ĐẠI HỌC HUẾ
TRƯỜNG ĐẠI HỌC KHOA HỌC

NGUYỄN HOÀNG HÀ, NGUYỄN VĂN TRUNG
NGUYỄN DŨNG, NGUYỄN MẬU HÂN

GIÁO TRÌNH
JAVA CƠ BẢN

NHÀ XUẤT BẢN ĐẠI HỌC HUẾ
Huế, 2020

Biên mục trên xuất bản phẩm của Thư viện Quốc gia Việt Nam

ĐTTS ghi: Đại học Huế. Trường Đại học Khoa học. - Thư mục: tr. 211

006.740711 - dc23

DUF0253p - CIP

Mã số sách: GT/38-2019

LỜI NÓI ĐẦU

Giới thiệu về NNLT Java: ....

Giáo trình "Java cơ bản" sẽ hệ thống toàn bộ những nội dung kiến thức cơ bản liên quan đến kiến thức về ngôn ngữ lập trình Java. Giáo trình được sử dụng cho sinh viên chuyên ngành Công nghệ Thông tin, là môn cơ sở để lập trình di động, lập trình web JSP, ... Giáo trình được chia làm 3 chương. Cuối mỗi chương sẽ có phần tổng kết chương, giúp người học nắm được các nội dung và ý nghĩa của chương. Phần câu hỏi và bài tập ở cuối mỗi chương cũng giúp người học nâng cao kỹ năng thực hành trên cơ sở phần lý thuyết được trang bị.

Các chương của giáo trình được tóm tắt như sau:

Chương 1: *Lập trình với Java*,....

Chương 2: *Lập trình hướng đối tượng trong Java*, ...

Chương 3: *Lập trình ứng dụng csdl*. ...

Mặc dù đã rất cố gắng để hoàn thiện giáo trình này với mong muốn đến tay người đọc nhưng giáo trình không thể tránh khỏi những thiếu sót về cách diễn đạt, bố cục, nội dung và các lỗi cú pháp. Rất mong được bạn đọc góp ý.

Để hoàn tất giáo trình này chúng tôi xin cám ơn các Thầy Cô giáo của Khoa Công nghệ Thông tin, Trường Đại học Khoa học, Đại học Huế đã góp ý, chỉnh sửa để giáo trình sớm được ra mắt bạn đọc.

Huế, tháng 02 năm 2019

Nhóm tác giả

MỤC LỤC

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

CHƯƠNG 3

LẬP TRÌNH ỨNG DỤNG CƠ SỞ DỮ LIỆU

Chương này tập trung vào sử dụng thư viện JDBC (Java DataBase Connectivity) để kết nối và truy xuất vào các sở dữ liệu. Nôi dung của chương bao gồm:

- Trình bày được khái niệm về JDBC; Các thành phần JDBC; Phân loại JDBC; Cơ chế hoạt động JDBC,...
- Thực hiện được việc tải và cài đặt JDBC driver cho project
- Trình bày được các phương pháp kết nối CSDL với các hệ QT CSDL SQL Server, Ocracle, MySQL,...
- Thực hiện được việc kết nối và truy xuất cơ sở dữ liệu
- Thực hiện được việc xử lý kết xuất kết quả truy xuất CSDL.
- Xây dựng được ứng dụng Quản lý CSDL

3.1. Giới thiệu JDBC

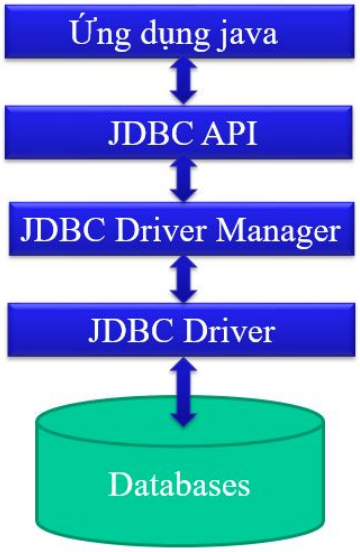
JDBC là một thư viện chuẩn cho phép các ứng dụng Java kết nối với nhiều cơ sở dữ liệu được cài đặt trên các hệ quản trị cơ sở dữ liệu như MS Access, SQL Server, Oracle, DB2,... JDBC hỗ trợ các chức năng như tạo một kết nối đến một cơ sở dữ liệu, tạo câu lệnh SQL (Structured Query Language), thực thi câu lệnh SQL, xem và thay đổi dữ liệu.

3.2. Các thành phần trong JDBC

JDBC bao gồm các thành phần như Hình 3.1, trong đó:

- JDBC API: là một API (Application Programming Interface) hoàn toàn dựa trên Java, giúp truy xuất cơ sở dữ liệu từ Java.
- JDBC Diver Manager: một lớp (class) giúp kết nối giữa ứng dụng Java đến các JDBC Driver.

- JDBC Driver: là thành phần cho phép ứng dụng Java tương tác với các cơ sở dữ liệu khác nhau.

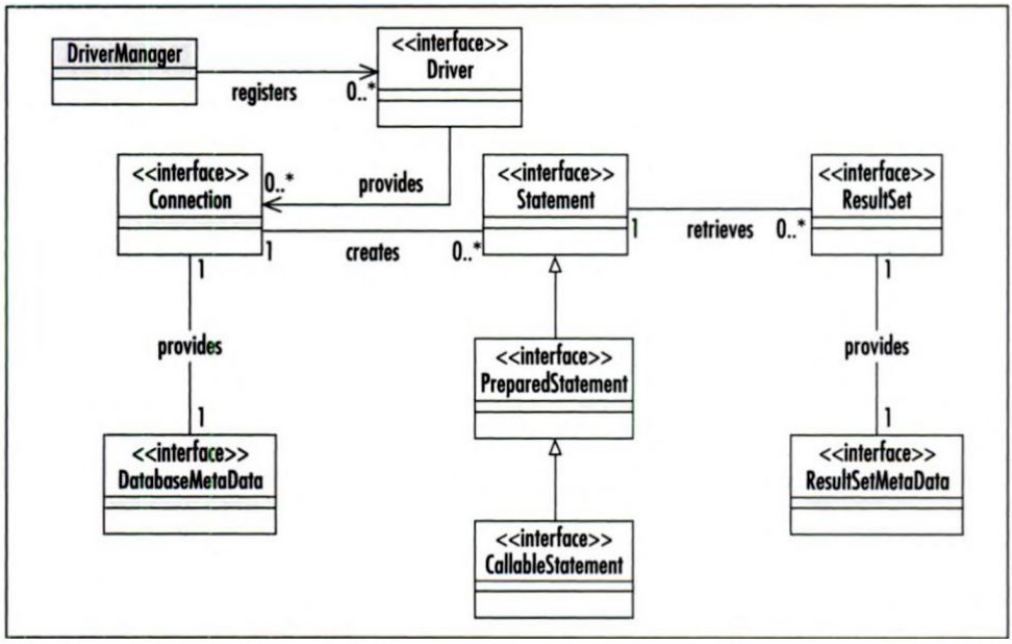


Hình 3.1. *Các thành phần trong JDBC*

3.3. Lược đồ lớp của JDBC

JDBC bao gồm các lớp (class) và giao diện (interface) nằm trong gói java.sql.* như thể hiện ở Hình 3.2, trong đó:

- **Driver:** Giao diện này chứa các hàm trừu tượng để xử lý các kết nối đến cơ sở dữ liệu. Trong lập trình, chúng ta chủ yếu sử dụng lớp DriverManager được cài đặt từ giao diện Driver để quản lý các kết nối đến cơ sở được liệu.
- **DriverManager:** Lớp này quản lý một danh sách trình điều khiển cơ sở dữ liệu (database drivers). Nó được dùng để nạp các Driver vào trong bộ nhớ và mở các kết nối tới một cơ sỡ dữ liệu nào đó.
- **Connection:** Đại diện cho một kết nối đến cơ sở dữ liệu. Được dùng để tạo ra các đối tượng Statement, PreparedStatement và CallableStatement.
- **Statement:** Đối tượng dùng để thực thi các câu lệnh SQL như



Hình 3.2. *Các lớp và giao diện của JDBC*

câu lệnh thêm dữ liệu (insert), câu lệnh thay đổi dữ liệu (update), câu lệnh xoá dữ liệu (delete), câu lệnh xem dữ liệu (select), ...

- **PreparedSatement:** Giống như Statement nhưng có thể truyền tham số vào câu lệnh SQL.
- **CallableStatement:** được sử dụng để thực thi một thủ tục (Stored Procedure) được lưu trữ trong hệ quản trị cơ sở dữ liệu.
- **ResultSet:** Đối tượng này sẽ quản lý dữ liệu sau khi chúng ta thực thi câu lệnh Select. Một đối tượng ResultSet duy trì một con trỏ trỏ tới hàng dữ liệu hiện tại của nó, sử dụng con trỏ này để lấy dữ liệu từ các bảng (Tables) trong cơ sở dữ liệu.
- **DatabaseMetaData:** cung cấp các phương thức để lấy siêu dữ liệu (metadata) của cơ sở dữ liệu như tên sản phẩm cơ sở dữ liệu, phiên bản sản phẩm cơ sở dữ liệu, tên Driver, tên của các bảng, tên view, ...
- **ResultSetMetaData:** cung cấp các phương thức để lấy siêu dữ liệu (metadata) của ResultSet như tổng số cột, tên cột, kiểu của

cột,...

Các bước truy xuất cơ sở dữ liệu

- **Bước 1:** Sử dụng lớp DirverManager để nạp trình điều khiển nhằm xác định hệ quản trị cơ sở dữ liệu cần sử dụng.
- **Bước 2:** Sử dụng lớp DirverManager để tạo đường kết nối đến cơ sở dữ liệu.
- **Bước 3:** Sử dụng đối tượng Connection để tạo ra các đối tượng Statement, PreparedStatement và CallableStatement.
- **Bước 4:** Sử dụng các đối tượng Statement, PreparedStatement và CallableStatement để thực thi các câu lênh SQL hoặc thủ tục lưu trữ.
- **Bước 5:** Xử lý dữ liệu lấy về từ cơ sở dữ liệu
- **Bước 6:** Giải phóng bộ nhớ và đóng kết nối

3.4. Lớp DirverManager

Lớp DirverManager hoạt động như một giao diện giữa người dùng và các trình điều kiển (Driver). Nó theo dõi các trình điều khiển có sẵn và xử lý việc thiết lập kết nối giữa một cơ sở dữ liệu và trình điều khiển thích hợp. Lớp DriverManager duy trì một danh sách các trình điều khiển được đăng ký bằng cách gọi phương thức DriverManager.registerDriver().

Để DriverManager nạp trình điều khiển thì trong project chúng ta phải thêm thư viện tương ứng với hệ quản trị cơ sở dữ liệu ta cần kết nối đến, Bảng 3.1 liệt kê một số thư viện điều khiển thường dùng.

