Spring Data - JPA

1. JPA là gì

- JPA là một chuẩn đặc tả cho việc ánh xạ giữa các đối tượng java và csdl quan hệ thông qua công nghệ ORM (Object Relational Mapping).
- JPA cung cấp một mô hình POJO persistence cho phép ánh xạ các table/các mối quan hệ giữa các table trong database sang các class/mối quan hệ giữa các object.
- Ví dụ: table Users với các column (Id, name, age…) sẽ tương ứng với class Users.java với các field Id, name, age… từ đó mỗi khi truy vấn table hay các column ta sẽ truy vấn trực tiếp trên các class, các field của class mà không cần quan tâm tới việc đang dùng loại database nào, dữ liệu database ra sao…

- Table **User**

#	Name	Type
1	id	int(11)
2	name	varchar(50)
3	age	int(11)

Class **Users.java**

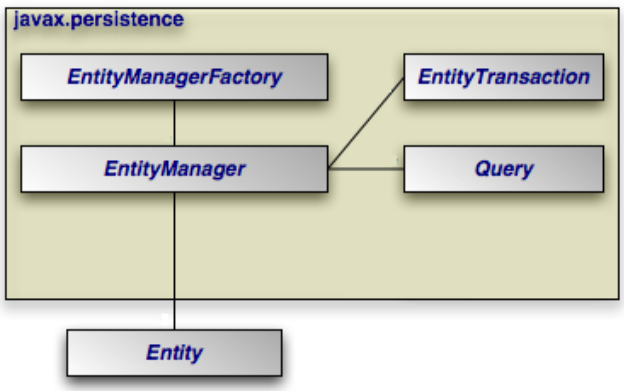
```
public class Users {

    private int id;
    private String name;
    private int age;

    // Constructor ...
    // Getter Setter ...
}
```

2. Kiến trúc JPA

JPA bao gồm 3 thành phần khác nhau:



2.1 Entities

2.1.1 Entity

- Là các đối tượng thể hiện tương ứng 1 table trong cơ sở dữ liệu. Khi lập trình, entity thường là các class POJO đơn giản, chỉ gồm các method getter, setter.
- Một số đặc điểm của một Entity:
  - Entity có thể tương tác với cơ sở dữ liệu quan hệ.
  - Entity được xác định thông qua một định danh(tương đương với khóa chình trong table của cơ sở dữ liệu quan hệ)
  - Entity hỗ trợ giao tác (transaction)
  - Entity hỗ trợ kế thừa giống như những lớp thường khác.

2.1.2 **EntityManager**:

- Là một giao diện (interface) cung cấp các API cho việc tương tác với các Entity.
- Một số chức năng cơ bản của EntityManager như:
  - **Persist**: phương thức này dùng để lưu một thực thể mới tạo vào cơ sở dữ liệu
  - **Merge**: dùng để cập nhật trạng thái của entity vào cơ sở dữ liệu.
  - **Remove**: xóa một thể hiện của entity.

2.1.3 **EntityManagerFactory**

- EntityManagerFactory được dùng để tạo ra một thể hiện của EntityManager.

2.2 Entity-Relationships

- Cung cấp anh xạ giữa các đối tượng java với tên bảng, tên các thuộc tính trong csdl. Điều này có thể làm bằng cách sử dụng tập tin cấu hình hoặc dùng các annotation.
- Các mối quan hệ cơ bản:
  - @ManyToOne
  - @OneToMany
  - @OneToOne
  - @ManyToMany

2.3 Java Persistence Query Language (JPQL)

- Khác với SQL là ngôn ngữ truy vấn trên trực tiếp trên database, thì JPQL là ngôn ngữ truy vấn trên Object. Khi thực thi chương trình thì JPQL sẽ được dịch sang SQL vì vậy phù hợp với các loại csdl quan hệ khác nhau.
- Cấu trúc và cú pháp của JPA Query Language thì rất tương tự như cấu trúc và cú pháp của câu SQL.
- Ví dụ cú pháp của JPQL như sau:
  - **Select**

```
SELECT ... FROM ... [WHERE ...]
```

  - **Update**

```
UPDATE ... SET ... [WHERE ...]
```

- **Delete**

```
DELETE FROM ... [WHERE ...]
```

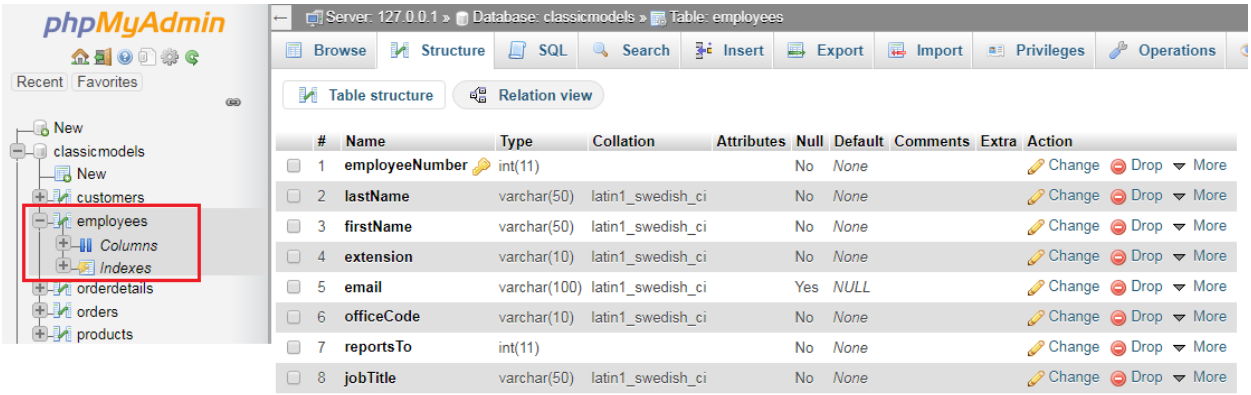
- Qua đó ta thấy cú pháp JPQL không khác gì so với SQL, ngoài việc thay các table name thành các object name

3. Tại sao nên dùng JPA

- Code ít và gọn hơn: đa số các business application đều có thao tác CRUD đến database. Với công nghệ JDBC trước đây, developers sẽ phải chủ động thực hiện các việc: từ mở kết nối vào Database, tạo các Statement, ResultSet cho đến đóng tất cả các thứ đó lại, và code sẽ trở nên cồng kềnh. Developers cần phải làm việc với cả java code và SQL. Thực tế, từng Database khác nhau thì câu SQL có một số phần khác nhau, nên đòi hỏi developers phải nắm được sự khác biệt này để viết code cho phù hợp. Khi ứng dụng muốn chuyển từ database sang database khác (ví dụ từ Oracle sang MS SQL Server) thì chắc chắn sẽ có một số phần của câu SQL cần phải đổi. Công việc sửa code lại đòi hỏi developers phải test lại ứng dụng. Điều này sẽ tốn thời gian phát triển. Để khắc phục nhược điểm này, đã có rất nhiều frameworks ra đời với mục đích giúp xóa đi vấn đề về tương thích giữa các Database, giúp developers tập trung vào phần xử lý nghiệp vụ.
- Độc lập về database
- Không phải làm việc với SQL

4. Sample

- Áp dụng **Spring Boot** và **Spring Data JPA** -> get list employee
- Cấu trúc table “employees” như sau:



- Source Code: https://github.com/java-training-tdt/SpringCoreExample

❖ **Tài liệu tham khảo**:
https://docs.spring.io/spring-data/jpa/docs/current/reference/html/