CHUYÊN ĐỀ JAVA

JDBC

(JAVA DATABASE CONNECTIVITY)

Nguyễn Hoàng Anh – nhanh@fit.hcmuns.edu.vn

Nội dung

- Giới thiệu JDBC
- Các Class và Interface của JDBC API
- Các bước cơ bản sử dụng JDBC
- Một số kỹ thuật khác
- Hỏi và đáp

GIỚI THIỆU JDBC

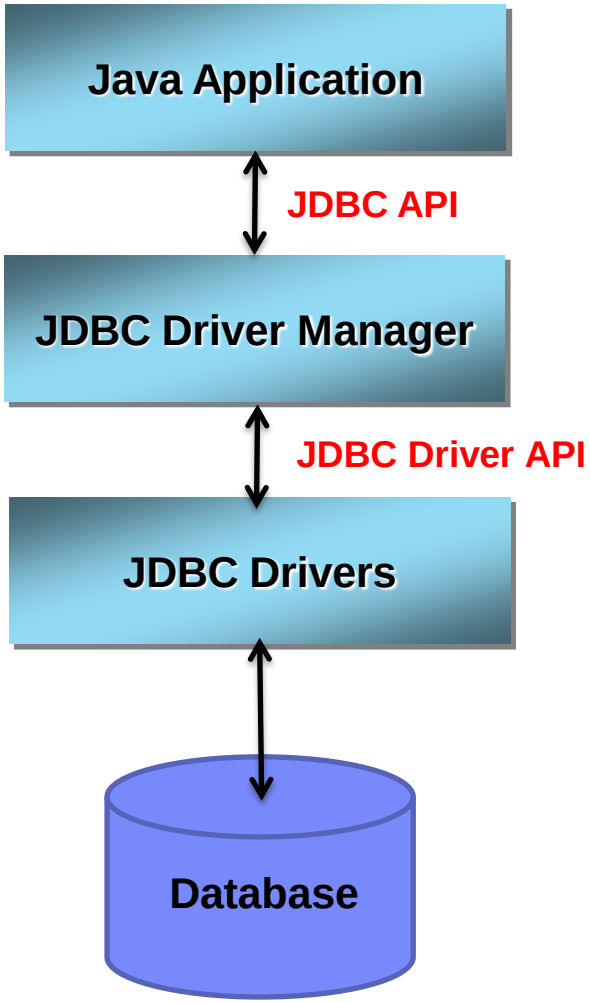
JDBC

- JDBC (Java database connectivity) cung cấp cho java developer tập các interface chuẩn dùng để truy xuất dữ liệu quan hệ.
- JDBC được phát triển bởi JavaSoft

