

08강 데이터베이스와 JDBC

예제 데이터를 만들어 보자

- 오라클에 테스트 계정(기존의 계정이 있다면 해당 계정을 사용한다)
 - 계정 생성

```
create user 계정명 identified by 비밀번호;  
alter user usertest01 quota 100m on users;  
alter user usertest01 default tablespace users;  
grant connect,resource,dba to 계정명;
```

- 테이블을 만들어 본다

```
create table member(  
    name varchar2(10) primary key,  
    userid varchar2(10),  
    userpwd varchar2(10),  
    email varchar2(20),  
    phone char(13)  
    admin number(1) default 0, --0:사용자, 1:관리자  
);
```

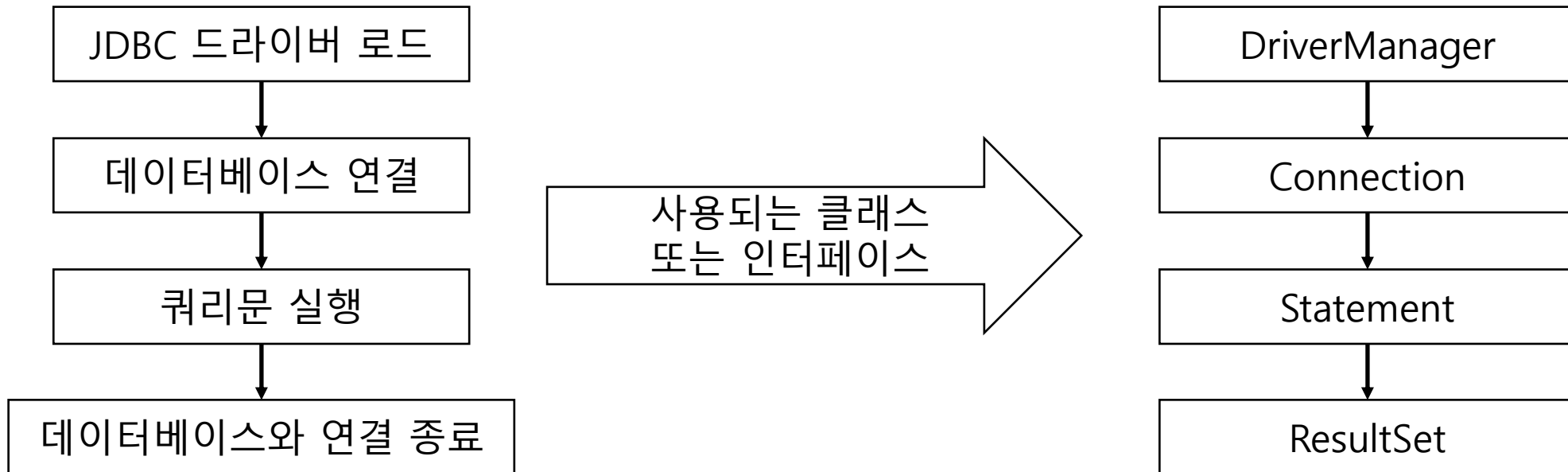
예제 데이터를 만들어 보자

- 오라클에 테스트 계정(기존의 계정이 있다면 해당 계정을 사용한다)
 - 테스트 레코드를 삽입한다.

```
insert into member
values ('고길동','ko','1234','ko@naver.com','010-1234-5678',0);
insert into member
values ('박길동','park','1234','park@naver.com','010-1234-5678',0);
insert into member
values ('이길동','lee','1234','lee@naver.com','010-1234-5678',1);
insert into member
values ('장길동','jang','1234','jang@naver.com','010-1234-5678',0);
insert into member
values ('홍길동','hong','1234','ko@naver.com','010-1234-5678',0);
```

JDBC를 통해서 데이터조작하기

- 자바는 JDBC를 통해서 데이터베이스에 접근하는 데 JDBC는 자바 프로그램에서 데이터베이스에 일관된 방식으로 접근할 수 있도록 API를 제공하는 클래스의 집합을 말합니다.
- JDBC를 이용해서 데이터베이스에 연결하는 방법은 4단계를 거쳐서 진행된다.



