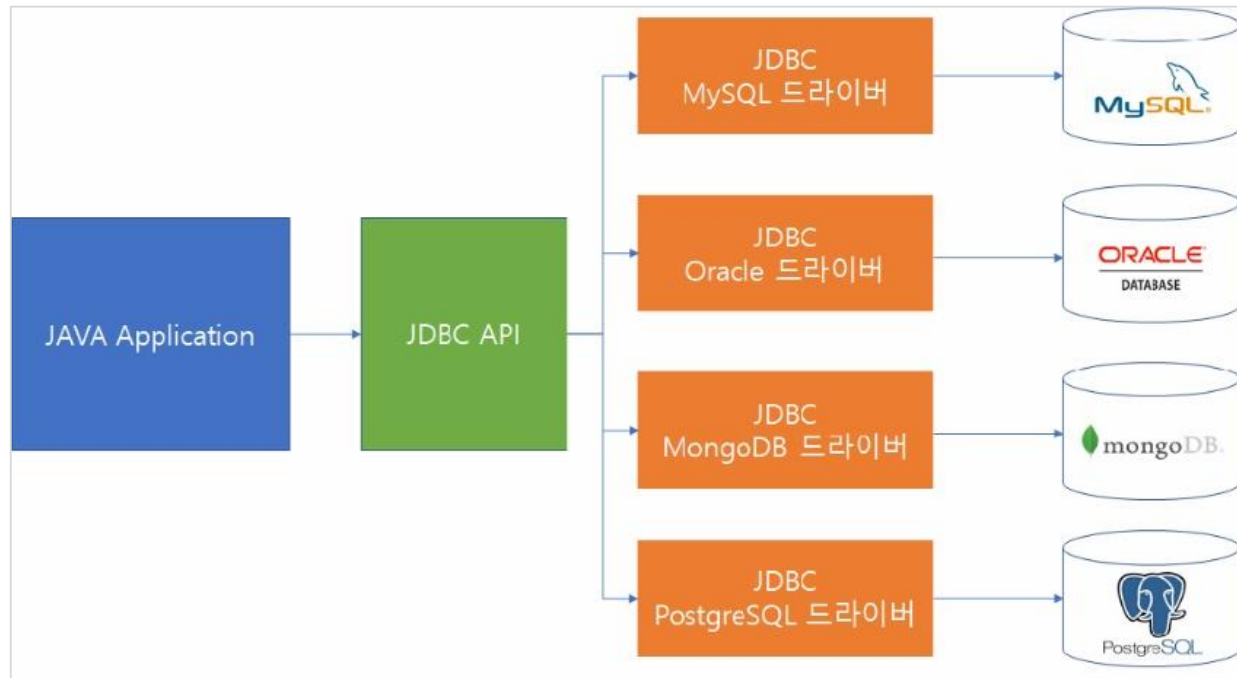
JDBC Connection



JDBC(Java Database Connectivity)

◆ JDBC(Java Database Connectivity) 정의와 사용

- 자바 애플리케이션에서 DBMS에 연결해주는 기능을 갖는 API이다.
- 오라클, MySQL, MS-SQL 개발사가 구현한 '드라이버'가 존재함



오라클 데이터 베이스

❖ 오라클 데이터베이스와 버전

▷ Oracle 데이터베이스

- 오라클사가 만든 DBMS 제품이다.
- 최신 버전은 2021년에 출시한 21c 버전이다.

현재 일반적으로 Express-Edition 버전을 많이 사용한다.

▷ Oracle 데이터베이스 설치

1. 계정 생성하고 로그인 하기
2. 다운로드 하기
3. 파일 압축 풀기
4. 설치 프로그램 실행하기



오라클 데이터 베이스

❖ oracle download > oracle Database XE download

데이터베이스

[Audit Vault and Database Firewall](#)

[Berkeley DB](#)

[Big Data Connectors](#)

[클러스터 검증 유틸리티](#)

[Database Enterprise/Standard Edition](#)

[Database Express Edition](#)

[데이터베이스 모바일 서버](#)

[Grid Infrastructure](#)

[Instant Client](#)

[MySQL](#)

Run Oracle Database 21c XE on Windows

[Download Oracle Database XE for Windows \(ZIP\)](#)

[Oracle Database XE Installation Guide for Windows](#)

[Oracle Database 21c XE Quick Start](#)

오라클 계정 로그인

사용자 이름

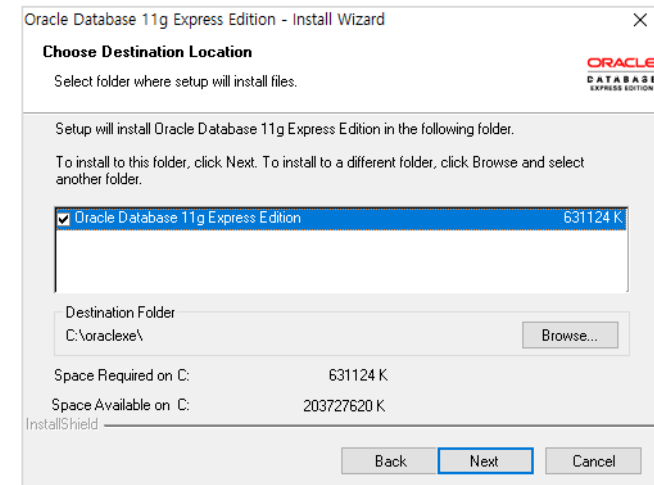
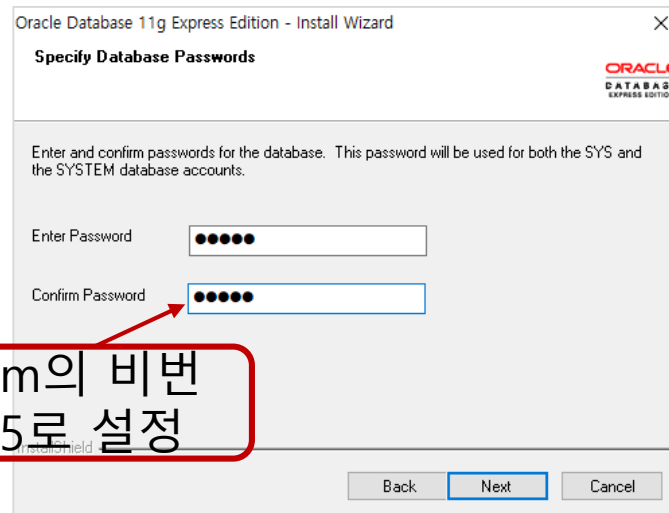
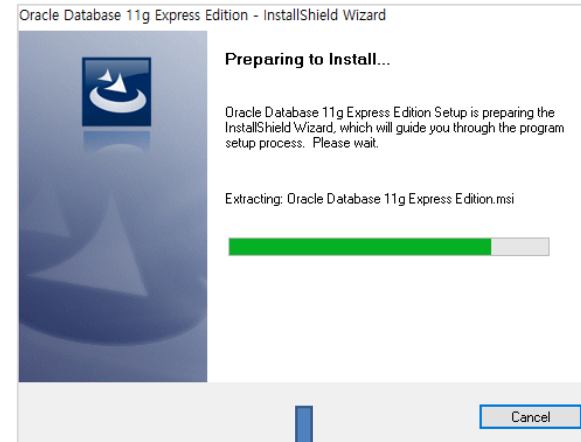
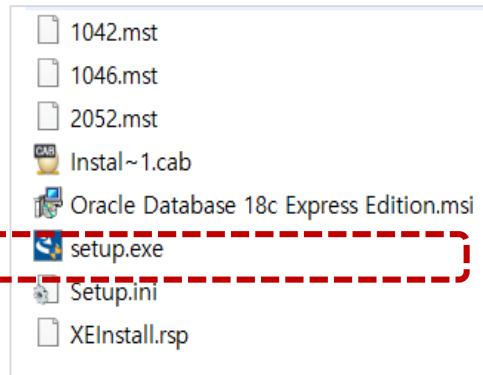


