ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

MẪU THIẾT KẾ HƯỚNG ĐỐI TƯỢNG VÀ ỨNG DỤNG

BÁO CÁO ĐỒ ÁN

XÂY DỰNG THƯ VIỆN DAM/ORM

GVTH: Thầy HỒ TUẤN THANH

Nhóm sinh viên

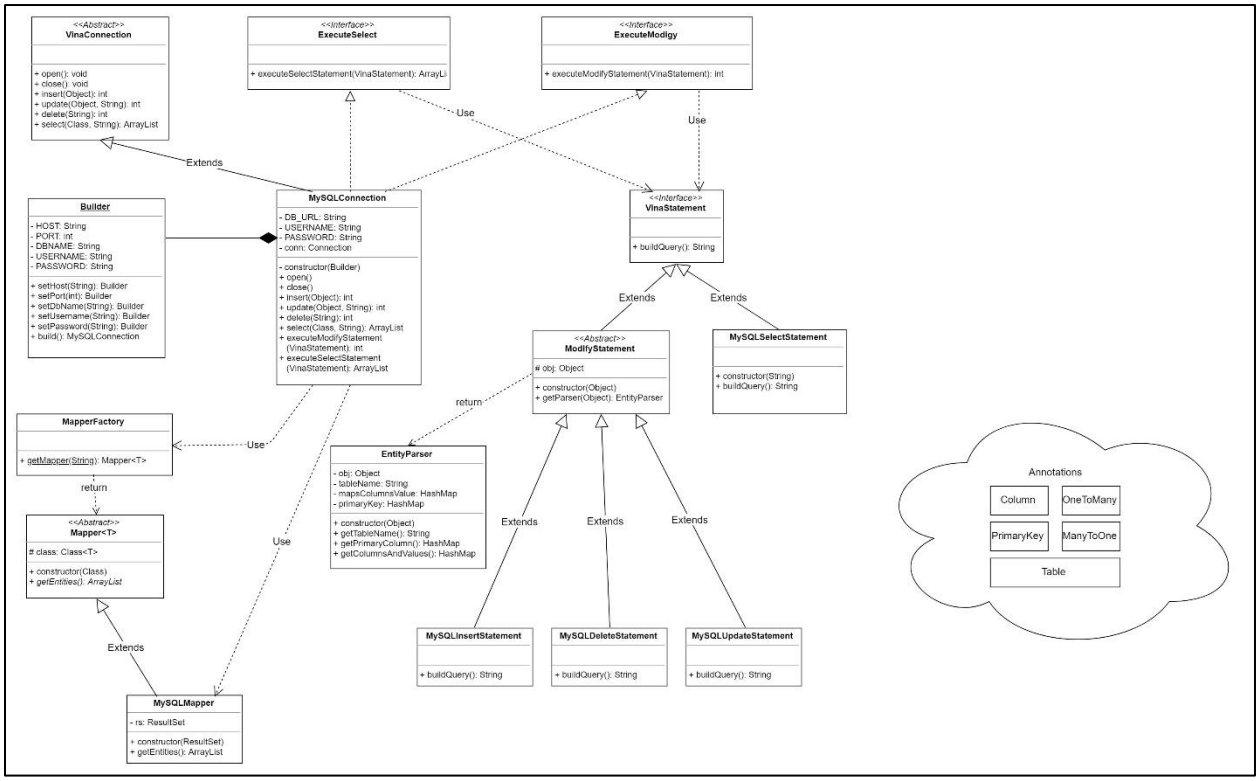
Nguyễn Văn Nghĩa 1512348

Đoàn Minh Nhật 1512378

TP. Hồ Chí Minh, 2018

PHẦN 1: BÁO CÁO SƠ ĐỒ LỚP

Class Diagram



Sơ đồ lớp của thư viện VinaORM

Giải thích ý nghĩa từng lớp

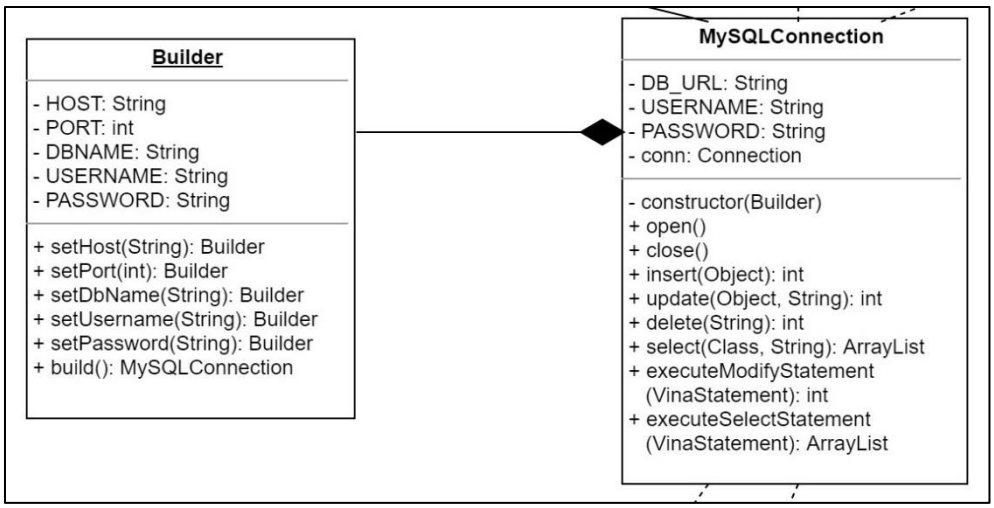
Tên lớp	Ý nghĩa
`VinaConnection`	Là lớp tổng quát cho các loại Connection của thư viện VinaORM. Mỗi loại Connection (thuộc một RDBMS) sẽ kế thừa từ lớp này
`MySQLConnection`	Kế thừa từ **VinaConnection**, là Connection dành cho CSDL MySQL
`MySQLConnection.Builder`	Dùng cho việc khởi tạo một đối tượng **MySQLConnection** dễ dàng hơn so với việc truyền nhiều tham số vào hàm khởi tạo

ExecuteSelect	Interface dành cho các **VinaConnection**, dùng cho truy vấn SELECT
ExecuteModify	Interface dành cho các **VinaConnection**, dùng cho các truy vấn INSERT, UPDATE, DELETE
