Hackademics Hanoi
Hack your passion, Code your life!

CHƯƠNG 10

KẾT NỐI CƠ SỞ DỮ LIỆU

CÁC TRÌNH ĐIỀU KHIỂN JDBC

- Hiện nay có rất nhiều Hệ quản trị CSDL (DBMS) khác nhau như SQL Sever, MySQL, Oracle, MS Access, FoxPro,…
- Để truy cập các DBMS khác nhau từ chương trình viết bằng Java thì ta cần có các JDBC driver tương ứng.
- Hãng Sun đã đưa ra 4 loại JDBC driver.
- Tham khảo tại: http://industry.java.sun.com/products/jdbc/drivers

CÁC TRÌNH ĐIỀU KHIỂN JDBC

- Tương tác căn bản nhất của JDBC được liệt kê sau đây:

  - Mở một kết nối đến CSDL (Open connection).
  - Thực thi các câu lệnh SQL (Execute SQL).
  - Xử lý dữ liệu (Process result).
  - Đóng kết nối (Close conection).

CÁC TRÌNH ĐIỀU KHIỂN JDBC

- Sử dụng JDBC API, chương trình ứng dụng có thể thiết lập kết nối đến hệ quản trị CSDL, giao tiếp với CSDL, thực thi các câu lệnh thao tác, truy vấn dữ liệu và nhận kết quả trả về.

- Kiến trúc của JDBC gồm hai tầng: tầng đầu tiên là JDBC API, có nhiệm vụ chuyển các câu lệnh SQL cho bộ quản lý trình điều khiển JDBC; tầng thứ hai là các JDBC Driver API, thực hiện nhiệm vụ liên hệ với trình điều khiển của hệ quản trị CSDL cụ thể như SQL-Server, MS-Access, MySQL, IBM-DB2, Oracle,…

Hackademics Hanoi
Hack your passion, Code your life!

CÁC TRÌNH ĐIỀU KHIỂN JDBC

Hackademics Hanoi
Hack your passion, Code your life!

CÁC TRÌNH ĐIỀU KHIỂN JDBC

- Các trình điều khiển có nhiệm vụ là yêu cầu hệ quản trị CSDL thực hiện các câu lệnh SQL. Các trình điều khiển trong JDBC là các đoạn chương trình do nhà sản xuất hệ quản CSDL cung cấp hoặc do các nhà cung cấp thứ ba đưa ra. Các trình điều khiển JDBC được chia ra làm bốn loại:
  - Cầu nối JDBC-ODBC (Bridge)
  - Trình điều khiển thuần túy Java (Native-API).
  - Trình điều khiển JDBC-Net, Pure Java.
  - Trình điều khiển Native-Protocol, Pure Java.

Hackademics Hanoi
Hack your passion, Code your life!

CƠ CHẾ HOẠT ĐỘNG VỚI JDBC

Hackademics Hanoi
Hack your passion, Code your life!

CẦU NỐI JDBC - ODBC (Bridge)

- Cầu nối JDBC-ODBC (Bridge): Trình điều khiển loại này kết nối với các hệ quản trị CSDL thông qua cầu nối ODBC. Trình điều khiển loại này luôn được cung cấp kèm theo bộ J2SE với tên: sun.jdbc.odbc.JdbcOdbcDriver.

Hackademics Hanoi
Hack your passion, Code your life!

TẠO NGUỒN DỮ LIỆU ODBC

- Trên Window, vào Start -> Control Panel -> Administrative Tools -> Data Sources (ODBC)

Hackademics Hanoi
Hack your passion, Code your life!

TẠO NGUỒN DỮ LIỆU ODBC

- Đặt tên nguồn dữ liệu ở mục “Data Source Name” (sẽ sử dụng trong chuỗi kết nối)
- Nhấp “Select” để chọn đường dẫn đến file cơ sơ dữ liệu.

Hackademics Hanoi
Hack your passion, Code your life!

