# **[Java数据库连接——JDBC基础知识（操作数据库：增删改查）](http://www.cnblogs.com/Qian123/p/5339164.html)**

**阅读目录**

* [一、JDBC简介](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label0)
* [二、JDBC使用详解](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label1)
* [1、打通数据库](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label2)
* [2、搭建模型层](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label3)
* [3、搭建模型层——CRUD操作](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label4)
* [4、搭建控制层](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label5)
* [5、搭建视图层](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label6)
* [三、JDBC总结](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_label7)

[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

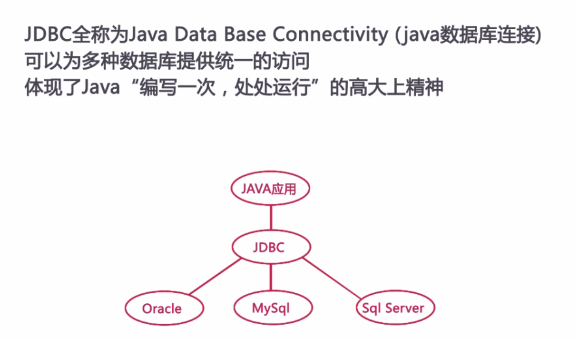
## **一、JDBC简介**

JDBC是连接java应用程序和数据库之间的桥梁。

**什么是JDBC?**

Java语言访问数据库的一种规范,是一套API。

JDBC (Java Database Connectivity) API，即Java数据库编程接口，是一组标准的Java语言中的接口和类，使用这些接口和类，Java客户端程序可以访问各种不同类型的数据库。比如建立数据库连接、执行SQL语句进行数据的存取操作。



JDBC代表Java数据库连接。

JDBC库中所包含的API任务通常与数据库使用：

连接到数据库

创建SQL或MySQL语句

在数据库中执行SQL或MySQL查询

查看和修改记录

比如：



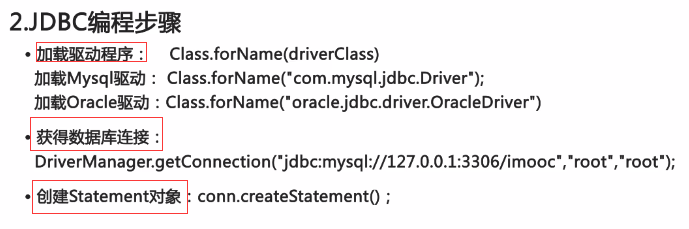
[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **二、JDBC使用详解**

[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **1、打通数据库**





先新建Java 工程，将jar包导进去：IMG_260

然后新建相应的数据库和表。

编写DBUtil.java

[IMG_261](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.db; 2 3 import java.sql.Connection; 4 import java.sql.DriverManager; 5 import java.sql.ResultSet; 6 import java.sql.Statement; 7 8 public class DBUtil { 9 10 private static final String URL="jdbc:mysql://localhost:3306/demo\_jdbc";11 private static final String NAME="root";12 private static final String PASSWORD="root";13 14 public static void main(String[] args) throws Exception{15 16 //1.加载驱动程序17 Class.forName("com.mysql.jdbc.Driver");18 //2.获得数据库的连接19 Connection conn = DriverManager.getConnection(URL, NAME, PASSWORD);20 //3.通过数据库的连接操作数据库，实现增删改查21 Statement stmt = conn.createStatement();22 ResultSet rs = stmt.executeQuery("select user\_name,age from imooc\_goddess");//选择import java.sql.ResultSet;23 while(rs.next()){//如果对象中有数据，就会循环打印出来24 System.out.println(rs.getString("user\_name")+","+rs.getInt("age"));25 }26 }27 }

[IMG_262](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

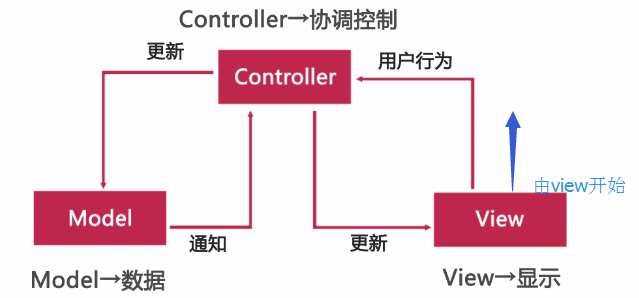
[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **2、搭建模型层**



采用MVC三层架构：建议由下往上开发（DB->view,需求读懂以后，设计数据库，然后根据数据库映射模型，再写控制层，再写视图层）

* View视图层——展示数据，反馈用户行为
* Control控制层——控制数据流通过程，协调视图层和数据层
* Model模型层——与数据库建立映射，与数据进行交互
* （DB数据库）