VinaStatement	Đại diện cho các lớp con kế thừa biểu thị một loại truy vấn khác nhau (SELECT, INSERT, UPDATE, DELETE). Các **VinaStatement** sẽ trả về một câu truy vấn SQL hoàn chỉnh
MySQLSelectStatement	Dùng cho truy vấn SELECT của MySQL
ModifyStatement	Được thiết kế để tách biệt với **MySQLSelectStatement**, vì tính chất của các truy vấn INSERT, UPDATE và DELETE là trả về số lượng dòng ảnh hưởng (khác với SELECT là trả về một Collection)
MySQLInsertStatement	Dùng để sinh ra truy vấn INSERT của MySQL
MySQLUpdateStatement	Dùng để sinh ra truy vấn UPDATE của MySQL
MySQLDeleteStatement	Dùng để sinh ra truy vấn DELETE của MYSQL
EntityParser	Nhận các tham số truyền vào từ **MySQLConnection**, phân tích một Entity thành các dữ liệu như Column-Value tương ứng để các **ModifyStatement** sử dụng sản xuất ra Query
MapperFactory	Tạo ra một đối tượng **Mapper** , tham số đầu vào là loại RDBMS (Ví dụ: “mysql”)
Mapper	Lớp tổng ánh xạ dữ liệu từ **ResultSet** từ kết quả truy vấn thành một **Collection** của các Entity
MySQLMapper	Lớp ánh xạ dành cho MySQL