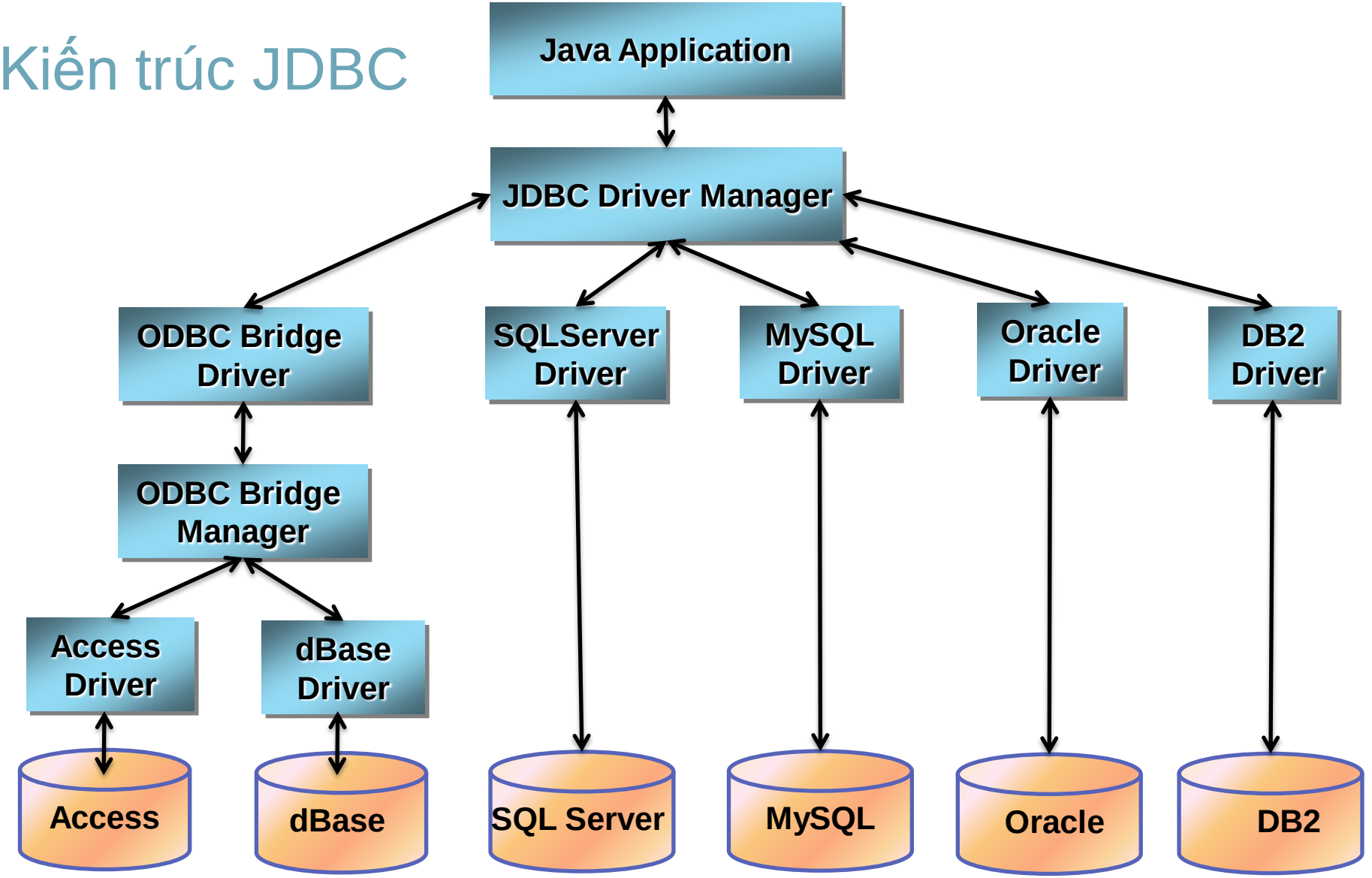
Kiến trúc JDBC

- Khi làm việc với JDBD, java developer sẽ làm việc như nhau đối với các hệ cơ sở dữ liệu khác nhau.
- Java developer không phụ thuộc vào một hệ cơ sở dữ liệu cụ thể nào.
- Java developer không cần phải quan tâm đến sự khác nhau khi giao tiếp với các HQTCSDL khác nhau.

