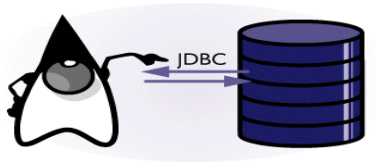
MYONGJI UNIVERSITY

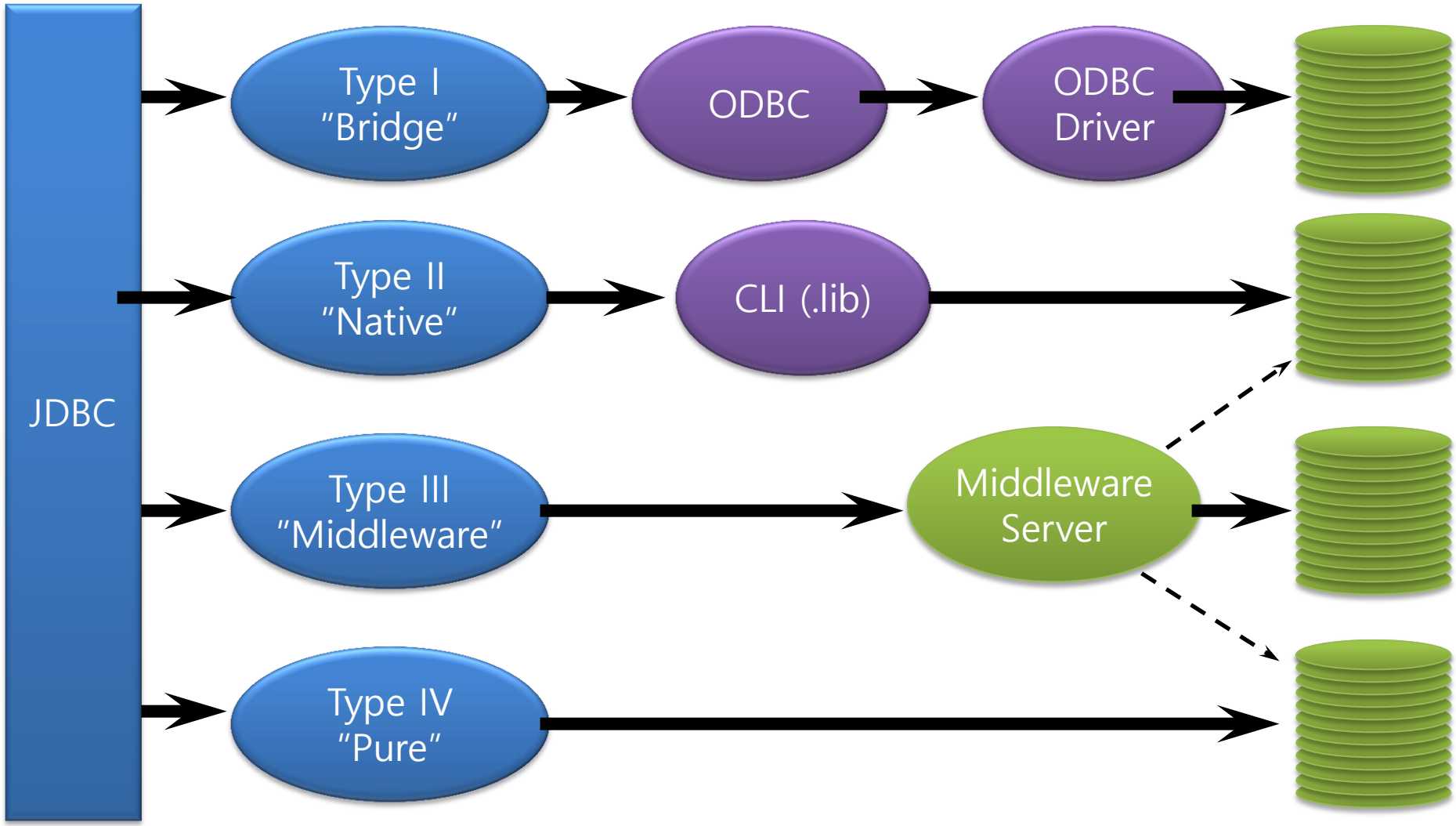
# JDBC

# JDBC




- JDBC(Java Database Connectivity)의 정의
  - 자바를 이용한 데이터베이스 접속과 SQL 문장의 실행, 그리고 실행 결과로 얻어진 데이터의 핸들링을 제공하는 방법과 절차에 관한 규약
  - 자바 프로그램내에서 SQL문을 실행하기 위한 자바 API
  - SQL과 프로그래밍 언어의 통합 접근 중 한 형태
- 개발자를 위한 표준 인터페이스인 JDBC API와 데이터베이스 벤더, 또는 기타 써드파티에서 제공하는 드라이버(driver)

# JDBC Drivers (Fig.)



# 환경 구성

- JDK 설치
  - http://java.sun.com
- JSP 환경 구비
  - http://jakarta.apache.org/
- JDBC 드라이버 설치
  - 오라클 JDBC 드라이버를 다운로드 받는다
    - http://otn.oracle.com/software/tech/java/sqlj_jdbc/index.html
    - ORACLE_HOME/jdbc/lib/ 아래에도 있음
  - JDBC 드라이버를 %JAVA_HOME%/jre/lib/ext 에 복사
  - 혹은 CLASSPATH에 추가

# 참고

- Java API Reference
  - http://docs.oracle.com/javase/7/docs/api/
- JDBC Reference
  - http://docs.oracle.com/javase/7/docs/technotes/guides/jdbc/index.html
- JDBC Tutorial
  - http://docs.oracle.com/javase/tutorial/jdbc/index.html

