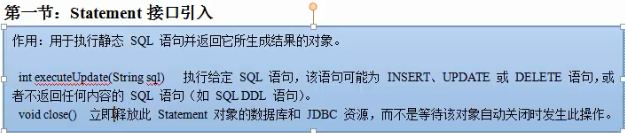
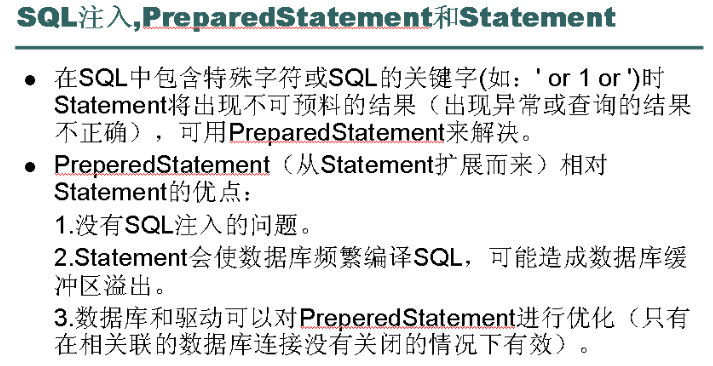
使用Statement和PreparedStatement进行增删改操作

1. **Statement是一个接口**，存在于**java.sql包**中，没有实现类。和Connection接口类似。
2. **增删改查又可以记为CRUD。create、read、update、delete。**
3. **Statement介绍**：



1. Statement和PreparedStatement的区别：



1. Statement中的方法介绍：
2. getXxxx方法：
   1. **getConnection方法**：返回一个**Connection接口**的实现对象。



1. **setXxxx方法**：
2. **executeUpdate方法**：返回值为**int值**，执行的语句数目。
3. PreparedStatement中的执行方法：

（1） execute()方法：执行语句，返回Boolean值；

（2）executeUpdate()方法：执行更新语句，返回更新的行数。

（3）executeQuery()方法：执行查询语句，返回ResultSet结果集。

Statement中的执行方法：必须有sql语句参数。

（1） execute(String sql)方法：执行语句，返回Boolean值；

（2）executeUpdate(String sql)方法：执行更新语句，返回更新的行数。

（3）executeQuery(String sql)方法：执行查询语句，返回ResultSet结果集。

PreparedStatement和Statement中的执行语句的区别：

前者是在创建psm时就将sql语句注入，而后者Statement是在执行时将**sql语句**注入。由于psm是statement的子类，所以在执行execute语句时，如果psm的execute中传入sql语句，则执行的是父类statement中的方法。即Connection中的**createStatement方法**是空参数，而prepareStatement方法由**参数sql**语句。

1. 利用**Statement接口**实现**增删改操作**示例：
   * + 1. **添加操作：insert into**

String sql = "insert into t\_book values("+book.id+",'"+book.bookName+"',"+book.price+",'"+book.author+"',"+book.bookId+")";

DbUtil du = new DbUtil();//自定义的工具类，连接数据库，获取Connection接口的实现对象

Connection con = du.getConnection();

String sql= "insert into t\_book values(null,'北京交通大学',23,'赵红',7)";

Statement sm = con.createStatement();

sm.executeUpdate(sql); sm.close();

con.close();//相当于du.close(con);

* + 1. 根据**变量**封装sql语句：

private static int addBook(int id,String bookName,int price,String author,int bookId) {String sql = "insert into t\_book values("+id+",'"+bookName+"',"+price+",'"+author+"',"+bookId+")";

* + 1. 把**book**封装成对象：

Book book = new Book(94448,"java",32,"lisi",43);

int num = addBook(**book)**;

* + - 1. **更新操作：update**

String sql = "update new\_table set id=?,name=? where id=?";

String sql = "update t\_book set bookName = '"+book.bookName+"',price ="+book.price+",author = '"+book.author+"',bookTypeId = "+book.bookId+" where id = "+book.id;

* + - 1. **删除操作：delete**

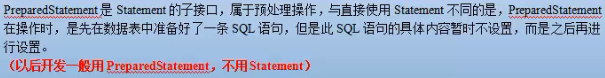
String sql = **"delete from t\_book where id ="+id;**

sm.executeUpdate(sql);