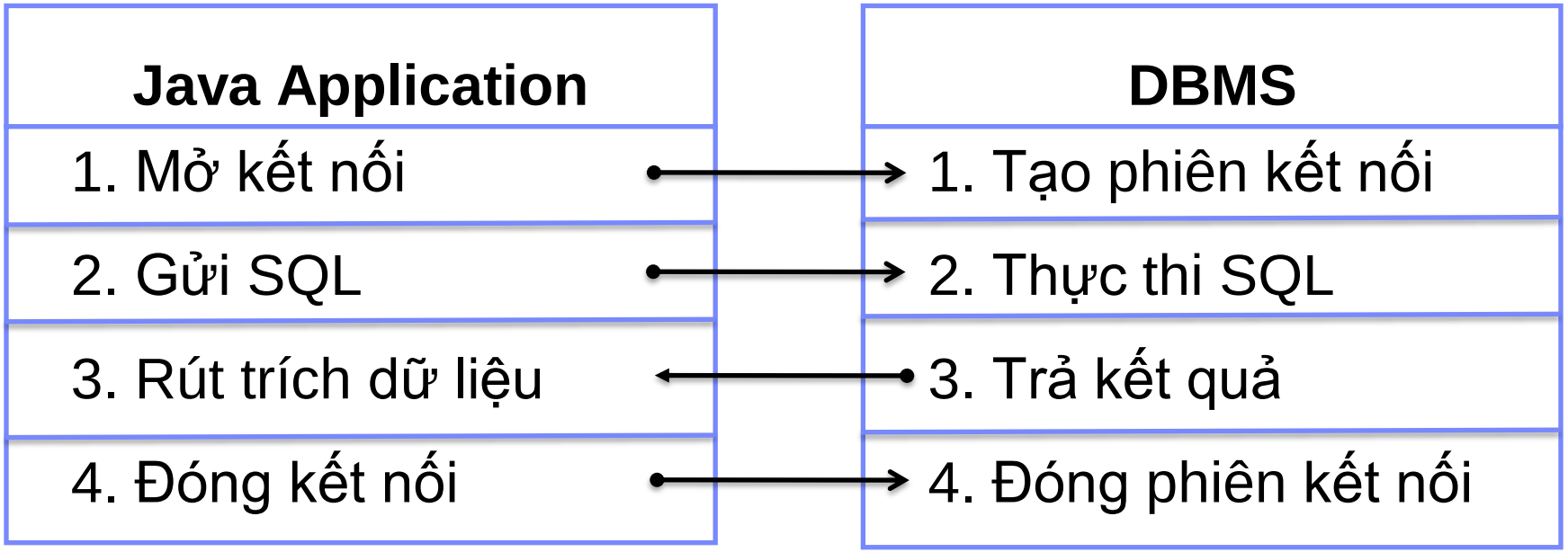
Kiến trúc JDBC



Kiến trúc JDBC



JDBC


Java Application
1. Mở kết nối
2. Gửi SQL
3. Rút trích dữ liệu
4. Đóng kết nối
DBMS
1. Tạo phiên kết nối
2. Thực thi SQL
3. Trả kết quả
4. Đóng phiên kết nối