# JDBC 이용방법

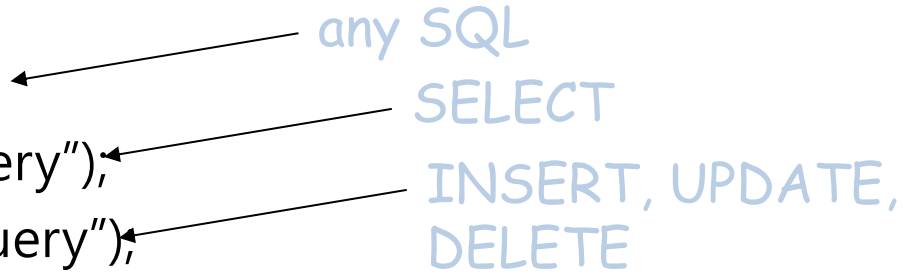
- JDBC를 이용한 데이터베이스 연결 방법
  - 1 단계 : import java.sql.*;
  - 2 단계 : 드라이버를 로드 한다.
  - 3 단계 : Connection 객체를 생성한다.
  - 4 단계 : Statement 객체를 생성 및 질의 수행
  - 5 단계 : SQL문에 결과물이 있다면 ResultSet 객체를 생성한다.
  - 6 단계 : 모든 객체를 닫는다.

# 단계별 설명 1

- IMPORT
  - import java.sql.*;
- 드라이버 로드
  - Class.forName("oracle.jdbc.driver.OracleDriver");
- Connection 받기
  - String dburl = "jdbc:oracle:thin:@localhost:1521:OID";
  - Connection con = DriverManager.getConnection (dburl, ID, PWD);

# 단계별 설명 2

- Statement 생성
  - Statement stmt = con.createStatement();
- 질의 수행
  - ResultSet rs = stmt.executeQuery("select ename from emp");
  - stmt.execute("query");
  - stmt.executeQuery("query");
  - stmt.executeUpdate("query");



# 질의수행 예

- executeQuery(String sql) – select

```
Statement stmt = con.createStatement();
ResultSet rs = stmt.executeQuery("SELECT * FROM emp");
```

