LẬP TRÌNH JAVA

Kết nối cơ sở dữ liệu (JDBC)

Nguyễn Thị Hồng Anh – Đỗ Ngọc Như Loan

PreparedStatement

- PreparedStatement kế thừa từ Statement.
- Sử dụng để thực hiện các câu truy vấn SQL động hoặc có tham số
- Thường dùng khi cần thực hiện nhiều câu lệnh SQL có cấu trúc tương tự nhau, chỉ có giá trị là thay đổi. Khi đó PreparedStatement được dùng để soạn trước câu lệnh có sẵn cấu trúc cần thiết, giá trị sẽ được đưa vào như những đối số khi câu lệnh được thực thi.
- Giúp tránh SQL Injection.

Ví dụ 1 - PreparedStatement

```
public void prepareStatement1(){
    String insertQuery = "INSERT INTO SinhVien
VALUES(?,?,?)";
    if (con != null){
        try{
            PreparedStatement prest =
    con.prepareStatement(insertQuery);
            prest.setInt(1, 30);
            prest.setString(2, “Minh");
            prest.setInt(3, 2001);
            prest.executeUpdate();
        } catch(SQLException e){
            System.out.println(e); }
} }
```

Ví dụ 2 - PreparedStatement

```
public void prepareStatement2(){
    String insertQuery = "INSERT INTO SinhVien VALUES(?,?,?)";
    int n = 4;
    int mssv[] = {40,41,42,43};
    String hoten[] = {“Lan”,”Hong”,”Son”,”Duy”};
    int namsinh[] = {2000,2002,2001,2002};
    if (con != null){
        try{
            PreparedStatement prest = con.prepareStatement(insertQuery);
```

Ví dụ 2 - PreparedStatement

```
            for (int i = 0; i<n; i++) {
                prest.setInt(1, mssv[i]);
                prest.setString(2, hoten[i]);
                prest.setInt(3, namsinh[i]);
                prest.executeUpdate();
            }
        } catch(SQLException e){
            System.out.println(e);}
    }
}
```

CallableStatement

- CallableStatement kế thừa từ PreparedStatement.
- CallableStatement được sử dụng để thực thi stored procedures.
- Stored procedure là một tập hợp các câu lệnh SQL dùng thực hiện 1 công việc nào đó, được gọi như 1 phương thức hay hàm.
- Stored procedure là khái niệm khá phổ biến và được hầu hết các DBMS hỗ trợ.

Ví dụ - CallableStatement

```
CREATE PROCEDURE insertStudent
    @MSSV INT,
    @HoTen VARCHAR(50),
    @NamSinh INT
AS
BEGIN
    INSERT INTO SinhVien
        VALUES(@MSSV,@HoTen,@NamSinh);
END
```

Ví dụ - CallableStatement

```
public void CallableStatement {
    if (con != null){
        try{
            CallableStatement callSt =
con.prepareCall("{call insertStudent(?,?,?)}");
            callSt.setInt(1,50);
            callSt.setString(2, “Nam”);
            callSt.setInt(3, 1999);
            callSt.execute();
        } catch (SQLException e) {
            System.out.println(e);}
} }
```

ResultSetMetaData

- ResultSetMetaData cung cấp các thông tin về cấu trúc cụ thể của ResultSet, ví dụ số lượng cột, tên và kiểu của chúng.
- Phương thức **getMetaData()** của ResultSet trả về đối tượng ResultSetMetaData.

Ví dụ:

ResultSetMetaData rsmd = rs.getMetaData();

Ví dụ - ResultSetMetaData

```
import java.sql.*;
public class ResultSetMetaDataExample {
    public static void main(String[] args) {
      try {
            … // connect database
            Statement stmt = con.createStatement();
            ResultSet rs = stmt.executeQuery(“SELECT * FROM SinhVien”);
            ResultSetMetaData rsmd = rs.getMetaData();
            System.out.println("Tong so cot cua bang: "
                  + rsmd.getColumnCount());
```