암호



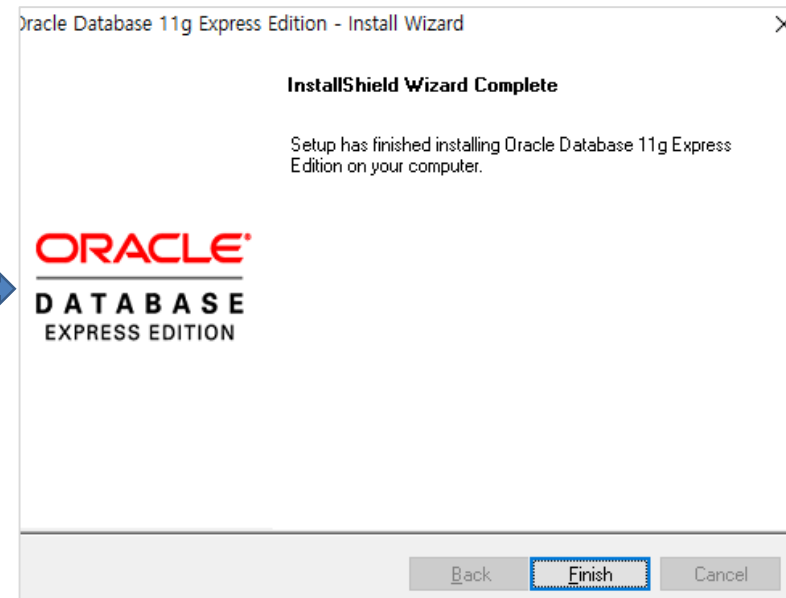
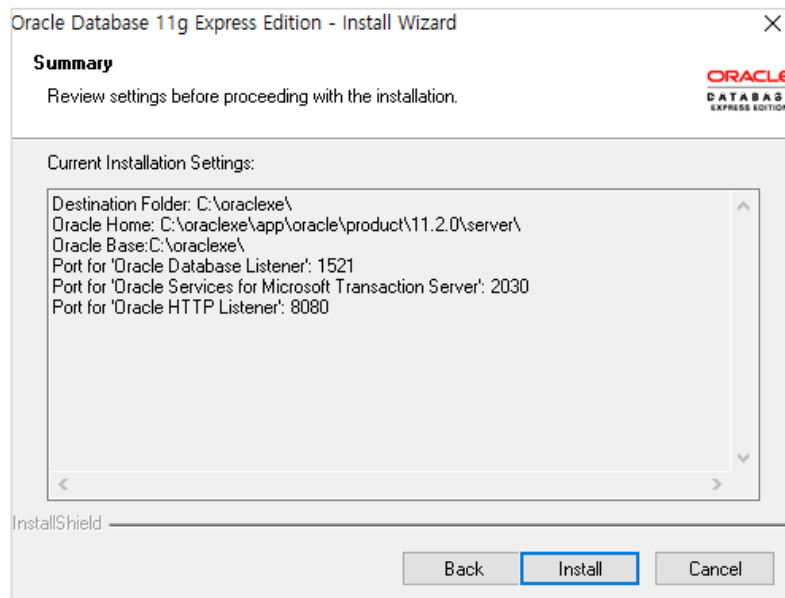
오라클 데이터 베이스

❖ 오라클 데이터베이스와 설치



오라클 데이터 베이스





❖ 오라클 데이터베이스와 설치



오라클 데이터 베이스

❖ 오라클 데이터베이스와 설치 후 확인

내컴퓨터 – 우측마우스 – 관리 – 서비스 및 응용프로그램 - 서비스

 OracleJobSchedulerXE		사용 안 함
 OracleOraDB21Home1MTSR...	실행 ...	자동
 OracleOraDB21Home1TNSLi...	실행 ...	자동
 OracleServiceXE	실행 ...	자동
 OracleVssWriterXE		자동



sqlplus – 개발 도구

■ sqlplus 사용하기

오라클 설치시 자동으로 지원되는 프로그램이다.

- ① 명령프롬프트(cmd) 열기
- ② **sqlplus**(DBMS 소프트웨어) 입력
- ③ 사용자명(user) : **system**
- ④ 비밀번호 : **12345**

```
C:\Users\kkyon>sqlplus

SQL*Plus: Release 21.0.0.0.0 - Production on 토 7월 2 08:28:19 2022
Version 21.3.0.0.0

Copyright (c) 1982, 2021, Oracle. All rights reserved.

사용자명 입력: system
비밀번호 입력:
마지막 성공한 로그인 시간: 토 7월 02 2022 08:21:30 +09:00

다음에 접속됨:
Oracle Database 21c Express Edition Release 21.0.0.0.0 - Production
Version 21.3.0.0.0

SQL>
```



SQL developer

◆ 오라클 SQL 디벨로퍼 : (sqldeveloper 다운로드 검색)

Oracle SQL developer는 오라클 데이터베이스에서 SQL 작업을 수행하는 통합 개발환경(IDE)이다.

SQL과 PL/SQL 코드 작성을 위해 다양한 기능을 제공하며 프리웨어이다.

Developer Resources

Tools

[APEX](#)

[Oracle REST Data Services \(ORDS\)](#)

[SQL Developer](#)

[SQL Developer Command Line \(SQLcl\)](#)

Drivers

[JDBC and UCP](#)

[ADO.NET and OLE DB Providers](#)




[Python Driver](#)

[Instant Client - C/C++ Drivers \(OCI Utilities\)](#)



오라클 SQL 디벨로퍼

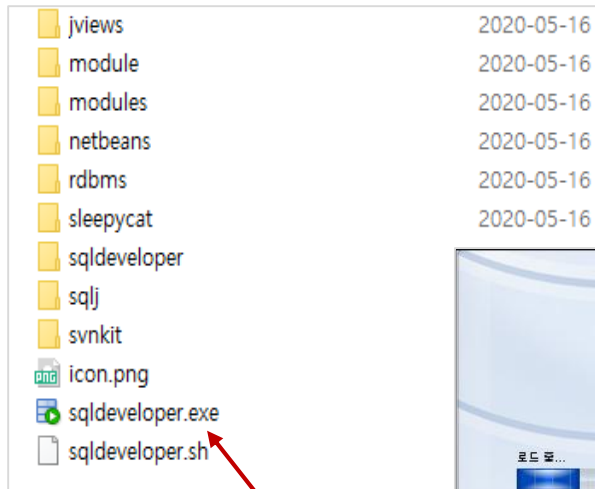
◆ 오라클 SQL 디벨로퍼 Download

Platform	Download
SQL Developer Extension for VSCode	 Download
Windows 64-bit with JDK 17 included	 Download (555 MB)
Windows 32-bit/64-bit	 Download (557 MB)