先新增对应的model层：按照数据库表对应的字段编写实体类。

[IMG_265](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.model; 2 3 import java.util.Date; 4 5 public class Goddess { 6 private Integer id; 7 private String userName; 8 private Integer sex; 9 private Integer age;10 private Date birthday;11 private String email;12 private String mobile;13 private String createUser;14 private Date createDate;15 private String updateUser;16 private Date updateDate;17 private Integer isDel;18 public Integer getId() {19 return id;20 }21 public void setId(Integer id) {22 this.id = id;23 }24 public String getUserName() {25 return userName;26 }27 public void setUserName(String userName) {28 this.userName = userName;29 }30 public Integer getSex() {31 return sex;32 }33 public void setSex(Integer sex) {34 this.sex = sex;35 }36 public Integer getAge() {37 return age;38 }39 public void setAge(Integer age) {40 this.age = age;41 }42 public Date getBirthday() {43 return birthday;44 }45 public void setBirthday(Date birthday) {46 this.birthday = birthday;47 }48 public String getEmail() {49 return email;50 }51 public void setEmail(String email) {52 this.email = email;53 }54 public String getMobile() {55 return mobile;56 }57 public void setMobile(String mobile) {58 this.mobile = mobile;59 }60 public String getCreateUser() {61 return createUser;62 }63 public void setCreateUser(String createUser) {64 this.createUser = createUser;65 }66 public Date getCreateDate() {67 return createDate;68 }69 public void setCreateDate(Date createDate) {70 this.createDate = createDate;71 }72 public String getUpdateUser() {73 return updateUser;74 }75 public void setUpdateUser(String updateUser) {76 this.updateUser = updateUser;77 }78 public Date getUpdateDate() {79 return updateDate;80 }81 public void setUpdateDate(Date updateDate) {82 this.updateDate = updateDate;83 }84 public Integer getIsDel() {85 return isDel;86 }87 public void setIsDel(Integer isDel) {88 this.isDel = isDel;89 }90 91 }

[IMG_266](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

然后开始新增Dao层，并修改DBUtil.java

DBUtil.java代码如下：

[IMG_267](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.db; 2 3 import java.sql.Connection; 4 import java.sql.DriverManager; 5 import java.sql.ResultSet; 6 import java.sql.SQLException; 7 import java.sql.Statement; 8 9 public class DBUtil {10 11 private static final String URL="jdbc:mysql://localhost:3306/demo\_jdbc";12 private static final String NAME="root";13 private static final String PASSWORD="root";14 15 private static Connection conn=null;16 //静态代码块（将加载驱动、连接数据库放入静态块中）17 static{18 try {19 //1.加载驱动程序20 Class.forName("com.mysql.jdbc.Driver");21 //2.获得数据库的连接22 conn = DriverManager.getConnection(URL, NAME, PASSWORD);23 } catch (ClassNotFoundException e) {24 e.printStackTrace();25 } catch (SQLException e) {26 e.printStackTrace();27 }28 }29 //对外提供一个方法来获取数据库连接30 public static Connection getConnection(){31 return conn;32 }33 34 public static void main(String[] args) throws Exception{35 36 //3.通过数据库的连接操作数据库，实现增删改查37 Statement stmt = conn.createStatement();38 ResultSet rs = stmt.executeQuery("select user\_name,age from imooc\_goddess");//选择import java.sql.ResultSet;39 while(rs.next()){//如果对象中有数据，就会循环打印出来40 System.out.println(rs.getString("user\_name")+","+rs.getInt("age"));41 }42 }43 }

[IMG_268](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

goddessDao.java 代码如下：

[IMG_269](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.dao; 2 3 import java.sql.Connection; 4 import java.sql.ResultSet; 5 import java.sql.SQLException; 6 import java.sql.Statement; 7 import java.util.ArrayList; 8 import java.util.List; 9 10 import com.study.db.DBUtil;11 import com.study.model.Goddess;12 13 public class GoddessDao {14 15 public void addGoddess(){16 17 }18 19 public void updateGoddess(){20 21 }22 public void delGoddess(){23 24 }25 26 public List<Goddess> query() throws Exception{27 Connection con=DBUtil.getConnection();28 Statement stmt=con.createStatement();29 ResultSet rs=stmt.executeQuery("select user\_name,age from imooc\_goddess");30 List<Goddess> gs=new ArrayList<Goddess>();31 Goddess g=null;32 while(rs.next()){//如果对象中有数据，就会循环打印出来33 g=new Goddess();34 g.setUserName(rs.getString("user\_name"));35 g.setAge(rs.getInt("age"));36 gs.add(g);37 }38 return gs;39 }40 41 //查询单个女神42 public Goddess get(){43 return null;44 }45 }

