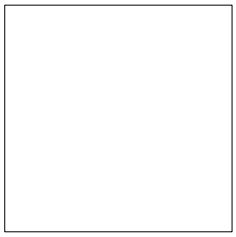
(8주차)
2015.08.25.(화)
---

cmd로 창 띄우고
sqlplus 입력하고 user name은 system 비밀번호는 지정한 것(설치하면서)으로

계정생성
create user "id" idenfied by "pwd";

생성한 계정에 권한을 부여
grant all privileges to scott;
--> 모든 권한을 부여

연결확인
conn "id"/"pwd"



DBMS
- **D**ata**B**ase **M**anagement **S**ystem
- data의 모음
- data를 저장/조회/삭제/수정

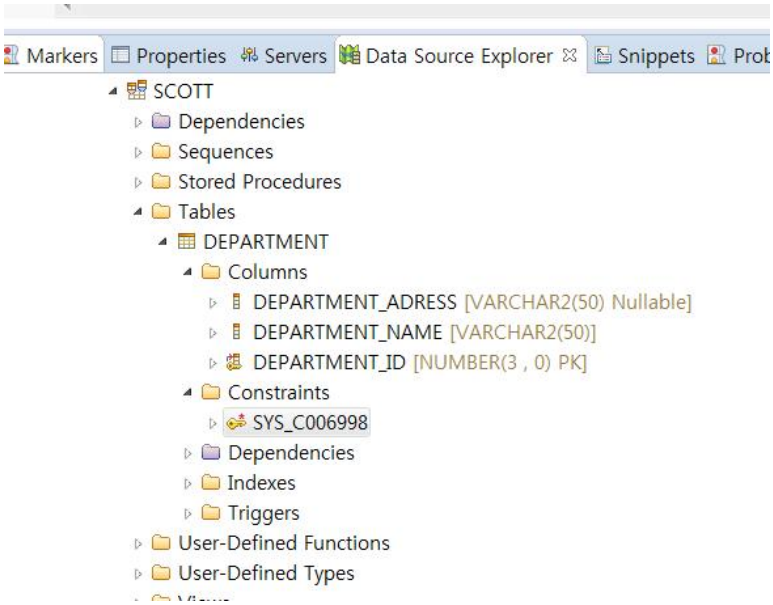
DB Client Program
- ex_ toad, sql Developer, Orange

이클립스 DB source 설정 pdf 참조

JDBC api

sql

테이블 생성
Create TABLE table_name
(colu,



scott 계정에 **Tables**에 보면 생성한 테이블들을 볼 수 있고 **Constraints**에 추가된 것이 제약자 이름(**SYS_C006998**는 pk를 자동생성된것)

(8주차)
2015.08.26.(수)

DB

테이블명
pk 칼럼 data type
일반 컬럼

ex)

DEPARTMENT
department_id nuber(3)
department_name varchar(50) location varchar2(50)

JDBC

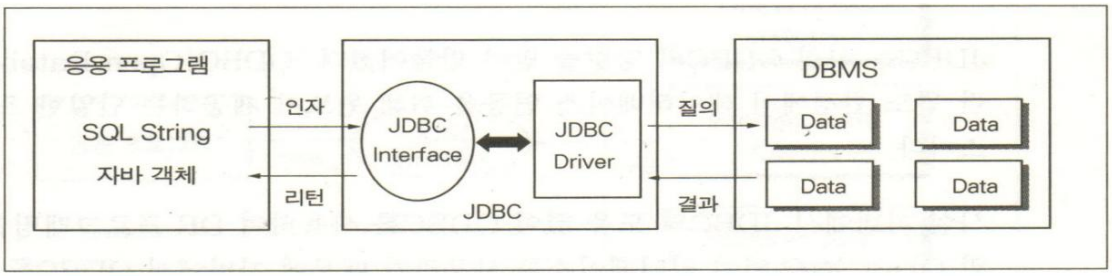
db는 네트워크를 통해서 서버로서 역할을 하게 됨..
db는 모든 컴퓨터?(사용자가 함께 써야하기 때문에 dB는 한군데서 관리한다.

