Oracle安装 ：Win7安装的时候会出现的问题，安装环境不能检测通过，需要以系统管理员的身份去运行我们Oracle的安装文件

今天课程：

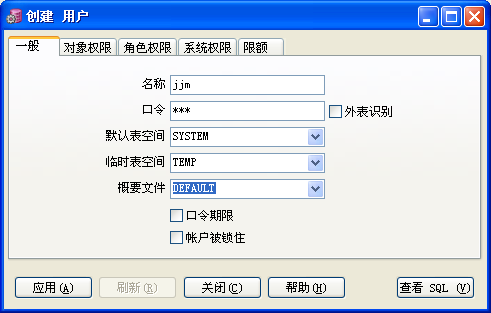
Oracle数据库的基本操作

Java连接Oracle数据库（JDBC）

Oracle的其他知识（Oracle中的一些函数、视图、PL/SQL程序设计语言、存储过程、函数、触发器）

一、Oracle的基本操作

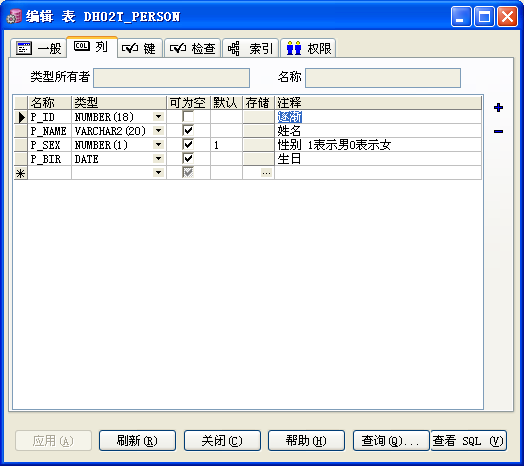
·创建用户

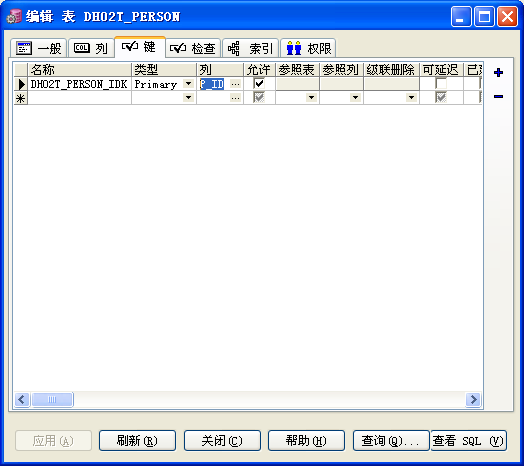




·创建表格：







创建表格的sql：

|  |
| --- |
| *-- Create table*  create table DH02T\_PERSON  (  P\_ID NUMBER(18) not null,  P\_NAME VARCHAR2(20),  P\_SEX NUMBER(1) default 1,  P\_BIR DATE  )  tablespace SYSTEM  pctfree 10  pctused 40  initrans 1  maxtrans 255  storage  (  initial 64K  minextents 1  maxextents unlimited  );  *-- Add comments to the columns*  comment on column DH02T\_PERSON.P\_ID  is '逐渐';  comment on column DH02T\_PERSON.P\_NAME  is '姓名';  comment on column DH02T\_PERSON.P\_SEX  is '性别 1表示男0表示女';  comment on column DH02T\_PERSON.P\_BIR  is '生日';  *-- Create/Recreate primary, unique and foreign key constraints*  alter table DH02T\_PERSON  add constraint DH02T\_PERSON\_IDK primary key (P\_ID)  using index  tablespace SYSTEM  pctfree 10  initrans 2  maxtrans 255  storage  (  initial 64K  minextents 1  maxextents unlimited  ); |

·新增数据

语法：insert into tableName （字段1，字段2.。。）values（值1，值2.。。）

|  |
| --- |
| insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (1,'朱永德',1,to\_date('1990-12-12','yyyy-mm-dd')); |