[IMG_270](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

新增Controller层：GoddessAction.java

[IMG_271](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.controller; 2 3 import java.util.List; 4 5 import com.study.dao.GoddessDao; 6 import com.study.model.Goddess; 7 8 public class GoddessAction { 9 10 public static void main(String[] args) throws Exception {11 GoddessDao gd=new GoddessDao();12 List<Goddess> gs=gd.query();13 for (Goddess goddess : gs) {14 System.out.println(goddess.getUserName()+","+goddess.getAge());15 }16 17 }18 19 }

[IMG_272](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

PS：CRUD：增加(Create)、读取(Retrieve)（重新得到数据-->高级应用：报表、分页）、更新(Update)和删除(Delete)

[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **3、搭建模型层——CRUD操作**

新增create操作：

[IMG_273](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 //添加女神 2 public void addGoddess(Goddess g) throws Exception{ 3 Connection con=DBUtil.getConnection();//首先拿到数据库的连接 4 String sql="" + 5 "insert into imooc\_goddess"+ 6 "(user\_name,sex,age,birthday,email,mobile,"+ 7 "create\_user,create\_date,update\_user,update\_date,isdel) "+ 8 "values("+ 9 "?,?,?,?,?,?,?,current\_date(),?,current\_date(),?)";//参数用?表示，相当于占位符;用mysql的日期函数current\_date()来获取当前日期10 //预编译sql语句11 PreparedStatement psmt = con.prepareStatement(sql);12 //先对应SQL语句，给SQL语句传递参数13 psmt.setString(1, g.getUserName());14 psmt.setInt(2, g.getSex());15 psmt.setInt(3, g.getAge());16 //注意：setDate()函数第二个参数需要的是java.sql.Date类型，我们传进来的是java.util.Date，类型不符，需要做一下转换17 psmt.setDate(4, new Date(g.getBirthday().getTime()));18 psmt.setString(5, g.getEmail());19 psmt.setString(6, g.getMobile());20 psmt.setString(7, g.getCreateUser());21 psmt.setString(8, g.getUpdateUser());22 psmt.setInt(9, g.getIsDel());23 //执行SQL语句24 psmt.execute();25 /\*\*

26 \* prepareStatement这个方法会将SQL语句加载到驱动程序conn集成程序中，但是并不直接执行

27 \* 而是当它调用execute()方法的时候才真正执行；

28 \*

29 \* 上面SQL中的参数用?表示，相当于占位符，然后在对参数进行赋值。

30 \* 当真正执行时，这些参数会加载在SQL语句中，把SQL语句拼接完整才去执行。

31 \* 这样就会减少对数据库的操作

32 \*/33 }

[IMG_274](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

测试：

[IMG_275](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

package com.study.controller;

import java.util.Date;import java.util.List;

import com.study.dao.GoddessDao;import com.study.model.Goddess;

public class GoddessAction {

public static void main(String[] args) throws Exception {

GoddessDao gd=new GoddessDao();

Goddess g1=new Goddess();

g1.setUserName("小夏");

g1.setAge(22);

g1.setSex(1);

g1.setBirthday(new Date());

g1.setEmail("xiaoxia@qq.com");

g1.setMobile("123456789");

g1.setCreateUser("admin");

g1.setUpdateUser("admin");

g1.setIsDel(1);

//添加女神 gd.addGoddess(g1);