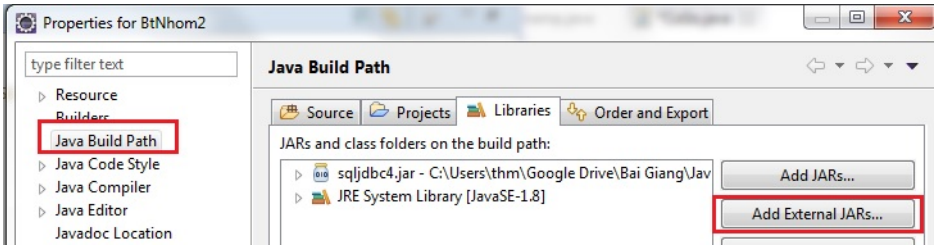
Bảng 3.1. *Một số thư viện điều khiển dùng trong JDBC*

STT	HQTCSDL	Tên thư viện điều khiển
1	MySQL	Mysql-connector-java-X.Y.Z.bin.jar (X.Y.Z là số hiệu của từng phiên bản)
2	SQL Server	sqljdbc.jar, sqljdbc4.jar
3	Oracle	jodbc5.jar, jodbc6.jar, ...
4	PostgreSQL	Postgresql-9.1-902.jdbc4.jar

5	SQLite	Sqlite-jdbc-3.7.2.jar

Ví dụ 3.4.1: Khi làm việc trên SQl Server, chúng ta cần thêm thư viện điều khiển sqljdbc4.rar vào Project theo các bước:

- Bước 1: Download thư viện điều khiển sqljdbc4.rar về máy.
- Bước 2: Tạo 1 project trên Eclipse.
- Bước 3: Nhấn Alt + Enter, chọn Java Build Path, chọn Add External JARs như Hình 3.3, sau đó chọn file sqljdbc4.rar download ở Bước 1.



Hình 3.3. *Thêm thư viện điều khiển vào Project của Eclipse*

Một số phương thức thường dùng trong lớp DirverManager

a. public static void registerDriver(Driver driver) throws SQLException: Phương thức này được sử dụng để đăng ký trình điều khiển đã cho với DriverManager.

Ví dụ 3.4.2: cần đăng ký trình điều khiển để làm việc với SqlServer:

```
1    Driver dr=new
         com.microsoft.sqlserver.jdbc.SQLServerDriver();
2    DriverManager.registerDriver(dr);
```

Ngoài cách trên, người ta thường dùng Class.forName() để đăng ký trình điều khiển đây là hướng tiếp cận chung và được dùng phổ biến để đăng ký một Driver. Cú pháp:

```
public static void Class.forName(String DatabaseDriver)
    throws ClassNotFoundException
```

Trong đó, DatabaseDriver là một chuỗi để xác đinh tên hệ quản trị cơ sỡ dữ liệu, Bảng 3.2 trình bày một số DatabaseDriver thường dùng:

Bảng 3.2. *Một số DatabaseDriver thường dùng*

STT	HQTCSDL	DatabaseDriver
1	MySQL	com.mysql.jdbc.Driver
2	SQL Server	com.microsoft.sqlserver.jdbc.SQLServerDriver
3	Oracle	oracle.jdbc.driver.OracleDriver
4	PostgreSQL	org.postgresql.Driver
5	Microsoft Access	sun.jdbc.odbc.JdbcOdbcDriver

Ví dụ 3.4.3:

Đăng ký trình điều khiển để làm việc với SqlServer

```
Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
```

Đăng ký trình điều khiển để làm việc với MySql

```
Class.forName("com.mysql.jdbc.Driver");
```

2. public static void deregisterDriver(Driver driver)

Phương thức này được sử dụng để xóa Driver đã cho khỏi danh sách đăng ký với DriverManager.

Ví dụ 3.4.4: Đăng ký trình điều khiển để làm việc với SqlServer, sau đó xóa trình điều khiển ra khỏi danh sách:

```
Driver dr=new com.microsoft.sqlserver.jdbc.SQLServerDriver();
DriverManager.registerDriver(dr);
```

```
DriverManager.deregisterDriver(dr);
```

3. public static Connection getConnection(String DatabaseURL)

Phương thức này được sử dụng để thiết lập kết nối đến cơ sở dữ liệu, các thông tin kết nối được lưu trong DatabaseURL. DatabaseURL là một chuỗi thường lưu tên Server chứa hệ quản trị cơ sở dữ liệu, tên cơ sở dữ liệu cần kết nối. Một số DatabaseURL có thể lưu username và password để truy cập vào cơ sở dữ liệu. Một số DatabaseURL thường dùng được lưu ở Bảng 3.3.

Bảng 3.3. *Một số DatabaseDriver thường dùng*

STT	HQTCSDL	DatabaseDriver
1	MySQL	jdbc:mysql://<Tên Server>:<port>/<Tên CSDL>
2	SQL Server	jdbc:sqlserver://<Tên server>:<port>; databaseName=<Tên CSDL>;user=...; password=...
3	Oracle	jdbc:oracle:thin:@<Tên server>:<port>: <Tên CSDL>
4	PostgreSQL	jdbc:postgresql://<Tên server>:<port>/<Tên CSDL>
5	Microsoft Access	jdbc:odbc:Driver=Microsoft Access Driver (*.mdb); DBQ=<Tên CSDL.mdb>

Ví dụ 3.4.5: Viết chương trình kết nối đến CSDL: QlNv của SQLServer; Tên Server: ADMIN; Username: sa; password:123

```
import java.sql.*;
public class ViDu345 {
public static void main(String[] args) {
    try {
        //b1 Nạp trình điều khiển
        String driver=
            "com.microsoft.sqlserver.jdbc.SQLServerDriver";
```

```
        Class.forName(driver);
        //b2: Kết nối vào CSDL
        String urlDatabase ="jdbc:sqlserver://ADMIN:1433;
            databaseName=QlNv; user=sa; password=123";
        Connection
            cn=DriverManager.getConnection(urlDatabase);
        System.out.println("Da ket noi");
    } catch (Exception e) {
        e.printStackTrace();
    }
  }
}
```

Khi chạy chương trình trên, một số lỗi thường gặp:

a. Chưa thêm thư viện sqljdbc4 vào Project: Khi chưa thêm thư viện sqljdbc4.rar vào Project sẽ hiển thị lỗi như Hình 3.4. Để khắc phục lỗi này ta thực hiện 3 bước như Ví dụ 3.4.1.

```
java.lang.ClassNotFoundException: com.microsoft.sqlserver.jdbc.SQLServerDriver
        at java.net.URLClassLoader.findClass(Unknown Source)
        at java.lang.ClassLoader.loadClass(Unknown Source)
        at sun.misc.Launcher$AppClassLoader.loadClass(Unknown Source)
        at java.lang.ClassLoader.loadClass(Unknown Source)
        at java.lang.Class.forName0(Native Method)
        at java.lang.Class.forName(Unknown Source)
        at ViDu345.main(ViDu345.java:10)
```

Hình 3.4. *Lỗi chưa thêm thư viện vào Project*

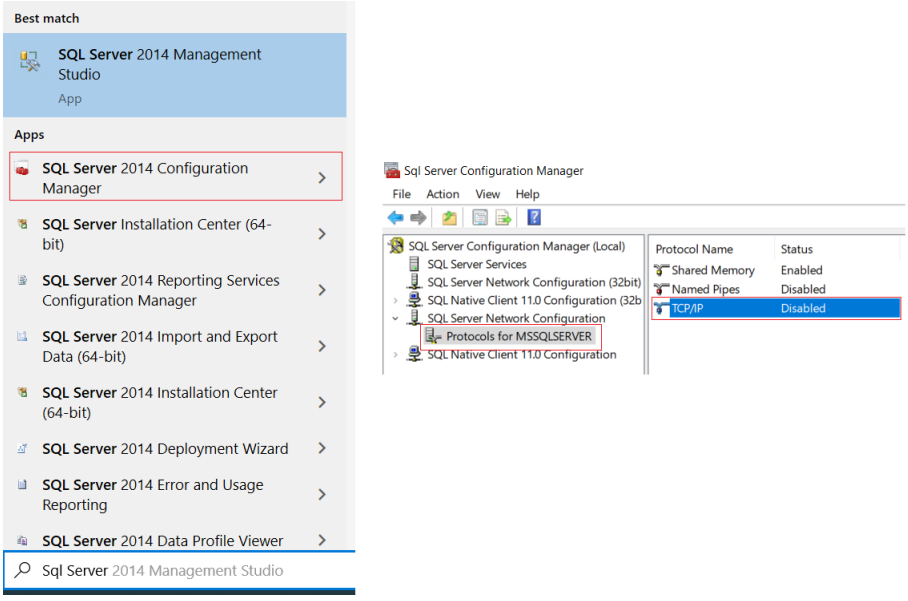
b. Chưa bật giao thức TCP và mở cổng trong SQLServer: Trong một số phiên bản của SQLServer khi cài lên sẽ chưa mở giao thức TCP và cổng 1433. Vì vậy, khi chạy chương trình sẽ báo lỗi như Hình 3.5.

```
com.microsoft.sqlserver.jdbc.SQLServerException: The TCP/IP connection to the host ADMIN, port 1433 has failed.
        at com.microsoft.sqlserver.jdbc.SQLServerException.makeFromDriverError(SQLServerException.java:130)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.connectHelper(SQLServerConnection.java:1195)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.loginWithoutFailover(SQLServerConnection.java:1054)
        at com.microsoft.sqlserver.jdbc.SQLServerConnection.connect(SQLServerConnection.java:758)
        at com.microsoft.sqlserver.jdbc.SQLServerDriver.connect(SQLServerDriver.java:842)
        at java.sql.DriverManager.getConnection(Unknown Source)
        at java.sql.DriverManager.getConnection(Unknown Source)
        at ViDu345.main(ViDu345.java:13)
```

Hình 3.5. *Lỗi chưa mở cổng và bật giao thức TCP*

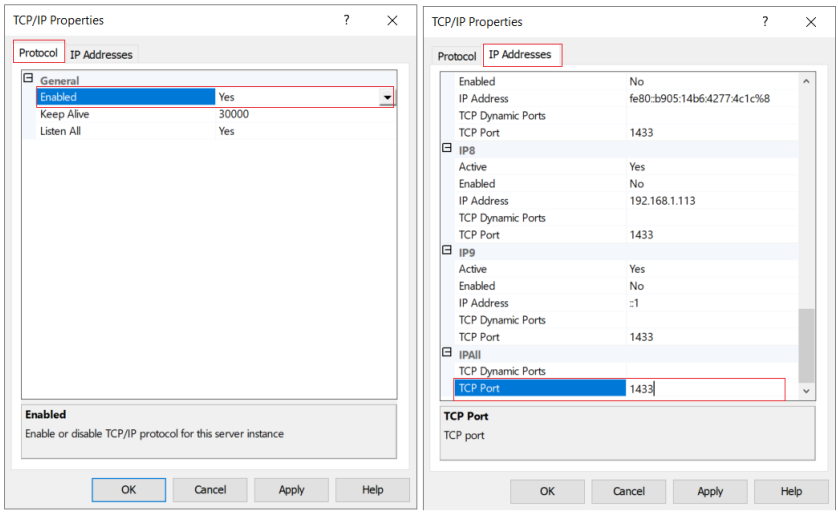
Để khắc phục lỗi này ta thực hiện theo các bước sau:

Bước 1: Nhấn Start và đi tìm và mở SQL Server Configuration Manager như Hình 3.6. Chọn SQL Server Network Configuration, chọn Protocols for ... và kích đôi lên TCP/IP.