JDBC API

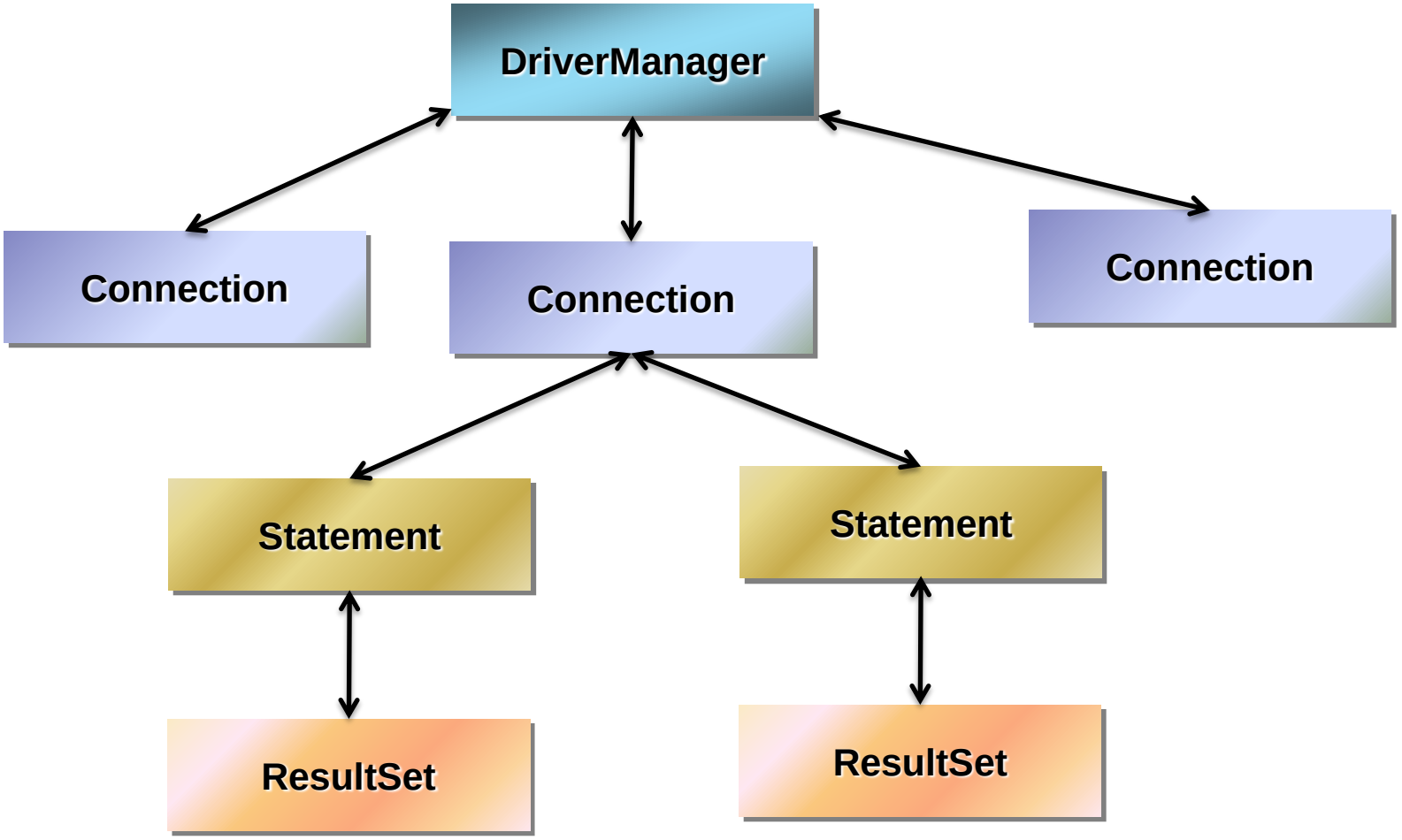
JDBC Interfaces

- Class và Interface của JDBC API thuộc gói java.sql
- DriverManager dùng để nạp các driver và tạo Connection đến cơ sở dữ liệu.
- Driver: Driver của cơ sở dữ liệu, mỗi JDBC Driver đều cài đặt lại Interface này.
- Connection :Thiết lập một Connection đến cơ sở dữ liệu và cho phép tạo các Statement .
- Statement: Gắn kết với một connection đến cơ sở dữ liệu và cho phép thực thi các câu lệnh SQL. CallableStatement tương tự Statement nhưng áp dụng cho Store procedures.

JDBC API

- PreparedStatement: Tương tự như Statement nhưng áp dụng cho Precompiled SQL.
- ResultSet: Cung cấp thông tin rút trích từ cơ sở dữ liệu , cho phép truy xuất các dòng dữ liệu.
- ResultSetMetaData: Cung cấp các thông tin như kiểu dữ liệu và các thuộc tính trong Resultset.
- DatabaseMetaData: Cung cấp các thông tin của cơ sở dữ liệu kết nối.
- SQLException: Cung cấp thông tin các ngoại lệ xảy ra khi tương tác với cơ sở dữ liệu.

JDBC API


DriverManager
Connection
Connection
Connection
Statement
Statement
ResultSet
ResultSet

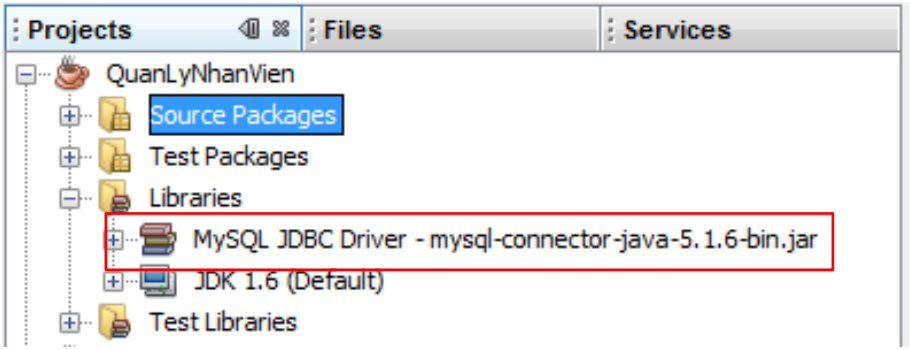
CÁC BƯỚC SỬ DỤNG JDBC

Các bước cơ bản sử dụng JDBC

- Đăng ký Driver
- Mở kết nối đến cơ sở dữ liệu
- Tạo và thực các câu lệnh SQL
- Xử lý các ngoại lệ
- Đóng kết nối cơ sở dữ liệu