PHẦN 2: BÁO CÁO ÁP DỤNG MẪU THIẾT KẾ HƯỚNG ĐỐI TƯỢNG

① Builder

Sơ đồ lớp



Code example

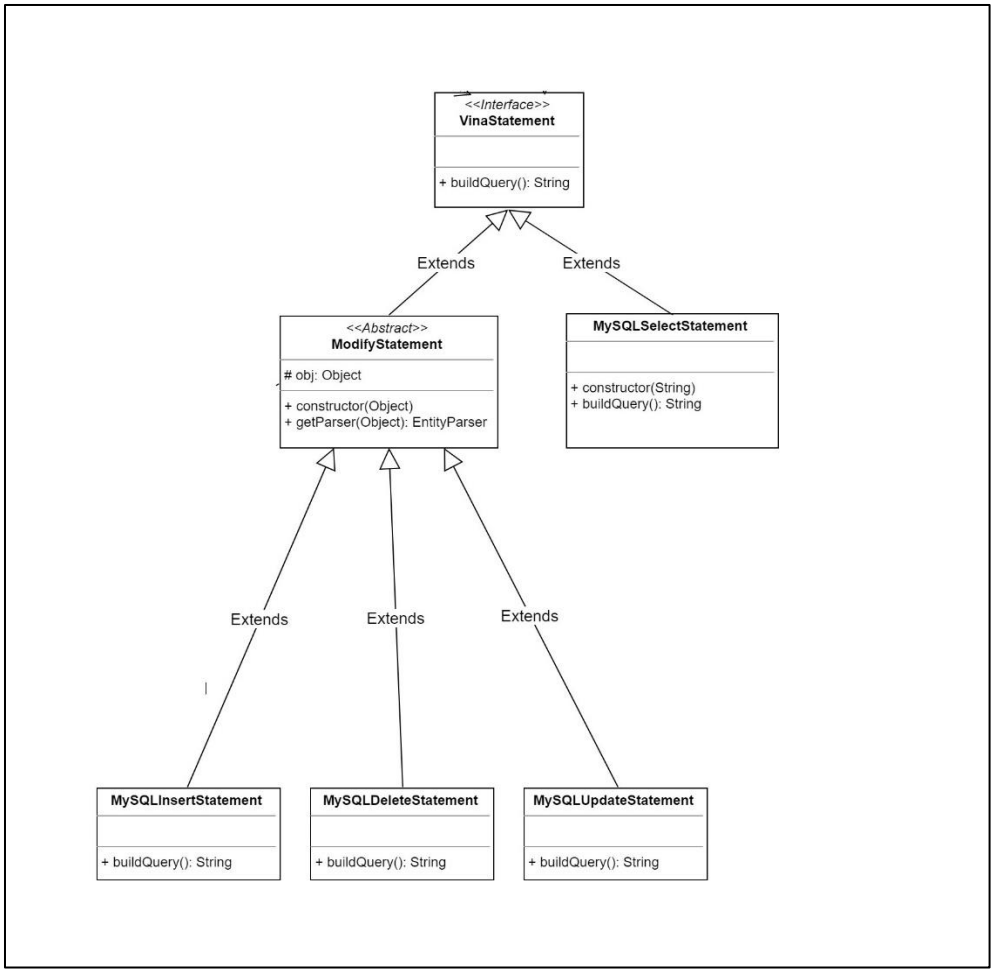
```
MySQLConnection.Builder mySQLConnectionBuilder = new MySQLConnection.Builder();
MySQLConnection conn = mySQLConnectionBuilder
                        .setHost("localhost")
                        .setPort(3306)
                        .setDbName("db")
                        .setUsername("root")
                        .setPassword("admin")
                        .build();
```

Ý nghĩa

Đơn giản hóa việc khởi tạo một Connection bằng cách sử dụng **Builder** (thay vì phải truyền từng tham số vào Constructor).

② Decorator

Sơ đồ lớp



Code example

```
@Override
public int insert(Object obj) throws SQLException, InvocationTargetException, IllegalAccessExce
    return executeModifyStatement(new MySQLInsertStatement(obj));
}

@Override
public int update(Object obj, String whereClause) throws IllegalAccessException, SQLException,
    return executeModifyStatement(new MySQLUpdateStatement(obj, whereClause));
}

@Override
public int delete(String tableName, String whereClause) throws IllegalAccessException, SQLExcep
    return executeModifyStatement(new MySQLDeleteStatement(tableName, whereClause));
}
```

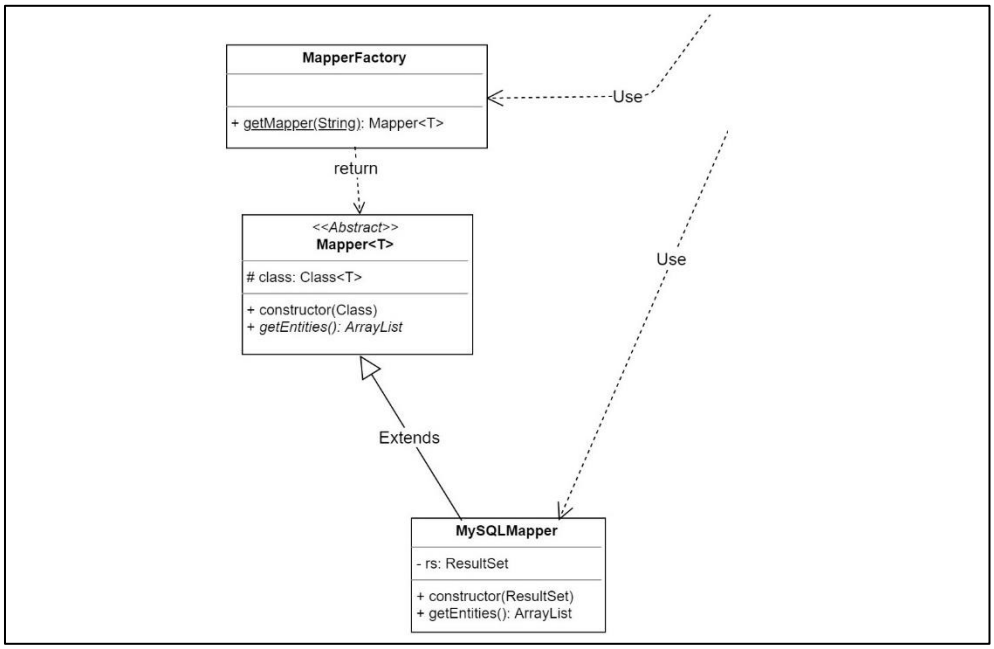
Ý nghĩa

Các lớp con của **VinaStatement** có nhiệm vụ chính là nhận vào các tham số thích hợp để tạo ra một câu truy vấn MySQL tương ứng. Khác với câu lệnh SELECT nhận một