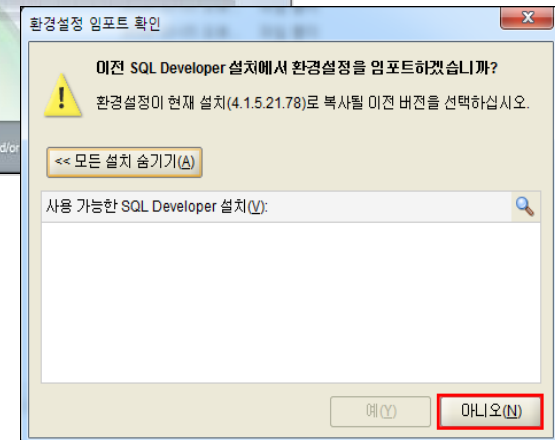


오라클 SQL 디벨로퍼

◆ Sqldeveloper 설치 – 다운로드후 압축풀기



실행

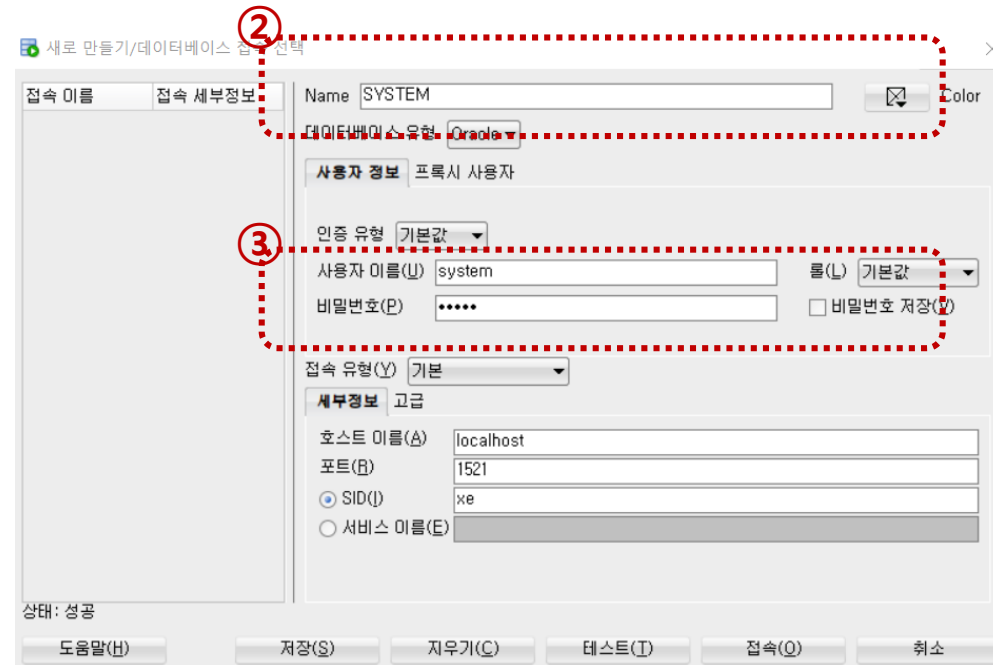
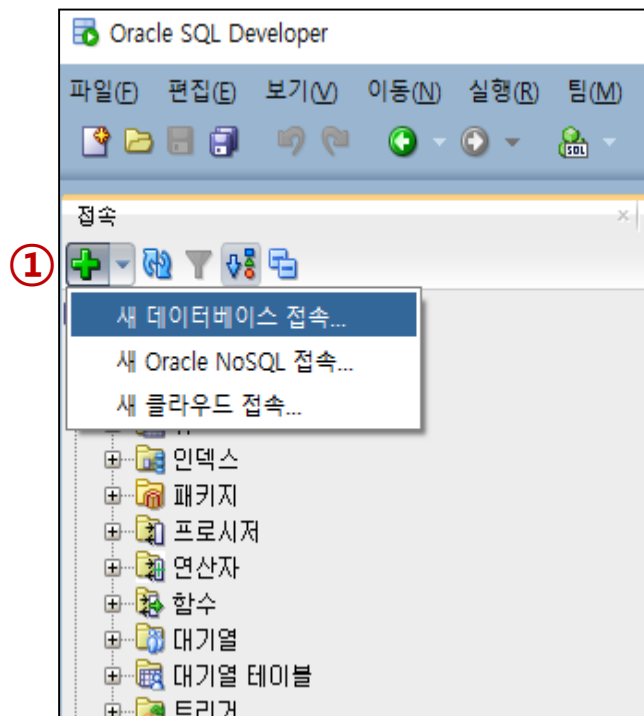


데이터베이스 생성

◆ 데이터베이스 만들기 및 접속

1. 관리자 (system) 계정만들기

Name-SYSTEM, 사용자이름- system, 비밀번호 - 12345

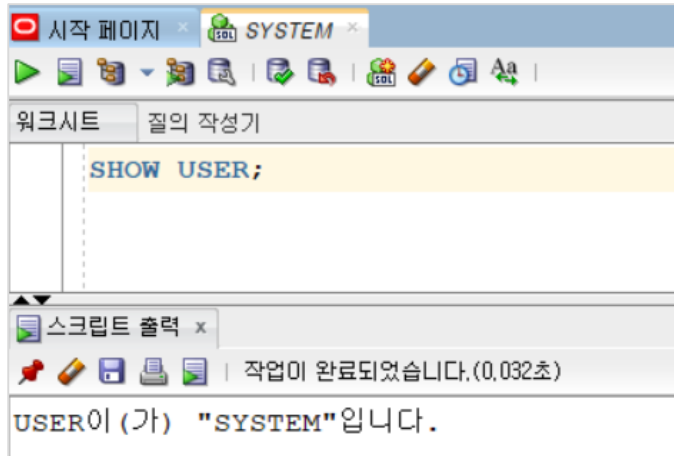


테스트후 성공
하면 접속



User 보기

USER 보기



system 파일 저장하기 – user.sql



JDBC(Java Database Connectivity)

◆ JDBC를 이용한 데이터베이스 연동하기

ojdbc 드라이버 구하기

1. 오라클 설치 경로 2. sql developer 설치 경로

로컬 디스크 (C:) > app > kiyon > product > 21c > dbhomeXE > jdbc > lib				
이름	수정된 날짜	유형	크기	
ojdbc8.jar	2021-07-21 오전 2:06	ALZip JAR File	4,936KB	
ojdbc8_g.jar	2021-07-21 오전 2:39	ALZip JAR File	8,213KB	
ojdbc8dms.jar	2021-07-21 오전 2:18	ALZip JAR File	6,852KB	
ojdbc8dms_g.jar	2021-07-21 오전 2:43	ALZip JAR File	8,215KB	
ojdbc11.jar	2021-07-21 오전 2:07	ALZip JAR File	5,027KB	
ojdbc11_g.jar	2021-07-21 오전 2:47	ALZip JAR File	8,357KB	

로컬 디스크 (C:) > sqldeveloper > jdbc > lib	
이름	
ojdbc8.jar	



JDBC(Java Database Connectivity)

◆ JDBC를 이용한 데이터베이스 연동하기 ojdbc 이클립스 프로젝트에 복사하기

The first screenshot shows the project structure of 'Chapter16 [javaWorks master]'. It includes a 'src' folder with subfolders 'dbconnection' and 'person', and a 'lib' folder containing 'ojdbc11.jar'. An arrow points from this 'lib' folder to the first step in the list below.

The second screenshot shows a right-click context menu on the 'lib' folder, with 'Build Path' > 'Configure Build Path...' selected.

The third screenshot shows the 'Java Build Path' dialog box. In the 'Libraries' tab, 'JRE System Library [JavaSE-10]' is selected. A red dashed box highlights the 'Add External JARs...' button, which is labeled with a circled '2'.

The 'JAR Selection' dialog box is also visible, showing a list of JAR files. 'ojdbc6.jar' is highlighted in the list.