}

}

[IMG_276](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

修改、删除、查询(根据id)操作：

[IMG_277](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 //更新女神 2 public void updateGoddess(Goddess g) throws SQLException{ 3 Connection con=DBUtil.getConnection();//首先拿到数据库的连接 4 String sql="" + 5 "update imooc\_goddess "+ 6 "set user\_name=?,sex=?,age=?,birthday=?,email=?,mobile=?,"+ 7 "update\_user=?,update\_date=current\_date(),isdel=?) "+ 8 "where id=?";//参数用?表示，相当于占位符;用mysql的日期函数current\_date()来获取当前日期 9 //预编译sql语句10 PreparedStatement psmt = con.prepareStatement(sql);11 //先对应SQL语句，给SQL语句传递参数12 psmt.setString(1, g.getUserName());13 psmt.setInt(2, g.getSex());14 psmt.setInt(3, g.getAge());15 //注意：setDate()函数第二个参数需要的是java.sql.Date类型，我们传进来的是java.util.Date，类型不符，需要做一下转换16 psmt.setDate(4, new Date(g.getBirthday().getTime()));17 psmt.setString(5, g.getEmail());18 psmt.setString(6, g.getMobile());19 psmt.setString(7, g.getUpdateUser());20 psmt.setInt(8, g.getIsDel());21 psmt.setInt(9, g.getId());22 //执行SQL语句23 psmt.execute(); 24 }25 26 //删除女神27 public void delGoddess(Integer id) throws SQLException{28 Connection con=DBUtil.getConnection();//首先拿到数据库的连接29 String sql="" + 30 "delete from imooc\_goddess "+31 "where id=?";//参数用?表示，相当于占位符;用mysql的日期函数current\_date()来获取当前日期32 //预编译sql语句33 PreparedStatement psmt = con.prepareStatement(sql);34 //先对应SQL语句，给SQL语句传递参数35 psmt.setInt(1, id);36 //执行SQL语句37 psmt.execute(); 38 }39 40 //查询单个女神(根据id去查询)41 public Goddess get(Integer id) throws SQLException{42 Goddess g=null;43 Connection con=DBUtil.getConnection();//首先拿到数据库的连接44 String sql="" + 45 "select \* from imooc\_goddess "+46 "where id=?";//参数用?表示，相当于占位符;用mysql的日期函数current\_date()来获取当前日期47 //预编译sql语句48 PreparedStatement psmt = con.prepareStatement(sql);49 //先对应SQL语句，给SQL语句传递参数50 psmt.setInt(1, id);51 //执行SQL语句52 /\*psmt.execute();\*///execute()方法是执行更改数据库操作（包括新增、修改、删除）;executeQuery()是执行查询操作53 ResultSet rs = psmt.executeQuery();//返回一个结果集54 //遍历结果集55 while(rs.next()){56 g=new Goddess();57 g.setId(rs.getInt("id"));58 g.setUserName(rs.getString("user\_name"));59 g.setAge(rs.getInt("age"));60 g.setSex(rs.getInt("sex"));61 //rs.getDate("birthday")获得的是java.sql.Date类型。注意：java.sql.Date类型是java.util.Date类型的子集，所以这里不需要进行转换了。62 g.setBirthday(rs.getDate("birthday"));63 g.setEmail(rs.getString("email"));64 g.setMobile(rs.getString("mobile"));65 g.setCreateUser(rs.getString("create\_user"));66 g.setCreateDate(rs.getDate("create\_date"));67 g.setUpdateUser(rs.getString("update\_user"));68 g.setUpdateDate(rs.getDate("update\_date"));69 g.setIsDel(rs.getInt("isdel"));70 }71 return g;72 }

[IMG_278](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

 查询（根据name\mobile等）操作

[IMG_279](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 //查询单个女神(根据姓名等信息去查询) 2 public List<Goddess> get(String name,String mobile) throws SQLException{ 3 List<Goddess> result=new ArrayList<Goddess>(); 4 Connection con=DBUtil.getConnection();//首先拿到数据库的连接 5 StringBuffer sb=new StringBuffer(); 6 sb.append("select \* from imooc\_goddess "); 7 sb.append("where user\_name like ? and mobile like ?"); 8 //预编译sql语句 9 PreparedStatement psmt = con.prepareStatement(sb.toString());10 //先对应SQL语句，给SQL语句传递参数11 psmt.setString(1, "%"+name+"%");12 psmt.setString(2, "%"+mobile+"%"); 13 System.out.println(sb.toString());14 //执行SQL语句15 /\*psmt.execute();\*///execute()方法是执行更改数据库操作（包括新增、修改、删除）;executeQuery()是执行查询操作16 ResultSet rs = psmt.executeQuery();//返回一个结果集17 Goddess g=null;18 //遍历结果集19 while(rs.next()){20 g=new Goddess();21 g.setId(rs.getInt("id"));22 g.setUserName(rs.getString("user\_name"));23 g.setAge(rs.getInt("age"));24 g.setSex(rs.getInt("sex"));25 //rs.getDate("birthday")获得的是java.sql.Date类型。注意：java.sql.Date类型是java.util.Date类型的子集，所以这里不需要进行转换了。26 g.setBirthday(rs.getDate("birthday"));27 g.setEmail(rs.getString("email"));28 g.setMobile(rs.getString("mobile"));29 g.setCreateUser(rs.getString("create\_user"));30 g.setCreateDate(rs.getDate("create\_date"));31 g.setUpdateUser(rs.getString("update\_user"));32 g.setUpdateDate(rs.getDate("update\_date"));33 g.setIsDel(rs.getInt("isdel"));34 35 result.add(g);//将结果封装到对象中36 }37 return result;38 }