câu truy vấn và trả về một Collection. Các câu lệnh như INSERT, UPDATE, DELETE nhận vào các tham số khác và trả về số lượng dòng được cập nhật trên cơ sở dữ liệu, do đó cần sự thay đổi để linh hoạt hơn. Áp dụng **Decorator Pattern** giúp vẫn giữ nguyên được bản chất của lớp cha nhưng cũng tích hợp được thêm những phương thức khác cần thiết cho việc thực thi các câu lệnh INSERT, UPDATE, DELETE.

③ Factory Method

Sơ đồ lớp



Code example

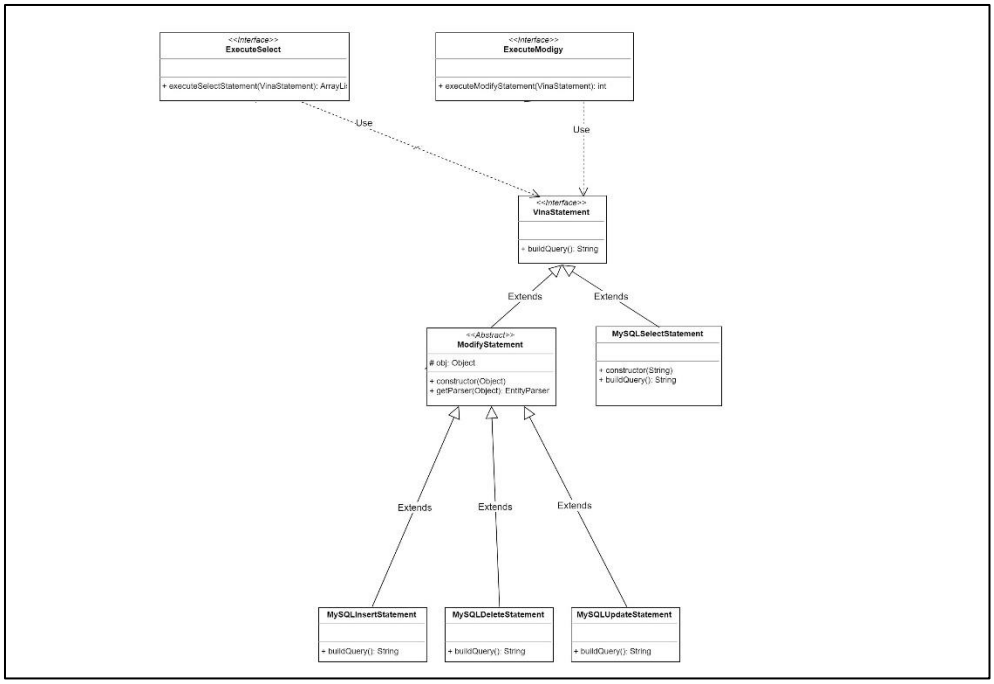
```
Statement stmt = conn.createStatement();
ResultSet rs = stmt.executeQuery(sql);

MySQLMapper<T> mapper = (MySQLMapper<T>) MapperFactory.getMapper("mysql", clazz, rs);
```

Ý nghĩa

Sau khi nhận về một **ResultSet**, ta sử dụng **MapperFactory** để nhận về một **MySQLMapper** dùng để chuyển đổi dữ liệu thô từ **ResultSet** thành một Entity Collection.

④ Strategy

Sơ đồ lớp



Code example

```
@Override
public int insert(Object obj) throws SQLException, InvocationTargetException, IllegalAccessExce
    return executeModifyStatement(new MySQLInsertStatement(obj));
}

@Override
public int update(Object obj, String whereClause) throws IllegalAccessException, SQLException,
    return executeModifyStatement(new MySQLUpdateStatement(obj, whereClause));
}

@Override
public int delete(String tableName, String whereClause) throws IllegalAccessException, SQLExcep
    return executeModifyStatement(new MySQLDeleteStatement(tableName, whereClause));
}
```

Ý nghĩa

Phương thức **executeModifyStatement** sẽ nhận các lớp con của **VinaStatement/ModifyStatement**, với mỗi lớp con (ứng với mỗi truy vấn INSERT, SELECT, UPDATE) sẽ tạo ra được câu truy vấn tương ứng.