- executeUpdate(String sql) –insert, update, delete

```
Statement stmt = con.createStatement();
int updateCount = stmt.executeUpdate("INSERT INTO test VALUES (1)");
```

- execute(String sql) – SQL문 모를경우

```
Statement stmt = con.createStatement();
if(stmt.execute(sql)){
   ResultSet rs = stmt.getResultSet();
...
}else{
   int rowCount = stmt.getUpdateCount();
...
}
```

# 단계별 설명 3

- ResultSet으로 결과 받기

```
ResultSet rs = stmt.executeQuery("select ename from emp");
while (rs.next()) {
    System.out.println(rs.getString("ename"));
}
```

  - 이동함수: **next**, previous, first, last, beforeFirst, afterLast, relative, absolute
  - 기본: TYPE_FORWARD_ONLY & CONCUR_READ_ONLY
  - 값 추출 함수: getXXX(숫자/이름) 데이터 타입에 따라 알맞게

# 단계별 설명 4

- Close
  - rs.close();
  - stmt.close();
  - con.close();

- 주의: Exception Handling!

# 예제 1: JDBCTest.java

```
import java.sql.*;


public class JDBCTest {
    static{
        try{
            Class.forName("oracle.jdbc.driver.OracleDriver");
            }catch (ClassNotFoundException cnfe) {
                cnfe.printStackTrace();
            }
    }
    public static void main(String[] args) throws SQLException{
        String dburl = "jdbc:oracle:thin:@localhost:1521:xe";
        Connection con = DriverManager.getConnection(dburl,"scott","tiger");
        Statement stmt = con.createStatement();

        ResultSet rs = stmt.executeQuery("SELECT ename FROM emp");

        while(rs.next()){
            System.out.println(rs.getString("ename"));
        }

        rs.close();
        stmt.close();
        con.close();
    }
```

# 예제 1

- 컴파일 방법
  - javac JDBCTest.java

- 실행 방법
  - java JDBCTest

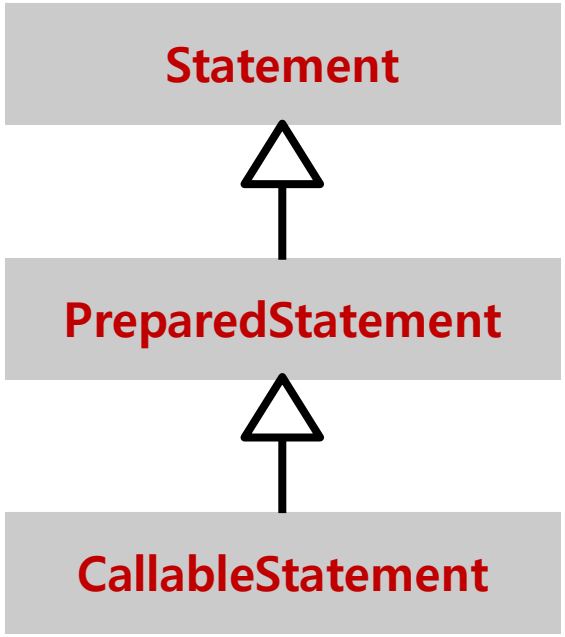
# 에러시 해결방법

- 에러 메시지를 확인하자
- 대소문자가 틀렸나?
- JDBC는 제대로 찾고 있나?
- CLASSPATH나 PATH는 설정이 잘 되어 있나?
- 오라클은 제대로 켜져 있는가?
- SQL*Plus로 접근이 가능한가?
- OID나 ID/PWD는 올바른가?