**for**(**int** i = 10;i>0;i--)

**int** num = *deleteBook*(id++);

1. 利用**PreparedStatement接口**实现**增删改操作：**



**示例：**

1. **增添：insert语句：**

**String sql= "insert into t\_book values(null,?,?,?,?)";**

PreparedStatement psm = con.prepareStatement(sql);

psm.setString(1,"北京交通大学");

psm.setInt(2,23);

psm.setString(3,"日中天");

psm.setInt(4,44);

int num = psm.executeUpdate();

进一步，把Book封装成对象：

String sql = "insert into t\_book values(null,?,?,?,?)";

PreparedStatement psm = con.prepareStatement(sql);

psm.setString(1,book.getBookName());

psm.setInt(2,book.getPrice());

psm.setString(3,book.getAuthor());

psm.setInt(4,book.getBookId());

int num = psm.executeUpdate();

1. **修改操作：update语句**：

String sql = "update t\_book set bookName =?,price =?,author = ?,bookTypeId =? where id =?";

PreparedStatement psm = con.prepareStatement(sql);

psm.setString(1,book.getBookName());

psm.setInt(2,book.getPrice());

psm.setString(3,book.getAuthor());

psm.setInt(4,book.getBookId());

psm.setInt(5,book.getId());

int num = psm.executeUpdate();

1. **删除语句：delete语句**

String sql = "delete from t\_book where id = ?";

PreparedStatement psm = con.prepareStatement(sql);

psm.setInt(1,book.getId());

int num = psm.executeUpdate();

1. 从上面可以看出PreparedStatement接口比Statement接口方便了很多，**Statement的sql语句**必须使用+号拼接成一个完整的语句，而PreparedStatement语句只需要把必要的东西封装成一个sql语句，对于语句中的内容利用**setXXX方法**添加即可。如：

Statement的sql语句String sql = "insert into t\_book values("+id+",'"+bookName+"',"+price+",'"+author+"',"+bookId+")";

PreparedStatement的sql语句为：String sql = "update t\_book set bookName =?,price =?,author = ?,bookTypeId =? where id =?";

PreparedStatement psm = con.prepareStatement(sql);

psm.setString(1,book.getBookName());

psm.setInt(2,book.getPrice());

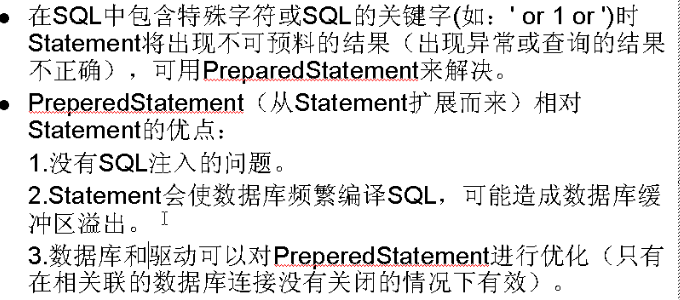
psm.setString(3,book.getAuthor());

psm.setInt(4,book.getBookId());

psm.setInt(5,book.getId());

注意：实际工程开发中，都使用**PreparedStatement接口**。

1. PreparedStatement比Statement的效率更高。



1. 利用PreparedStatement插入时间和日期的方法：

由于java本身和不同的数据库的日期和时间类型不一致，所以需要进行转换，具体转换的话，利用的就是相同毫秒值，由于都是从1970年1月1日开始计时的，所以，通过java的Date对象获取毫秒值，作为参数传入相应的数据库**提供的时间类**从而new出数据库的新的对象。

如：

String sql = "insert into table\_ttt(id,name,birthday) values(?,?,?)";

PreparedStatement ps = con.prepareStatement(sql);

**Date birthday = new Date(11111111);**

**ps.setDate(3,new java.sql.Date(birthday.getTime()));**

1. 获取时间和日期：

Connection con = du.getConnection();

String sql = "select \* from table\_ttt";

PreparedStatement ps = con.prepareStatement(sql);

ResultSet rs = ps.executeQuery();

while(rs.next()) {

System.out.println(rs.getInt("id")+rs.getString("name")+**new Date(rs.getDate("birthday").getTime()));//通过java中util包中的Date创建新的类，从而输出java规定的字符串形式**

}

du.close(con,ps);