JDBC를 통해서 데이터조작하기

- JDBC에 연결하기 위해서 사용하는 클래스는 java.sql. 패키지에 포함되어 있다.

```
import java.sql.*
```

- 데이터베이스 연결에 사용된 인터페이스는 Connection
- 질의, 갱신과 관련된 인터페이스는 Statement
- 결과물을 가져오는 인터페이스는 ResultSet

JDBC를 통해서 데이터조작하기

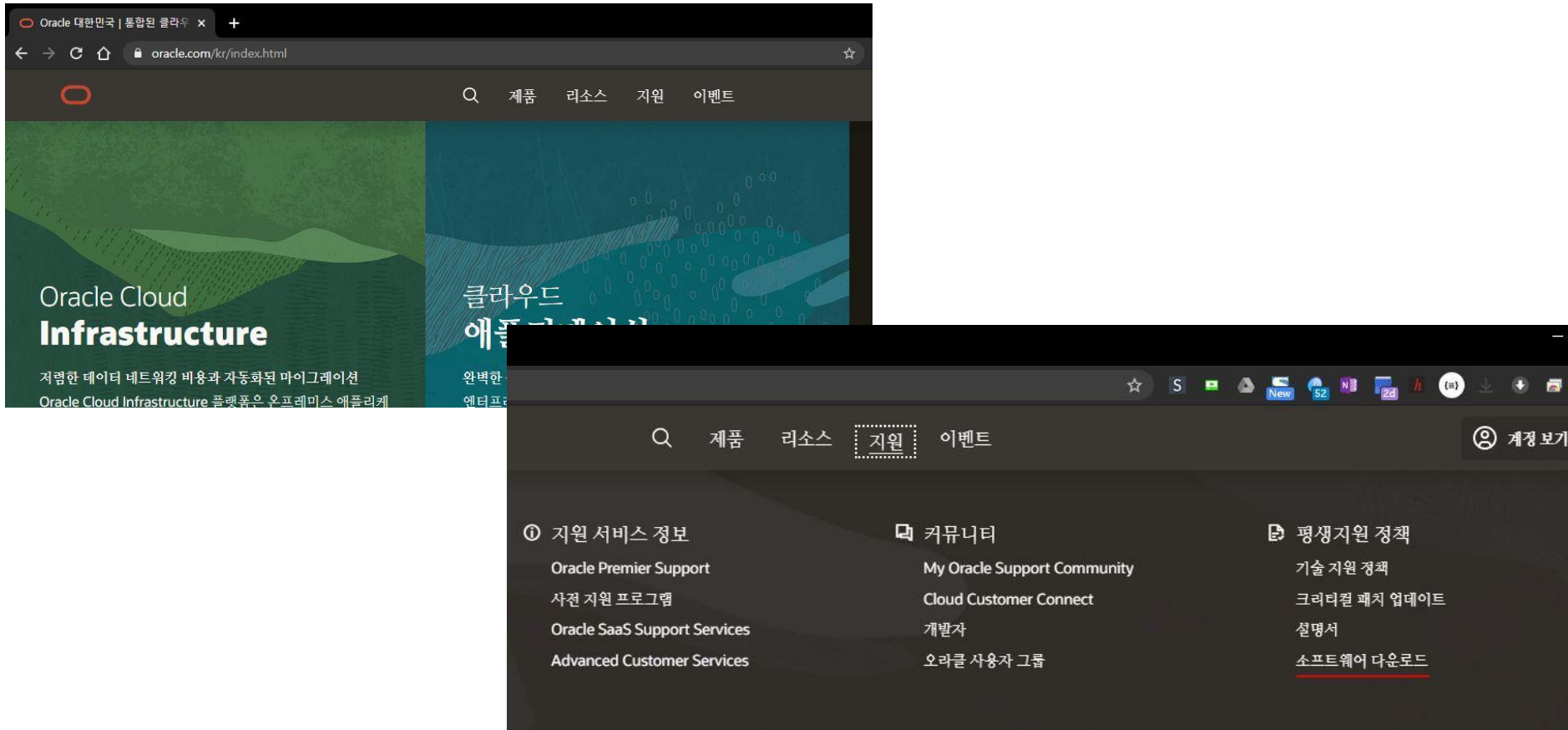
- JDBC에 연결하기 위해서 사용하는 클래스는 java.sql. 패키지에 포함되어 있다.

```
import java.sql.*
```