·查询数据

|  |
| --- |
| select \* from dh02t\_person t;  select p\_id,p\_name,p\_sex,p\_bir from dh02t\_person;  *--发现性别没有出现男女*  select p\_id,p\_name,decode(p\_sex,1,'男',0,'女'),p\_bir from dh02t\_person;  *--还可以对数据库表和字段从新命名*  select t.p\_id,t.p\_name,decode(t.p\_sex,1,'男',0,'女') sex ,t.p\_bir from dh02t\_person t;  *--还可以进行排序查询*  *--asc ：升序排列*  *--desc：降序排列*  select t.p\_id,t.p\_name,decode(t.p\_sex,1,'男',0,'女') sex ,t.p\_bir from dh02t\_person t order by t.p\_bir asc;*--默认升序*  select t.p\_id,t.p\_name,decode(t.p\_sex,1,'男',0,'女') sex ,t.p\_bir from dh02t\_person t order by t.p\_bir desc;*--降序*  *--如果某个字段为空，在查询的时候需要赋默认值，如果现在性别为空，查询数据我们给默认值为0*  select p\_id,p\_name,nvl(p\_sex,0) sex,p\_bir from dh02t\_person;  *--现在我还是需要将男女在查询中显示出现*  select p\_id,p\_name,decode(nvl(p\_sex,0) ,1,'男',0,'女') sex,p\_bir from dh02t\_person; |

·修改操作：

update tableName set 字段1=值1，。。。。。where。。。。

|  |
| --- |
| update dh02t\_person set p\_bir = to\_date('1989-12-31','yyyy-mm-dd') where p\_name='张军师'; |

·删除操作：

|  |
| --- |
| delete dh02t\_person where p\_id = 1; |

二、JDBC

·JDBC的分类

·JDBC的基本使用（基本操作，预处理、批处理）

·事务处理

JDBC的概念：Java DataBase Connective，是java连接数据库，是java专门用于数据连接的标准，在这整个JDBC操作中，提供了大量的接口，针对不同的数据库生产厂商，只要想使用java进行连接操作，就必须要对这写接口有所支持。

·JDBC的分类：

·JDBC-ODBC：叫做JDBC桥连接，是sun公司在JDK中提供的最标准的一套JDBC的操作，使用的时候是JDBC—ODBC——数据库，就是说中间需要使用一个ODBC的操作，这种操作会降低连接速度，所以在以后的开发中，是绝对不会去使用这一套连接标准的。