1. 프로젝트 만든후, lib폴더 만들기
2. 오라클 드라이버 .jar파일 복사
3. 클래스 패스 설정

프로젝트 > 우측마우스 > Build Path > Cofigure Build Path > Libraries(Classpath) > Add JARs



연결 테스트(Connection Test)

◆ 오라클 DBMS에 연결하기

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class ConnectionTest {
    public static void main(String[] args) {
        Connection conn = null;

        try {
            // jdbc 드라이버 등록
            Class.forName("oracle.jdbc.OracleDriver");
            // 연결
            conn = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521/xe", //db url
                "system", //user 계정
                "pw1234"); //password
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```



연결 테스트(Connection Test)

◆ 오라클 DBMS에 연결하기

```
        System.out.println(conn + "DB 연결 성공!!");
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        if(conn != null) {
            try {
                conn.close();
                System.out.println("연결 끊음");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

```
oracle.jdbc.driver.T4CConnection@16aa0a0aDB 연결 성공!!
연결 끊음
```



SQL – DDL, DML

◆ 테이블 생성 및 데이터 삽입, 검색

```
-- users (회원) 테이블 생성
CREATE TABLE users(
    userid    VARCHAR2(10) PRIMARY KEY,      -- 아이디
    userpassword VARCHAR2(100) NOT NULL,    -- 비밀번호
    username  VARCHAR2(20) NOT NULL,        -- 이름
    usage     NUMBER(3)                     -- 나이
);

-- 회원 추가
INSERT INTO users(userid, userpassword, username, usage)
VALUES ('today', 't1234$#', '이정후', 26);

INSERT INTO users(userid, userpassword, username)
VALUES ('hangang', 'h1234$#', '한강');

-- 전체 회원 검색
SELECT * FROM users;
```

	USERID	USERPASSWORD	USERNAME	USAGE
1	today	t1234\$#	이정후	26
2	hangang	h1234\$#	한강	(null)



오라클 DB – 데이터 타입

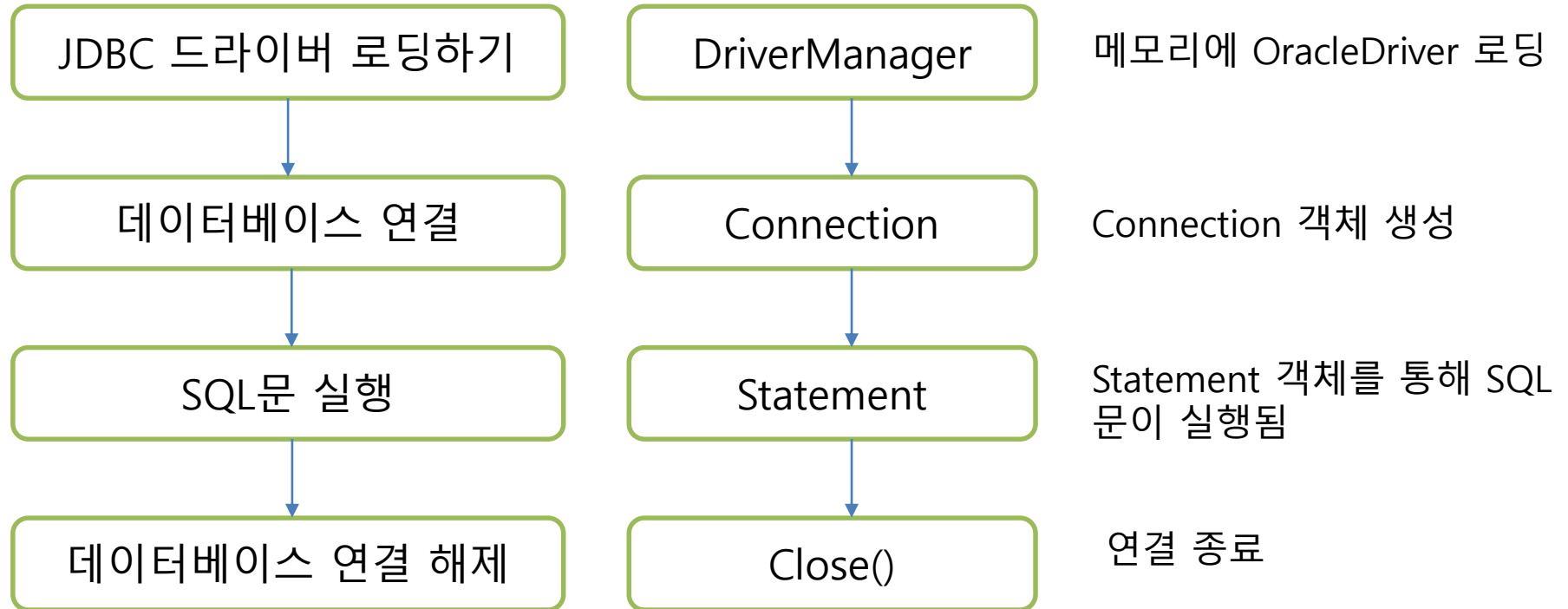
◆ 오라클 데이터 타입(Data Type)

데이터 타입	설명
CHAR	고정길이 문자, 최대 2000byte
VARCHAR2	가변길이 문자, 최대 4000byte
CLOB(Character)	매우 큰 크기의 문자열 저장, 최대 4GB
BLOB(Binary)	매우 큰 크기의 바이너리 파일 저장, 최대 4GB
NUMBER	가변 숫자, 십진수 기준 최대 220byte
DATE	날짜 – 연, 월, 일
TIMESTAMP	날짜 – 연, 월, 일, 시, 분, 초, 밀리초



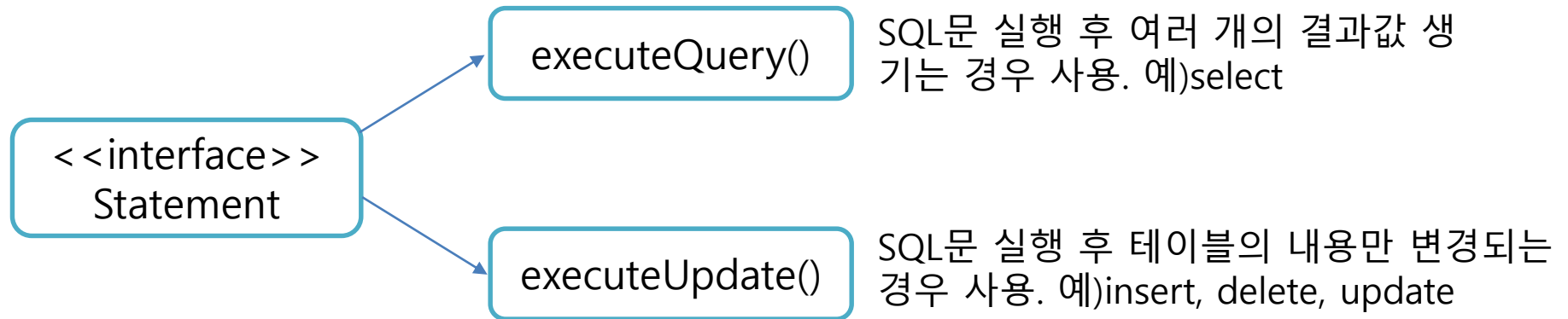
JDBC(Java Database Connectivity)

➤ 데이터베이스 연결 순서

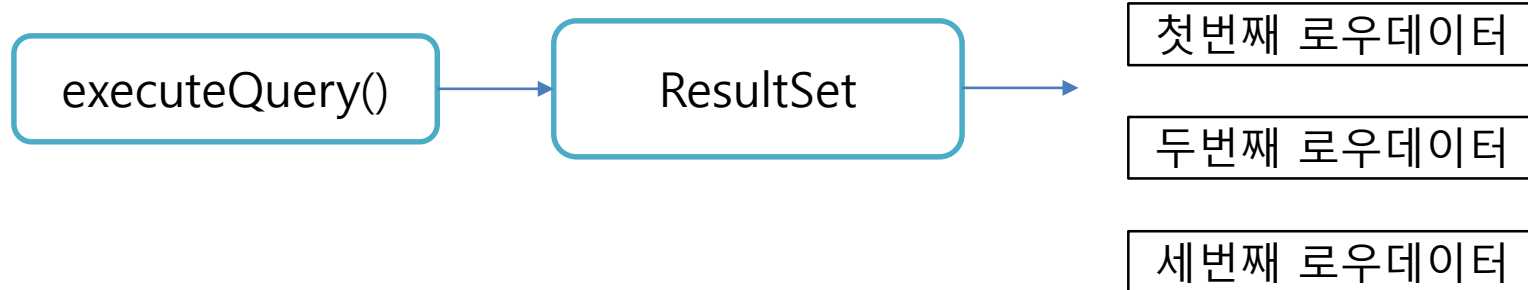


JDBC(Java Database Connectivity)

➤ Statement 객체 살펴보기



executeQuery() 실행 후 반환 되는 레코드셋



JDBC(Java Database Connectivity)

Module java.sql

Package java.sql

Interface Connection

All Superinterfaces:

AutoCloseable, Wrapper

