**Bài 13**

**JDBC QUERYING & TRANSACTION**

Module: JWBD

Kiểm tra bàitrước

Hỏivàtraođổivềcáckhókhăngặpphảitrongbài“JDBC & CRUD"

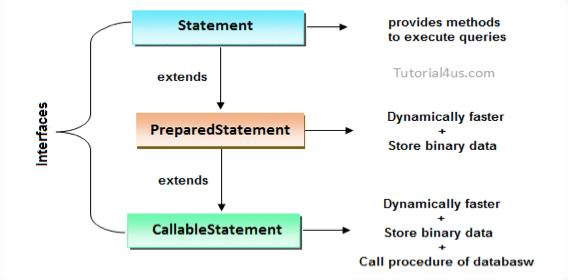
Tómtắtlạicácphầnđãhọctừbài“JDBC & CRUD”

# Mụctiêu

* Sử dụng được JDBC gọiStored Procedures
* Sử dụng được JDBCTransaction

## CallableStatement interface trong JDBC

• CallableStatementInterfaceđượcsửdụngđểthựcthiStored Procedure



## Đối tượng CallableStatement

* CallableStatement là đối tượng kế thừa từ đối tượng PreparedStatement
* CallableStatementcócáctínhchấtcủa1PreparedStatement
* Sử dụng CallableStatement để gọi các stored procedure

## Cú pháp tạo CallableStatement

Chúng ta tạo 1 CallableStatement bằng cú pháp sau:

CallableStatement prepareCall(String sql) throws SQLException Hoặc:

CallableStatement prepareCall(String sql, int resultSetType, int resultSetConcurrency) throws SQLException

# Gọi Stored Procedure

Chúngtatriệugọicácstoredprocedurebằngcáccúphápsau:

* Đốivớiprocedurekhôngcóthamsố:{callprocedure\_name}
* Đốivớiprocedurecóthamsố:{callprocedure\_name(?,?,…)}
* Đốivớiprocedurecógiátrịtrảvề(function):{?=call procedure\_name(?,?,…)} Vídụ:

CallableStatement cstmt = con.prepareCall(“{callprocedureName(?, ?)}”);

Trongđócácdấu?thaythếchocácthamsốkiểuIN,OUT,INOUT

# Tham số kiểu IN

* Tham số IN là một tham số mà giá trị của nó chưa được biết khi lệnh SQL đượctạo.
* Đối với tham số kiểu IN, ta sử dụng phương thức setXXX của đối tượngPrepareStatementđểchuyểnthamsốvào.

Vídụ: cstmt.setString(1, “c%”);

# Tham số kiểu OUT

* ThamsốOUTlàmộtthamsốmàgiátrịcủanóđượccungcấpbởi lệnh SQLnó trả về.Giá trị của chúng được thu nhận thông qua phươngthứcgetXXX().
* Trongđó,kiểucủatấtcảthamsốOUTphảiđượcđăngkýtrướckhi thựcthistoredprocedure.

Vídụ:

stmt.registerOutParameter(1,java.sql.Types.INTEGER);

# Tham số kiểu INOUT

Tham số INOUT là một tham số mà cung cấp cả giá trị input và output.BạngắnkếtcácbiếnvớiphươngthứcsetXXX()vàthunhận giátrịvớiphươngthứcgetXXX().

# Đóng đối tượng CallableStatement

Để đóng đối tượng CallableStatement, bạn sử dụng phương thứcclose()

Vídụ:

CallableStatement cstmt = null;

try{

String SQL = "{call getTenSV (?,?)}"; cstmt = conn.prepareCall(SQL);

. ..

}

catch (SQLException e){

. ..

}

finally {

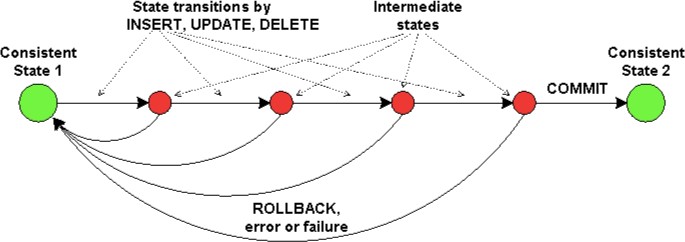
cstmt.close();

}

**Transaction là gì ?**

* Transaction (giao tác) là một tiến trình xử lý, có điểm bắt đầu và điểm kết thúc
* Transaction gồm nhiều phép thực thi nhỏ, trong đó mỗi phép thực thi được thực thi một cách tuần tự và độc lập theo nguyên tắc là tất cả thành công hoặc một phép thực thi thất bại thì cả tiến trình thất bại.

# Transaction (minh hoạ)



# Các thuộc tính ACID

Các thuộc tính ACID miêu tả rõ ràng nhất về Transaction. 4 thuộc tính này bao gồm Atomicity, Consistency, Isolation và Durability, trong đó:

* **Atomicity** nghĩa là tất cả thành công hoặc không.
* **Consistency** bảo đảm rằng tính đồng nhất của dữ liệu.
* **Isolation** bảo đảm rằng Transaction này là độc lập với Transaction khác.
* **Durability** nghĩa là khi một Transaction đã được ký thác thì nó sẽ vẫn tồn tại như thế cho dù xảy ra các lỗi, …

# Transaction trong JDBC

Trong JDBC, Connection Interface cung cấp các phương thức sau để quản lý transaction:

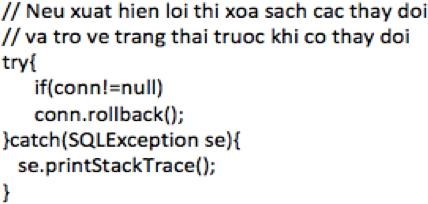
* void setAutoCommit(boolean status)
* void commit()
* void rollback()
* setSavepoint(String ten\_cua\_savepoint)
* releaseSavepoint(Savepoint ten\_cua\_savepoint)
* rollback (String ten\_cua\_savepoint)

# Phương thức commit()

Phương thức commit() được sử dụng khi bạn đã thực hiện các thay đổi với cơ sở dữ liệu và muốn ký thác các thay đổi đó

# Phương thức rollback()

Phươngthứcrollback()sửdụngđểxóacácthayđổiđãđượcthực hiệntrướcđóđểquayvềtrạngtháitrướckhithựchiệnthayđổikhi thấyrằngcólỗixảyra



# Tổng kết

* CallableStatement Interface được sử dụng để thực thi Stored Procedure
* CallableStatement là đối tượng kế thừa từ đối tượng PreparedStatement
* CallableStatement có các tính chất của 1 PreparedStatement (tức có các tính chất của Statement(public interface CallableStatement extends PreparedStatement)
* Transaction (giao tác) là một tiến trình xử lý, có điểm bắt đầu và điểm kết thúc
* Transaction gồm nhiều phép thực thi nhỏ, trong đó mỗi phép thực thi được thực thi một cách tuần tự và độc lập theo nguyên tắc là tất cả thành công hoặc một phép thực thi thất bại thì cả tiến trình thất bại.

**Hướngdẫn**

Hướng dẫnlàmbàithựchànhvàbàitập

Chuẩnbịbàitiếptheo:Packaging Dependences Management



R a i s i n g t h e b a r