·JDBC连接：是由各个数据库生产厂商提供的数据库驱动程序进行连接

·JDBC网络连接：主要是通过网络连接到数据库

JDBC连接数据库的步骤：

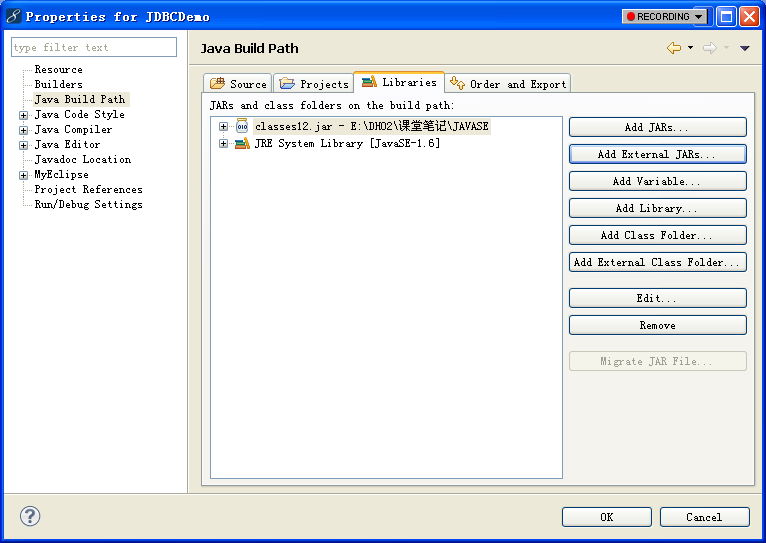
·加载驱动程序

·通过Connection接口和DriverManager类完成数据库操作

·进行数据库操作，通过Statement、PreparedStatement，ResultSet完成数据库操作

·关闭数据库连接，在实际开发中，数据库连接是非常有限的，操作完成之后就必须要关闭，不关闭会造成数据库的崩溃。

JDBC的准备工作：

·加载驱动程序（D:\oracle\product\10.2.0\db\_1\jdbc\lib）

范例：进行数据库连接

|  |
| --- |
| **package** com.wanczy.jdbcDemo;  **import** java.sql.Connection;  **import** java.sql.DriverManager;  **import** java.sql.SQLException;  **public** **class** JDBCDemo01 {  //配置驱动程序  **public** **static** **final** String *DRIVER* = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  **public** **static** **final** String *URL* = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  **public** **static** **final** String *USERNAME* = "jjm";  //数据库密码  **public** **static** **final** String *PASSWORD* = "jjm";  **public** **static** **void** main(String[] args) {  Connection con = **null**;  **try** {  Class.*forName*(*DRIVER*);//本程序中加载驱动  con = DriverManager.*getConnection*(*URL*,*USERNAME*,*PASSWORD*);  System.*out*.println(con);  } **catch** (Exception e) {  e.printStackTrace();  }**finally**{  **try** {  con.close();  } **catch** (SQLException e) {  e.printStackTrace();  }  }  }  } |

连接对象正常打印，说明数据库连接成功。

·java连接数据库的基本操作

·范例：新增操作，现在我们向dh02T\_PERSON表中添加一笔数据

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.sql.Statement;  public class JDBCDemo02 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  Statement stat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  stat = con.createStatement();//取得数据库操作对象  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (4,'方倩',0,to\_date('1990-10-22','yyyy-mm-dd'))";  stat.execute(sql);//操作sql  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  stat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

·范例：修改操作

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.sql.Statement;  public class JDBCDemo03 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  Statement stat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  stat = con.createStatement();//取得数据库操作对象  String sql = "update dh02t\_person set p\_bir = to\_date('1989-12-31','yyyy-mm-dd') where p\_name='方倩'";  stat.execute(sql);//操作sql  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  stat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

·范例：删除操作

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.sql.Statement;  public class JDBCDemo04 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  Statement stat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  stat = con.createStatement();//取得数据库操作对象  String sql = "delete dh02t\_person where p\_name='刘雪峰'";  stat.execute(sql);//操作sql  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  stat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

·数据库查询操作，在数据库查询中，使用ResultSet接口去接收数据库查询结果，ResultSet表示的是数据库查询结果的数据集。

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.Date;  import java.sql.DriverManager;  import java.sql.ResultSet;  import java.sql.SQLException;  import java.sql.Statement;  public class JDBCDemo05 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  Statement stat = null;  ResultSet res = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  stat = con.createStatement();//取得数据库操作对象  String sql = "select p\_id,p\_name,decode(nvl(p\_sex,0),0,'女',1,'男') sex,p\_bir from dh02t\_person";//在开发过程中，在查询数据库的时候一般不会去使用\*查询全部，而是需要什么字段就查询什么字段  res = stat.executeQuery(sql);//进行查询操作  while(res.next()){//将光标移向下一行，并且判断是否有下一行  // int pId = res.getInt("p\_id");//通过字段名称取得查询的值  // String pName = res.getString("p\_name");//通过字段名称取得查询的值  // String pSex = res.getString("sex");//通过字段名称取得查询的值  // Date pBir = res.getDate("p\_bir");//通过字段名称取得查询的值    int pId = res.getInt(1);//通过字段位置取得查询的值  String pName = res.getString(2);//通过字段位置取得查询的值  String pSex = res.getString(3);//通过字段位置取得查询的值  Date pBir = res.getDate(4);//通过字段位置取得查询的值  System.out.println(pId+"\t" + pName+"\t" + pSex+"\t" + pBir);  }  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  res.close();  stat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