TRÌNH ĐIỀU KHIỂN THUẦN TÚY JAVA

- Trình điều khiển thuần túy Java (Native-API): Trình điều khiển loại này sẽ chuyển các lời gọi của JDBC API sang thư viện hàm (API) tương ứng với từng hệ CSDL cụ thể. Trình điều khiển loại này do nhà xây dựng hệ quản trị CSDL cung cấp.
- Loại này cho phép JDBC giao tiếp trực tiếp với các driver hay các hàm API của CSDL.

Hackademics Hanoi
Hack your passion, Code your life!

Trình điều khiển JDBC - Net, Pure Java

- Có thể sử dụng cùng một trình điều khiển để truy cập đến nhiều hệ quản trị CSDL khác nhau
- Chuyển các lời gọi JDBC API sang một dạng chuẩn độc lập với các hệ quản trị CSDL, sau đó được chuyển sang lời gọi của một hệ quản trị CSDL cụ thể bởi một chương trình trung gian
- Có thể giao tiếp với nhiều loại CSDL.
- Không phải của nhà cung cấp csdl, mà là của các nhà cung cấp thứ ba.
- Tất cả bằng mã java.
- Có thể chuyển các yêu cầu đến các csdl nằm ở xa.

Hackademics Hanoi
Hack your passion, Code your life!

Trình điều khiển Native-Protocol, Pure Java

- Trình điều khiển loại này chuyển các lời gọi JDBC API sang mã lệnh của hệ quản trị CSDL cụ thể, đây là các trình điều khiển thuần Java, có nghĩa là không cần phải có mã lệnh của hệ quản trị CSDL cụ thể khi thi hành chương trình.

Hackademics Hanoi
Hack your passion, Code your life!

CÁC BƯỚC LẬP TRÌNH TRUY XUẤT CSDL

- Nạp trình điều khiển.
- Thiết lập kết nối.
- Tạo đối tượng Statement
- Thực hiện vấn tin
- Xử lý kết quả trả về
- Đóng kết nối

NẠP TRÌNH ĐIỀU KHIỂN

- Sử dụng phương thức tĩnh forName() của lớp Class với tham số là tên trình điều khiển cơ sở dữ liệu.

- ***<u>Cách dùng:</u>***

```
try{
    Class.forName("Database driver name");
}
catch(ClassNotFoundException e){
    System.out.println("Driver not found: " +e.getMessage());
}
catch(SQLException e){
    System.out.println("SQL Exception: " + e.getMessage());
}
```


Hackademics Hanoi
Hack your passion, Code your life!

NẠP TRÌNH ĐIỀU KHIỂN

- Trình điều khiển của MySQL:
  - **Class.forName(“org.gjf.mm.mysql.Driver”);**
- Trình điều khiển của Oracle:
  - **Class.forName(“oracle.jdbc.driver.OracleDriver”);**
- Trình điều khiển của Sybase:
  - **Class.forName("com.sybase.jdbc.SybDriver");**
- Trình điều khiển qua cầu nối ODBC:
  - **Class.forName(“sun.jdbc.odbc.JdbcOdbcDriver”);**

THIẾT LẬP KẾT NỐI

- Để thiết lập kết nối ta gọi phương thức tĩnh getConnection() của lớp DriverManager, khi đó trả về một thể hiện của lớp Connection, theo dạng như sau:
  - **String user = “sa”**
  - **String password = “secret”**
  - **Connection con = DriverManager.getConnection(dbUrl, username, password);**
- Trong đó:
  - **dbUrl**: là chuỗi kết nối đến cơ sở dữ liệu.
  - **username** : tên người dùng đăng nhập
  - **password** : mật khẩu đăng nhập.

THIẾT LẬP KẾT NỐI

Định nghĩa chuỗi kết nối:

```
String host = “dbhost.yourcompany.com”;
String dbName = “someName”;
int port = 1234;
String oracleURL = “jdbc:oracle:thin:@” + host + “:”
                                        + port + “:” +dbName;
String sybaseURL = “jdbc:sybase:Tds:“ + host + “:”
                + port + “:” + “?SERVICEid=” + dbName;
```


Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ VỀ THIẾT LẬP KẾT NỐI

- Nạp jdbc driver cần sử dụng

  Class.forName(

  “sun.jdbc.odbc.JbdcOdbcDriver”);

- Tạo kết nối (sử dụng cầu nối JDBC-ODBC)

Connection conn=DriverManager.getConnection(

“jdbc:odbc:<DataSourceName>”);

- DataSourceName là tên của ODBC data source được tạo trên MS Windows

Hackademics Hanoi
Hack your passion, Code your life!