DBMS

oracle / ms / mysql /
⇨ core api 는 회사마다 다르기 때문에 DB와 연동하는 API가 자바가 제공하지 못하고 DBMS 각 회사들이 만듦. java랑 연동하는 dbms를 만드려면 각 각의 DBMS회사들이 만들 수 밖에 없음. 그래서 sun은 ①**DBMS** 라는 표준, 가이드를 제공해서 dbms들이 저 표준을 활용해서 DBMS를 만들어서 제공해서 dbms가 바뀌더다라도 db를 사용하는데..문제없음?

oracle의 경우
C:₩oraclexe₩app₩oracle₩product₩11.2.0₩server₩jdbc₩lib₩**ojdbc6.jar** **(oracle에서 만든 api)**

C:₩Program Files₩Java₩jdk1.8.0_45₩jre₩lib₩ext(자바 외부 api, jar) 갔다 놓거나 class path를 잡아야 실행가능 --> 다른 dbms를 이용하려면 해당의 api를 java에 적용해야 한다.



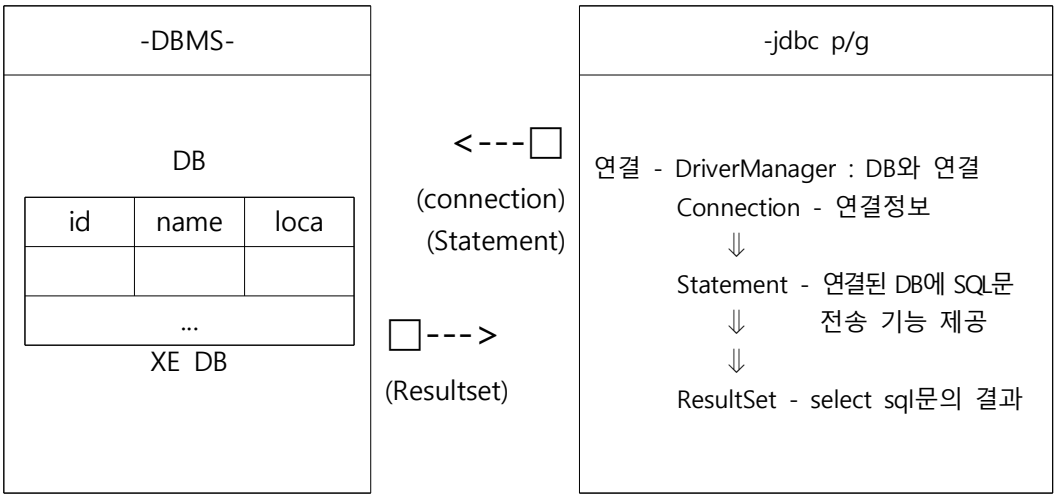
(8주차)
2015.08.27.(목)
---

data->row->table->DB
-DB를 관리하는 것이 dbms
-newtwork 서비스를 제공(서버개념)
- 외부 있는 DB에 접근하기 위해서 URL형식으로 연결주소를 제공
(pdf. 35page)

URI(indicator) : 상위 개념
↓
URL
- Uniform Resource Locator : 통일된 자원의 위치
- protocal://ip:port/자원의 경로 /.../../자원
**port를 생략하면 80port가 디폴트 값(웹의 경우)

Protocal
+-+ㅈ

java.sql
- ① Connection
- ① Statement
- ① ResultSet(select)



**결국
connetion이 socket의 역할
Statement가 outputstream의 역할
ResultSet이 InputStream의 역할
을 한다고 볼 수 있다.

-Statement를 통해 select쿼리를 DBMS에게 전송하면 ResultSet을 통해서 결과를 리턴.
-socket과 패턴과 유사하여 연결을 끊는 작업도 이행해야함

①Statement : Query를 막 날릴 수 있음,
⇓(상속받은, 기능추가)
①PreparedStatement : 미리 저장, 지정된 Query문을 보낸다. 성능은 좋지만, 융통성은 떨어짐
⇓
①Callerable Statement : DB에 있는 query(sp : stored procedure)를 활용, 특정 DB에 종속적 --> PLSQL

1. driver Loading : program 시작 시 한번만 한다.
sql작업

try{
① DB와 연결 -(결과)--> Connection
② Statement 생성
③ Sql문 전송

-> select squl문 경우 Resultset으로 리턴 받음
->ResultSet으로 조회 결과 처리
}finally{
④ 연결닫기 : ResultSet, Statement, Connetion
close;
}