```
public interface Connection
extends Wrapper, AutoCloseable
```

A connection (session) with a specific database. SQL sta

A **Connection** object's database is able to provide inform obtained with the **getMetaData** method.

Note: When configuring a **Connection**, JDBC application commands directly to change the connection's configura changes after executing each statement. If auto-commit

Module java.sql

Package java.sql

Class DriverManager

```
java.lang.Object
java.sql.DriverManager
```

```
public class DriverManager
extends Object
```

The basic service for managing a set of JDBC drivers.

NOTE: The **DataSource** interface, provides another w

As part of its initialization, the **DriverManager** class w

- The **jdbc.drivers** system property which contain
 - **jdbc.drivers=foo.bah.Driver:wombat.s**
- Service providers of the **java.sql.Driver** class,

Implementation Note:

DriverManager initialization is done lazily and looks u loader of the thread that triggers driver initialization l



Java – Oracle 연동

■ 회원 추가(삽입)

```
public class UsersInsertTest {  
    public static void main(String[] args) {  
        Connection conn = null;           //연결 객체 선언  
        PreparedStatement pstmt = null;    //sql 작업 객체 선언  
  
        try {  
            // db 연결  
            Class.forName("oracle.jdbc.OracleDriver");  
            conn = DriverManager.getConnection(  
                "jdbc:oracle:thin:@localhost:1521/xe",  
                "javauser",  
                "pwjava");  
  
            // sql 작성 - 회원 추가('?' - 동적 바인딩 기호)  
            String sql = "INSERT INTO users(userid, userpassword, username, userage) "  
                + "VALUES (?, ?, ?, ?)";  
            pstmt = conn.prepareStatement(sql); //sql을 매개변수로 전달  
            pstmt.setString(1, "cloud"); //setString(인덱스, 문자값)  
            pstmt.setString(2, "c3355@!");  
            pstmt.setString(3, "임시현");  
            pstmt.setInt(4, 22);           //setInt(인덱스, 숫자값)
```



Java – Oracle 연동

■ 회원 추가(삽입)

```
        int rows = pstmt.executeUpdate(); // sql 실행(삽입, 수정, 삭제시)
        System.out.println("저장된 행의 수: " + rows);
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        if(conn != null) {
            try {
                conn.close();
                System.out.println("연결 끊음");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}
```

USERID	USERPASSWORD	USERNAME	USERAGE
1 today	t1234\$#	이정후	26
2 hangang	h1234\$#	한강	(null)
3 cloud	c3355@!	임시현	22



DTO 정의와 사용법

- DTO(Data Transfer Object)의 정의와 사용법

- 여러 다른 타입의 데이터를 다른 클래스로 전달할 때 사용
- 만드는 방법

- DB 테이블의 필드명을 속성으로 선언한다.
- 생성자를 구현한다.
- 각 속성에 대한 getter/setter 메서드를 구현한다.



DAO와 DTO 정의와 사용법

▪ DTO(Data Transfer Object)의 정의

```
package dto;
//DTO 정의
public class Users {
    private String userId;        //아이디
    private String userPassword;  //비밀번호
    private String userName;      //이름
    private int userAge;          //나이

    //getter, setter 메서드 정의
    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getUserId() {
        return userId;
    }

    public void setUserId(String userId) {
        this.userId = userId;
    }
}
```



DAO와 DTO 정의와 사용법

▪ DTO(Data Transfer Object)의 정의

```
public String getUserPassword() {  
    return userPassword;  
}  
  
public void setUserPassword(String userPassword) {  
    this.userPassword = userPassword;  
}  
  
public int getUserAge() {  
    return userAge;  
}  
  
public void setUserAge(int userAge) {  
    this.userAge = userAge;  
}  
  
//객체 정보를 문자열로 리턴  
@Override  
public String toString() {  
    return "Users [userId=" + userId + ", userPassword=" + userPassword  
        + ", userName=" + userName + ", userAge=" + userAge + "];"  
}  
}
```



DB 연동 - 회원 관리

■ 회원 전체 검색(Select)

```
public class UsersSelectAllTest {
    public static void main(String[] args) {
        Connection conn = null;
        PreparedStatement pstmt = null;

        try {
            // jdbc 드라이버 등록
            Class.forName("oracle.jdbc.OracleDriver");
            // 연결
            conn = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521/xe",
                "system",
                "pw1234");

            // sql - 전체 회원 검색
            String sql = "SELECT * FROM users";
            pstmt = conn.prepareStatement(sql);

            ResultSet rs = pstmt.executeQuery(); //검색된 데이터 셋 객체 생성
            List<Users> userList = new ArrayList<>(); //ArrayList 자료구조 생성
        }
    }
}
```



DB 연동 - 회원 관리

■ 회원 전체 검색(Select)

```
while(rs.next()) {
    Users user = new Users(); //회원 객체 생성

    user.setUserId(rs.getString("userid"));
    user.setUserPassword(rs.getString("userpassword"));
    user.setUserName(rs.getString("username"));
    user.setUserAge(rs.getInt("userage"));

    userList.add(user); //리스트에 회원 객체 추가 저장
}
for(int i=0; i<userList.size(); i++) {
    Users user = userList.get(i); //리스트에서 객체 가져와서
    System.out.println(user);      //객체 정보 출력

    /*System.out.println("=====");
    System.out.println("userId: " + user.getUserId());
    System.out.println("userPassword: " + user.getUserPassword());
    System.out.println("userName: " + user.getUserName());
    System.out.println("userAge: " + user.getUserAge());*/
}
```



DB 연동 - 회원 관리

■ 회원 전체 검색(Select)

```
        rs.close();    //rs 종료
        pstmt.close(); //pstmt 종료
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        if(conn != null) {
            try {
                conn.close(); //conn 종료
                System.out.println("연결 끊음");
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

Users [userId=today, userPassword=t1234$#, userName=이정후, userAge=26]
Users [userId=hangang, userPassword=h1234$#, userName=한강, userAge=0]
Users [userId=cloud, userPassword=c3355@!, userName=임시현, userAge=22]
연결 끊음
```



DB 연동 - 회원 관리

■ 회원 1건 검색(SelectOne)

```
public class UsersSelectOneTest {  
    public static void main(String[] args) {  
        Connection conn = null;  
        PreparedStatement pstmt = null;  
  
        try {  
            // db 연결  
            Class.forName("oracle.jdbc.OracleDriver");  
  
            conn = DriverManager.getConnection(  
                "jdbc:oracle:thin:@localhost:1521/xe",  
                "system",  
                "pw1234");  
  
            // sql - 1건 검색  
            String sql = "SELECT * FROM users WHERE userid = ?";  
            pstmt = conn.prepareStatement(sql);  
            pstmt.setString(1, "cloud"); //? - 1개  
  
            ResultSet rs = pstmt.executeQuery(); //검색된 객체 생성
```



DB 연동 - 회원 관리

■ 회원 1건 검색(SelectOne)

```
if(rs.next()) {  
    Users user = new Users(); //회원 객체 생성  
  
    user.setUserId(rs.getString("userid")); //db 필드에서 가져와서 아이디 설정  
    user.setPassword(rs.getString("userpassword"));  
    user.setUserName(rs.getString("username"));  
    user.setUserAge(rs.getInt("userage"));  
  
    System.out.println("userId: " + user.getUserId());  
    System.out.println("userPassword: " + user.getPassword());  
    System.out.println("userName: " + user.getUserName());  
    System.out.println("userAge: " + user.getUserAge());  
  
    /*  
    String userId = rs.getString("userid");  
    String userPassword = rs.getString("userpassword");  
    String userName = rs.getString("username");  
    int userAge = rs.getInt("userage");  
    */  
}
```



DB 연동 - 회원 관리

■ 회원 1건 검색(SelectOne)

