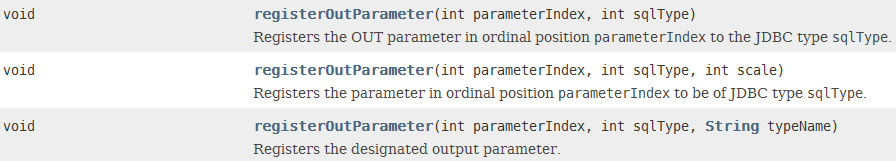
利用CallableStatement接口调用存储过程和java.sql.Types

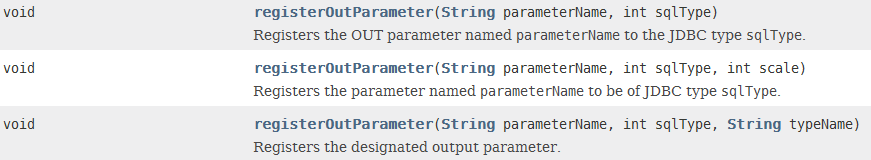
1. java.sql.Types是一个**类**，存在于**java.sql包**中，里面存储的都是字段，包含了SQL中涉及的数据类型，每个数据类型都是一个常量int值代替，且这些字段都是**静态的**，直接利用**类名Types**进行调用。主要有int、bigint、float、double、varchar、bit、binary、char、blob、Clob等等。
2. **CallableStatement接口**是PreparedStatement接口的子接口，而**PreparedStatement接口**又是接口**Statement的子接口**。故CallableStatement接口是对PreparedStatement接口**功能的扩展**。
3. **CallableStatement接口的方法介绍：（Field和Statement接口一样）**
4. **CallableStatement接口除了和PreparedStatement接口有相同的getXxxx方法和setXxxx方法外，还具有自己特有的方法，最重要的是registerOutParameter方法。**
5. **registerOutParameter方法：**

**sqlType利用int值表示，当然还可以利用SQLType类型表示。**

1. **利用索引：**



1. **利用标签：**



1. 其他方法和PreparedStatement方法类似。
2. **CallableStatement接口**的作用：主要是调用**数据库中的存储过程（**SQL stored procedures**）**，可以利用**registerOutParameter方法**获取**存储过程的返回值**。
3. **CallableStatement接口**的使用：



1. 在数据库中创建存储过程，并调用：

DELIMITER &&

CREATE PROCEDURE getBookNameById(IN bookId INT,OUT bName VARCHAR(30))

BEGIN

SELECT bookName INTO bName FROM t\_book WHERE id = bookId;

END

&&

DELIMITER ;

CALL getBookNameById(3,@bookName);

SELECT @bookName;

1. 在**Eclipse中**调用存储过程：

private static DbUtil du = new DbUtil();

public static String getBookNameById (int bookId)throws Exception {

Connection con = du.getConnection();

String sql = "{CALL getBookNameById(?,?)}";

CallableStatement csm = con.prepareCall(sql);

csm.setInt(1,bookId);

csm.registerOutParameter(2,Types.VARCHAR);

csm.execute();

String bookName = csm.getString("bName");

return bookName;

}

调用：

System.out.println("名称为 "+getBookNameById(3));//返回。