[IMG_280](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

测试：

[IMG_281](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.controller; 2 3 import java.util.Date; 4 import java.util.List; 5 6 import com.study.dao.GoddessDao; 7 import com.study.model.Goddess; 8 9 public class GoddessAction {10 11 public static void main(String[] args) throws Exception {12 GoddessDao gd=new GoddessDao();13 /\*//查询所有女神14 List<Goddess> gs=gd.query();15 for (Goddess goddess : gs) {16 System.out.println(goddess.getUserName()+","+goddess.getAge());17 }\*/18 19 Goddess g1=new Goddess();20 g1.setUserName("小夏");21 g1.setAge(22);22 g1.setSex(1);23 g1.setBirthday(new Date());24 g1.setEmail("xiaoxia@qq.com");25 g1.setMobile("123456789");26 g1.setCreateUser("admin");27 g1.setUpdateUser("admin");28 g1.setIsDel(1);29 //添加女神30 // gd.addGoddess(g1);31 32 //查询单个女神（根据id）33 /\*Goddess g2 = gd.get(4);34 System.out.println(g2.toString());\*/35 36 //查询单个女神（根据姓名等信息）37 List<Goddess> list = gd.get("小溪", "139");38 //遍历结果集39 for(int i=0;i<list.size();i++){40 System.out.println(list.get(i).toString());41 }42 }43 44 }

[IMG_282](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

运行结果：

select \* from imooc\_goddess where user\_name like ? and mobile like ?

Goddess [id=1, userName=小溪, sex=1, age=22, birthday=2000-12-12, email=xiaoxi@163.com, mobile=13911111111, createUser=ADMIN, createDate=2015-01-08, updateUser=ADMIN, updateDate=2015-01-08, isDel=0]

 查询操作的进一步完善：

[IMG_283](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 //查询单个女神(根据姓名等信息去查询)----->使用Map存储条件信息；防止条件为空可加一条where 1=1 2 public List<Goddess> get(List<Map<String, Object>> params) throws SQLException{ 3 List<Goddess> result=new ArrayList<Goddess>(); 4 Connection con=DBUtil.getConnection();//首先拿到数据库的连接 5 StringBuffer sb=new StringBuffer(); 6 sb.append("select \* from imooc\_goddess where 1=1 ");//注意where 1=1 的小技巧 7 if(params !=null && params.size()>0){//先判断集合是否为空 8 //遍历集合 9 for(int i=0;i<params.size();i++){10 Map<String, Object> map=params.get(i);11 sb.append("and "+map.get("name")+" " +map.get("rela")+" "+map.get("value")+" ");//查询什么？关系是什么？以及值是什么，我们都可以通过参数传进来。12 }13 14 }15 //预编译sql语句16 PreparedStatement psmt = con.prepareStatement(sb.toString());17 System.out.println(sb.toString());18 //执行SQL语句19 /\*psmt.execute();\*///execute()方法是执行更改数据库操作（包括新增、修改、删除）;executeQuery()是执行查询操作20 ResultSet rs = psmt.executeQuery();//返回一个结果集21 Goddess g=null;22 //遍历结果集23 while(rs.next()){24 g=new Goddess();25 g.setId(rs.getInt("id"));26 g.setUserName(rs.getString("user\_name"));27 g.setAge(rs.getInt("age"));28 g.setSex(rs.getInt("sex"));29 //rs.getDate("birthday")获得的是java.sql.Date类型。注意：java.sql.Date类型是java.util.Date类型的子集，所以这里不需要进行转换了。30 g.setBirthday(rs.getDate("birthday"));31 g.setEmail(rs.getString("email"));32 g.setMobile(rs.getString("mobile"));33 g.setCreateUser(rs.getString("create\_user"));34 g.setCreateDate(rs.getDate("create\_date"));35 g.setUpdateUser(rs.getString("update\_user"));36 g.setUpdateDate(rs.getDate("update\_date"));37 g.setIsDel(rs.getInt("isdel"));38 39 result.add(g);//将结果封装到对象中40 }41 return result;42 }