·预处理：如果数据库是新增操作，我们首先在数据库中执行这条语句，但是新增的数据为空，后面再一个一个的单独设置。

PreparedStatement:这个节接口是Statement的子接口，在实际开发中最为广泛使用，是采用预处理的形式进行数据库操作的。

范例：预处理的好处

·不使用于处理

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.SQLException;  import java.sql.Statement;  public class JDBCDemo06 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  Statement stat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  stat = con.createStatement();//取得数据库操作对象  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (4,'mr'smith',0,to\_date('1990-10-22','yyyy-mm-dd'))";  stat.execute(sql);//操作sql  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  stat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

不使用预处理在新增的时候名字为mr'smith不能进行保存，而且出现异常

·使用预处理

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.PreparedStatement;  import java.sql.SQLException;  public class JDBCDemo07 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  PreparedStatement pstat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (?,?,?,to\_date(?,'yyyy-mm-dd'))";  pstat = con.prepareStatement(sql);//取得数据库操作预处理对象  pstat.setInt(1, 5);//设置第一个？的值为int类型  pstat.setString(2, "mr'smith");  pstat.setInt(3, 1);  pstat.setString(4, "1987-01-31");  pstat.execute();//不需要绑定sql，直接运行  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  pstat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

·采用预处理进行查询：

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.Date;  import java.sql.DriverManager;  import java.sql.PreparedStatement;  import java.sql.ResultSet;  import java.sql.SQLException;  public class JDBCDemo08 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  PreparedStatement stat = null;  ResultSet res = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象    String sql = "select p\_id,p\_name,decode(nvl(p\_sex,0),0,'女',1,'男') sex,p\_bir from dh02t\_person where p\_name=?";//在开发过程中，在查询数据库的时候一般不会去使用\*查询全部，而是需要什么字段就查询什么字段  stat = con.prepareStatement(sql);//取得数据库操作对象  stat.setString(1, "方倩");  res = stat.executeQuery();//进行查询操作  while(res.next()){//将光标移向下一行，并且判断是否有下一行  // int pId = res.getInt("p\_id");//通过字段名称取得查询的值  // String pName = res.getString("p\_name");//通过字段名称取得查询的值  // String pSex = res.getString("sex");//通过字段名称取得查询的值  // Date pBir = res.getDate("p\_bir");//通过字段名称取得查询的值    int pId = res.getInt(1);//通过字段位置取得查询的值  String pName = res.getString(2);//通过字段位置取得查询的值  String pSex = res.getString(3);//通过字段位置取得查询的值  Date pBir = res.getDate(4);//通过字段位置取得查询的值  System.out.println(pId+"\t" + pName+"\t" + pSex+"\t" + pBir);  }  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  res.close();  stat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

批处理：可想而知，就是一次性执行多条语句

范例：进行数据插入，一次性插入10笔数据

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.PreparedStatement;  import java.sql.SQLException;  public class JDBCDemo09 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  PreparedStatement pstat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (?,?,?,to\_date(?,'yyyy-mm-dd'))";  pstat = con.prepareStatement(sql);//取得数据库操作预处理对象  for (int i = 0; i < 10 ;i++) {  pstat.setInt(1, i+10);//设置第一个？的值为int类型  pstat.setString(2, "刘春燕"+i);  pstat.setInt(3, 0);  pstat.setString(4, "1987-01-31");  pstat.addBatch();//加入批处理  }  pstat.executeBatch();//执行批处理  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  pstat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

事务管理：

在开发过程中，批处理执行到某一个sql的很有可能会出现错误，如果出现错误之后，我们希望的是此批处理所有的操作都回滚掉。但是现在我们试一下能不能进行回滚：

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.PreparedStatement;  import java.sql.SQLException;  public class JDBCDemo10 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  PreparedStatement pstat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (?,?,?,to\_date(?,'yyyy-mm-dd'))";  pstat = con.prepareStatement(sql);//取得数据库操作预处理对象  for (int i = 0; i < 10 ;i++) {  if(i==5){  pstat.setInt(1, 5);//设置第一个？的值为int类型  }else{  pstat.setInt(1, i+10);//设置第一个？的值为int类型  }  pstat.setString(2, "刘春燕"+i);  pstat.setInt(3, 0);  pstat.setString(4, "1987-01-31");  pstat.addBatch();//加入批处理  }  pstat.executeBatch();//执行批处理  } catch (Exception e) {  e.printStackTrace();  }finally{  try {  pstat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