```
        System.out.println("userId: " + userId);
        System.out.println("userPassword: " + userPassword);
        System.out.println("userName: " + userName);
        System.out.println("userAge: " + userAge);
        */
    }else {
        System.out.println("아이디가 존재하지 않습니다.");
    }
} catch (ClassNotFoundException e) {
    e.printStackTrace();
} catch (SQLException e) {
    e.printStackTrace();
} finally {
    if(conn != null) {
        try {
            conn.close();
            System.out.println("연결 끊음");
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}
```

```
userId: cloud
userPassword: c3355@!
userName: 임시현
userAge: 22
연결 끊음
```



DB 연동 – 게시판 관리

◆ 테이블 만들기(생성)

```
CREATE TABLE boards (  
    bno                number        primary key,  
    btitle             varchar2(100)  not null,  
    bcontent           clob           not null,  
    bwriter            varchar2(50)   not null,  
    bdate              date           default sysdate,  
    bfilename          varchar2(50)   null,  
    bfiledata          blob           null  
);  
  
CREATE SEQUENCE SEQ_BNO NOCACHE;  -- 시퀀스: 일련번호  
  
INSERT INTO boards(bno, btitle, bcontent, bwriter)  
VALUES (SEQ_BNO.NEXTVAL, 'smartphone', '삼성 갤럭시 s21입니다.', 'today');  
INSERT INTO boards(bno, btitle, bcontent, bwriter, bdate, bfilename, bfiledata)  
VALUES (SEQ_BNO.NEXTVAL, 'smartphone', '삼성 갤럭시 s21입니다.', 'today', SYSDATE, null, null);  
COMMIT;
```



Java – Oracle 연동

■ 데이터 저장(Create)

```
public class BoardWithFileInsertTest {
    public static void main(String[] args) {
        Connection conn = null;
        PreparedStatement pstmt = null;
        try {
            Class.forName("oracle.jdbc.OracleDriver");
            conn = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521/xe",
                "c##khsql",
                "pwkhsql");
            System.out.println("연결 성공");

            //db 작업
            String sql = "INSERT INTO boards(bno, btitle, bcontent, bwriter, bdate, bfilename, bfiledata) "
                + "VALUES(SEQ_BNO.NEXTVAL, ?, ?, ?, SYSDATE, ?, ?)";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, "notebook");
            pstmt.setString(2, "LG 그램 노트북입니다.");
            pstmt.setString(3, "sky123");
            pstmt.setString(4, "phone.jpg");
            pstmt.setBlob(5, new FileInputStream("src/dbdml/phone.jpg"));
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```



Java – Oracle 연동

■ 데이터 저장(Create)

```
//sql문 실행
int rows = pstmt.executeUpdate();
System.out.println("저장된 행 수: " + rows);
//자동으로 auto commit이 됨
} catch (ClassNotFoundException e) {
    e.printStackTrace();
} catch (SQLException e) {
    e.printStackTrace();
} catch (FileNotFoundException e) {
    e.printStackTrace();
} finally {
    if(conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
        }
        System.out.println("연결 끊음");
    }
}
}
```



Java – Oracle 연동

■ 데이터 저장(Create)

```
public class BoardWithFileInsertTest2 {
    public static void main(String[] args) {
        Connection conn = null;
        PreparedStatement pstmt = null;
        try {
            Class.forName("oracle.jdbc.OracleDriver");
            conn = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521/xe",
                "c##khsql",
                "pwkhsql");
            System.out.println("연결 성공");

            //db 작업
            String sql = "INSERT INTO boards(bno, btitle, bcontent, bwriter, bdate, bfilename, bfiledata) "
                + "VALUES(SEQ_BNO.NEXTVAL, ?, ?, ?, SYSDATE, ?, ?)";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, "notebook2");
            pstmt.setString(2, "LG 그램 노트북입니다.");
            pstmt.setString(3, "today");
            pstmt.setString(4, null);
            Blob blob = null;
            pstmt.setBlob(5, blob);
```



Java – Oracle 연동

■ 데이터 수정(Update)

```
public class BoardUpdateTest {  
  
    public static void main(String[] args) {  
        Connection conn = null;  
        PreparedStatement pstmt = null;  
        try {  
            Class.forName("oracle.jdbc.OracleDriver");  
            conn = DriverManager.getConnection(  
                "jdbc:oracle:thin:@localhost:1521/xe",  
                "c##khsql",  
                "pwkhsql");  
            System.out.println("연결 성공");  
  
            //db 작업, 매개 변수화된 SQL 문 작성  
            String sql = "UPDATE boards SET "  
                + "btitle=?, bcontent=?, bfilename=?, bfiledata=? "  
                + "WHERE bno=";  
            pstmt = conn.prepareStatement(sql);  
            pstmt.setString(1, "notebook2");  
            pstmt.setString(2, "LG 그램 노트북");  
            pstmt.setString(3, "P1235.png");
```



Java – Oracle 연동

■ 데이터 수정(Update)

```
pstmt.setBlob(4, new FileInputStream("src/dbdml/P1235.png"));
pstmt.setInt(5, 4);

int rows = pstmt.executeUpdate();
System.out.println("수정된 행 수: " + rows);

pstmt.close();
//자동으로 auto commit이 됨
} catch (Exception e) {
    e.printStackTrace();
} finally {
    if(conn != null) {
        try {
            conn.close();
        } catch (SQLException e) {
        }
        System.out.println("연결 끊음");
    }
}
}
```



Java – Oracle 연동

■ 데이터 삭제(Delete)

```
public class BoardDeleteTest {  
  
    public static void main(String[] args) {  
        Connection conn = null;  
        PreparedStatement pstmt = null;  
        try {  
            Class.forName("oracle.jdbc.OracleDriver");  
            conn = DriverManager.getConnection(  
                "jdbc:oracle:thin:@localhost:1521/xe",  
                "c##khsql",  
                "pwkhsql");  
            System.out.println("연결 성공");  
  
            //db 작업, 매개 변수화된 SQL 문 작성  
            String sql = "DELETE FROM boards WHERE bwriter = ?";  
            pstmt = conn.prepareStatement(sql);  
            pstmt.setString(1, "cloud");  
  
            int rows = pstmt.executeUpdate();  
            System.out.println("삭제된 행 수: " + rows);  
  
            pstmt.close();  
        } catch (Exception e) {
```



Java – Oracle 연동

▪ Board 데이터 조회(Select)

```
public class BoardSelectTest {
    public static void main(String[] args){
        Connection conn = null;
        PreparedStatement pstmt = null;
        try {
            Class.forName("oracle.jdbc.OracleDriver");
            conn = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521/xe",
                "c##khsql",
                "pwkhsql");
            System.out.println("연결 성공");

            String sql = "SELECT bno, btitle, bcontent, bwriter, bdate, bfilename, bfiledata "
                + "FROM boards WHERE bwriter = ?";
            pstmt = conn.prepareStatement(sql);
            pstmt.setString(1, "sky123");

            //검색된 데이터 가져오기
            ResultSet rs = pstmt.executeQuery();
            List<Board> boardList = new ArrayList<>();
            while(rs.next()) { //데이터가 있는 동안 반복
                Board board = new Board();
                board.setBno(rs.getInt("bno"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```



Java – Oracle 연동

■ Board 데이터 조회(Select)

```
board.setBcontent(rs.getString("bcontent"));
board.setBwriter(rs.getString("bwriter"));
board.setBdate(rs.getDate("bdate"));
board.setBfileName(rs.getString("bfilename"));
board.setBfileData(rs.getBlob("bfiledata"));
boardList.add(board);

Blob blob = board.getBfileData();
if(blob != null) {
    InputStream is = blob.getBinaryStream();
    OutputStream os = new FileOutputStream("c:/Temp/" + board.getBfileName());
    is.transferTo(os);

    /*byte[] data = new byte[1024];
    while(true) {
        int num = is.read(data);
        if(num == -1) break;
        os.write(data, 0, num);
    }*/
    os.flush();
    os.close();
    is.close();
}
```



게시판 구현

◆ 게시판 화면 - 1단계

[게시물 목록]

no writer date title

1 today12 2023.10.13 오늘도 좋은하루 되세요~

1.Create | 2.Read | 3.Clear | 4.Exit

선택: 1

create() 메서드 실행됨



게시판 구현

◆ 게시판 화면