[IMG_284](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

测试：

[IMG_285](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 //查询单个女神(根据姓名等信息去查询)----->使用Map存储条件信息 2 List<Map<String, Object>> params=new ArrayList<Map<String,Object>>(); 3 Map<String, Object> param=new HashMap<String, Object>(); 4 param.put("name", "user\_name"); 5 // param.put("rela", "="); 6 // param.put("value", "'小溪'");//注意加上单引号，因为这个变量是字符串的形式 7 param.put("rela", "like"); 8 param.put("value", "'%小溪%'");//注意加上单引号，因为这个变量是字符串的形式 9 params.add(param);10 param=new HashMap<String, Object>();11 param.put("name", "mobile");12 param.put("rela", "like");13 param.put("value", "'%139%'");14 params.add(param);15 List<Goddess> list1 = gd.get(params);16 //遍历结果集17 for(int i=0;i<list1.size();i++){18 System.out.println(list1.get(i).toString());19 }

[IMG_286](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

运行结果：

select \* from imooc\_goddess where 1=1 and user\_name like '%小溪%' and mobile like '%139%'

Goddess [id=1, userName=小溪, sex=1, age=22, birthday=2000-12-12, email=xiaoxi@163.com, mobile=13911111111, createUser=ADMIN, createDate=2015-01-08, updateUser=ADMIN, updateDate=2015-01-08, isDel=0]

这个查询的亮点在于：使用get(List<Map<String,Object>> params) 和 where 1=1；

jdbc传参数过多的时候：用List<Map<String,Object>> params双泛型作为参数.

我们查询的每一个条件都是一个键值对。键值对的数据结构就是Map。  
然后是因为查询的条件可能是多个，数量不确定，所以用List 比较好，所以就会是List<Map<key,value>>这样的数据结构了。

上面使用where 1=1的原因主要是下面的语句直接就跟着一个and。SQL语句后面需要跟着where，但不确定where后面是否有语句，就要加上1=1，保证语法不会出错。

一般用于构造动态SQL语句，"SELECT ... FROM ... WHERE 1=1 "+动态构造条件子句。

[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **4、搭建控制层**

控制层起到一个桥梁的作用，连接模型层和视图层。

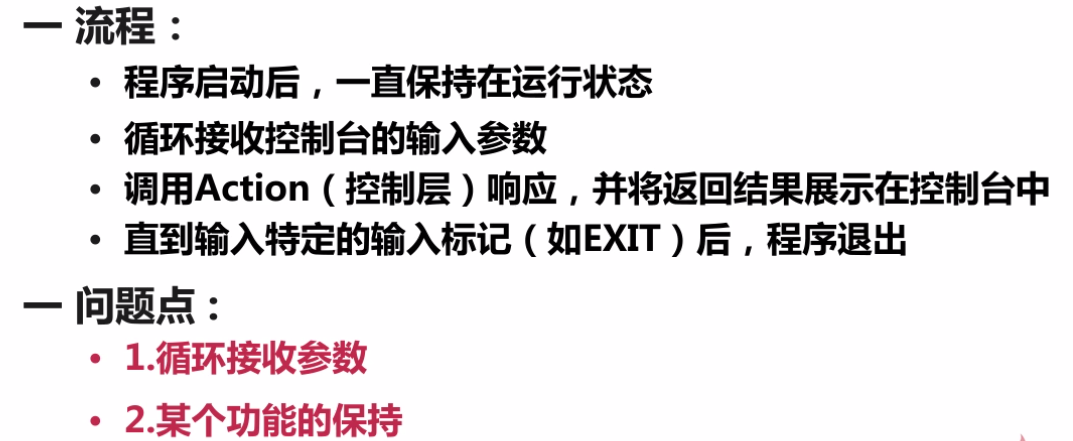
[IMG_287](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.controller; 2 3 import java.util.List; 4 import java.util.Map; 5 6 import com.study.dao.GoddessDao; 7 import com.study.model.Goddess; 8 9 public class GoddessAction1 {10 11 //添加女神12 public void add(Goddess goddess) throws Exception{13 GoddessDao dao=new GoddessDao();14 goddess.setSex(1);15 goddess.setIsDel(0);16 dao.addGoddess(goddess);17 }18 19 //修改女神20 public void edit(Goddess goddess) throws Exception{21 GoddessDao dao=new GoddessDao();22 dao.updateGoddess(goddess);23 }24 25 //删除女神26 public void del(Integer id) throws Exception{27 GoddessDao dao=new GoddessDao();28 dao.delGoddess(id);29 }30 31 //查询所有女神信息32 public List<Goddess> query() throws Exception{33 GoddessDao dao=new GoddessDao();34 return dao.query();35 }36 37 //查询单个女神信息(根据id)38 public Goddess get(Integer id) throws Exception{39 GoddessDao dao=new GoddessDao();40 return dao.get(id);41 }42 43 //查询单个女神信息（根据姓名等信息）44 public List<Goddess> get(List<Map<String, Object>> params) throws Exception{45 GoddessDao dao=new GoddessDao();46 return dao.get(params);47 }48 }