Đăng ký Driver

- Bước 1: Thêm thư viện JDBC Driver vào ứng dụng

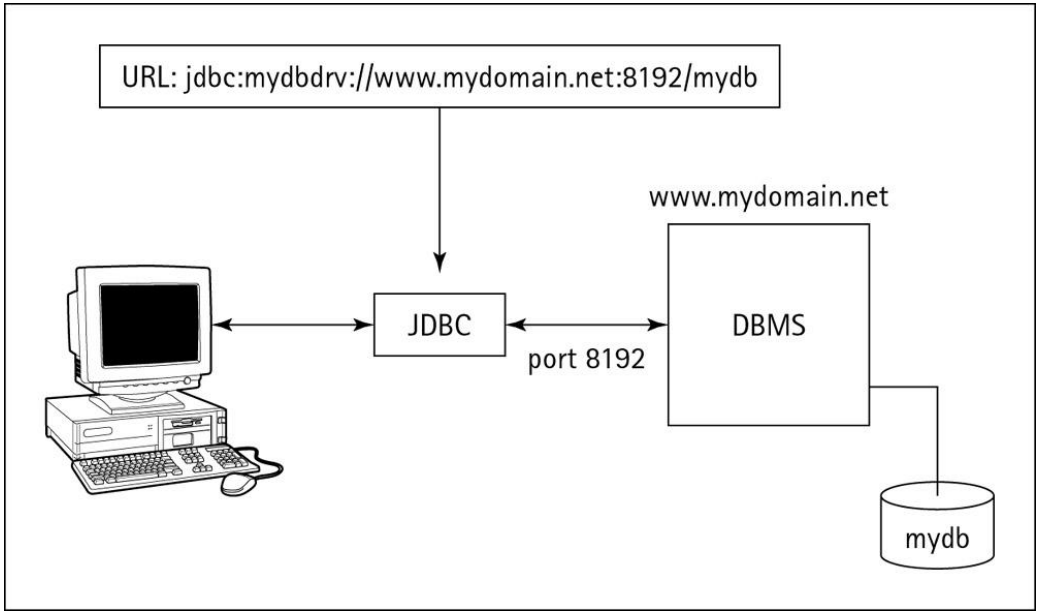


- Bước 2: Đăng ký:
  - Class.forName : Class.forName("org.gjt.mm.mysql.Driver");
  - DriverManager.registerDriver

```
Driver  = new org.gjt.mm.mysql.Driver()
DriverManager.registerDriver(new org.gjt.mm.mysql.Driver());
```

Mở kết nối cơ sở dữ liệu

- Đối tượng Connection được tạo từ getConnection của DriverManager
- Connection getConnection (String url);
- Connection getConnection (String url, String user, String password);
- Connection getConnection (String url, java.util.Properties info);



Mở kết nối cơ sở dữ liệu

```
Driver driver = new org.gjt.mm.mysql.Driver()
DriverManager.registerDriver(driver);
String conString =
  "jdbc:mysql://localhost:3306/QuanLyNhanVien";
Properties info= new Properties();
info.setProperty("characterEncoding", "utf8");
info.setProperty("user", "root");
info.setProperty("password", "root");
Connection connection =
  DriverManager.getConnection(conString, info);
```

Các câu lệnh SQL

- INSERT : Thêm dữ liệu vào bảng
- DELETE : Xóa dữ liệu từ bảng
- UPDATE : Cập nhật dữ liệu vào bảng
- SELECT : Truy vấn dữ liệu từ bảng
- …

Tạo và thực thi các câu lệnh SQL

- Statement statement = connection.createStatement() ;
- ResultSet executeQuery (String sql)
- int executeUpdate (String sql)
- boolean execute (String sql)

Thêm dữ liệu vào bảng

```
Statement statement = connection.createStatement();
String sql = "INSERT INTO PHONGBAN (MAPHONGBAN,
  TENPHONGBAN) VALUES ('PBHC', 'Hành Chính')";
int n = statement.executeUpdate (sql);
if (n == 1){
  System.out.println("Thêm phòng ban thành công");
}else{
  System.out.println("Thêm phòng ban thất bại");
}
```

Cập nhật dữ liệu từ bảng

```
Statement statement = connection.createStatement();
String sql = "UPDATE PHONGBAN SET TENPHONGBAN = 'Kỹ
  Thuật' WHERE MAPHONGBAN = 'PBHC'";
int n = statement.executeUpdate (sql);
if (n == 1){
  System.out.println("Cập nhật thành công");
}else{
  System.out.println("Cập nhật thất bại");
}
```

