数据库基本配置

• 在src/jdbc/BaseDao.java中配置你的数据库信息

```
private final static String url = "jdbc:mysql://localhost:3306/goodbye?characterEncoding=UTF-8";
private final static String user = "root"; //数据库登录用户名
private final static String pw = "liuyuan.xing"; //数据库登录密码
```

执行 mysql -u root -p 然后输入你的数据库密码,进入mysql数据库

然后依次执行下列命令, 创建数据库和数据表

create database goodbye;

```
DROP TABLE IF EXISTS `likeme`;

CREATE TABLE `likeme` (
    `like_id` int(10) NOT NULL AUTO_INCREMENT,
    `like_topicId` int(10) DEFAULT NULL,
    `like_userId` int(10) DEFAULT NULL,

PRIMARY KEY (`like_id`),

KEY `like_topic_id` (`like_topicId`),

KEY `like_user_id` (`like_userId`),

CONSTRAINT `like_topic_id` FOREIGN KEY (`like_topicId`) REFERENCES `topic` (`topic_id`),

CONSTRAINT `like_user_id` FOREIGN KEY (`like_userId`) REFERENCES `user` (`user_id`)
);
```

```
DROP TABLE IF EXISTS `likeme`;

CREATE TABLE `likeme` (
   `like_id` int(10) NOT NULL AUTO_INCREMENT,
   `like_topicId` int(10) DEFAULT NULL,
   `like_userId` int(10) DEFAULT NULL,

PRIMARY KEY (`like_id`),

KEY `like_topic_id` (`like_topicId`),

KEY `like_user_id` (`like_userId`),

CONSTRAINT `like_topic_id` FOREIGN KEY (`like_topicId`) REFERENCES `topic` (`topic_id`),

CONSTRAINT `like_user_id` FOREIGN KEY (`like_userId`) REFERENCES `user` (`user_id`)
);
```

```
`user_like` varchar(255) DEFAULT NULL,

`user_label` varchar(255) DEFAULT NULL,

PRIMARY KEY (`user_id`),

UNIQUE KEY `user_loginname` (`user_loginname`)

)ENGINE=InnoDB DEFAULT CHARSET=gbk;
```

使用JDBC的PreparedStatement对数据库进行增删改查

关键代码

• 使用preparedstatement编写你要执行的sql语句,涉及非固定的内容的地方用占位符?来代替。如下例所示,当我们不知道要插入的内容是什么的时候,就可以用"?"来占位,然后后面代码给"?"赋值。

```
pst = conn.prepareStatement("insert into user values(null,?,?,?)");
```

- 你甚至可以把上例中的user表也用占位符来代替,使之变成一个更普通的模版。当然在实际开发中看情况考虑是 否需要将表也用占位符替代。
- 下面使用setString函数对你刚刚设置的?占位符进行赋值。它有两个参数,一个是?开始的序号(从1开始),一个是 string类型变量用来替换前面那个参数位置上的占位符。(类似的也有setInt,参见后面例子)

```
pst.setString(1, users.getUserName());
```

上面的两行代码懂了之后其他的就很简单了,其他部分的代码都是固定的(比如try catch捕获异常),你只需要替换这两行就可以了。下面我们看几个例子,方便你的理解,虽然代码可能略有不同,但是核心是一样的。

插入数据

```
public boolean isRegister(User users) {
     Connection conn = null;
     PreparedStatement pst = null;
     ResultSet rs = null;
     try {
         conn = super.getConn();
         pst = conn.prepareStatement("insert into user values(null,?,?,?)");
         pst.setString(1, users.getUserName());
         pst.setString(2, users.getPassWord());
         pst.setString(3, users.getNiceName());
         System.out.println(pst);
                                     //可以输出看看你的sql语句是否正确
         pst.executeUpdate();
         return true;
     } catch (SQLException e) {
                                  //输出异常
         e.printStackTrace();
         return false;
     } finally {
         super.close(conn, pst, rs);
 }
```

修改数据

```
public boolean XiuGai3(User user){
     boolean flag=true;
     Connection conn=null;
      PreparedStatement ps=null;
      String sql="update user set pwd=? where name=?";
      conn=DBConnUtil.getConn();
      try {
          ps=conn.prepareStatement(sql);
          ps.setString(1, user.getPwd());
          ps.setString(2, user.getName());
          int i= ps.executeUpdate();
          if(i==0){
              flag=false;
          }
      } catch (SQLException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
      }finally{
          DBConnUtil.closeAll(null, ps, conn);
      return flag;
 }
```

删除数据

```
public boolean ShanChu3(int id){
      boolean flag=true;
     Connection conn=null;
      PreparedStatement ps=null;
      String sql="delete from user where id=?";
      conn=DBConnUtil.getConn();
      try {
          ps=conn.prepareStatement(sql);
          ps.setInt(1, id);
          int i=ps.executeUpdate();
          if(i==0){
              flag=false;
          }
      } catch (SQLException e) {
          // TODO Auto-generated catch block
          e.printStackTrace();
      }finally{
          DBConnUtil.closeAll(null, ps, conn);
      return flag;
  }
```

查询数据

```
public List<User> ChaKan3(){
   List<User> list= new ArrayList<User>();
   Connection conn=null;
   PreparedStatement ps=null;
   ResultSet rs=null;
   String sql="select * from user";
   conn=DBConnUtil.getConn();
   try {
```

```
ps=conn.prepareStatement(sql);
    rs=ps.executeQuery();
    while(rs.next()){
        User user= new User();
        user.setName(rs.getString("name"));
        list.add(user);
    }
} catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}finally{
    DBConnUtil.closeAll(rs, ps, conn);
}
    return list;
}
```