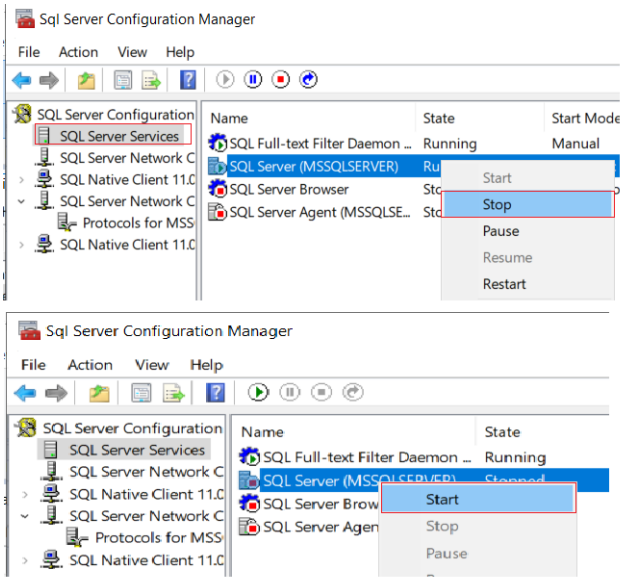
Hình 3.6. *Mở SQL Server Configuration Manager*

Bước 2: Bật giao thức TCP và mở cổng 1433 như Hình 3.7



Hình 3.7. *Bật giao thức TCP và mở cổng 1433*

Bước 3: Tắt và khởi động lại dịch vụ của SQL Server như Hình 3.8. Kích phải lên dịch vụ của SQL Server chọn Stop để tắt dịch vụ, sau đó kích phải lên dịch vụ của SQL Server chọn Start để khởi động lại dịch vụ.



Hình 3.8. *Tắt và khởi động lại dịch vụ của SQL Server*

c. Chưa bật cấu hình user và password để login vào SQL Server: Trong quá trình cài SQLServer ta chưa cấu hình tài khoản để login vào SQL Server. Vì vậy, khi chạy chương trình sẽ báo lỗi như Hình 3.9.

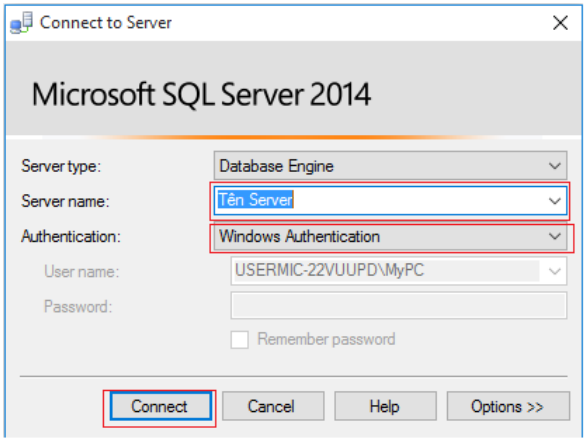
```
com.microsoft.sqlserver.jdbc.SQLServerException: Login failed for user 'sa'.
        at com.microsoft.sqlserver.jdbc.SQLServerException.makeFromDatabaseError(S
        at com.microsoft.sqlserver.jdbc.TDSTokenHandler.onEOF(tdsparser.java:240)
        at com.microsoft.sqlserver.jdbc.TDSParser.parse(tdsparser.java:78)
```

Hình 3.9. *Chưa cấu hình tài khoản để login vào SQL Server*

Để khắc phục lỗi này ta lần lượt thực hiện theo các bước để cấu hình user và password như sau:

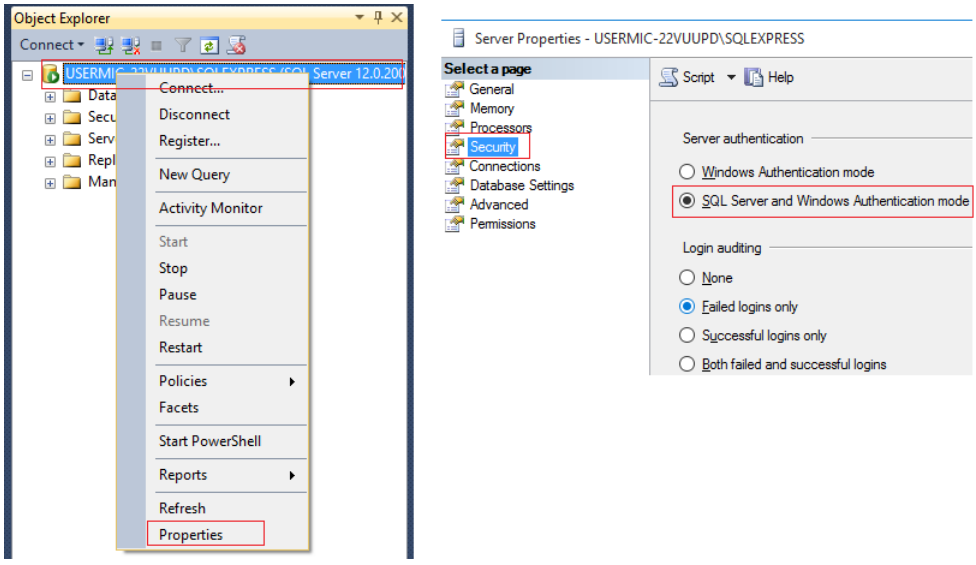
Bước 1: Khởi động SQL Server Management Studio và đăng nhập vào theo tài khoản của Window như Hình 3.10

Bước 2: Cấu hình để người dùng đăng nhập bằng tài khoản trong



Hình 3.10. *Đăng nhập vào SQL Server*

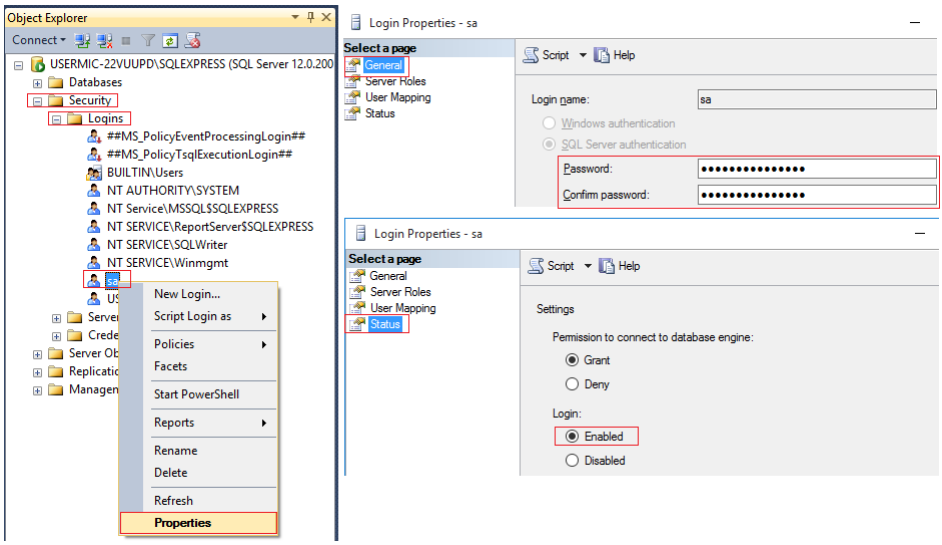
SQL Server như Hình 3.11. Kích phải lên Tên Server, chọn Properties, tại Security chọn SQL Server and Window Authentication mode.



Hình 3.11. *Cấu hình để người dùng đăng nhập bằng tài khoản*

Bước 3: Đặt mật khẩu và cho phép user: sa đăng nhập vào SQL Server như Hình 3.12. Kích phải lên user: sa, chọn Properties, Chọn General, sau đó nhập mật khẩu và nhập lại mật khẩu. Chọn Status và chọn Enabled để cho phép tài khoản sa đăng nhập vào hệ thống. **Bước 4:** Tắt và mở lại dịch vụ SQL Server. Kích phải lên Tên Server chọn

Stop, sau đó kích phải lên Tên Server chọn Start.



Hình 3.12. *Đặt mật khẩu và cho phép user: sa đăng nhập*

4. public static Connection getConnection(String url,String userName,String password)

Trong trường hợp một số DatabaseURL không chứa username, và password thì dùng phương thức này để thiết lập kết nối đến CSDL.

Ví dụ 3.4.6. Viết chương trình kết nối đến CSDL: QlNv của MySql; Tên Server: ADMIN; Username: root; password:123

```
import java.sql.Connection;
import java.sql.DriverManager;
public class ViDu346 {
public static void main(String[] args) {
    try {
        //b1 Nạp trình điều khiển
        String driver="com.mysql.jdbc.Driver";
        Class.forName(driver);
        //b2: Kết nối vào CSDL
        String urlDatabase ="jdbc:mysql://localhost/QlNv";
        Connection  cn=DriverManager.getConnection(
```

```
            urlDatabase,"root","123");
12          System.out.println("Da ket noi");
13      } catch (Exception e) {
14      e.printStackTrace();
15      }
16   }
17  }
```

Sau khi kết nối vào CSDL thì phương thức getConnection sẽ trả về một đối tượng Connection, sử dụng đối tượng Connection để tạo ra các đối tượng Statement, PreparedStatement và CallableStatement.

5. public static void setLoginTimeout(int second)

Phương thức này thiết lập thời gian (bằng giây) tối đa mà một Driver sẽ đợi trong khi cố gắng kết nối với một CSDL.

6. public static int getLoginTimeout()

Phương thức này lấy thời gian (bằng giây) tối đa mà một Driver sẽ đợi trong khi cố gắng truy cập vào một CSDL.

3.5. Đối tượng Connection

Sau khi đã nạp trình điều khiển để xác định hệ quản trị cơ sở dữ liệu và đã kết nối vào cơ sở dữ liệu, bước tiếp theo chúng ta cần lấy dữ liệu về để xử lý và lưu lại dữ liệu vào cơ sở dữ liệu. Để thực hiện được các việc này, chúng ta sử dụng các phương thức của Connection, một số phương thức của Connection:

1. public Statement createStatement()

Phương thức này tạo đối tượng Statement để thực thi các truy vấn SQL. Khi lấy dữ liệu từ cơ sở dữ liệu về chúng ta chỉ duyệt các bản ghi từ trên xuống, không thể di chuyển con trỏ đến bản ghi bất kỳ và không thể cập nhật lại dữ liệu.