**catch 직접처리하거나 throws : 간접처리

Driver load

driver loading : 어떤 DB를 쓸지를 등록해주는 작업,
- driver class 객체 생성
- Class.forName("driverclass이름"), 문자열 (driver를 바꿨을 때 설정을 바꿀 필요가 없다?)
  java.lang.Class : class의 정보를 알려주는
  class ?? = ???.class
  .getClass() class객체를 가져오거나
  .forName("패키지,클래스")

①표준API -
oracle.jdbc.driver.OracleDriver Class

①DB와 연결

URL / UserName / Pwd
Connection 객체 =
Connecton con = DriverManager.getConnection(url, id, pwd)
DriverManager.getConnection(url, id, pwd)
- 연결해주는 클래스
- 매개변수는 String 값으로

oracle 연결문 url
jdbc:oracle:thin:@127.0.0.1:1521:XE
protocal ip port sid(DB명)

②Statement

- Statement stmt = conn.createStatement();
- PreparedStatement = pstmt = conn.prepareStatement("SQL")

③SQL문 전송

- int count = stmt.executeUpdate(String sql문)
    ⇒ DB의 내용을 변경하는 SQL문 전송, select문 제외한 나머지 SQL문
    ⇒ 리턴 값 : 변경된 결과 수
- ResultSet rset = stmt.executeQuery(String select SQL문)
    ⇒ select SQL문 전송
    ⇒ 리턴 값 : ResultSet : select 실행결과 조회 기능 제공 객체

④ ResultSet으로 조회결과 조회 = select

** 첫 번째 행은 컬럼명이기 때문에

- next() : boolean
    ⇒ 다음 행(row)으로 이동
    ⇒ 값(행)이 잇는지 여부 retrun

- get컬럼type명(column 구분자) : String
    ⇒ 컬럼 type 별 getString, getLong, getInt 등등
    ⇒ 매개변수(coulumn구분자)는 순서(int)<-[1부터 시작]와 컬럼명(String)->별칭(alias)으로
    ⇒ Strng 값으로 리턴

-패턴

```
while (rset.next()){
int id = rset.getInt(1);
int String = rset.getString("dept_name"(;
}
```

- pk 제약조건 조회시 (pk로 하면 값이 하나 있거나 없거나 둘중하나이기에 if문사용

```
if(rset.next){
컬럼조회
]
```

⑤ 연결 닫기

- finally에서
- close() : 생성된 역순으로 close
    ResultSet -> Statement -> Connection

(8주차)
2015.08.28.(금)

오라클DB 다른사람이 쓰게 설정하는

1) 오라클 열어주기
C:₩oraclexe₩app₩oracle₩product₩11.2.0₩server₩network₩ADMIN

tnsnames.ora 파일 수정

xe

```
REMOTE_XE =
  (DESCRIPTION =
    (ADDRESS = (PROTOCOL = TCP)(HOST =  해당 IP)(PORT = 1521))
    (CONNECT_DATA =
      (SERVER = DEDICATED)
      (SERVICE_NAME = XE)
    )
  )
```

2) 방화벽 해제
방화벽 - 고급설정 - 인바운드 규칙 - 새규칙 - 포트 - 특정로컬포트(오라클 1521) 로 추가

(9주차)
2015.08.31.(월)

DB를 활용해서 업무처리를 하는~Service 클래스를 만들고 DAO를 테이블 당 하나씩 만들어서 DB를 관리할 때 불러와서 용이하게 사용한다.

DAO
insert테이블(vo)
① Connector - url. id, password
② statemnet - create
③ insert sql문 전송 - executeupdate()
④ close - stmt, conn
⑤ insert 수 리턴

결과적으로 같은 DB처리가 있다면 어떤 객체를 통해서 하더라도 DB처리되는 결과도 같기 때문에 객체를 여러 개의 객체를 생성할 필요 없다
즉, dao클래스의 인스턴스 변수들을 private으로 제공하거나 매서드를 static 제공해서 사용하는 것이 좋음

변수 저장영역
instance - heap
static - class, method **static을 남발하는 것은 좋지 않다.