```
public class BoardExample1 {  
  
    private Scanner scanner = new Scanner(System.in);  
    private Connection conn;  
  
    //생성자  
    public BoardExample1() {  
        try {  
            Class.forName("oracle.jdbc.OracleDriver");  
            conn = DriverManager.getConnection(  
                "jdbc:oracle:thin:@localhost:1521/xe",  
                "c##khsql",  
                "pwkhsql");  
            //System.out.println("db 연결 성공!!");  
        } catch (Exception e) {  
            e.printStackTrace();  
            exit();  
        }  
    }  
}
```



게시판 구현

◆ 게시판 화면

```
public void list() {  
    System.out.println();  
    System.out.println("[게시물 목록]");  
    System.out.println("-----");  
    System.out.printf("%-4s%-12s%-12s%-40s\n", "no", "writer", "date", "title");  
    System.out.println("-----");  
    System.out.printf("%-4s%-12s%-12s%-40s\n",  
        "1", "today12", "2023.10.13", "오늘도 좋은하루 되세요~");  
    System.out.println();  
  
    //메인 메뉴 호출  
    mainMenu();  
}
```



게시판 구현

◆ 게시판 화면

```
public void mainMenu() {  
    System.out.println("-----");  
    System.out.println("1.Create | 2.Read | 3.Clear | 4.Exit");  
    System.out.print("선택: ");  
    String menuNo = scanner.nextLine();  
  
    switch(menuNo) {  
        case "1":  
            create(); break;  
        case "2":  
            read(); break;  
        case "3":  
            clear(); break;  
        case "4":  
            exit(); break;  
    }  
}
```



게시판 구현

◆ 게시판 화면

```
public void create() {
    System.out.println("create() 메서드 실행됨");
    list();
}
public void read() {
    System.out.println("read() 메서드 실행됨");
    list();
}
public void clear() {
    System.out.println("clear() 메서드 실행됨");
    list();
}
public void exit() {
    System.out.println("*** 게시판을 종료합니다. ***");
    System.exit(0);
}
public static void main(String[] args) {
    BoardExample1 boardExample = new BoardExample1();
    boardExample.list();
}
```



게시판 구현

◆ Board 테이블 생성

```
CREATE TABLE board (  
    bno          number          primary key,  
    btitle       varchar2(100)   not null,  
    bcontent     clob            not null,  
    bwriter      varchar2(50)    not null,  
    bdate        date            default sysdate  
);  
  
CREATE SEQUENCE seq_bno NOCACHE; -- 시퀀스: 일련번호  
  
INSERT INTO board(bno, btitle, bcontent, bwriter)  
VALUES (seq_bno.NEXTVAL, 'smartphone', '삼성 갤럭시 S21입니다.', 'today');  
  
SELECT bno, btitle, bcontent, bwriter, bdate  
FROM board  
ORDER BY bno DESC;
```



게시판 구현

◆ Board 클래스 생성

```
public class Board implements Serializable{

    private static final long serialVersionUID = 1L;

    private int bno;
    private String btitle;
    private String bcontent;
    private String bwriter;
    private Date bdate;

    public int getBno() {
        return bno;
    }
    public void setBno(int bno) {
        this.bno = bno;
    }
    public String getBtitle() {
        return btitle;
    }
    public void setBtitle(String btitle) {
        this.btitle = btitle;
    }
}
```



게시판 구현

◆ Board 클래스 생성

```
public String getBcontent() {  
    return bcontent;  
}  
public void setBcontent(String bcontent) {  
    this.bcontent = bcontent;  
}  
public String getBwriter() {  
    return bwriter;  
}  
public void setBwriter(String bwriter) {  
    this.bwriter = bwriter;  
}  
public Date getBdate() {  
    return bdate;  
}  
public void setBdate(Date bdate) {  
    this.bdate = bdate;  
}  
}
```



게시판 구현

◆ 게시판 화면 - list() 메뉴 개발

[게시물 목록]

```
-----  
no  writer      date      title  
-----  
1   today       2023-10-16  smartphone
```

```
-----  
1.Create | 2.Read | 3.Clear | 4.Exit
```

선택:



게시판 구현

◆ 게시판 화면 - list() 메뉴 개발

```
public void list() {
    System.out.println("[게시물 목록]");
    System.out.println("-----");
    System.out.printf("%-4s%-12s%-12s%-40s\n", "no", "writer", "date", "title");
    System.out.println("-----");

    //board 테이블에서 게시물 정보를 가져와서 출력하기
    try {
        String sql = "SELECT bno, btitle, bcontent, bwriter, bdate"
            + " FROM board ORDER BY bno DESC";
        pstmt = conn.prepareStatement(sql);
        ResultSet rs = pstmt.executeQuery();
        while(rs.next()) {
            Board board = new Board();
            board.setBno(rs.getInt("bno"));
            board.setBwriter(rs.getString("bwriter"));
            board.setBtitle(rs.getString("btitle"));
            board.setBdate(rs.getDate("bdate"));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```



게시판 구현

◆ 게시판 화면 - list() 메뉴 개발

```
        System.out.printf("%-4s%-12s%-12s%-40s \n",
                           board.getBno(),
                           board.getBwriter(),
                           board.getBdate(),
                           board.getBtitle()
                           );
    }
    rs.close();
    pstmt.close();
} catch (SQLException e) {
    e.printStackTrace();
    exit(); //종료 함수 호출
}
//메인 메뉴 호출
mainMenu();
}
```



게시판 구현

◆ 게시판 화면 - Create 메뉴 개발

```
-----  
1.Create | 2.Read | 3.Clear | 4.Exit  
선택: 1
```

[새 게시물 입력]

제목: 가입 인사

내용: 가입 인사 드립니다. 잘 부탁드립니다~

작성자: cloud

```
-----  
1.Ok | 2.Cancel
```

선택: 1

[게시물 목록]

```
-----  
no  writer      date      title  
-----  
4   cloud       2023-10-16  가입 인사  
3   김기용      2023-10-16  제목 수정  
1   today       2023-10-16  smartphone
```



게시판 구현

◆ 게시판 화면 - Create 메뉴 개발

```
public void create() {  
    Board board = new Board();  
    System.out.println("[새 게시물 입력]");  
    System.out.print("제목: ");  
    board.setBtitle(scanner.nextLine());  
    System.out.print("내용: ");  
    board.setBcontent(scanner.nextLine());  
    System.out.print("작성자: ");  
    board.setBwriter(scanner.nextLine());  
  
    //확인, 취소 메뉴 생성  
    System.out.println("-----");  
    System.out.println("1.Ok | 2.Cancel");  
    System.out.print("선택: ");  
    String menuNo = scanner.nextLine();  
}
```



게시판 구현

◆ 게시판 화면 - Create 메뉴 개발

```
if(menuNo.equals("1")) { //1번 선택
    try {
        String sql = "INSERT INTO board(bno, btitle, bcontent, bwriter) "
            + "VALUES(SEQ_BNO.NEXTVAL, ?, ?, ?)";
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, board.getBtitle());
        pstmt.setString(2, board.getBcontent());
        pstmt.setString(3, board.getBwriter());
        pstmt.executeUpdate();

        pstmt.close();
    } catch (SQLException e) {
        e.printStackTrace();
        exit();
    }
}
//목록 함수 호출
list();
}
```



게시판 구현

◆ 게시판 화면 - Read 메뉴 개발

[게시물 목록]

no	writer	date	title
2	cloud	2023-10-17	notebook
1	today	2023-10-16	smartphone

1.Create | 2.Read | 3.Clear | 4.Exit

선택: 2

[게시물 읽기]

글번호 입력: 2

번호: 2

제목: notebook

내용: LG 노트북입니다.