# 참고

- Java에서 Printf와 유사한 출력 (JDK1.5이상)
  - String.format → sprintf와 유사
  - System.out.format나 System.out.printf도 가능

# Statement 상속관계



# PreparedStatement

- 수행방법
  - SQL 미리 준비 : 파싱, 실행 계획 저장
  - 실시간에 Parameter 바인딩
- 바인딩
  - 바인딩변수: ?
  - setXXX 함수: 숫자 1부터
- 장점
  - 효율성
  - 보안 (SQL Injection)

# 예제 2. JDBCTest2.java

- PreparedStatement 실습

```
import java.sql.*;

class JDBCTest2 {
 static{
  try{
   Class.forName("oracle.jdbc.driver.OracleDriver");
  }catch(ClassNotFoundException cnfe){
   cnfe.printStackTrace();
  }
 }
 public static void main (String args []) throws SQLException {
  String dburl  = "jdbc:oracle:thin:@localhost:1521:OID";
  Connection con =  DriverManager.getConnection (dburl, "scott" , "tiger");

  int deptid = Integer.parseInt(args[0]);

  System.out.print("DEPT ID " + deptid + "=" );
  String sql = "SELECT dname, loc FROM dept WHERE deptno = ?";
  PreparedStatement pstmt = con.prepareStatement(sql);
  pstmt.setInt(1, deptid);
  ResultSet rs = pstmt.executeQuery();

  while (rs.next()) {
   System.out.println( rs.getString(1) + " (" + rs.getString(2) + ")");
  }
  rs.close();
  pstmt.close();
  con.close();
 }
}
```

# CallableStatement

- 저장 프로시저/함수 호출
- 사용법
  - CallableStatement cstmt =
    con.prepareCall("{call adjust(?,?)}");
    - 1. {?= call <procedure-name>([<arg1>,<arg2>, ...])}
    - 2. {call <procedure-name>([<arg1>,<arg2>, ...])}
  - 저장함수 리턴값
    - registerOutParameter()

# 예제 3. JDBCTest3.java

- CallableStatement 실습

아래와 같은 PL/SQL Stored Function을 생성한 후 작업

```
--#############################################################
-- Stored Function
--#############################################################


SQL>edit f1

CREATE OR REPLACE FUNCTION get_grade
   ( p_sal  IN  NUMBER)
   RETURN NUMBER
IS
   v_grade    NUMBER;
BEGIN
   SELECT grade INTO v_grade
   FROM salgrade
   WHERE  p_sal  BETWEEN losal AND hisal;
   RETURN v_grade;
END get_grade;
/
```

```
import java.sql.*;

class JDBCTest3 {
 static{
  try{
   Class.forName("oracle.jdbc.driver.OracleDriver");
  }catch(ClassNotFoundException cnfe){
   cnfe.printStackTrace();
  }
 }

 public static void main (String args []) throws SQLException {
  String dburl  = "jdbc:oracle:thin:@localhost:1521:BIT01";
  Connection con =  DriverManager.getConnection (dburl, "scott" , "tiger");

  CallableStatement cstmt = con.prepareCall("{? = call get_grade(?)}");
  cstmt.registerOutParameter(1,Types.INTEGER);

  for (int sal=1000; sal<5000; sal = sal + 1000) {
   cstmt.setInt(2, sal);
   cstmt.execute();
   int grade = cstmt.getInt(1);
   System.out.println(" Sal " + sal + " = Grade " + grade);
  }

  cstmt.close();
  con.close();
 }
}
```

# Transaction

- con.setAutoCommit(true/false);
- con.commit();
- con.rollback();

# Savepoint

```
Statement stmt = conn.createStatement();
int rows = stmt.executeUpdate("INSERT INTO TAB1 (COL1) VALUES ('AAA')");
...
//  set savepoint
Savepoint svpt1 = conn.setSavepoint("SAVEPOINT_1");
rows = stmt.executeUpdate("INSERT INTO TAB1 (COL1) VALUES ('AAA')");
 ...
conn.rollback(svpt1);
...
conn.commit();
```

# JDBC 접속 guide

- MySQL: http://dev.mysql.com/doc/refman/6.0/en/connector-j-reference.html
- MS SQL Server : http://www.microsoft.com/sql/technologies/jdbc/default.mspx
- PostgreSQL : http://jdbc.postgresql.org/