[IMG_288](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **5、搭建视图层**



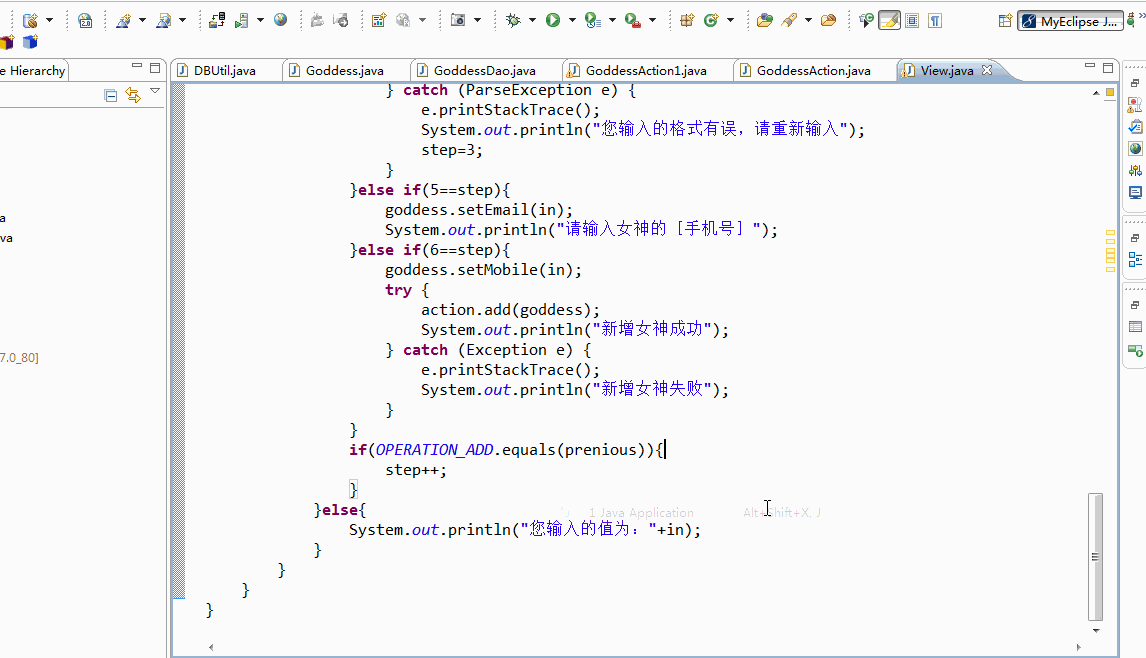
代码实例：

[IMG_290](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

1 package com.study.view; 2 3 import java.text.ParseException; 4 import java.text.SimpleDateFormat; 5 import java.util.Date; 6 import java.util.List; 7 import java.util.Scanner; 8 9 import com.study.controller.GoddessAction1; 10 import com.study.model.Goddess; 11 12 public class View { 13 14 //提示语 15 private static final String CONTEXT="欢迎来到女神禁区：\n" + 16 "下面是女神禁区的功能列表：\n" + 17 "[MAIN/M]:主菜单\n" + 18 "[QUERY/Q]:查看全部女神的信息\n" + 19 "[GET/G]:查看某位女神的详细信息\n" + 20 "[ADD/A]:添加女神信息\n" + 21 "[UPDATE/U]:更新女神信息\n" + 22 "[DELETE/D]:删除女神信息\n" + 23 "[SEARCH/S]:查询女神信息(根据姓名、手机号来查询)\n" + 24 "[EXIT/E]:退出女神禁区\n" + 25 "[BREAK/B]:退出当前功能，返回主菜单"; 26 27 //操作标记 28 private static final String OPERATION\_MAIN="MAIN"; 29 private static final String OPERATION\_QUERY="QUERY"; 30 private static final String OPERATION\_GET="GET"; 31 private static final String OPERATION\_ADD="ADD"; 32 private static final String OPERATION\_UPDATE="UPDATE"; 33 private static final String OPERATION\_DELETE="DELETE"; 34 private static final String OPERATION\_SEARCH="SEARCH"; 35 private static final String OPERATION\_EXIT="EXIT"; 36 private static final String OPERATION\_BREAK="BREAK"; 37 38 public static void main(String[] args) { 39 //输出提示 40 System.out.println(CONTEXT); 41 //怎么保持程序的一直运行:while(true){} 42 Scanner scanner=new Scanner(System.in);//接受控制台的输入 43 Goddess goddess=new Goddess(); 44 GoddessAction1 action=new GoddessAction1(); 45 46 String prenious=null;//标记，记住上一次请求 47 Integer step=1;//步骤标记 48 while(scanner.hasNext()){//scanner.hasNext()有数值时才进行循环，没有就不进行循环 49 String in=scanner.next().toString(); 50 if(OPERATION\_EXIT.equals(in.toUpperCase()) 51 || OPERATION\_EXIT.substring(0, 1).equals(in.toUpperCase())){ 52 System.out.println("您已成功退出女神禁区。"); 53 break;//退出当前while循环 54 }else if(OPERATION\_QUERY.equals(in.toUpperCase()) 55 ||OPERATION\_QUERY.substring(0, 1).equals(in.toUpperCase())){ 56 try { 57 List<Goddess> list=action.query(); 58 for (Goddess go : list) { 59 