2. public Statement createStatement(int resultSetType,int resultSetConcurrency) throws SQLException

Phương thức này cũng tạo ra một đối tượng Statement khắc phục các nhược điểm của của phương thức createStatement(), với phương thức reateStatement(int resultSetType,int resultSetConcurrency) ta có

thể di chuyển con trỏ đến vị trí bất kỳ, cho phép người dùng cập nhật lại dữ liệu trên các bản ghi được lấy về, trong đó:

resultsetType là 1 trong 3 kiểu:

- ResultSet.TYPE _FORWARD _ONLY: chỉ di chuyển con trỏ đi tới.
- ResultSet.TYPE _SCROLL _INSENSITIVE: cho phép di chuyển con trỏ nhưng không cho phép cập nhật lại dữ liệu
- ResultSet.TYPE _SCROLL _SENSITIVE: cho phép di chuyển con trỏ và cho phép cập nhật lai dữ liệu.

resultSetConcurrency là 1 trong 2 kiểu:

- ResultSet.CONCUR _READ _ONLY: dữ liệu lấy về chỉ đọc không thể cập nhật được.
- ResultSet.CONCUR _UPDATABLE: Cho phép cập nhật lại dữ liệu.

3. public PreparedStatement prepareStatement(String sql) throws SQLException Trong đó: chuỗi sql chứa câu lệnh SQL.

Giống như phương thức Statement() nó được dùng để thực thi truy vấn SQL nhưng có thể sử dụng tham số trong câu lênh SQL. Thực tế, ta nên dùng phương này ngay cả khi không cần phải truyền tham số bởi nó thực thi nhanh hơn do được nạp trước (preload) khi thực thi. PreparedStatement được kế thừa từ Statement, vì vậy mọi phương thức của Statement đều có thể dùng ở đây.

4. public PreparedStatement prepareStatement(String sql, int resultSetType, int resultSetConcurrency)throws SQLException

Phương thức này và các tham số của nó giống như phương thức createStatement(int resultSetType,int resultSetConcurrency) như ta có thể truyền tham số vào câu lệnh sql.

Ví dụ 3.4.7. Đã có cơ sở dữ liệu QlNv trong SqlServer, trong cơ sở dữ liệu này đã có bảng DonVi(**MaDonVi**, TenDonVi), đoạn mã dưới đây sử dụng phương thức prepareStatement để tạo ra một câu lệnh

SQL với các tham số nhằm mục đích chuẩn bị thêm vào 1 đơn vị vào bảng DonVi.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
public class ViDu347 {
public static void main(String[] args) {
    try {
        //b1 Nạp trình điều khiển
        String driver=
            "com.microsoft.sqlserver.jdbc.SQLServerDriver";
        Class.forName(driver);
        //b2: Kết nối vào CSDL
        String urlDatabase ="jdbc:sqlserver://ADMIN:1433;
            databaseName=Qlnv;user=sa;password=123";
        Connection
            cn=DriverManager.getConnection(urlDatabase);
        //b3: Tạo câu lệnh PreparedStatement
        String sql="insert into DonVi values (?,?)";
        PreparedStatement cmd=cn.prepareStatement(sql);
    } catch (Exception e) {
        e.printStackTrace();
    }
 }
}
```

5. public CallableStatement prepareCall(" call tên_thủ_tục(các tham số của thủ tục)");

Được sử dụng để thực thi một thủ tục lưu trữ (Stored Procedure) trong SQL Server. Ta có thể sử dụng dấu ? để truyền vào các tham số của thủ tục. CallableStatement được thừa kế từ PreparedStatement.

5. public void setAutoCommit(boolean autoCommit) throws

SQLException

Phương thức này thiết lập Connection trong chế độ auto-commit. Nếu một Connection trong chế độ tự động ký thác thì tất cả các lệnh SQL của nó sẽ được thực thi và ký thác sau mỗi giao tác. Nếu tham số autoCommit được thiết lập là true tức là kích hoạt chế độ auto-commit, nếu là false là vô hiệu hóa chế độ này.

6. public void commit() throws SQLException

Phương thức này lưu các thay đổi đã được thực hiện trước đó. Phương thức này nên chỉ được sử dụng khi chế độ auto-commit đã bị vô hiệu hóa.

7. public void rollback()

Phương thức này xóa tất cả các thay đổi đã được thực hiện trước đó và quay về trạng thái trước khi thực hiện thay đổi. Phương thức này nên chỉ được sử dụng khi chế độ auto-commit đã bị vô hiệu hóa.

8. public void close()

Phương thức này đóng kết nối và giải phóng tài nguyên.

Ví dụ 3.4.8. Ví dụ này trình bày các bước để quyết định thực hiện các công việc hoặc phục hồi lại các công việc nếu chương trình bị lỗi.

```
1  import java.sql.Connection;
2  import java.sql.DriverManager;
3  import java.sql.PreparedStatement;
4  public class ViDu348 {
5  public static void main(String[] args) {
6      Connection  cn=null;
7      try {
8          //b1 Nạp trình điều khiển
9          String driver=
               "com.microsoft.sqlserver.jdbc.SQLServerDriver";
10         Class.forName(driver);
11         //b2: Kết nối vào CSDL Qlnv, tên Server: ADMIN,
               user:sa, password=123
12         String urlDatabase ="jdbc:sqlserver://ADMIN:1433;
```

```
            databaseName=Qlnv;user=sa; password=123";
        cn=DriverManager.getConnection(urlDatabase);
        //b3: Đặt chế độ AutoCommi=false
        cn.setAutoCommit(false);
        //b4: Thực thi các công việc như thêm, xóa, cập nhật
            vào cơ sở dữ liệu
          ...
        //b5: Quyết định thực thi các công việc ở B4
        cn.commit();
        cn.close();//Đóng kết nối
        System.out.println("Đã hoàn thành các công việc");
    } catch (Exception e) {
        System.out.print("Chương trình bị lỗi");
        try {
            //b6: Phục hồi lại công việc ở ở B4
            cn.rollback();
            cn.close();//Đóng kết nối
        } catch (Exception e2) {
        System.out.print("Không thể phục hồi");
    }
  }
 }
}
```

Ví dụ 3.4.9. Để đơn giản B1 và B2 ta viết một lớp DungChung và xây dựng một hàm tĩnh để kết nối và trả về đường kết nối đến cơ sở dữ liệu Qlnv trên máy chủ ADMIN với user và password để login vào CSDL là: sa và 123.

```
import java.sql.Connection;
import java.sql.DriverManager;
public class DungChung {
   public static Connection KetNoi() throws Exception{
    //b1 Nạp trình điều khiển
```

```
6     String
          driver="com.microsoft.sqlserver.jdbc.SQLServerDriver";
7     Class.forName(driver);
8      //b2: Kết nối vào CSDL Qlnv, tên Server: ADMIN, user:sa,
           password=123
9     String urlDatabase ="jdbc:sqlserver://ADMIN:1433;
          databaseName=Qlnv;user=sa; password=123";
10    return DriverManager.getConnection(urlDatabase);
11   }
12  }
```

3.6. Đối tượng Statement

Statment được tạo ra bởi đối tượng Connection dùng để thực thi các câu lệnh SQL, một số phương thức thường dùng:

1. public ResultSet executeQuery(String sql)

Thực thi một lệnh SQL đã cho và trả về một đối tượng Resultset. Tham số sql là một chuỗi chứa câu lệnh SQL: SELECT.

2. public int executeUpdate(String sql)

Thực thi lệnh sql đã cho, câu lệnh sql có thể là INSERT, UPDATE, DELETE hoặc một lệnh SQL mà không trả về giá trị như lệnh SQL về định nghĩa dữ liệu: CREATE TABLE, CREATE DATABASE, ...

Phương thức trả về số mẫu tin có hiệu lực, ví dụ trả về số mẫu tin thêm được hoặc xóa được hoặc cập nhật được. Nếu câu lệnh SQL chứa các câu lệnh SQL về định nghĩa dữ liệu hàm sẽ trả về bằng 0.

Ví dụ 3.6.1: Chèn thêm 1 dòng vào bảng DonVi có cấu trúc như ví dụ 3.4.7. Đoạn mã sau chưa kiểm tra trùng MaDonVi.

```
1 import java.sql.*;
2 public class ViDu361 {
3 public static void main(String[] args) {
4 Connection  cn=null;
5 try {
```