Ví dụ - ResultSetMetaData

```
            System.out.println("Ten cot thu 2: "
                  + rsmd.getColumnName(2));
            System.out.println("Kieu du lieu cua cot thu 2: "
                  + rsmd.getColumnTypeName(2));
            con.close();
        } catch (Exception e) {
            System.out.println(e);}
    }
}
```

Kết quả:

Tong so cot cua bang: 3

Ten cot thu 2: HoTen

Kieu du lieu cua cot thu 2: varchar

Transactions

- Mặc định sau khi mỗi câu lệnh SQL được thực thi qua JDBC, dữ liệu sẽ được cập nhật ngay vào CSDL (AutoCommit).
- Có những trường hợp, ta muốn dữ liệu chỉ được cập nhật vào CSDL sau khi 1 nhóm câu lệnh SQL được thực hiện.
- Một nhóm các câu lệnh như thế được gọi là một giao dịch (transaction).

Transactions

- Ví dụ:

Chuyển 5 triệu từ tài khoản A sang tài khoản B, các bước thực hiện:

1) Trừ 5 triệu trong tài khoản A

2) Cộng 5 triệu trong tài khoản B

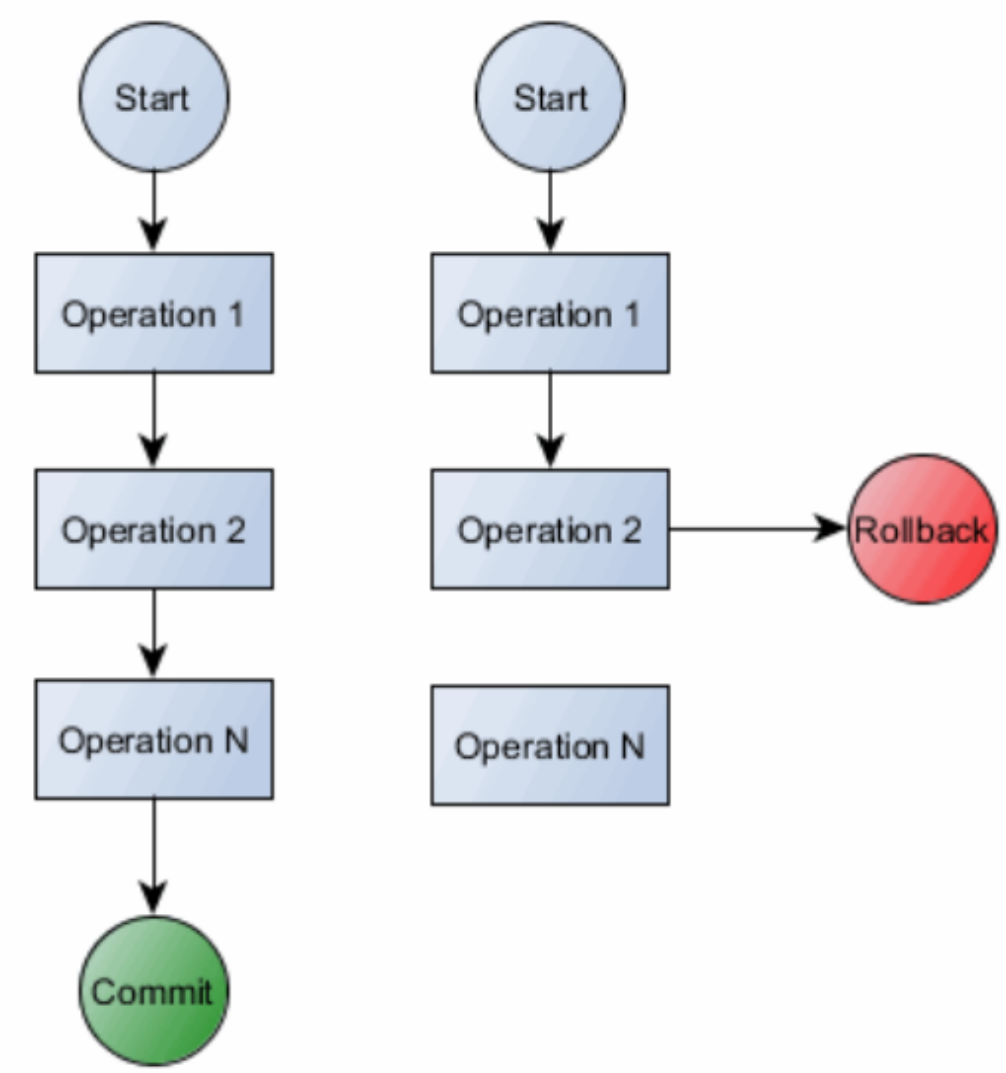
3) Lưu lại thông tin giao dịch để rà soát sau