System.out.println(go.getId()+",姓名："+go.getUserName()); 60 } 61 } catch (Exception e) { 62 e.printStackTrace(); 63 } 64 }else if(OPERATION\_ADD.equals(in.toUpperCase()) 65 || OPERATION\_ADD.substring(0, 1).equals(in.toUpperCase()) 66 || OPERATION\_ADD.equals(prenious)){ 67 prenious=OPERATION\_ADD; 68 //新增女神 69 if(1==step){//如果是第一次进来 70 System.out.println("请输入女神的［姓名］"); 71 }else if(2==step){ 72 goddess.setUserName(in); 73 System.out.println("请输入女神的［年龄］"); 74 }else if(3==step){ 75 goddess.setAge(Integer.valueOf(in));//string转换为int型 76 System.out.println("请输入女神的［生日］，格式如：yyyy-MM-dd"); 77 }else if(4==step){ 78 SimpleDateFormat sf=new SimpleDateFormat("yyyy-MM-dd"); 79 Date birthday=null; 80 try { 81 birthday = sf.parse(in); 82 goddess.setBirthday(birthday); 83 System.out.println("请输入女神的［邮箱］"); 84 } catch (ParseException e) { 85 e.printStackTrace(); 86 System.out.println("您输入的格式有误，请重新输入"); 87 step=3; 88 } 89 }else if(5==step){ 90 goddess.setEmail(in); 91 System.out.println("请输入女神的［手机号］"); 92 }else if(6==step){ 93 goddess.setMobile(in); 94 try { 95 action.add(goddess); 96 System.out.println("新增女神成功"); 97 } catch (Exception e) { 98 e.printStackTrace(); 99 System.out.println("新增女神失败");100 }101 }102 if(OPERATION\_ADD.equals(prenious)){103 step++; 104 }105 }else{106 System.out.println("您输入的值为："+in);107 }108 }109 }110 }

[IMG_291](https://www.cnblogs.com/Qian123/p/javascript:void(0);" \o "复制代码)

运行结果:



[回到顶部](https://www.cnblogs.com/Qian123/p/5339164.html" \l "_labelTop)

## **三、JDBC总结**

1、JDBC的基本概念：JDBC是连接数据库的桥梁；

是由Java语言编写的类和接口组成，可以为多种数据库提供统一的访问。

2、采用MVC三层结构:

DB（数据库）、Model（模型层）、Control（控制层）、View（视图层）

模型层（DB + DAO）：封装了实现类  
控制层：负责调用模型层  
视图层调用控制层

3、JDBC各种连接方式的对比：

* 1、JDBC + ODBC桥的方式。特点：需要数据库的ODBC驱动，仅适用于微软的系统

这种方式，JDBC将调用传递给ODBC，然后ODBC再调用本地的数据库驱动代码。

* 2、JDBC + 厂商API的形式。特点：厂商API一般使用C编写

这种方式，JDBC将调用直接传递给厂商API的服务，然后在调用本地的数据库驱动。

* 3、JDBC + 厂商Database Connection Server + DataBase的形式。

特点：在JAVA与DATABASE之间架起了一台专门用于数据库连接的服务器（一般有数据库厂商提供）  
这种方式，JDBC将调用传递给中间服务器，中间服务器再将调用转换成数据库能够被调用的形式，在调用数据库服务器。中间增设数据库服务器能够提升效率，但不如直接操作数据库便捷。

* 4、JDBC + DATABASE的连接方式。

特点：这使得Application与数据库分开，开发者只需关心内部逻辑的实现而不需注重数据库连接的具体实现。（没有中间环节，是推荐方式！）