Date : 22/01/2021
Spring Boot 9AM
Mr. RAGHU

Spring Boot : Data JPA

-> PagingAndSortingRepository(I)
   This inteface extends CrudRepository(I) internally
   and supports 2 additional methods
   a) findAll(Sort s):Iterable<T>
   b) findAll(Pageable pageable):Page<T>

---

*) in SQL, order by columns [ASC|DESC] is used to fetch
   data in Sorting order.

Ex: select * from student order by sfee desc;
    select * from student order by sfee; //default is ASC.

select pid, pcod, pcost, pvendor from product ;

*) ToDo Sorting, enum : Direction is provided that has two possible
   values : ASC, DESC. Default value is ASC.
   This enum is Part of Sort(C) [org.springframework.data.domain]

```
class Sort {

  static Sort by(String... properties) {___}
  static Sort by(Direction direction, String... properties) {___}

  static enum Direction {
     ASC, DESC;
  }

}
```

*) Using static method 'by()' we can create Sort object and pass
   it to findAll(Sort) method, that gets data in Sorting order.

---code---

Name : SpringBoot2PageAndSort
Dep : Data Jpa, Lombok, MySQL.

1. application.yml

```
spring:
  datasource:
    driver-class-name: com.mysql.cj.jdbc.Driver
    password: root
    url: jdbc:mysql://localhost:3306/boot9am
    username: root
  jpa:
    database-platform: org.hibernate.dialect.MySQL8Dialect
    hibernate:
      ddl-auto: update
    show-sql: true
```

2. Model class

```
package in.nareshit.raghu.model;

import javax.persistence.Entity;
import javax.persistence.Id;

import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
@Entity
public class Product {
	@Id
	private Integer pid;
	private String pcode;
	private Double pcost;
	private String pvendor;
}
```

3. Repository Interface

```
package in.nareshit.raghu.repo;

import org.springframework.data.repository.PagingAndSortingRepository;

import in.nareshit.raghu.model.Product;

public interface ProductRepository
	extends PagingAndSortingRepository<Product, Integer> {

}
```

4. Runner class#1 for insert

```
package in.nareshit.raghu.runner;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;

import in.nareshit.raghu.model.Product;
import in.nareshit.raghu.repo.ProductRepository;

//@Component
public class ProductInsertRunner implements CommandLineRunner {
	@Autowired
	private ProductRepository repo;

	@Override
	public void run(String... args) throws Exception {
		repo.save(new Product(101, "PEN", 25.0, "SNTY"));
		repo.save(new Product(102, "BTL", 125.0, "SNTY"));
		repo.save(new Product(103, "KYBRD", 2500.0, "NIT"));
		repo.save(new Product(104, "MOUSE", 180.0, "NIT"));
	}

}
```

Runner class#2 for Data Sort

```
package in.nareshit.raghu.runner;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.data.domain.Sort;
import org.springframework.data.domain.Sort.Direction;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.model.Product;
import in.nareshit.raghu.repo.ProductRepository;

@Component
public class ProductDataSortRunner implements CommandLineRunner {
	@Autowired
	private ProductRepository repo;

	@Override
	public void run(String... args) throws Exception {
		//Iterable<Product> itr =  repo.findAll();
		//itr.forEach(System.out::println);

		//SQL:select * from product order by pcode asc
		//Sort sort = Sort.by("pcode"); // ASC
		//Sort sort = Sort.by(Direction.ASC,"pcode");

		//SQL:select * from product order by pcode desc
		//Sort sort = Sort.by(Direction.DESC,"pcode");

		//Sort sort = Sort.by("pcode","pcost");
		Sort sort = Sort.by(Direction.DESC,"pcode","pcost");

		Iterable<Product> itr = repo.findAll(sort);
		itr.forEach(System.out::println);


	}

}
```

Pagination using Data JPA

Pagination is process of fetching database table data -- Page by Page called as parts (Equally divided parts-Pages).

*) We need to call method: findAll(Pageable):Page<T>

Here Pageable means Input passed by Programmer for pagination.
Pageable(I) has impl class PageRequest(C).

```
Pageable pageable =  PageRequest.of(pageNum,pageSize);
```
[package : org.springframework.data.domain]

----Runner#3----

```
package in.nareshit.raghu.runner;
```

```
import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Pageable;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.model.Product;
import in.nareshit.raghu.repo.ProductRepository;

@Component
public class ProductPagingRunner implements CommandLineRunner {
	@Autowired
	private ProductRepository repo;

	@Override
	public void run(String... args) throws Exception {
		//Provide Input Pageable(pageNum,pageSize)
		Pageable pageable = PageRequest.of(0, 5);

		//call findAll() Method
		Page<Product> page =  repo.findAll(pageable);

		//---print output---------------------
		//read content
		List<Product> list = page.getContent();
		list.forEach(System.out::println);
		//--meta data---
		System.out.println("First Page : " + page.isFirst());
		System.out.println("Last Page : " + page.isLast());
		System.out.println("Next Page? : " + page.hasNext());
		System.out.println("Previous Page? : " +
page.hasPrevious());
		System.out.println("Empty Page? : " + page.isEmpty());
		System.out.println("Page Size? : " + page.getSize());
		System.out.println("Page Number? : " +
page.getNumber());
		System.out.println("Total Pages? : " +
page.getTotalPages());
		System.out.println("Total Rows? : " +
page.getTotalElements());

	}

}
```