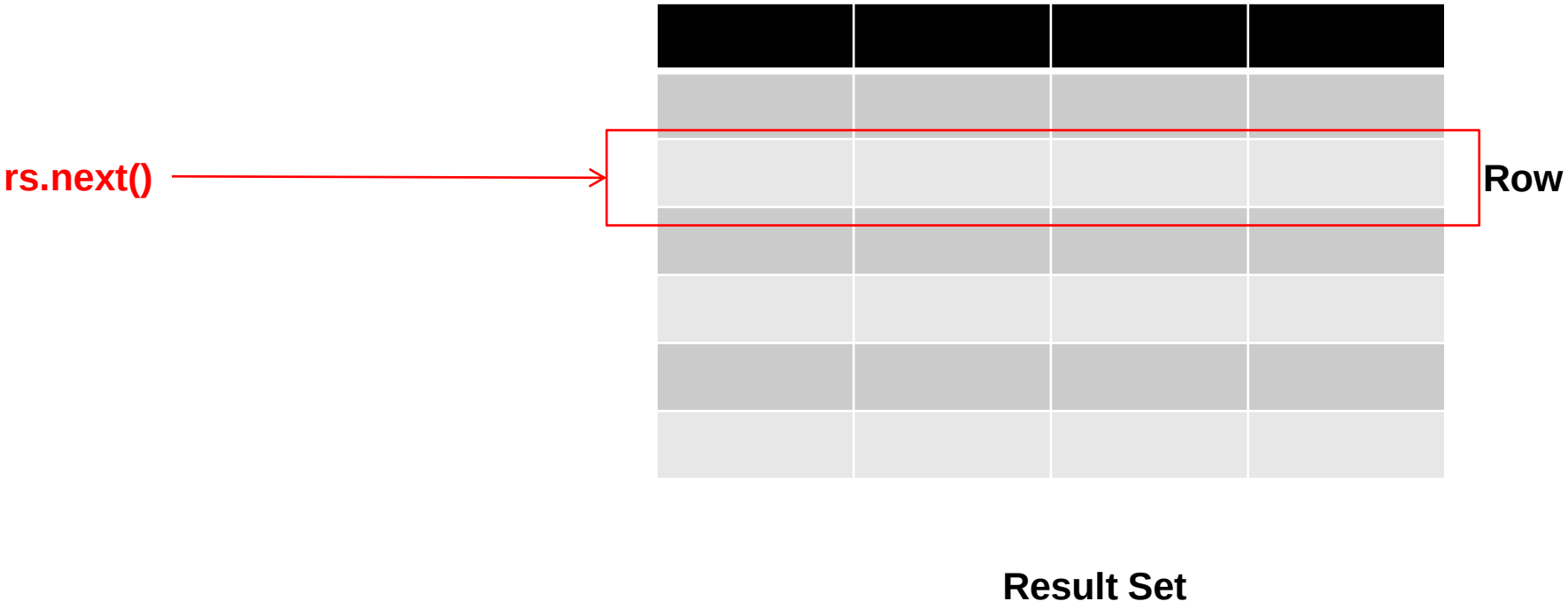
Xóa dữ liệu từ bảng

```
Statement statement = connection.createStatement();
String sql = "DELETE FROM PHONGBAN WHERE MAPHONGBAN
  = 'PBHC'";
int n = statement.executeUpdate (sql);
if (n == 1){
  System.out.println("Xóa thành công");
}else{
  System.out.println("Xóa thất bại");
}
```

Lấy dữ liệu từ bảng

```
Statement statement = connection.createStatement();
String sql = "SELECT * FROM PHONGBAN";
ResultSet rs = statement.executeQuery(sql);
while (rs.next()){
    System.out.println(rs.getString("MAPHONGBAN");
     System.out.println(rs.getString("TENPHONGBAN");
}
```

Lấy dữ liệu từ bảng



Đóng kết nối cơ sở dữ liệu

- connection.close()

Xử lý ngoại lệ

- java.sql.SQLException kế thừa từ java.lang.Exception
- Một số phương thức thông dụng của SQLException
  - int getErrorCode ()
  - String getSQLState ()
  - SQLException getNextException ()
  - String getMessage ()
- Xử lý ngoại lệ bằng cách sử dụng try catch finally

Xử lý ngoại lệ

```
try{
  - Đăng ký driver, Mở kết nối cơ sở dữ liệu
  - Tạo và thực thi các câu lệnh SQL, Đóng kết nối
}catch (SQLException ex){
  while (ex != null) {
    System.out.println("SQLState: "  + ex.getSQLState());
    System.out.println("Message: "   + ex.getMessage());
    System.out.println("ErrorCode: " + ex.getErrorCode());
    System.out.println("");
    ex.getNextException();
  }
}catch(java.lang.Exception ex) {
  ex.printStackTrace();
}
```

