MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- Tạo CSDL mẫu và viết các câu lệnh SQL cần thiết
- Lập trình JDBC với Statement, PreparedStatement, CallableStatement

PHẦN I: CSDL MẪU VÀ LẬP TRÌNH JDBC SỬ DỤNG STATEMENT

BÀI 1: CƠ SỞ DỮ LIỆU

1.1 Tạo CSDL HRM chứa 2 bảng Departments (phòng) và Employees (nhân viên)

```
CREATE TABLE Departments( --Phòng ban
    Id CHAR(3) NOT NULL, --Mã phòng
    Name NVARCHAR(50) NOT NULL, --Tên phòng
    Description NVARCHAR(255) NULL, --Mô tả phòng
    PRIMARY KEY(Id)
);

CREATE TABLE Employees( --Nhân viên
    Id VARCHAR(20) NOT NULL, --Mã nhân viên
    Password NVARCHAR(50) NOT NULL, --Mật khẩu
    Fullname NVARCHAR(50) NOT NULL, --Họ và tên
    Photo NVARCHAR(50) NOT NULL, --Hình ảnh
    Gender BIT NOT NULL, --Giới tính
    Birthday DATE NOT NULL, --Ngày sinh
    Salary FLOAT NOT NULL, --Lương cơ bản
    DepartmentId CHAR(3) NOT NULL, --Mã phòng
    PRIMARY KEY(Id),
    FOREIGN KEY(DepartmentId) REFERENCES Departments(Id)
        ON DELETE CASCADE --Xóa dây chuyền theo DepartmentId
        ON UPDATE CASCADE --Sửa dây chuyền theo DepartmentId
);
```

1.2 Viết các câu lệnh SQL truy vấn và thao tác dữ liệu cho mỗi bảng

- ✓ Truy vấn tất cả
- ✓ Truy vấn theo khóa chính
- ✓ Thêm mới
- ✓ Cập nhật theo khóa chính

- ✓ Xóa theo khóa chính

BÀI 2: LẬP TRÌNH JDBC SỬ DỤNG STATEMENT

2.1 Khai báo thư viện phụ thuộc SQL Server Driver

```
<dependency>
        <groupId>com.microsoft.sqlserver</groupId>
        <artifactId>mssql-jdbc</artifactId>
        <version>12.8.0.jre8</version>
</dependency>
```

2.2 Xây dựng lớp tiện ích chứa các phương thức thực hiện truy vấn và thao tác dữ liệu theo hướng dẫn sau:

```
public class Jdbc {
        static String driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
        static String dburl = "jdbc:sqlserver://localhost;database=HRM";
        static String username = "sa";
        static String password = "123456";

        static {
                try { // nạp driver
                        Class.forName(driver);
                } catch (ClassNotFoundException e) {
                        throw new RuntimeException(e);
                }
        }
        /**Mở kết nối*/
        public static Connection getConnection() throws SQLException {
                return DriverManager.getConnection(dburl, username, password);
        }
        /**Thao tác dữ liệu*/
        public static int executeUpdate(String sql) throws SQLException {
                Connection connection = getConnection();
                Statement statement = connection.createStatement();
                return statement.executeUpdate(sql);
        }
        /**Truy vấn dữ liệu*/
        public static ResultSet executeQuery(String sql) throws SQLException {
                Connection connection = getConnection();
                Statement statement = connection.createStatement();
                return statement.executeQuery(sql);
        }
}
```

2.3 Sử dụng thư viện tiện ích vừa xây dựng ở trên để chạy thử các câu lệnh SQL đã viết trong bài 1 như hướng dẫn sau

```
try {
        String sql = "SELECT...";
        ResultSet resultSet = Jdbc.executeQuery(sql);
        while(resultSet.next()) {
                String value = resultSet.getString("column");
                //...
                System.out.println(value);
        }
} catch (Exception e) {
        e.printStackTrace();
}

try {
        String sql = "INSERT..., UPDATE..., DELETE…";
        int rows = Jdbc.executeUpdate(sql);
        System.out.println(rows);
} catch (Exception e) {
        e.printStackTrace();
}
```

PHẦN II: LẬP TRÌNH JDBC VỚI PREPAREDSTATEMENT VÀ CALLABLESTATEMENT

BÀI 3: LẬP TRÌNH JDBC SỬ DỤNG PREPAREDSTATEMENT

3.1 Xây dựng lớp tiện ích chứa các phương thức thực hiện truy vấn và thao tác dữ liệu theo hướng dẫn sau:

```
public class Jdbc {
        static String driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
        static String dburl = "jdbc:sqlserver://localhost;database=HRM";
        static String username = "sa";
        static String password = "123456";

        static {
                try { // nạp driver
                        Class.forName(driver);
                } catch (ClassNotFoundException e) {
                        throw new RuntimeException(e);
```

```
            }
        }
        /**Mở kết nối*/
        public static Connection getConnection() throws SQLException {
            return DriverManager.getConnection(dburl, username, password);
        }
        /**Thao tác dữ liệu*/
        public static int executeUpdate(String sql, Object... values) throws SQLException {
            Connection connection = getConnection();
            PreparedStatement statement = connection.prepareStatement(sql);
            for (int i = 0; i < values.length; i++) {
                statement.setObject(i + 1, values[i]);
            }
            return statement.executeUpdate();
        }
        /**Truy vấn dữ liệu*/
        public static ResultSet executeQuery(String sql, Object... values) throws
SQLException {
            Connection connection = getConnection();
            PreparedStatement statement = connection.prepareStatement(sql);
            for (int i = 0; i < values.length; i++) {
                statement.setObject(i + 1, values[i]);
            }
            return statement.executeQuery();
        }
}
```