```
    //b1, b2
    cn=DungChung.KetNoi();
    //b3: Đặt chế độ AutoCommi=false
    cn.setAutoCommit(false);
    //b4: Thêm vào bảng đơn vị 1 bản ghi với mã đơn vị là cntt
        và tên đơn vị là Công nghệ thông tin
    String madv="cntt";
    String tendv="Công nghệ thông tin";
    //Thiết lập câu lệnh sql
    String sql="insert into DonVi values ('" +madv+"',N'"+
        tendv+"')";
    Statement cmd=cn.createStatement();
    int smt=cmd.executeUpdate(sql);
    System.out.print("Số mẫu tin thêm được: "+ smt);
    //b5: như trong ví dụ 3.4.8
    ...
} catch (Exception e) {
    System.out.print("Chương trình bị lỗi");
    try {
        //b6: Phục hồi lại công việc ở ở B4
        cn.rollback();
        cn.close();//Đóng kết nối
    } catch (Exception e2) {
        System.out.print("Không thể phục hồi");
    }
  }
 }
}
```

Chú ý: Theo ví dụ trên nếu Tendv="Baker's Chocolate" thì câu lệnh SQL sẽ bị lỗi cú pháp vì sau khi ghép câu lệnh sql sẽ là "insert into DonVi values ('cntt',N'Baker's Chocolate'). Để khắc phục nhược điểm này ta dùng đối tượng PreparedStatement thay vì dùng đối tượng Statement.

3. void close() throws SQLException

Phương thức được sử dụng để đóng đối tượng Statement và giải phóng tài nguyên ngay lập tức. Khi đã đóng đối tượng Statement rồi, thì các lời gọi phương thức nào tới đối tượng đó sẽ không hoạt động. Khi một đối tượng Statement đã bị đóng thì đối tượng ResultSet của nó cũng bị đóng theo.

3.7. Hạn chế của đối tượng Statement

Khi sử dụng đối tượng Statement, thường nối (ghép) các chuỗi lại với nhau để thiết lập câu lệnh SQL.

Ví dụ 3.7.1: Để tìm kiếm một đơn vị trong bảng DonVi thì câu lệnh SQL là "Select * from DonVi where TenDonVi=N'"+**tendv**+"'", trong đó **tendv** do người dùng nhập vào.

Giả sử người dùng nhập **tendv=Công nghệ thông tin** thì sẽ tìm ra các đơn vị có tên là Công nghệ thông tin. Nhưng nếu người dùng nhập vào **tendv=Baker's Chocolate** thì câu lệnh SQL sẽ bị sai cú pháp vì bị dư dấu ' trong câu lệnh SQL.

Ví dụ 3.7.2: Giả sử đã có bảng Login(**username**, password) để người dùng đăng nhập vào hệ thống. Câu lệnh SQL để kiểm tra người dùng đăng nhập thành công hay không là: "select * from Login where username ='"+**un**+ "' and password='"+**pass**+"'", trong đó **un** và **pass** do người dùng nhập vào. Như vậy nếu người dùng nhập vào **un=nhập bất kỳ** và **pass= nhập bất kỳ' or '1'='1** khi đó điều kiện của câu lệnh SQL luôn đúng: select * from Login where username ='nhập bất kỳ' and password='nhập bất kỳ' **or '1'='1'**". Điều này có nghĩa là cho dù người dùng không có tài khoản để đăng nhập họ vẫn đăng nhập thành công vào hệ thống. Dây được gọi là SQL Injection:

SQL injection là một kỹ thuật cho phép những kẻ tấn công lợi dụng lỗ hổng của việc kiểm tra dữ liệu đầu vào trong các ứng dụng và các thông báo lỗi của hệ quản trị cơ sở dữ liệu trả về để inject (tiêm vào) và thi hành các câu lệnh SQL bất hợp pháp. SQL injection có thể cho phép những kẻ tấn công thực hiện các thao tác, delete, insert, update, v.v. trên cơ sở dữ liệu của ứng dụng, thậm chí là server mà ứng dụng đó đang chạy. SQL injection thường được biết đến như là một vật trung gian tấn

công trên các ứng dụng có dữ liệu được quản lý bằng các hệ quản trị cơ sở dữ liệu như SQL Server, MySQL, Oracle, DB2, Sysbase...

Qua hai ví dụ trên ta thấy được các tác hại và hạn chế khi sử dụng Statement. Statement không có tham số hóa ? nên muốn truyền vào tham số chúng ta cần thêm dấu + để nối các chuỗi lại với nhau. Từ đó hacker hay người dùng xấu có thể cố tình truy cập vào hệ thống.

Vì vậy PreparedStatement sinh ra để khắc phục nhược điểm trên, PreparedStatement không nối chuỗi và dùng các tham số ? để truyền vào câu lệnh SQL, ví dụ:

String sql = "SELECT * FROM users WHERE username = ? AND password = ?";

Từ đó ta có thể đặt giá trị cho các biến được tham số hóa dấu ?. PreparedStatement sẽ tự loại bỏ những ký tự đặc biệt và phù hợp với câu lệnh truy vấn, tránh xảy ra SQL Injection.

3.8. Đối tượng PreparedStatement

Như đã trình bày ở trên, PreparedStatement được tạo ra để khắc phục các hạn chế của Statement. Đối tượng này được sử dụng khi chúng ta muốn thực thi 1 truy vấn SQL có tham số. Nó là 1 đối tượng được kế thừa từ Statement(public interface PreparedStatement extends Statement) cho nên mọi phương thức của Statement đều có thể dùng cho PreparedStatement. Một số phương thức thường dùng trong PreparedStatement:

1. public int executeUpdate()

Thực thi truy vấn SQL trong đối tượng PreparedStatement, câu lệnh SQL gồm INSERT, UPDATE hoặc DELETE, hoặc một lệnh SQL mà không trả về bất cứ cái gì, chẳng hạn như câu lệnh SQL định nghĩa dữ liệu như CREATE, ALTER,...

2. public ResultSet executeQuery() throws SQLException Phương thức này thực thi truy vấn SQL trong đối tượng PreparedStatement, câu lệnh SQL thường dùng là SELECT. Phương thức trả về đối tượng ResultSet được tạo bởi truy vấn.

3. public void setString(int paramIndex, String giaTri)

Thiết lập tham số đã cho thành giá trị String trong Java đã cung

cấp. Trình điều khiển sẽ chuyển đổi giá trị này thành một kiểu varchar hoặc nvarchar khi nó gửi giá trị tới CSDL.

Ví dụ 3.8.1: Thêm vào bảng DonVi(MaDonVi,TenDonVi) một bảng ghi và đổi tên một đơn vị dựa vào MaDonVi. MaDonVi và TenDonVi có kiểu nvarchar(50). Đoạn mã dưới đây chưa kiểm tra trùng MaDonVi khi thêm.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
public class ViDu381 {
public static void main(String[] args) {
Connection  cn=null;
try {
    //b1, b2
    cn=DungChung.KetNoi();
    //b3: Đặt chế độ AutoCommi=false
    cn.setAutoCommit(false);
    String madv="toan";
    String tendv="Khoa Toán";
//b4: Thêm vào một đơn vị có mã đơn vị là: toan, tên đơn vị
    là Khoa Toán
    //Thiết lập câu lệnh SQL
    String sql="insert into DonVi values(?,?)";
    PreparedStatement cmd= cn.prepareStatement(sql);
    cmd.setString(1,madv);//Truyền tham số vào câu lệnh SQL
    cmd.setString(2,tendv);
    int smt=cmd.executeUpdate();//Thực thi câu lệnh
    System.out.println("Số mẫu tin thêm được: "+ smt);
//B5: Sửa tên đơn vị thành Tin học của đơn vị có mã là: cntt
    madv="cntt";
    tendv="Tin học";
    //Thiết lập câu lệnh SQL
```

```
27      sql="update DonVi set TenDonVi=? where MaDonVi=?";
28      cmd=cn.prepareStatement(sql);
29      cmd.setString(1,tendv);//Truyền tham số vào câu lệnh SQL
30      cmd.setString(2,madv);
31      smt=cmd.executeUpdate();//Thực thi câu lệnh
32      System.out.println("Số mẫu tin sửa được: "+ smt);
33      cn.commit();//Quyết định thực thi các công việc ở B4,B5
34      cmd.close();//Giải phóng bộ nhớ
35      cn.close();//Đóng kết nối
36      System.out.println("Đã hoàn thành các công việc")
37    } catch (Exception e) {
38      System.out.print("Chương trình bị lỗi");
39      try {
40   //b6: Phục hồi lại công việc ở ở B4
41          cn.rollback();
42          cn.close();//Đóng kết nối
43      } catch (Exception e2) {
44         System.out.print("Không thể phục hồi");
45      }
46    }
47   }
48  }
```

4. public void setInt(int paramIndex, int giaTri)

Thiết lập tham số đã cho tới giá trị nguyên trong Java đã cung cấp. Driver sẽ chuyển đổi giá trị này thành một giá trị nguyên trong SQL khi nó gửi giá trị tới CSDL.

5. public void setFloat(int paramIndex, float giaTri)

Thiết lập tham số đã cho thành giá trị float trong Java đã cung cấp. Driver chuyển đổi giá trị này thành một giá trị REAL trong SQL khi nó gửi giá trị tới CSDL.

6. public void setDouble(int paramIndex, double giaTri)

Thiết lập tham số đã cho thành giá trị double trong Java đã cung

cấp. Driver chuyển đổi giá trị này thành một giá trị DOUBLE trong SQL khi nó gửi giá trị tới CSDL.

Ngoài các phương thức trên, PreparedStatement hỗ trợ các phương thức thiết lập tham số cho các kiểu Boolean, Date, Byte, Long như setBoolean(int paramIndex, int giaTri), setDate(int paramIndex, int giaTri),setByte(int paramIndex, int giaTri), setLong(int paramIndex, int giaTri).

Ví dụ 3.8.2: Viết chương trình để tạo ra bảng NhanVien trong CSDL QlNv có cấu trúc như Hình 3.13, sau đó nhập vào 1 nhân viên.

	Column Name	Data Type	Allow Nulls
🔑	Manv	nvarchar(50)	☐
	HoTen	nvarchar(50)	☑
	gioitinh	bit	☑
	NgaySinh	date	☑
	Hsl	float	☑
	MaDv	nvarchar(50)	☑

Hình 3.13. *Cấu trúc của bảng NhanVien*

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import java.util.Date;
public class ViDu382 {
public static void main(String[] args) {
Connection  cn=null;
try {
    //b1, b2
    cn=DungChung.KetNoi();
    //b3: Đặt chế độ AutoCommi=false
    cn.setAutoCommit(false);
    //b4: Tạo bảng
        NhanVien(MaNv,HoTen,GioiTinh,NgaySinh,HSL,Madv)
```

```
16      //Tạo câu lệnh SQL để tạo bảng NhanVien
17      String sql="CREATE TABLE NhanVien(MaNv nvarchar(15) NOT
            NULL,HoTen nvarchar(50) NULL,GioiTinh bit NULL,
            NgaySinh date NULL, Hsl real NULL, MaDv [nvarchar](50)
            NULL, PRIMARY KEY( MaNv ))";
18      PreparedStatement cmd= cn.prepareStatement(sql);
19      cmd.executeUpdate();//Thực thi câu lệnh
20      //b5: Thêm vào một nhân viên
21      //Tạo câu lệnh SQL
22      sql="insert into nhanvien(Manv,HoTen,GioiTinh,NgaySinh,
            Hsl, Madv) values(?,?,?,?,?,?)";
23      cmd= cn.prepareStatement(sql);
24      //Thiết lập các tham số
25      cmd.setString(1,"Nv1");
26      cmd.setString(2,"Nguyễn Hoàng Hà");
27      cmd.setBoolean(3,true);
28      String ngaysinh ="18/11/1976";
29      //Đổi chuỗi ngaysinh ra kiểu ngày java.util.Date;
30      SimpleDateFormat dd= new SimpleDateFormat("dd/MM/yyyy");
31      Date NsUtil=dd.parse(ngaysinh);
32      //Đổi ngày java.util.Date ra java.sql.Date
33      java.sql.Date NsSql= new java.sql.Date(NsUtil.getTime());
34      //Thiết lập tham số cho trường NgaySinh
35      cmd.setDate(4,NsSql);
36      cmd.setDouble(5, 2.34);
37      cmd.setString(6, "cntt");
38      cmd.executeUpdate();//Thực thi câu lệnh
39      //b6 Quyết định thực thi các công việc ở b4,b5
40      cn.commit();//
41      cmd.close();//Giải phóng bộ nhớ
42      cn.close();//Đóng kết nối
43      System.out.println("Đã hoàn thành các công việc");
44
45  } catch (Exception e) {
```