- 데이터베이스 연결에 사용된 인터페이스는 Connection
- 질의, 갱신과 관련된 인터페이스는 Statement
- 결과물을 가져오는 인터페이스는 ResultSet

JDBC를 통해서 데이터조작하기

- JDBC에 연결하기 위해서 사용할 드라이버 파일을 다운로드 받는다.





JDBC를 통해서 데이터조작하기


- JDBC에 연결하기 위해서 사용할 드라이버 파일을 다운로드 받는다.


개발자 다운로드


모든 소프트웨어 다운로드 는 무료이며 어플리케이션의 개발 및 프로토타이핑을 위해, 또는 자가 학습 용도로 엄격히 제한하여 제품의 전체 버전을 무료로 사용할 수 있도록 해 주시지 않는 이상, 오라클 기술 지원 조직은 본 계약 하에 사용권을 부여 받은 프로그램에 대한 기술 지원, 전화 지원 또는 업데이트를 제공하지 않습니다.) 온라인 [Oracle Store](#)에 포함된 제품을 언제든지 구매할 수 있습니다.


 데이터베이스

 미들웨어

 애플리케이션

 Java

 개발자 툴

 드라이버 및 유틸리티

드라이버 및 유틸리티

[클러스터 검증 유틸리티](#)

[Database Migration Assistant for Unicode](#)

[Enterprise Manager Grid Control Plugins](#)

[Failsafe](#)

[Help Technologies](#)

[JDBC 드라이버](#)

[JDeveloper Extensions](#)

[JDeveloper Extension SDK](#)

[node-oracledb Oracle Database Driver](#)

JDBC를 통해서 데이터조작하기

- JDBC에 연결하기 위해서 사용할 드라이버 파일(ojdbc6.jar)을 다운로드 받는다.
- (오라클 로그인 이 필요하다)

Oracle Database 19c and 18c JDBC driver

[Oracle Database 19c \(19.7\) drivers - NEW !!](#)

[Oracle Database 19c \(19.6\) drivers](#)

[Oracle Database 19c \(19.3\) drivers](#)

[Oracle Database 18c \(18.3\) drivers](#)

[Online JDBC Javadoc](#)

[Online UCP Javadoc](#)

Older Releases - 12.2,12.1, and 11.2 JDBC drivers

[Oracle Database 12c Release 2 \(12.2.0.1\) drivers](#)




[Oracle Database 12c Release 1 \(12.1.0.2\) drivers](#)

[Oracle Database 12c Release 1 \(12.1.0.1\) drivers](#)

[Oracle Database 11g Release 2 \(11.2.0.4\) drivers](#)

Unzipped JDBC Driver and Companion JARs

The JARs included in the **ojdbc-full.tar.gz** are also available as individual downloads in this section.

Download	Release Notes
 ojdbc6.jar	(2,739,670 bytes) - (SHA1 Checksum: a483a046eee2f404d864a6ff5b09dc0e1be3fe6c) Certified with <u>JDK8</u>, JDK7, and JDK6;
 ojdbc5.jar	(2,091,137 bytes) - (SHA1 Checksum: 5543067223760fc2277fe3f603d8c4630927679c) For use with <u>JDK1.5</u>;
 ucp.jar	(506,301 bytes) - (SHA1 Checksum: 5520b4e492939b477cc9ced90c03bc72710dcaf3)

JDBC를 통해서 데이터조작하기

- 프로젝트를 생성 후 다운 받은 Odbc6.jar 파일을 WebContent\WEB-INF\lib폴더에 저장한다.