# 실습 과제 1.

- EMP 테이블의 결과를 다음과 같이 출력해보시오. (컬럼 이름 등은 문자열로 출력)

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|---|---|---|---|---|---|---|---|
| 7369 | SMITH | CLERK | 7902 | 80/12/17 | 880 | | 20 |
| 7499 | ALLEN | SALESMAN | 7698 | 81/02/20 | 1760 | 300 | 30 |
| 7521 | WARD | SALESMAN | 7698 | 81/02/22 | 1375 | 500 | 30 |
| 7566 | JONES | MANAGER | 7839 | 81/04/02 | 2975 | | 20 |
| 7654 | MARTIN | SALESMAN | 7698 | 81/09/28 | 1375 | 1400 | 30 |
| 7698 | BLAKE | MANAGER | 7839 | 81/05/01 | 2850 | | 30 |
| 7782 | CLARK | MANAGER | 7839 | 81/06/09 | 2450 | | 10 |
| 7788 | SCOTT | ANALYST | 7566 | 87/04/19 | 3000 | | 20 |
| 7839 | KING | PRESIDENT | | 81/11/17 | 5000 | | 10 |
| 7844 | TURNER | SALESMAN | 7698 | 81/09/08 | 1650 | 0 | 30 |
| 7876 | ADAMS | CLERK | 7788 | 87/05/23 | 1210 | | 20 |
| 7900 | JAMES | CLERK | 7698 | 81/12/03 | 1045 | | 30 |
| 7902 | FORD | ANALYST | 7566 | 81/12/03 | 3000 | | 20 |
| 7934 | MILLER | CLERK | 7782 | 82/01/23 | 1430 | | 10 |

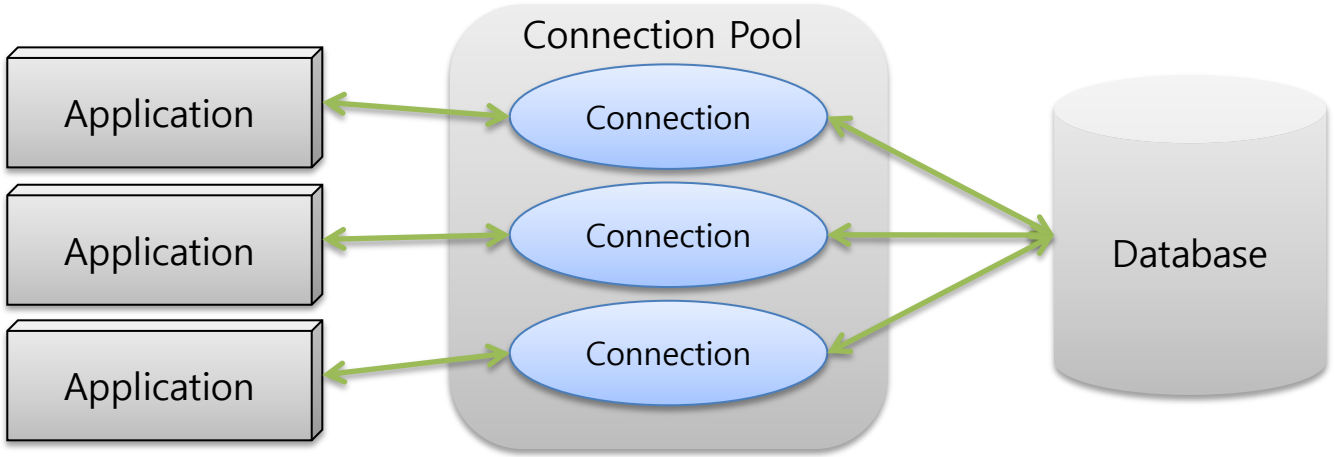
14 개의 행이 선택되었습니다.