```
46      System.out.print("Chương trình bị lỗi");
47      try {
48          //b6: Phục hồi lại công việc ở ở B4
49          cn.rollback();
50          cn.close();//Đóng kết nối
51      } catch (Exception e2) {
52          System.out.print("Không thể phục hồi");
53      }
54     }
55    }
56  }
```

3.9. Đối tượng ResultSet

Phương thức executeQuery của PrepareStatement hoặc Statement trả về một con trỏ trỏ đến một tập kết quả lấy về từ cơ sở dữ liệu, con trỏ này có kiểu là ResultSet. Người lập trình sử dụng con trỏ này để lấy dữ liệu về, di chuyển con trỏ, cập nhật lại dữ liệu, ...

3.9.1. Lấy giá trị của mẫu tin tại vị trí của con trỏ.

Để lấy giá trị của mẫu tin tại vị trí của con trỏ ta sử dụng các phương thức getXXX. Cú pháp chung của các phương thức này có dạng:

public XXX getXXX(int columnIndex)

hoặc

public XXX getXXX(String columnName)

Trong đó, **XXX** là tên kiểu dữ liệu, **columnIndex** và **columnName** là chỉ số của cột hoặc tên cột cần lấy dữ liệu ra.

Ví dụ: getInt, getString, getDouble, getBoolean, getDate, ...

Các phương thức sau lấy ra số nguyên và chuỗi tại vị trí con trỏ:

1. public int getInt(int columnIndex): được sử dụng để lấy về

một số nguyên trên cột thứ columnIndex tại vị trí của con trỏ. Chú ý: cột đầu tiên có số thứ tự là 1.

2. public int getInt(String columnName): được sử dụng để lấy về một số nguyên trên cột có tên columnName tại vị trí của con trỏ.

3. public String getString(int columnIndex): được sử dụng để lấy về một chuỗi trên cột thứ columnIndex tại vị trí của con trỏ.

4. public String getString(String columnName): được sử dụng để lấy về một chuỗi trên cột có tên columnName tại vị trí của con trỏ.

3.9.2. Các phương thức di chuyển con trỏ mẫu tin

1. public boolean next(): được sử dụng để di chuyển con trỏ đến mẫu tin tiếp theo từ vị trí hiện tại. Hàm trả về false nếu con trỏ đang đứng mẫu tin cuối cùng mà tiếp tục di chuyển, ngược lại hàm trả về true.

Chú ý: Để duyệt từ mẫu tin đầu tiên đến cuối cùng trong ResultSet: rs ta thường dùng vong lặp while:

```
1  while (rs.next()){
2    //Xử lý dữ liệu
3  }
```

Ví dụ 3.9.1: Giả sử bảng NhanVien đã có dữ liệu như Hình 3.14

	Manv	HoTen	gioiti...	NgaySinh	Hsl	MaDv
1	Nv1	Nguyễn Hoàng Hà	1	1976-11-18	2.34	cntt
2	Nv2	Lê Văn Trung	1	1980-10-10	3.33	cntt
3	Nv3	Trần Nguyên Phong	1	1976-12-10	3.33	cntt
4	Nv4	Lê Viết Mẫn	1	1992-10-12	3.66	toan
5	Nv5	Hà Lê Nguyên	0	1980-04-03	2.67	van
6	Nv6	Phạm Bình Minh	1	1979-02-09	4.32	van

Hình 3.14. *Dữ liệu của bảng NhanVien*

Viết chương trình hiển thị thông tin của tất cả các nhân viên trong bảng NhanVien.

```java
import java.sql.*;
import java.text.SimpleDateFormat;
import java.util.Date;
public class ViDu391 {
public static void main(String[] args) {
    Connection  cn=null;
    try {
        cn=DungChung.KetNoi();
        //Tạo câu lệnh SQL Lấy về tất cả các nhân viên
        String sql="select * from nhanvien";
        PreparedStatement cmd= cn.prepareStatement(sql);
        ResultSet rs= cmd.executeQuery();//Thực thi câu lệnh
        int i=1;
        //Duyệt qua từng mẫu tin trong rs
        while(rs.next()){
            String Manv=rs.getString("Manv");//Lấy ra MaNv
            String Hoten=rs.getString("HoTen");//Lấy ra HoTen
            Boolean GioiTinh=rs.getBoolean("GioiTinh");
            Date NgaySinh =rs.getDate("NgaySinh");
            String Madv=rs.getString("MaDv");
            System.out.println("Thông tin của nhân viên
                thứ: "+ i++);
            System.out.print(Manv);
            System.out.print(" "+ Hoten);
            System.out.print(" "+ GioiTinh);
            System.out.print(" "+ NgaySinh);
            System.out.println(" "+Madv);
            System.out.println("--------------------");
        }
        rs.close();// Đóng ResultSet
        cn.close();//Đóng kết nối
        System.out.println("Đã hoàn thành các công việc");

```

```
33      } catch (Exception e) {
34          System.out.print("Chương trình bị lỗi");
35          try {
36              cn.close();//Đóng kết nối
37          } catch (Exception e2) {
38              System.out.print("Không thể phục hồi");
39          }
40      }
41    }
42  }
```

Kết quả hiển thị của chương trình:

```
Thông tin của nhân viên thứ:  1
Nv1 Nguyễn Hoàng Hà true 1976-11-18 cntt
--------------------
Thông tin của nhân viên thứ:  2
Nv2 Lê Văn Trung true 1980-10-10 cntt
--------------------
Thông tin của nhân viên thứ:  3
Nv3 Trần Nguyên Phong true 1976-12-10 cntt
--------------------
Thông tin của nhân viên thứ:  4
Nv4 Lê Viết Mẫn true 1992-10-12 toan
--------------------
Thông tin của nhân viên thứ:  5
Nv5 Hà Lê Nguyên false 1980-04-03 van
--------------------
Thông tin của nhân viên thứ:  6
Nv6 Phạm Bình Minh true 1979-02-09 van
--------------------
Đã hoàn thành các công việc
```

Hình 3.15. *Kết quả chương trình*

2. public boolean previous(): được sử dụng để di chuyển con trỏ đến một mẫu tin trước đó từ vị trí hiện tại.

3. public boolean first(): được sử dụng để di chuyển con trỏ đến mẫu tin đầu tiên.

4. public boolean last(): được sử dụng để di chuyển con trỏ đến mẫu tin cuối cùng.

5. public boolean absolute(int row): được sử dụng để di chuyển

con trỏ đến số mẫu tin thứ row.

6. public boolean relative(int row): được sử dụng để di chuyển con trỏ đến mẫu tin tương đối. Nếu row là số âm thì di chuyển con trỏ đến mẫu tin trước mẫu tin hiện tại, ngược lại di chuyển mẫu tin đến sau mẫu tin hiện tại.

7. public void moveToInsertRow()

Di chuyển con trỏ tới một hàng đặc biệt trong ResultSet mà có thể được sử dụng để chèn một hàng mới vào trong cơ sở dữ liệu.

Chú ý: Để di chuyển con trỏ mẫu tin bằng các phương thức previous(), first(), last(), absolute(), relative(), moveToInsertRow() ta phải chỉ rõ loại ResultSet (resultsetType) và kiểu của ResultSet (resultSetConcurrency) trong phương thức prepareStatement:

public PreparedStatement prepareStatement(String sql, int resultSetType, int resultSetConcurrency)throws SQLException

Trong đó: resultsetType: chỉ dùng 1 trong 2 kiểu:

- ResultSet.TYPE _SCROLL _INSENSITIVE
- ResultSet.TYPE _SCROLL _SENSITIVE

resultSetConcurrency là 1 trong 2 kiểu:

- ResultSet.CONCUR _READ _ONLY:
- ResultSet.CONCUR _UPDATABLE:

Ví dụ 3.9.2. Giả sử bảng NhanVien đã có dữ liệu như Hình 3.14. Hãy hiển thị họ tên của người đầu tiên, người cuối cùng, người thứ 4, người đứng trước 2 mẫu tin tính từ mẫu tin số 4. người đứng sau 2 mẫu tin tính từ mẫu tin số 4.

```
1 import java.sql.Connection;
2 import java.sql.DriverManager;
3 import java.sql.PreparedStatement;
4 import java.sql.ResultSet;
5
```

```
6  import dao.DungChung;
7  public class ViDu392 {
8  public static void main(String[] args) {
9      Connection  cn=null;
10     try {
11         cn=DungChung.KetNoi();
12         //Tạo câu lệnh SQL Lấy về tất cả các nhân viên
13         String sql="select * from nhanvien";
14         PreparedStatement cmd= cn.prepareStatement(sql,
               ResultSet .TYPE_SCROLL_INSENSITIVE,
               ResultSet.CONCUR_READ_ONLY);
15         //Thực thi câu lệnh
16         ResultSet rs= cmd.executeQuery();
17         //Đưa con trỏ về mẫu tin đầu tiên;
18         rs.first();
19         //Lấy ra HoTen mẫu tin thứ 1
20         String Hoten=rs.getString("HoTen");
21         System.out.println("Họ tên người đầu tiên: "+ Hoten);
22         System.out.println("--------------------");
23
24         //Đưa con trỏ về mẫu tin cuối cùng;
25         rs.last();
26         //Lấy ra HoTen mẫu tin thứ cuối cùng
27         Hoten=rs.getString("HoTen");
28         System.out.println("Họ tên người cuối cùng: "+ Hoten);
29         System.out.println("--------------------");
30         //Đưa con trỏ về mẫu tin thứ 4;
31         rs.absolute(4);
32         //Lấy ra HoTen mẫu tin thứ 4
33         Hoten=rs.getString("HoTen");
34         System.out.println("Họ tên người người thứ
               4: "+ Hoten);
35         System.out.println("--------------------");
36
```

```
37          //Đưa con trỏ về trước 2 mẫu tin tính từ mẫu tin số 4
38          rs.absolute(4);
39          rs.relative(-2);
40          //Lấy ra HoTen mẫu tin thứ 2
41          Hoten=rs.getString("HoTen");
42          System.out.println("Họ tên người thứ 2: "+ Hoten);
43          System.out.println("--------------------");
44
45          //Đưa con trỏ về sau 2 mẫu tin tính từ mẫu tin số 4
46          rs.absolute(4);
47          rs.relative(2);
48          //Lấy ra HoTen mẫu tin thứ 6
49          Hoten=rs.getString("HoTen");
50          System.out.println("Họ tên người thứ 6: "+ Hoten);
51          System.out.println("--------------------");
52          rs.close();// Đóng ResultSet
53          cn.close();//Đóng kết nối
54          System.out.println("Đã hoàn thành các công việc");
55
56      } catch (Exception e) {
57          e.printStackTrace();
58          System.out.print("Chương trình bị lỗi");
59      }
60   }
61  }
```

Kết quả hiển thị của chương trình:

3.9.3. Các phương thức cập nhật dữ liệu

Để cập nhật dữ liệu trong ResultSet, ta sử dụng phương thức có dạng:

public void updateXXX(int columnIndex, XXX value);

hoặc