JDBC를 통해서 데이터조작하기

- Class.forName()메소드를 통해서 JDBC 드라이버를 로딩한다.

```
Class.forName("oracle.jdbc.driver.OracleDriver")
```

- 로딩된 드라이버는 DriverManager에 등록된다.
- DriverManager객체에서 getConnection()메소드를 통해서 오라클 데이터베이스에 연결 객체를 생성한다.

```
Connection conn = DriverManager.getConnection(url, uid, upw);
```

- 이때 매개값으로 url과 uid와 upw를 전달하는데 각각의 의미는 다음과 같다
 - url : 데이터 베이스 주소
 - uid : 데이터베이스에 접속할 아이디
 - upw : 접속할 아이디의 비밀번호

The image shows a screenshot of the Oracle SQL Developer connection configuration dialog. The 'Credentials' tab is active, displaying the following fields: '인증 유형' (Authentication Type) set to '기본값' (Default), '사용자 이름(U)' (Username) set to 'HONGTEAM', and '비밀번호(P)' (Password) masked with dots. Below these, '접속 유형(Y)' (Connection Type) is set to '기본' (Basic). The '세부정보' (Advanced) tab is also visible, showing '호스트 이름(S)' (Host Name) as 'localhost', '포트(P)' (Port) as '1521', and 'SID(I)' (SID) as 'xe'. The '서비스 이름(E)' (Service Name) field is empty.

JDBC를 통해서 데이터조작하기

- URL 주소는 다음과 같은 형식에 맞춰서 작성한다.

```
jdbc:oracle:thin:[hostname][:port]:dbname
```

```
"jdbc:oracle:thin:@localhost:1521:XE"
```

- 다음은 커넥션 객체를 얻는 예제이다

```
Connection conn = DriverManager.getConnection(  
    'jdbc:oracle:thin:@localhost:1521:XE',  
    '계정명',  
    '비밀번호');
```

- 연결이 종료되면 연결을 끊어 주어야 한다.

```
conn.close();
```

JDBC를 통해서 데이터조작하기

- 연결이 완료되면 표준 쿼리문을 수행시키기 위한 객체(Statement)를 생성해야한다.

```
Statement stmt = conn.createStatement();
```

- 모든 작업이 완료되면 해당 객체도 닫아 주어야 한다.

```
stmt.close();
```

- 쿼리를 작성하면 쿼리를 실행할 메소드를 호출해야 한다.

메소드	설명
executeQuery()	select문과 같이 결과값이 여러 개의 레코드로 구해지는 경우에 사용
executeUpdate()	insert,update,delete문과 같은 내부적으로 테이블의 내용이 변경만 되고 결과값이 없는 경우에 사용

JDBC를 통해서 데이터조작하기

- select 쿼리를 실행하면 결과를 받아줄 객체가 필요하게 되는데 이때 사용하는 객체가 ResultSet 객체이다.

```
String sql = "select * from member";  
ResultSet rs = stmt.executeQuery(sql);
```

- ResultSet 객체는 여러 레코드를 저장하는데 한번에 하나의 레코드만 처리할 수 있다.(cursor)

name	userid	userpwd	email	phone	admin	Begin of File
고길동	ko	1234	ko@naver.com	010-1234-5678	0	
박길동	park	1234	park@naver.com	010-1234-5678	0	
이길동	lee	1234	lee@naver.com	010-1234-5678	1	
장길동	jang	1234	jang@naver.com	010-1234-5678	0	
홍길동	hong	1234	hong@naver.com	010-1234-5678	0	End of File

JDBC를 통해서 데이터조작하기

- ResultSet객체에서 각 레코드를 선택하기 위한 메소드를 제공한다.