3.2 Sử dụng thư viện tiện ích vừa xây dựng ở trên để chạy thử các câu lệnh SQL đã viết trong bài 1 như hướng dẫn sau

```
try {
        String sql = "SELECT...?";
        String values = {...};
        ResultSet resultSet = Jdbc.executeQuery(sql, values);
        while(resultSet.next()) {
            String value = resultSet.getString("column");
            //...
            System.out.println(value);
        }
} catch (Exception e) {
        e.printStackTrace();
}

try {
```

```
        String sql = "INSERT...?, UPDATE...?, DELETE...?";
        String values = {...};
        int rows = Jdbc.executeUpdate(sql, values);
        System.out.println(rows);
} catch (Exception e) {
        e.printStackTrace();
}
```

BÀI 4: LẬP TRÌNH JDBC SỬ DỤNG CALLABLESTATEMENT

4.1 Trong CSDL HRM hãy tạo các stored procedure thực hiện thao tác và truy vấn dữ liệu Departments.

```
--Thêm mới: {CALL spInsert(?, ?, ?)}--
CREATE PROCEDURE spInsert(
        @Id VARCHAR(20),
        @Name NVARCHAR(50),
        @Description NVARCHAR(100)
) AS BEGIN
        INSERT INTO Departments(Id, Name, Description)
                VALUES(@Id, @Name, @Description)
END
--Cập nhật: {CALL spUpdate(?, ?, ?)}--
CREATE PROCEDURE spUpdate(
        @Id VARCHAR(20),
        @Name NVARCHAR(50),
        @Description NVARCHAR(100)
) AS BEGIN
        UPDATE Departments
                SET Name=@Name, Description=@Description
                WHERE Id=@Id
END
--Xóa theo khóa chính: {CALL spDeleteById(?)}--
CREATE PROCEDURE spDeleteById(
        @Id VARCHAR(20)
) AS BEGIN
        DELETE FROM Departments WHERE Id=@Id
END
--Truy vấn tất cả: {CALL spSelectAll()}--
CREATE PROCEDURE spSelectAll()
```

```
AS BEGIN
	SELECT * FROM Departments
END
--Truy vấn theo khóa chính: {CALL spSelectById(?)}--
CREATE PROCEDURE spSelectById(
	@Id VARCHAR(20)
) AS BEGIN
	SELECT * FROM Departments WHERE Id=@Id
END
```

4.2 Xây dựng lớp tiện ích chứa các phương thức thực hiện truy vấn và thao tác dữ liệu theo hướng dẫn sau:

```
public class Jdbc {
	static String driver = "com.microsoft.sqlserver.jdbc.SQLServerDriver";
	static String dburl = "jdbc:sqlserver://localhost;database=HRM";
	static String username = "sa";
	static String password = "123456";

	static {
		try { // nạp driver
			Class.forName(driver);
		} catch (ClassNotFoundException e) {
			throw new RuntimeException(e);
		}
	}
	/**Mở kết nối*/
	public static Connection getConnection() throws SQLException {
		return DriverManager.getConnection(dburl, username, password);
	}
	/**Thao tác dữ liệu*/
	public static int executeUpdate(String sql, Object... values) throws SQLException {
		Connection connection = getConnection();
		CallableStatement statement = connection.prepareCall(sql);
		for (int i = 0; i < values.length; i++) {
			statement.setObject(i + 1, values[i]);
		}
		return statement.executeUpdate();
	}
	/**Truy vấn dữ liệu*/
	public static ResultSet executeQuery(String sql, Object... values) throws
SQLException {
		Connection connection = getConnection();
		CallableStatement statement = connection.prepareCall(sql);
		for (int i = 0; i < values.length; i++) {
```

```
                statement.setObject(i + 1, values[i]);
            }
            return statement.executeQuery();
        }
}
```

4.3 Sử dụng thư viện tiện ích vừa xây dựng ở trên để chạy thử các câu lệnh gọi thủ tục lưu ở phần 4.1 đã viết trong bài 1 như hướng dẫn sau

```
try {
        String sql = "{CALL spSelect…}";
        String values = {…};
        ResultSet resultSet = Jdbc.executeQuery(sql, values);
        while(resultSet.next()) {
            String value = resultSet.getString("column");
            //...
            System.out.println(value);
        }
} catch (Exception e) {
        e.printStackTrace();
}

try {
        String sql = "{CALL spInsert…, spUpdate...?, spDelete…?}";
        String values = {…};
        int rows = Jdbc.executeUpdate(sql, values);
        System.out.println(rows);
} catch (Exception e) {
        e.printStackTrace();
}
```

BÀI 5: GIẢNG VIÊN CHO THÊM