```
Họ tên người đầu tiên: Nguyễn Hoàng Hà
---------------------------------
Họ tên người cuối cùng: Phạm Bình Minh
---------------------------------
Họ tên người người thứ 4: Lê Viết Mẫn
---------------------------------
Họ tên người thứ 2: Lê Văn Trung
---------------------------------
Họ tên người thứ 6: Phạm Bình Minh
---------------------------------
Đã hoàn thành các công việc
```

Hình 3.16. *Kết quả chương trình*

public void updateXXX(String columnName, XXX value);

Trong đó, **XXX** là tên kiểu dữ liệu; **columnIndex** và **columnName** là chỉ số của cột hoặc tên cột cần cập nhật dữ liệu; value là giá trị cần cập nhật. ví dụ: updateInt, updateString, updateDouble, updateDate, ...

Để cập nhật dữ liệu vào CSDL, khi lấy dữ liệu về ta phải sử dụng kiểu con trỏ (resultSetType) là ResultSet.TYPE_SCROLL_ SENSITIVE và loại con trỏ (resultSetConcurrency) là ResultSet.CONCUR_ UPDATABLE.

Các phương thức update này không cập nhật vào cơ sở dữ liệu. Để cập nhật dữ liệu vào cơ sở dữ liệu, ta sử dụng thêm phương thức updateRow() hoặc insertRow() nếu thêm mới 1 mẫu tin.

Một số phương thức thường dùng để cập nhật dữ liệu:

1. public void updateInt(int columnIndex hoặc String columnName, int x):

Cập nhật giá trị nguyên **x** vào mẫu tin hiện tại của cột có chỉ số **columnIndex** hoặc có tên cột là **columnName**.

2. public void updateString(int columnIndex hoặc String columnName, String x)

Cập nhật chuỗi **x** vào mẫu tin hiện tại của cột có chỉ số **columnIndex** hoặc có tên cột là **columnName**.

3. public void updateBoolean(int columnIndex hoặc String columnName, Boolean x)

Cập nhật giá trị **x** vào mẫu tin hiện tại của cột có chỉ số **columnIndex** hoặc có tên cột là **columnName**

4. public void updateDate(int columnIndex hoặc String columnName, Date x)

Cập nhật ngày **x** vào mẫu tin hiện tại của cột có chỉ số **columnIndex** hoặc có tên cột là **columnName**

5. public void updateDouble(int columnIndex hoặc String columnName, Double x)

Cập nhật số thực **x** vào mẫu tin hiện tại của cột có chỉ số **columnIndex** hoặc có tên cột là **columnName**

Ví dụ 3.9.3. Viết chương trình để nhập vào 1 mã nhân viên. Tìm xem có nhân viên có mã này hay không. Nếu có thì nhập vào 1 ngày sinh và cập nhập lại ngày sinh cho nhân viên tìm được.

```
1  import java.sql.Connection;
2  import java.sql.PreparedStatement;
3  import java.sql.ResultSet;
4  import java.text.SimpleDateFormat;
5  import java.util.Date;
6  import java.util.Scanner;
7
8  import dao.DungChung;
9  public class ViDu3_9_3 {
10
11 public static void main(String[] args) {
12 Connection  cn=null;
13 try {
14     //Kết nối vào CSDL
15     cn=DungChung.KetNoi();
16     //Nhập vào từ bàn phím 1 mã nhân viên
17     System.out.println("Nhập vào 1 mã nhân viên:");
18     Scanner nhap= new Scanner(System.in);
19     String manv=nhap.next();
```

```
20
21      //Tạo câu lệnh SQL để lấy về nhân viên có mã là manv
22      String sql="select * from nhanvien where Manv=?";
23      //Tạo câu lệnh prepareStatement với:
24      //Kiểu con trỏ: TYPE_SCROLL_SENSITIVE
25      // và loại con trỏ: CONCUR_UPDATABLE
26      PreparedStatement cmd=
            cn.prepareStatement(sql,ResultSet.TYPE_SCROLL_SENSITIVE,
            ResultSet.CONCUR_UPDATABLE);
27      //Truyền tham số vào câu lệnh
28      cmd.setString(1, manv);
29      //Thực thi câu lệnh
30      ResultSet rs= cmd.executeQuery();
31
32      //Nếu trong rs không có dữ liệu
33      if (!rs.next())
34          System.out.println("Không tim ra nhân viên này");
35      else{
36          System.out.println("Nhập vào 1 ngày sinh theo dạng:
                yyyy-MM-dd");
37          String ngay=nhap.next();
38          //Đổi chuỗi sang ngày của Util
39          SimpleDateFormat dd= new SimpleDateFormat(
                "yyyy-MM-dd");
40          Date ngayUtil=dd.parse(ngay);
41          //Đổi ngày của Util sang ngày của sql
42          //và cập nhật lại ngày sinh
43          rs.updateDate("NgaySinh", new java.sql.Date(
                ngayUtil.getTime()));
44          rs.updateRow();
45      }
46      rs.close();// Đóng ResultSet
47      cn.close();//Đóng kết nối
48  } catch (Exception e) {
```

```
49      e.printStackTrace();
50      System.out.print("Chương trình bị lỗi");
51     }
52    }
53  }
```

3.9.4. Thêm và xóa một mẫu tin

Để thêm một mẫu tin vào cơ sở dữ liệu thì ta thực hiện các bước:

- Di chuyển con trỏ về sau mẫu tin cuối cùng và tạo ra 1 mẫu tin: moveToInsertRow();
- Cập nhật lại dữ liệu vào mẫu tin mới thêm, sử dụng các phương thức updateXXX(int columnIndex, XXX value);
- Lưu dữ liệu vào cơ sở dữ liệu, sử dụng phương thức insertRow();

Ví dụ 3.9.4. Viết chương trình để thêm vào 1 nhân viên.

```
1  import java.sql.*;
2  import java.text.SimpleDateFormat;
3  import java.util.Date;
4  import java.util.Scanner;
5
6  import dao.DungChung;
7  public class ViDu3_9_4 {
8
9  public static void main(String[] args) {
10 Connection  cn=null;
11 try {
12     //Kết nối vào CSDL
13     cn=DungChung.KetNoi();
14     //Nhập vào từ bàn phím 1 mã nhân viên
15     System.out.println("Nhập vào 1 mã nhân viên:");
16     Scanner nhap= new Scanner(System.in);
17     String manv=nhap.nextLine();
```

```
18      //Tạo câu lệnh SQL để lấy về nhân viên có mã là manv
19      String sql="select * from nhanvien where Manv=?";
20      //Tạo câu lệnh prepareStatement
21      PreparedStatement cmd=
            cn.prepareStatement(sql,ResultSet.TYPE_SCROLL_SENSITIVE,
            ResultSet.CONCUR_UPDATABLE);
22      //Truyền tham số vào câu lệnh
23      cmd.setString(1, manv);
24      //Thực thi câu lệnh
25      ResultSet rs= cmd.executeQuery();
26      //Nếu có dữ liệu trong rs
27      if (rs.next())
28          System.out.println("Đã tồn tại nhân viên có mã:
                "+manv);
29      else{
30          System.out.println("Nhập họ tên nhân viên:");
31          String HoTen=nhap.nextLine();
32
33          System.out.println("Nhập giới tính (true/false):");
34          Boolean GioiTinh=Boolean.parseBoolean(nhap.nextLine());
35          System.out.println("Nhập vào 1 ngày sinh theo dạng:
                yyyy-MM-dd");
36          String ngay=nhap.nextLine();
37          //Đổi chuỗi sang ngày của Util
38          SimpleDateFormat dd= new
                SimpleDateFormat("yyyy-MM-dd");
39          Date ngayUtil=dd.parse(ngay);
40
41          System.out.println("Nhập hệ số lương:");
42          Double hsl= Double.parseDouble(nhap.nextLine());
43
44          System.out.println("Nhập mã đơn vị:");
45          String Madv=nhap.nextLine();
46          //Di chuyển con trỏ về sau mẫu tin cuối cùng và tạo ra
                1 mẫu tin
```

```
47          rs.moveToInsertRow();
48          //Cập nhật lại dữ liệu vào mẫu tin mới thêm
49          rs.updateString("Manv", manv);
50          rs.updateString("HoTen", HoTen);
51          rs.updateBoolean("GioiTinh", GioiTinh);
52          rs.updateDate("NgaySinh", new
                java.sql.Date(ngayUtil.getTime()));
53          rs.updateDouble("Hsl", hsl);
54          rs.updateString("MaDv", Madv);
55          //Lưu vào CSDL
56          rs.insertRow();
57      }
58
59      rs.close();// Đóng ResultSet
60      cn.close();//Đóng kết nối
61  } catch (Exception e) {
62      e.printStackTrace();
63      System.out.print("Chương trình bị lỗi");
64     }
65    }
66  }
```

Để xóa một mẫu tin ra khỏi cơ sở dữ liệu thì ta thực hiện các bước:

- Di chuyển con trỏ đến mẫu tin cần xóa (tìm mẫu tin cần xóa)
- Xóa và lưu dữ liệu vào cơ sở dữ liệu, sử dụng phương thức deleteRow().

Ví dụ 3.9.5. Viết chương trình để nhập vào 1 mã nhân viên và xóa nhân viên có mã này ra khỏi CSDL.

```
1  import java.sql.Connection;
2  import java.sql.PreparedStatement;
3  import java.sql.ResultSet;
4  import java.util.Scanner;
```

```
import dao.DungChung;
public class ViDu3_9_5 {
public static void main(String[] args) {
Connection  cn=null;
try {
        cn=DungChung.KetNoi();//Kết nối vào CSDL
    //Nhập vào từ bàn phím 1 mã nhân viên
    System.out.println("Nhập vào 1 mã nhân viên:");
    Scanner nhap= new Scanner(System.in);
    String manv=nhap.nextLine();
    //Tạo câu lệnh SQL để lấy về nhân viên có mã là manv
    String sql="select * from nhanvien where Manv=?";
    //Tạo câu lệnh prepareStatement
    PreparedStatement cmd=
        cn.prepareStatement(sql,ResultSet.TYPE_SCROLL_SENSITIVE,
        ResultSet.CONCUR_UPDATABLE);
    cmd.setString(1, manv);//Truyền tham số vào câu lệnh
    //Thực thi câu lệnh
    ResultSet rs= cmd.executeQuery();
    //Nếu không tim thấy
    if (!rs.next())
        System.out.println("Không tìm ra nhân viên có
            mã: "+manv);
    else{
        // Xóa dòng hiện tại và lưu vào CSDL
        rs.deleteRow();
    }
    rs.close();// Đóng ResultSet
    cn.close();//Đóng kết nối
} catch (Exception e) {
    e.printStackTrace();
    System.out.print("Chương trình bị lỗi");
    }
  }
```

```
}
```

3.9.5. Lấy về bảng mô tả trong ResultSet

Các phần trên tập trung vào truy xuất dữ liệu trong ResultSet, để lấy thông tin (metadata) trong ResultSet như tên bảng, tổng số cột, tên cột, kiểu của cột, ...ResultSet hỗ trợ hàm:

public ResultSetMetaData getMetaData();

Một số phương thức trong ResultSetMetaData:

1. public int getColumnCount()throws SQLException: trả về tổng số cột trong đối tượng ResultSet.

2. public String getColumnName(int index)throws SQLException: trả về tên cột tại chỉ mục đã cho.

3. public String getColumnTypeName(int index)throws SQLException : trả về tên kiểu của cột tại chỉ mục đã cho.

4. public Boolean isAutoIncrement(int column)

Xác định xem cột đã cho có phải là tự động tăng (auto-increment) hay không.

Ví dụ 3.9.6. Viết hàm GetBang(String TenBang) để hiển thị tên cột và dữ liêu của bảng: TenBang. Sau đó viết chương trình gọi lại hàm GetBang để hiển thi tên cột vào dữ liệu của bảng NhanVien và DonVi.