MỘT SỐ KỸ THUẬT KHÁC

Tạo SQL bằng phương pháp cộng chuỗi

```
String maPhongBan = Lấy giá trị mã phòng ban;

String tenPhongBan = Lấy giá trị tên phòng ban;

String sql = " INSERT INTO PHONGBAN (MAPHONGBAN,
  TENPHONGBAN) VALUES ('" +
  maPhongBan + "', '" +
  tenPhongBan + "');

int n = statement.executeUpdate(sql);
```

Tạo SQL bằng String.format

```
String maPhongBan = Lấy giá trị mã phòng ban;
String tenPhongBan = Lấy giá trị tên phòng ban;
String sql =
 String.format("INSERT INTO PHONGBAN
                (MAPHONGBAN, TENPHONGBAN VALUES
                ('%s', '%s')",
                 maPhongBan, tenPhongBan );
int n = statement.executeUpdate(sql);
```

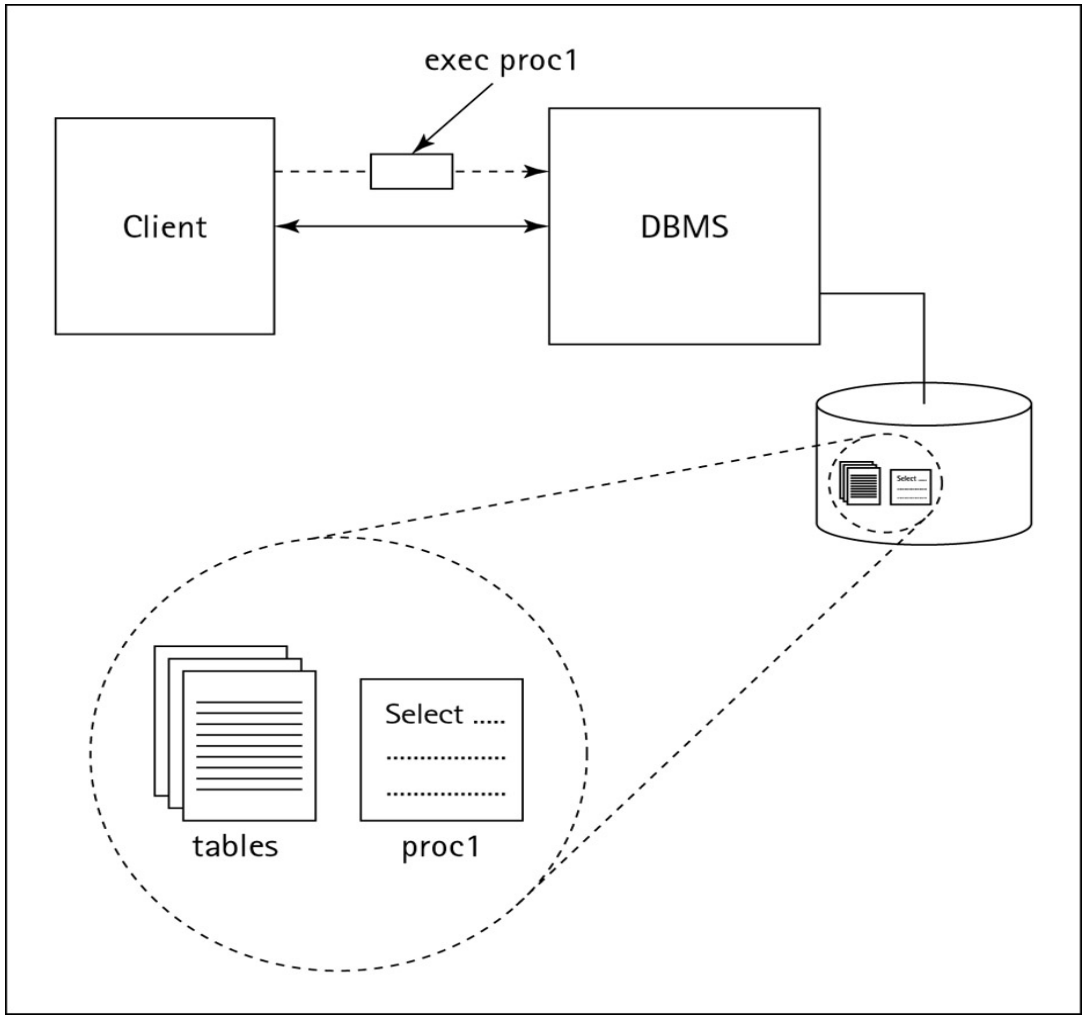
Tạo SQL sử dụng PreparedStatement

```
String maPhongBan = Lấy giá trị mã phòng ban;

String tenPhongBan = Lấy giá trị tên phòng ban;

String sql = "INSERT INTO PHONGBAN (MAPHONGBAN,
  TENPHONGBAN) VALUES (?, ?)";

PreparedStatement statement =
  connection.prepareStatement(sql);

statement.setString(1, maPhongBan);

statement.setString(2, tenPhongBan);

Statement.executeUpdate();
```