TẠO ĐỐI TƯỢNG Statement

- Sử dụng đối tượng Connection để tạo đối tượng Statement.
  - **Statement s = con.createStatement();**
- Đối tượng này có nhiệm vụ gửi các câu lệnh sql đến CSDL
- Cùng một đối tượng Statement có thể sử dụng cho nhiều câu lệnh sql khác nhau.
- Có 3 phương thức thực thi
  - executeQuery()
  - executeUpdate()
  - execute()

Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ (tạo Statement và vấn tin)

```
import java.sql.*;

public class TestDatabase {
   public static void main(String[] arguments) {
      //String data = "jdbc:odbc:AddressBook";
      String data = "jdbc:odbc:Driver={Microsoft Access Driver (*.mdb)};DBQ="
     + "D:/JAVA/LamBTJava/AddressBook.mdb"; //;DriverID=22";

     try {
         Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
         Connection conn = DriverManager.getConnection(
            data, "Vo Tan Dung", "abc");
         Statement st = conn.createStatement();
         ResultSet rec = st.executeQuery(
            "SELECT * FROM AddressBook");

//          (còn tiếp ở slide kế tiếp)
```


Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ (tạo Statement và vấn tin)

```
//    (tiếp theo)
        while(rec.next()) {
            System.out.println(rec.getString("NumberID") +  "\t"
                + rec.getString("HomeNumber") + "\t\t"
                + rec.getString("MobileNumber") + "\t"
                + rec.getString("WorkNumber"));
        }
        conn.close();
    } catch (SQLException s) {
        System.out.println("SQL Error: " + s.toString() + " "
            + s.getErrorCode() + " " + s.getSQLState());
    } catch (Exception e) {
        System.out.println("Error: " + e.toString()
            + e.getMessage());
    }
  }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

TẠO BẢNG QUA JDBC

```
public void createTable(){
    String sqlQuery = "CREATE TABLE PRICELIST(NAME
                                VARCHAR(20), PRICE NUMBER);";
    if (conn != null){
        try{
            Statement st = conn.createStatement();
            st.execute(sqlQuery);
            st.close();
        }catch(SQLException e){
            e.printStackTrace();
        }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

NHẬP DỮ LIỆU QUA JDBC

- Sử dụng phương thức executeUpdate() của Statement để cập nhật dữ liệu qua JDBC
- executeUpdate() trả về số dòng bị ảnh hưởng sau khi câu lệnh được thực thi

Ví dụ:

```
public void insertData(String name, double val){
    String insertQuery = "INSERT INTO PRICELIST VALUES
                                        ('" + name + "'," + val + ");";
    if (conn != null){
        try{
            Statement st = conn.createStatement();
            int d = st.executeUpdate(insertQuery);
            System.out.println("So dong duoc cap nhat: " + d);
            st.close();
        }catch(SQLException e){
            e.printStackTrace();
        }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

ĐỌC DỮ LIỆU TỪ CSDL

- Sử dụng phương thức **executeQuery()** của Statement để đọc dữ liệu từ cơ sở dữ liệu.
- Dữ liệu đọc được sẽ được lưu trong ResultSet.
- Có thể di chuyển tới lui trong ResultSet để lấy dữ liệu ra.

VÍ DỤ

```
public void selectAll(){
    String selectQuery = "SELECT * FROM PRICELIST;";
    if (conn != null){
    try{
        Statement st =
                conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
        ResultSet.CONCUR_READ_ONLY);
        ResultSet rs = st.executeQuery(selectQuery);
        System.out.println("NAME\t\tPRICE");

        while (rs.next()){
                String name = rs.getString("NAME");
                double val = rs.getDouble("PRICE");
                System.out.println(name + "\t\t" + val);
        }
        st.close();
    }catch(SQLException e){
        e.printStackTrace();
    }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