local - stack

Singleton Design Pattern

- 하나의 객체만 생성하는 class를 구현하는 방법.
- 객체 여러 개 생성 후 메소드로 출시 결과가 다 같은 경우 굳이 객체를 여러개 생성할 필요 없기에 싱글톤 패턴class 구현
- 업무흐름(Business Logic)을 처리하는 class들 (동작 - 메서드 위주 class)
    -> Business Service
    -> DAO(DB연동)
    ->

prepared statement

prepared statement는
sq문의 변술 f
??

set
set null(int null
null

(9주차)
2015.09.01.(화)

.
prpared Statement
setSTring(1, null) : String은 null값이 올 수 있지만
setInt(1, int값) : int 값에는 null이 들어 갈수 없기에
setNull(1, int)값을 사용해서 두 번째 매개변수는 sql내의 아래 속성을 활용한다.
//정수: java.sql.Types.INTERGER, 실수: java.sql.Types.Double
//숫자: java.sql.Types.NUMERIC

Oracle Date Type

날짜시간
date - 년 / 월 / 일 / 시 / 분 / 초
timestamp - 년 / 월 / 일 / 시 / 분 / 초 - /밀리초

char(8)- YYYYMMDD
char(14) - YYYYMMDD HHMMSS

java.util.date를 상속 받은
① java.sql.Date 날짜 / ② java.sql.time 시간 / ③java.sql.TimeStamp 날짜/시간

get / set 메서드

Transaction

- business logic(업무처리), DB작업 -변경
- 작업처리 단위,
- 중간에 문제 발생 시 해온 처리작업을 원상복귀(roll back), 끝까지 정상적 으로 실행 시(commit)
- begin / commit / rollbakc;

Connection Pool

XXX Pool - XXX객체들을 모아서 관리, 그 객체들을 필요한 곳에 대여 처리,
- 객체 생성 시 비용(시간)이 많이 드는 객체들은 pool이용해 관리
- 미리 여러 개 만들어 놓고 사용시점에서는 만들어 진 것을 대여해서 사용하도록 한다.

Connection Pool은 객체가 실행되는 시점이 실제 시점이 아니라 사용 전에 만들어 놓고 여러 번 쓰게 됨

①DataSource

- javax.sql.①**DataSource** - DriverManager의 업글버전
- DB연결정보를 한 번만 설정하도록 설정.
- Conection Pool을 지원이 가능.

api다운 아파치님들이 하는 프로젝트를 통해 제작한 api를 build path를 통해 프로젝트에 적용하고~~
www.apache.org > commoms / dbcp, logging, pool

(9주차)
2015.09.03.(수)

- 하나의 프로그램이 basicData source를 통해 접근하는 DB는 하나이기에 DB당 하나의 basicDataSource를 활용하는 것이 가장 효율적으로 좋다.
- 즉, 이를 통합적으로 관리할 class로 DatabaseManage Class를 만들어서 하나의 DB(여러 테이블이 들어있는)를 관리 할 수 있게 해준다. basicdataSource는 하나만 만드는 클래스이기에 싱글패턴을 사용해서 객체를 생성한다., 하나의 객체를 통해서, 효율적으로 Connection을 사용할 수 있게 한다.

customerManager dao 적용하기

- CustomerDao(customer 테이블과 DB 작업)
  +어떤 메소드가 필요할지?
    ->메소드 리스트 문서를 작성(메소드들의 구문을 목록으로 작성
  +구현
    ->싱글톤 패턴**(完)**
    ->BasicDataSource를 이용해 디비 연결**(完)**
- CustomerService(고객관리 처리 클래스)**(完)**
  +구현**(完)(完)**
    ->데이터 저장소가 ArrayList에서 db 변경 - dao를 이용해 고객 데이터 관리
    -> 싱글콘 패턴**(完)**
  + 사용자 정의 exception 구현
    ->어떤 Exception 구현**(**내맘대로
    ->언제 발생시킬까? 내맘대로

메서드 리스트
추가 insertCustomer
아이디로 제거 deleteCustomerbyId
아이디로 고객 조회 selectCustomerById
이름으로 고객 조회 selectCustomerByName
전체 조회 selectAllCustomer
수정 updateCustmorById

예외처리
난중에 언젠가

xhem

(9주차)
2015.09.02.(목)
---
coustomer ㄴ
트랜스액션을 해야된다면 connection을 service클래스에서 열고 dao로 연결해야한다
트랜스액션이 connection을 받아서 rollback하고 commit해야되기 때문