PreparedStatement

- `statement.setInt (int parameterIndex, int x)`
- `statement.setString (int parameterIndex, String x)`
- `statement.setDouble (int parameterIndex, Double x)`
- `statement.setDate (int parameterIndex, Date x)`
- …
- Lưu ý: parameterIndex : Bắt đầu từ 1

CallableStatement - Xử lý Store Procedure



CallableStatement - Xử lý Store Procedure

```
DELIMITER$$
CREATE PROCEDURE spThemPhongBan (
                        ma VARCHAR (20),
                        ten VARCHAR (45)
)
BEGIN
  INSERT INTO PHONGBAN(MAPHONGBAN, TENPHONGBAN)
  VALUES (ma, ten);
END$$
DELIMITER;
```

CallableStatement - Xử lý Store Procedure

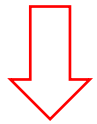
```
String maPhongBan = Lấy giá trị mã phòng ban;
String tenPhongBan = Lấy giá trị tên phòng ban;
String sql = "{call spThemPhongBan (?, ?)}";
CallableStatement statement =
  connection.prepareStatement(sql);
statement.setString("ma", maPhongBan);
statement.setString("ten", tenPhongBan);
Statement.executeUpdate();
```

CallableStatement - Xử lý Store Procedure

- `statement.setInt (int parameterIndex, int x)`
- `statement.setInt (String parameterName, int x)`
- `statement.setString (int parameterIndex, String x)`
- `statement.setString (String parameterName, String x)`
- `statement.setDouble (int parameterIndex, Double x)`
- `statement.setDouble (String parameterName, Double x)`
- `statement.setDate (int parameterIndex, Date x)`
- `statement.setDate (String parameterName, Date x)`
- …
- Lưu ý: parameterIndex : Bắt đầu từ 1
  parameterName : Tên tham số

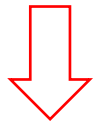
MySQLDataAccessHelper.java

```
public class MySqlDataAccessHelper {
    private Connection connection;
    public void open() {
        try {
            Driver driver = new org.gjt.mm.mysql.Driver();
            DriverManager.registerDriver();
            String conString = "jdbc:mysql://localhost:3306/QuanLyNhanVien";
            Properties pros = new Properties();
            info.setProperty("characterEncoding", "utf8");
            info.setProperty("user", "root");
            this.connection = DriverManager.getConnection(conString, info);
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
    }
```



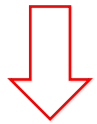
MySQLDataAccessHelper.java

```
public class MySqlDataAccessHelper {
  - - - - - - - - - - - - - - - - - -
  /**
   * Đóng kết nối cơ sở dữ liệu
   */
  public void close() {
          try {
              this.connection.close();
          } catch (Exception ex) {
              System.out.println(ex.getMessage());
          }
  }
  - - - - - - - - - - - - - - - - - -
```



MySQLDataAccessHelper

```
public class MySqlDataAccessHelper {
    - - - - - - - - - - - - - - - - - -
    /**
     * Rút trích dữ liệu
     */
    public ResultSet executeQuery(String sql) {
        ResultSet rs = null;
        try {
            Statement sm = this.connection.createStatement();
            rs = sm.executeQuery(sql);
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
        return rs;
    }
```



MySQLDataAccessHelper

```
public class MySqlDataAccessHelper {
  - - - - - - - - - - - - - - - - - -
  /**
   * Thêm, xóa, cập nhật dữ liệu
   */
    public int executeUpdate(String sql) {
        int num = -1;
        try {
            Statement sm = this.connection.createStatement();
            num = sm.executeUpdate(sql);
        } catch (Exception ex) {
            System.out.println(ex.getMessage());
        }
        return num;
    }
```

TÀI LIỆU THAM KHẢO

Tài liệu tham khảo

- Bernard Van Haecke (2001) : JDBC 3.0 - Java Database Connectivity

HỎI VÀ ĐÁP