메소드	설명
next()	현재 행에서 다음 행으로 이동
previous()	현재 행에서 이전 행으로 이동
first()	현재 행에서 첫번째 행으로 이동
last()	현재 행에서 마지막 행으로 이동

- 위 메소드는 성공적으로 실행시 true 반환, 실패시 false를 반환한다.
- 보통 모든 레코드를 순서대로 검색하기 위해서 while문과 next()메소드를 사용한다.

```
while(rs.next()){  
  
}
```

JDBC를 통해서 데이터조작하기

- 하나의 레코드를 선택했으면 이제 레코드에서 컬럼의 데이터를 꺼내 와야 한다.
- 컬럼의 데이터를 꺼내 오기 위해서는 `get타입("컬럼명")`메소드를 사용한다.

```
int admin = rs.getInt("admin");  
String name = rs.getString("name");
```

- 모든 컬럼의 데이터를 꺼내오는 예제(컬럼 이름만 일치하면 순서는 상관없다.)

```
String name = rs.getString("name");  
String userid = rs.getString("userid");  
String userpwd = rs.getString("userpwd");  
String email = rs.getString("email");  
String phone = rs.getString("phone");  
int admin = rs.getInt("admin");
```


JDBC를 통해서 데이터조작하기

- 컬럼을 꺼내올때는 컬럼 이름 뿐 아니라 테이블에 명시된 컬럼의 인덱스 값으로 가져올수도 있다.(시작은 1번 부터)

name	userid	userpwd	email	phone	admin
고길동	ko	1234	ko@naver.com	010-1234-5678	0
박길동	park	1234	park@naver.com	010-1234-5678	0
이길동	lee	1234	lee@naver.com	010-1234-5678	1
장길동	jang	1234	jang@naver.com	010-1234-5678	0
홍길동	hong	1234	hong@naver.com	010-1234-5678	0
1	2	3	4	5	6

```
String name = rs.getString(1);
String userid = rs.getString(2);
String userpwd = rs.getString(3);
String email = rs.getString(4);
String phone = rs.getString(5);
int admin = rs.getInt(6);
```

JDBC를 통해서 데이터조작하기

- 데이터 삽입

- 이제 데이터를 저장하는 방법을 알아보자.
- 데이터를 저장하기 위해서는 다음과 같은 insert문을 사용해야 한다.

```
insert into member  
values ('최길동','choi','1234','choi@naver.com','010-1234-5678',0);
```

- 위 쿼리를 전송하기 위해서는 Statement객체의 executeUpdate메소드를 사용하면 된다.

```
String sql = "insert into member values ('최길동','choi','1234','choi@naver.com','010-1234-5678',0)";  
stmt.executeUpdate(sql);
```

JDBC를 통해서 데이터조작하기

- 데이터 삽입

- 다만 저장해야 하는 값은 고정이 아니라 유동적으로 바뀐다.
- 그러므로 쿼리문을 다음과 같이 바꿔 주어야 한다.

```
String name = "정길동";
String userid = "jeong";
String userpwd = "1234";
String email = "jeong@naver.com";
String phone = "010-1234-5678";
String admin = "0";
String sql = "insert into member values ('"+name+"','"+userid
                                     + "','"+userpwd+"','"+email
                                     + "','"+phone+"','"+admin+"')";

stmt.executeUpdate(sql);
```

JDBC를 통해서 데이터조작하기

- 데이터 삽입

- 기존의 방법은 sql문을 작성하기 대단히 복잡하므로 좀더 개선된 방법이 제공된다.
- preparedStatement인터페이스를 활용한 방법을 사용한다
- preparedStatement객체를 활용하기 위해 preparedStatement()메소드를 호출합니다.

```
PreparedStatement psmt = conn. preparedStatement(sql);
```

- preparedStatement에서 사용하는 쿼리 문은 다음과 같이 작성한다.

```
String sql = "insert into member values(?,?,?,?,?,?)";
```

JDBC를 통해서 데이터조작하기

- 데이터 삽입

- 위 쿼리문에서 ?부분은 바인드 변수라고 해서 나중에 값을 넣어서 쿼리를 완성합니다.
? 부분을 채우는 방법

```
psmt.setString(인덱스, 문자열);  
psmt.setInt(인덱스, 숫자);
```

- 바인드 변수를 모두 채운후 쿼리를 실행하는 메소드를 호출하면 된다.

```
psmt.executeUpdate();  
psmt.close();
```

JDBC를 통해서 데이터조작하기

- 데이터 삽입

- 회원 가입 폼으로부터 데이터를 입력받아서 데이터베이스에 저장하는 코드를 작성해보자.