发现在出现问题之前的所有的操作都会直接写入到数据库中，但是如果这些数据是关联的，我们希望是如果有一笔数据插入出现失败，则所有的数据都回滚掉，这个时候就需要使用我们的事务处理，事务处理即提交和回滚，在我们之前的JDBC的操作中，所有的提交和回滚都是自动进行的，我们进行事务处理就是将提交和回滚进行手动的管理。

事务管理步骤：

·取消自动提交（回滚）：void setAutoCommit(boolean autoCommit) throws SQLException，设置是否自动提交，设置为false时，取消自动提交。

·进行手动提交:commit()

·进行手动回滚：rollback（）；

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.PreparedStatement;  import java.sql.SQLException;  public class JDBCDemo10 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  PreparedStatement pstat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  con.setAutoCommit(false);//取消自动提交  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (?,?,?,to\_date(?,'yyyy-mm-dd'))";  pstat = con.prepareStatement(sql);//取得数据库操作预处理对象  for (int i = 0; i < 10 ;i++) {  if(i==5){  pstat.setInt(1, 5);//设置第一个？的值为int类型  }else{  pstat.setInt(1, i+10);//设置第一个？的值为int类型  }  pstat.setString(2, "刘春燕"+i);  pstat.setInt(3, 0);  pstat.setString(4, "1987-01-31");  pstat.addBatch();//加入批处理  }  pstat.executeBatch();//执行批处理  con.commit();  } catch (Exception e) {  try {  con.rollback();  } catch (SQLException e1) {  e1.printStackTrace();  }  e.printStackTrace();  }finally{  try {  pstat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

发现如果出现异常一笔数据都不会写入到数据中。

|  |
| --- |
| package com.wanczy.jdbcDemo;  import java.sql.Connection;  import java.sql.DriverManager;  import java.sql.PreparedStatement;  import java.sql.SQLException;  public class JDBCDemo11 {  //配置驱动程序  public static final String DRIVER = "oracle.jdbc.driver.OracleDriver";  //配置数据库连接池  public static final String URL = "jdbc:oracle:thin:@172.26.64.44:1521:orcl";  //数据库用户名  public static final String USERNAME = "jjm";  //数据库密码  public static final String PASSWORD = "jjm";  public static void main(String[] args) {  Connection con = null;  PreparedStatement pstat = null;  try {  Class.forName(DRIVER);//本程序中加载驱动  con = DriverManager.getConnection(URL,USERNAME,PASSWORD);//取得数据库连接对象  con.setAutoCommit(false);//取消自动提交  String sql = "insert into dh02t\_person (p\_id,p\_name,p\_sex,p\_bir) values (?,?,?,to\_date(?,'yyyy-mm-dd'))";  pstat = con.prepareStatement(sql);//取得数据库操作预处理对象  for (int i = 0; i < 10 ;i++) {  pstat.setInt(1, i+10);//设置第一个？的值为int类型  pstat.setString(2, "刘春燕"+i);  pstat.setInt(3, 0);  pstat.setString(4, "1987-01-31");  pstat.addBatch();//加入批处理  }  pstat.executeBatch();//执行批处理  con.commit();  } catch (Exception e) {  try {  con.rollback();  } catch (SQLException e1) {  e1.printStackTrace();  }  e.printStackTrace();  }finally{  try {  pstat.close();  con.close();  } catch (SQLException e) {  e.printStackTrace();  }  }  }  } |

没有错误的时候，数据全部写入到数据库中。

总结：

·预处理

·事务管理

练习：接口和数据库的练习，新建一个表，进行增删改查，保存到数据库。