CÁC KIỂU Resultset

- TYPE_FORWARD_ONLY
  - Con trỏ của ResultSet kiểu này chỉ được di chuyển theo một hướng từ đầu đến cuối
- TYPE_SCROLL_INSENSITIVE
  - Con trỏ có thể di chuyển tới lui tương đối với vị trí hiện tại của nó, và cũng có thể di chuyển đến một vị trí cụ thể, không bị ảnh hưởng nếu kết quả được thay đổi ở nơi khác
- TYPE_SCROLL_SENSITIVE
  - Con trỏ có thể di chuyển tới lui tương đối với vị trí hiện tại của nó, và cũng có thể di chuyển đến một vị trí cụ thể, sẽ bị ảnh hưởng nếu kết quả bị thay đổi nơi khác

CÁC CHẾ ĐỘ HOẠT ĐỘNG CỦA RESULTSET

Các chế độ hoạt động đồng thời của Resultset:

- CONCUR_READ_ONLY
  - Xác định chế độ hoạt động đồng thời, kết quả lưu trong đối tượng ResultSet không được thay đổi
- CONCUR_UPDATABLE
  - Xác nhận chế độ hoạt động đồng thời, kết quả lưu trong đối tượng ResultSet được thay đổi

CÁC PHƯƠNG THỨC CỦA RESULTSET

- next(): di chuyển con trỏ đến dòng kế, trả về true nếu có dòng kế tiếp, false nếu đến cuối ResultSet
- previous(): di chuyển con trỏ đến dòng trước
- first(): di chuyển con trỏ đến dòng đầu tiên
- last(): di chuyển con trỏ đến dòng cuối cùng
- beforeFirst(): di chuyển con trỏ đến vị trí trước dòng đầu tiên
- afterLast(): di chuyển con trỏ đến sau dòng cuối cùng
- relative(int rows): di chuyển con trỏ tương đối với vị trí hiện tại của nó với số dòng là rows
- absolute(int row): di chuyển con trỏ đến dòng thứ row

Hackademics Hanoi
Hack your passion, Code your life!

LẤY DỮ LIỆU TỪ RESULTSET

- Dùng phương thc getXXX(String colname)
  - XXX là kiểu dữ liệu được trả về
  - colname là tên của cột cần lấy dữ liệu ra
- Ví d:
  - String name = rs.getString("NAME");
  - double val = rs.getDouble("PRICE");

Hackademics Hanoi
Hack your passion, Code your life!

CÁC PHƯƠNG THỨC getXXX

Kiểu của SQL	Kiểu của Java	Hàm getXXX()
CHAR	String	getString()
VARCHAR	String	getString()
LONGVARCHAR	String	getString()
NUMBERIC	java.math.BigDecimal	getBigDecimal()
DECIMAL	java.math.BigDecimal	getBigDecimal()
BIT	Boolean (boolean)	getBoolean()
TINYINT	Integer (byte)	getByte()
SMALLINT	Integer (short)	getShort()
INTEGER	Integer (int)	getInt()
BIGINT	Long (long)	getLong()
REAL	Float (float)	getFloat()
FLOAT	Double (double)	getDouble()
DOUBLE	Double (double)	getDouble()
BINARY	byte[]	getBytes()
VARBINARY	byte[]	getBytes()
LONGVARBINARY	byte[]	getBytes()
DATE	java.sql.Date	getDate()
TIME	java.sql.Time	getTime()
TIMESTAMP	java.sql.Timestamp	getTimestamp()

Hackademics Hanoi
Hack your passion, Code your life!