작성자: cloud

작성일: 2023-10-17



게시판 구현

◆ 게시판 화면 - Read 메뉴 개발

```
public void read() {  
    //입력 받기  
    System.out.println("[게시물 읽기]");  
    System.out.print("글번호 입력: ");  
    int bno = Integer.parseInt(scanner.nextLine());  
  
    try {  
        String sql = "SELECT bno, btitle, bcontent, bwriter, bdate"  
            + " FROM board WHERE bno = ?";  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setInt(1, bno);  
        ResultSet rs = pstmt.executeQuery();  
        if(rs.next()) {  
            Board board = new Board();  
            board.setBno(rs.getInt("bno"));  
            board.setBtitle(rs.getString("btitle"));  
            board.setBcontent(rs.getString("bcontent"));  
            board.setBwriter(rs.getString("bwriter"));  
            board.setBdate(rs.getDate("bdate"));  
        }  
    }  
}
```



게시판 구현

◆ 게시판 화면 - Read 메뉴 개발

```
//게시글 출력
System.out.println("*****");
System.out.println("번호: " + board.getBno());
System.out.println("제목: " + board.getBtitle());
System.out.println("내용: " + board.getBcontent());
System.out.println("작성자: " + board.getBwriter());
System.out.println("작성일: " + board.getBdate());
System.out.println("*****");
}
rs.close();
pstmt.close();
} catch (SQLException e) {
    e.printStackTrace();
    exit();
}
//목록 함수 호출
list();
}
```



게시판 구현

◆ 게시판 화면 - Clear 메뉴 개발

```
-----  
1.Create | 2.Read | 3.Clear | 4.Exit  
선택: 3
```

[전체 게시물 삭제]

```
-----  
1.Ok | 2.Cancel  
선택: 1  
[게시물 목록]
```

```
-----  
no  writer      date      title  
-----
```



게시판 구현

◆ 게시판 화면 - Clear 메뉴 개발

```
public void clear() {  
    System.out.println("[전체 게시물 삭제]");  
    System.out.println("-----");  
    System.out.println("1.Ok | 2.Cancel");  
    System.out.print("선택: ");  
    String menuNo = scanner.nextLine();  
    if(menuNo.equals("1")) {  
        String sql = "TRUNCATE TABLE board";  
        try {  
            pstmt = conn.prepareStatement(sql);  
            pstmt.executeUpdate();  
            pstmt.close();  
        } catch (SQLException e) {  
            e.printStackTrace();  
            exit();  
        }  
    }  
    //목록 함수 호출  
    list();  
}
```



게시판 구현

◆ 게시판 화면 - Clear 메뉴 개발

```
if(menuNo.equals("1")) {  
    try {  
        String sql = "TRUNCATE TABLE board"; //모든 데이터 삭제  
        pstmt = conn.prepareStatement(sql);  
        pstmt.executeUpdate();  
  
        //데이터 생성시 글번호 이전 번호 초기화하여 1부터 시작하기  
        sql = "DROP SEQUENCE seq_bno"; //시퀀스 삭제  
        pstmt = conn.prepareStatement(sql);  
        pstmt.executeUpdate();  
  
        sql = "CREATE SEQUENCE seq_bno"; //시퀀스 생성  
        pstmt = conn.prepareStatement(sql);  
        pstmt.executeUpdate();  
  
        pstmt.close();  
    } catch (SQLException e) {
```



게시판 구현

◆ 게시판 화면 - Exit 메뉴 개발

[게시물 목록]

no	writer	date	title
2	cloud	2023-10-17	notebook
1	today	2023-10-16	smartphone

1.Create | 2.Read | 3.Clear | 4.Exit

선택: 4

*** 게시판을 종료합니다. ***



게시판 구현

◆ 게시판 화면 - Exit 메뉴 개발

```
public void exit() {  
    if(conn != null) {  
        try {  
            conn.close();  
        } catch (SQLException e) {  
        }  
    }  
    System.out.println("*** 게시판을 종료합니다. ***");  
    System.exit(0);  
}
```



게시판 구현

◆ Read 화면 - Update 메뉴 개발

```
번호: 2
제목: 날씨가 쌀쌀해요
내용: 날이 춥네요.. 감기 조심하세요
작성자: cloud120
작성일: 2023-10-17
*****

1.Update | 2.Delete | 3.List
선택: 1

[수정할 내용 입력]
제목: 좋은 아침입니다.
내용: 날씨가 쌀쌀해요.. 감기 조심하세요
작성자: cloud100
1.Ok | 2.Cancel
선택: 1
[게시물 목록]

-----
no  writer      date      title
-----
2   cloud100    2023-10-17  좋은 아침입니다.
1   todav10     2023-10-17  가입인사 드려요
```



◆ Read() 메서드 - Update 메뉴 개발

```
System.out.println("*****");
System.out.println("번호: " + board.getBno());
System.out.println("제목: " + board.getBtitle());
System.out.println("내용: " + board.getBcontent());
System.out.println("작성자: " + board.getBwriter());
System.out.println("작성일: " + board.getBdate());
System.out.println("*****");

//수정, 삭제, 목록가기 메뉴 생성
System.out.println("1.Update | 2.Delete | 3.List");
System.out.print("선택: ");
String menuNo = scanner.nextLine();
System.out.println();

if(menuNo.equals("1")) {
    update(board);
} else if(menuNo.equals("2")) {
    delete(board);
}
}
rs.close();
pstmt.close();
```



게시판 구현

◆ Read() 메서드 - Update 메뉴 개발

```
public void update(Board board) {
    System.out.println("[수정할 내용 입력]");
    System.out.print("제목: ");
    board.setBtitle(scanner.nextLine());
    System.out.print("내용: ");
    board.setBcontent(scanner.nextLine());
    System.out.print("작성자: ");
    board.setBwriter(scanner.nextLine());

    //확인, 취소 메뉴 생성
    System.out.println("1.Ok | 2.Cancel");
    System.out.print("선택: ");
    String menuNo = scanner.nextLine();
    if(menuNo.equals("1")) {
        try {
            String sql = "UPDATE board "
                + "SET btitle = ?, bcontent = ?, bwriter = ? "
                + "WHERE bno = ?";
```



게시판 구현

◆ Read() 메서드 - Update 메뉴 개발

```
        pstmt = conn.prepareStatement(sql);
        pstmt.setString(1, board.getBtitle());
        pstmt.setString(2, board.getBcontent());
        pstmt.setString(3, board.getBwriter());
        pstmt.setInt(4, board.getBno());
        pstmt.executeUpdate();
        pstmt.close();
    } catch (SQLException e) {
        e.printStackTrace();
        exit();
    }
}
//게시물 목록 출력
list();
}
```



게시판 구현

◆ Read() 메서드 - Delete 메뉴 개발

```
1.Create | 2.Read | 3.Clear | 4.Exit
```

```
선택: 2
```

```
[게시물 읽기]
```

```
글번호 입력: 1
```

```
*****
```

```
번호: 1
```

```
제목: 가입인사 드려요
```

```
내용: 안녕하세요~ 만나서 반갑습니다.
```

```
작성자: today10
```

```
작성일: 2023-10-17
```

```
*****
```

```
1.Update | 2.Delete | 3.List
```

```
선택: 2
```

```
[게시물 목록]
```

```
-----  
no  writer      date      title  
-----  
2   cloud100    2023-10-17  좋은 아침입니다.
```



게시판 구현

◆ Read() 메서드 - Delete 메뉴 개발

```
public void delete(Board board) {  
    //System.out.println("[게시글 삭제]");  
    try {  
        String sql = "DELETE FROM board WHERE bno = ?";  
        pstmt = conn.prepareStatement(sql);  
        pstmt.setInt(1, board.getBno());  
        pstmt.executeUpdate();  
        pstmt.close();  
    } catch (SQLException e) {  
        e.printStackTrace();  
        exit();  
    }  
    //게시물 목록 출력  
    list();  
}
```