```
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.ResultSetMetaData;

public class ViDu3_9_6 {
public static void GetBang(String TenBang) throws Exception{
    Connection cn=DungChung.KetNoi();//Tạo đường kết nối
    // Tạo câu lênh SQl để lấy về dữ liệu của bảng: TenBang
    String sql="select * from "+ TenBang;
    //Tạo và thực thi câu lệnh
```

```
11      PreparedStatement cmd=cn.prepareStatement(sql);
12      ResultSet rs=cmd.executeQuery();
13      // Lấy về thông tin của rs (metadata)
14      ResultSetMetaData meta=rs.getMetaData();
15      //' Lấy về tổng số cột
16      int sc=meta.getColumnCount();
17      // Hiển thị tên các cột của bảng: TenBang
18      for(int i=1;i<=sc;i++)
19          System.out.printf("%-20s",meta.getColumnName(i));
20      System.out.println();
21      // Duyệt qua các mẫu tin trong rs
22      while(rs.next()){
23          // Duyệt qua các cột trên mỗi mẫu tin
24          for(int i=1;i<=sc;i++)
25              // Hiển thị giá trị trên cột i
26              System.out.printf("%-20s",rs.getString(i));
27          System.out.println();
28      }
29      rs.close();
30      cn.close();
31  }
32  public static void main(String[] args) {
33  try {
34      System.out.println("Bảng Nhân viên:");
35      DungChung.GetBang("NhanVien");
36      System.out.println("Bản Đơn Vị:");
37      DungChung.GetBang("DonVi");
38  } catch (Exception e) {
39      e.printStackTrace();
40    }
41   }
42  }
```

Kết quả hiển thị của chương trình như Hình 3.17

Bảng Nhân viên:

MaNv	HoTen	GioiTinh	NgaySinh	Hsl	MaDv
Nv1	Nguyễn Hoàng Hà	1	1976-11-18	2.34	cntt
Nv2	Lê Văn Trung	1	1980-10-10	3.33	cntt
Nv3	Trần Nguyên Phong	1	1976-12-10	3.33	cntt
Nv4	Lê Viết Mẫn	1	1992-10-12	3.66	toan
Nv5	Hà Lê Nguyên	0	1980-04-03	2.67	van
Nv6	Phạm Bình Minh	1	1979-02-09	4.32	van

Bản Đơn Vị:

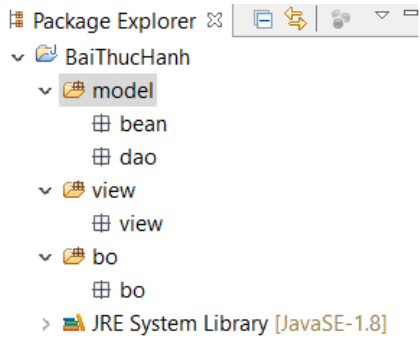
MaDonVi	TenDonVi
cntt	Tin học
cntt3	Công nghệ thông tin2
toan	Khoa Toán

Hình 3.17. *Dữ liệu bảng Nhân viên và Đơn vị*

3.10. Bài tập thực hành

3.10.1. Bài thực hành 1

Sử dụng Eclipse để tạo các Source Folder và các gói như Hình 3.18



Hình 3.18. *Mô hình 3 mức*

Một số quy tắc khi lập trình:

model: Chứa các gói và lớp tách riêng với tầng bo (business objects) và tầng view:

model.bean: Chứa các thực thể, gồm các dữ liệu (private), kèm theo các phương thức set/get.

model.dao: Thực hiện các công việc liên quan đến CSDL như kết nối, lấy dữ liệu, truy vấn, chỉnh sửa, thêm xóa dữ liệu trực tiếp với database

bo: Truyền yêu cầu từ view chuyển qua dao, lấy dữ liệu từ dao về cho view. Nhiệm vụ của tầng này là xử lý các yêu cầu nghiệp vụ.

view: Dùng để nhập xuất dữ liệu của người dùng

bean chứa các thực thể, có thể sử dụng ở bất cứ nơi nào cần thiết: dao, bo và view

Các kết nối đến CSDL chỉ thực hiện ở **model.dao**, các tầng khác không liên quan đến CSDL

dao chỉ cho phép gọi từ **bo**, các nơi khác không được gọi đến **dao**

bo chỉ cho phép gọi từ **view**, các nơi khác không được gọi **bo**

view: chỉ nhận và gửi dữ liệu từ **bo**

3.10.2. Bài thực hành 2

Tại **model.dao**, hãy tạo ra lớp DungChung và viết các hàm:

- Kết nối vào cơ sỡ dữ liệu;
- Lấy dữ liệu của một bảng bất kỳ;
- Lấy dữ liệu của một bảng bất kỳ và nạp toàn bộ dữ liệu vào 1 DefaultTableModel.

Hướng dẫn giải:

Tại gói dao, tạo ra lớp DungChung.

```
1  package dao;
2
3  import java.sql.Connection;
4  import java.sql.DriverManager;
5  import java.sql.PreparedStatement;
6  import java.sql.ResultSet;
7  import java.sql.ResultSetMetaData;
8
9  import javax.swing.table.DefaultTableModel;
10
11 public class DungChung {
12     public static Connection cn;
13     // Hàm kết nối vào máy chủ: tenServer; cơ sở dữ liệu:
14     //  database; tên đăng nhập: user và mật khẩu pass
15     //  nếu kết nối thành công, đường kết nối được lưu trong cn
```

```
16      public static void KetNoi(String tenServer, String
            database, String user, String pass) throws Exception{
17      // tham khảo ví dụ 3.4.9 để viết:
18          // Nạp trình điều khiển
19          ...
20          //  tạo chuỗi urlDatabase để kết nối vào CSDL
21          ...
22          //  cn= DriverManager.getConnection(urlDatabase)
23          ...
24  }
25  //  Hàm trả về 1 ResultSet chứa dữ liệu của 1 bảng
26  public static ResultSet getBang(String TenBang)throws
        Exception{
27      // tạo câu lệnh sql để lấy về toàn bộ dữ liệu của
28      // bảng có tên: TenBang
29      ...
30      // tạo câu lệnh PreparedStatement
31      ...
32      //  cho thực hiện câu lệnh và trả về 1 ResultSet
33      ...
34  }
35
36  // Hàm lấy dữ liệu của bảng có tên: TenBang. Sau đó nạp toàn
37  //  bộ dữ liệu của bảng vào 1 DefaultTableModel
38  public static DefaultTableModel loadBang(String TenBang)
        throws Exception{
39      // Tạo ra 1 DefaultTableModel
40      DefaultTableModel dt= new DefaultTableModel();
41      // Lấy dữ liệu về
42      ResultSet rs=getBang(TenBang);
43      // Lấy về thông tin của rs (metadata)
44      ResultSetMetaData meta=rs.getMetaData();
45      // Lấy về tổng số cột
46      int socot=meta.getColumnCount();
```

```
47      // Nạp tên các cột của bảng: TenBang vào DefaultTableModel
48      for(int i=1;i<=socot;i++)
49          dt.addColumn( meta.getColumnName(i));
50
51      // Duyệt qua các mẫu tin trong rs
52      while(rs.next()){
53          Object[] t=new Object[socot];
54          // Duyệt qua các cột trên mỗi mẫu tin
55          for(int i=1;i<=socot;i++)
56              t[i-1]=rs.getString(i);
57          dt.addRow(t);
58      }
59      rs.close();
60      cn.close();
61      return dt;
62    }
63  }
```

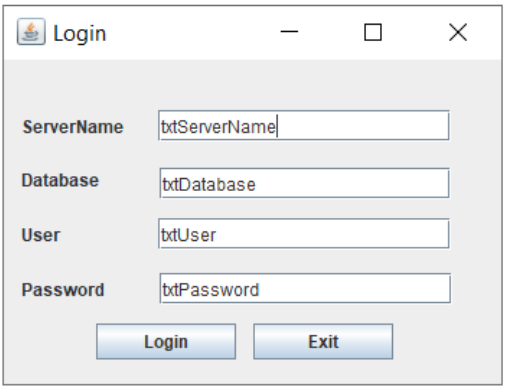
3.10.3. Bài thực hành 3

Tại **bo**, hãy tạo ra lớp HeThongBo, sau đó gọi hàm KetNoi và loadBang từ lớp DungChung của gói model.dao.

```
1  package bo;
2
3  import javax.swing.table.DefaultTableModel;
4
5  import dao.DungChung;
6
7  public class HeThongBo {
8      public static void KetNoi(String tenServer, String
           database, String user, String pass) throws Exception{
9          DungChung.KetNoi(tenServer, database, user, pass);
10     }
```

```
11      public static DefaultTableModel loadBang(String TenBang)
            throws Exception{
12          return DungChung.loadBang(TenBang);
13      }
14  }
```

3.10.4. Bài thực hành 4



Hình 3.19. *Frame đăng nhập hệ thống*

3.11. Giới thiệu Hibernate

NHÀ XUẤT BẢN ĐẠI HỌC HUẾ

07 Hà Nội, Huế - Điện thoại: 0234.3834486; Fax: 0234.3819886

Website: http://nhaxuatban.hueuni.edu.vn

Chịu trách nhiệm xuất bản

Quyền Giám đốc: TS. Trần Bình Tuyên

Chịu trách nhiệm nội dung

Quyền Tổng biên tập: TS. Nguyễn Chí Bảo

Phản biện giáo trình

PGS. TS. Trương Công Tuấn

TS. Nguyễn Công Hào

Biên tập viên

Tôn Nữ Quỳnh Chi

Biên tập kỹ thuật

Ngô Văn Cường

Trình bày, minh họa

Minh Hoàng

Sửa bản in

Nguyễn Hoàng Hà

Đối tác liên kết xuất bản

Đại học Huế, 03 Lê Lợi, thành phố Huế

GIÁO TRÌNH
XML VÀ ỨNG DỤNG

In 180 bản khổ 17 × 25 cm tại công ty TNHH MTV in và dịch vụ Thanh Minh, 99 Phan Văn Trường, phường Vĩ Dạ, thành phố Huế. Số xác nhận đăng ký xuất bản: 1548 - 2019/CXBIPH/07-12/ĐHH. Quyết định xuất bản số: 38/QĐ/ĐHH-NXB cấp ngày 21/05/2019. In xong và nộp lưu chiểu năm 2019.

Mã số ISBN: 978-604-974-152-4