CHỈNH SỬA DỮ LIỆU BẰNG RESULTSET

- Dữ liệu của một bảng trong CSDL có thể được chỉnh sửa bằng cách sử dụng câu lệnh SQL UPDATE
- Ta cũng có thể chỉnh sửa dữ liệu trên các hàng của một bảng từ ResultSet
- Để làm được điều này, ResultSet phải được đặt ở chế độ CONCUR_UPDATABLE
- Sử dụng các phương thức updateXXX() để chỉnh sửa dữ liệu trên ResultSet
  - XXX là kiểu dữ liệu của cột cần được sửa

Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ CHỈNH SỬA DỮ LIỆU

```
public void updateTable(){
    String selectQuery = "SELECT * FROM PRICELIST;";
    if (conn != null){
        try{
            Statement st =
        conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
                ResultSet.CONCUR_UPDATABLE);
            ResultSet rs = st.executeQuery(selectQuery);
            rs.next();
            rs.updateDouble("PRICE", 1.5);
            rs.updateRow();
            st.close();
        }catch(SQLException e){
            e.printStackTrace();
        }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

SỬ DỤNG PreparedStatement

- Có những trường hợp ta cần thực hiện nhiều câu lệnh SQL có cấu trúc tương tự nhau, chỉ có giá trị là thay đổi
- PreparedStatement có thể được sử dụng 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

VÍ DỤ PreparedStatement

```
public void prepareStatement(){
    String insertQuery = "INSERT INTO PRICELIST VALUES(?,?);";
    if (conn != null){
    try{
        PreparedStatement prest = conn.prepareStatement(insertQuery);
        prest.setString(1, "Biscuit");
        prest.setDouble(2, 1.2);
        prest.executeUpdate();
        prest.setString(1, "Pen");
        prest.setDouble(2, 0.5);
        prest.executeUpdate();
        prest.close();
    }catch(SQLException e){
        e.printStackTrace();
    }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

SỬ DỤNG Transaction (Giao Dịch)

- Mặc định, sau khi mỗi câu lệnh SQL được thực thi qua JDBC, dữ liệu sẽ được cập nhật ngay vào CSDL
- Có những trường hợp, ta muốn dữ liệu chỉ được cập nhật vào CSDL sau khi một số câu lệnh SQL được thực hiện
  - Ví dụ: đối với trang ứng dụng bán hàng qua mạng, để CSDL được thống nhất, ta chỉ muốn lưu các dữ liệu liên quan tới một đơn đặt hàng cùng một lúc
- Một nhóm các câu lệnh như thế được gọi là một giao dịch (transaction)

Hackademics Hanoi
Hack your passion, Code your life!

CÀI ĐẶT GIAO DỊCH

- Trước hết, phải không dùng chế độ COMMIT tự động từ Connection object
  - conn.setAutoCommit(false);
- Thực hiện các câu lệnh trong một giao dịch
- Thực hiện COMMIT (lưu) CSDL
  - conn.commit();
- Nếu không cần dùng ở chế độ giao dịch nữa, ta nên trả lại chế độ COMMIT tự động
  - conn.setAutoCommit(true);

Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ Transaction

```
public void transaction(){
    String insertQuery = "INSERT INTO PRICELIST VALUES(?,?);";
    if (conn != null){
    try{
        conn.setAutoCommit(false);
        PreparedStatement prest = conn.prepareStatement(insertQuery);
        prest.setString(1, "Biscuit");
        prest.setDouble(2, 1.2);
        prest.executeUpdate();
        prest.setString(1, "Pen");
        prest.setDouble(2, 0.5);
        prest.executeUpdate();
        prest.close();
        conn.commit();
        conn.setAutoCommit(true);
    }catch(SQLException e){
        e.printStackTrace();
    }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

HỦY MỘT GIAO DỊCH

- Trong quá trình thực hiện một giao dịch, nếu có sai sót nào xảy ra, ta có thể hủy giao dịch đang được thực hiện nửa chừng bằng cách sử dụng rollback
- conn.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

Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ rollback