Nếu bị lỗi giữa chừng thì sao?

Transactions

- Transaction là cơ chế đảm bảo tính toàn vẹn của dữ liệu trong CSDL.
- Transaction kết thúc với **commit** hoặc **rollback.**
  - **commit** (thực hiện giao dịch) nếu thực hiện thành công tất cả các lệnh trong giao dịch
  - **rollback** (hủy giao dịch) nếu có lỗi xảy ra trong quá trình thực hiện giao dịch

Transactions



Transactions

- Khi rollback sẽ hủy dữ liệu đến thời điểm nó được commit gần nhất.
- Tùy thuộc vào hệ quản trị CSDL mà cách rollback có thể khác nhau.
- Dữ liệu sau khi được commit sẽ không hủy được bằng rollback.
- Dùng phương thức **commit()** và **rollback()** của đối tượng Connection để thực thi / hủy giao dịch.

Ví dụ: **con.commit()** //thực thi giao dịch

Transactions

- Dùng phương thức **setAutoCommit(boolean)** của đối tượng Connection để bật/tắt chế độ tự động commit.

Ví dụ: Tắt chế độ tự động commit để thực hiện giao dịch

con.setAutoCommit(false);

- Nếu không cần dùng ở chế độ giao dịch nữa, ta nên trả lại chế độ commit tự động

Ví dụ - Transactions

```
public void transaction(){
    String insertQuery = "INSERT INTO SinhVien VALUES(?,?,?)";
    if (con != null){
        try{
            con.setAutoCommit(false);
            PreparedStatement prest = con.prepareStatement(insertQuery);
            prest.setInt(1, 60);
            prest.setString(2, “Hai");
            prest.setInt(3, 2001);
            prest.executeUpdate();
            prest.setInt(1, 61);
```

Ví dụ - Transactions

```
            prest.setString(2, “Linh");
            prest.setInt(3, 2002);
            prest.executeUpdate();
            con.commit();
            con.setAutoCommit(true);
        } catch(SQLException e){
            try {
                con.rollback();
            } catch (SQLException e1) {
                System.out.println(e1); }
            System.out.println(e); }
    }
}
```

Bài tập

1) Tạo bảng Users trong CSDL gồm các cột ID (int, auto-increment, primary key), Username (varchar), Password (varchar).

2) Xây dựng form **Đăng nhập**, kết nối CSDL để kiểm tra thông tin đăng nhập (sử dụng PreparedStatement). Nếu thông tin đăng nhập đúng thì tắt cửa sổ đăng nhập, mở ra cửa sổ **Home** (frame) với dòng chữ Hello + username.

3) Xây dựng form **Đăng ký**: Kiểm tra username không được trùng, không chứa khoảng trắng, password ít nhất 6 ký tự, nếu hợp lệ thì thêm vào CSDL (sử dụng stored procedure và CallableStatement). Sau khi thêm, hiện dialog thông báo, khi click OK sẽ chuyển đến form Đăng nhập.

Bài tập

4) Màn hình Home có nút Đăng xuất, click vào chuyển đến form Đăng nhập.

5) Thêm nút Đăng ký vào form đăng nhập, click vào chuyển đến form Đăng ký.

6) Khi bắt đầu chạy ứng dụng sẽ hiển thị form Đăng nhập.

7) Khi tắt bất kỳ cửa sổ làm việc nào ứng dụng sẽ kết thúc.