# 실습 과제 2.

- 과제 1에서 사용자로부터 최소 봉급과 최대 봉급을 입력 받아 봉급이 이 범위 내에 속하는 사원의 정보를 출력하는 프로그램을 작성하라.

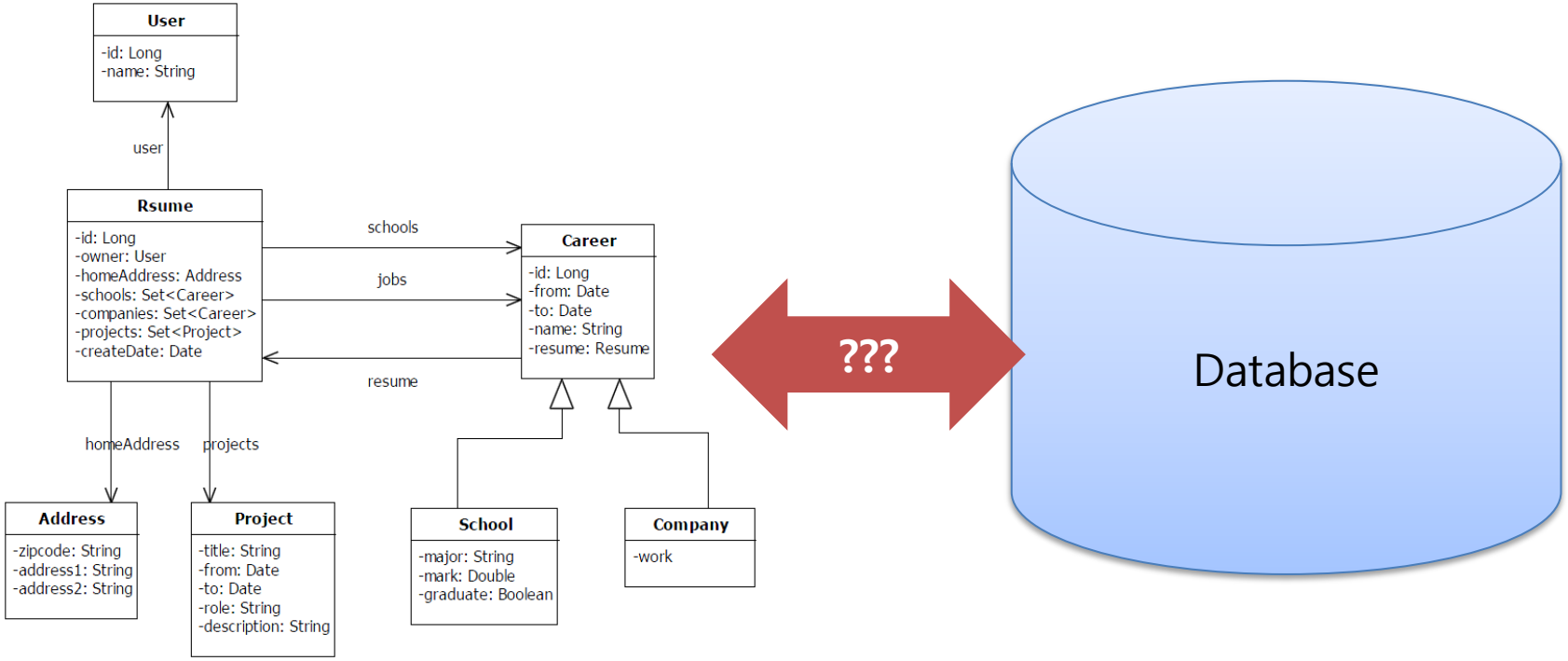
# Connection Pooling

- DB Connection을 pool로 유지하여 DB connection을 획득하고 반납하는 overhead를 최소화




- Popular Implementations
  - Apache DBCP: http://commons.apache.org/dbcp/
  - C3P0: http://www.mchange.com/projects/c3p0/

# Object-Relational Mapping




- ORM framework: Hibernate, iBatis, JDO …