```
public void rollbackTransaction(){
    String insertQuery = "INSERT INTO PRICELIST VALUES(?,?);";
    if (conn != null){
        try{    conn.setAutoCommit(false);
                PreparedStatement prest =
                        conn.prepareStatement(insertQuery);
                prest.setString(1, "Biscuit");
                prest.setDouble(2, 1.2);
                prest.executeUpdate();
                conn.commit();
                prest.setString(1, "Pen");
                prest.setDouble(2, 0.5);
                prest.executeUpdate();
                prest.close();
                conn.rollback();  conn.commit();
                conn.setAutoCommit(true);
        }catch(SQLException e){e.printStackTrace();}
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

ResultSetMetaData

- *ResultSetMetaData* cung cấp các thông tin về cấu trúc cụ thể của *ResultSet*, bao gồm cả số cột, tên và giá trị của chúng

VÍ DỤ ResultSetMetaData

```
public void showMetadata(){
    String selectQuery = "SELECT * FROM PRICELIST;";
    if (conn != null){
        try{
            Statement st = conn.createStatement
                (ResultSet.TYPE_SCROLL_INSENSITIVE,
                                ResultSet.CONCUR_READ_ONLY);
            ResultSet rs = st.executeQuery(selectQuery);
            ResultSetMetaData meta = rs.getMetaData();
            int iColumnCount = meta.getColumnCount();
            for (int i =1 ; i <= iColumnCount ; i++){
                System.out.println("Column Name: " + meta.getColumnName(i));
                System.out.println("Column Type: " + meta.getColumnTypeName(i));
                System.out.println("Display Size: " + meta.getColumnDisplaySize(i) );
                System.out.println("Precision: " + meta.getPrecision(i));
                System.out.println("Scale: " + meta.getScale(i) );
            }
            st.close();
        }catch(SQLException e){
                e.printStackTrace();
        }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ VỀ HIỂN THỊ MỘT TABLE

Ví dụ: Chương trình hiển thị một bảng dữ liệu.

```
import java.sql.*; import java.util.StringTokenizer;
public class TableViewer {
    final static String jdbcURL = "jdbc:odbc:StudentDB";
    final static String jdbcDriver = "sun.jdbc:odbc:JdbcOdbcDriver";
    final static String table = "STUDENT";
    public static void main(java.lang.String[]args) {
      System.out.println("---Table Viewer ---");
      try { Class.forName(jdbcDriver);
            Connection con =
                        DriverManager.getConnection(jdbcURL, "", "");
            Statement stmt = con.createStatement();
            // Đọc ra cả bảng Student và đưa vào đối tượng rs
            ResultSet rs = stmt.executeQuery("SELECT * FROM " + table);
            // Đọc ra các thông tin về rs
            ResultSetMetaData rsmd = rs.getMetaData();
            // Xác định số cột của rsmd
            int colCount = rsmd.getColumnCount();
//xem tiếp ở slide tiếp theo
```


Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ VỀ HIỂN THỊ MỘT TABLE

```
for(int col = 1; col <= colCount; col++)
{           //  In ra tên và kiểu của từng trường dữ liệu trong rsmd
            System.out.print(rsmd.getColumnLabel(col));
            System.out.print(" (" + rsmd.getColumnTypeName(col) + ")");
            if(col < colCount) System.out.print(", ");
}
System.out.println();
while(rs.next()){
            //  In ra dòng dữ liệu trong rsmd
            for(int col = 1; col <= colCount; col++)
            {
                        System.out.print(rs.getString(col));
                        if(col < colCount) System.out.print("  ");
            }
            System.out.println();
} //xem tiếp ở slide tiếp theo
```


Hackademics Hanoi
Hack your passion, Code your life!

VÍ DỤ VỀ HIỂN THỊ MỘT TABLE

```
        rs.close();
        stmt.close();
        con.close();
        }
        catch (ClassNotFoundException e) {
                System.out.println("Unable to load database driver class");
        }
        catch (SQLException se) {
                System.out.println("SQL Exception: " + se.getMessage());
        }
    }
}
```


Hackademics Hanoi
Hack your passion, Code your life!

ĐÓNG KẾT NỐI

- Để đóng kết nối ta sử dụng phương thức close của đối tượng connection:
  - connetion.close()

Hackademics Hanoi
Hack your passion, Code your life!

HẾT
CHƯƠNG 10