Hibernate實現的JPA規範

ORM 框架僅實現 JPA 完整規範中的大部分

> *JPA（Java Persistence API）目的是為簡化持久化的開發工作，以及整合各家 ORM 框架*

blocked URL

Spring Boot 預設整合的 ORM 框架為 Hibernate

> - *Hibernate 是實現了 JPA 規範的 ORM 框架之一*
> - *Hibernate 底層仍是使用 JDBC，所以專案內也必須要有 JDBC 驅動程式*

blocked URL

資料表結構

標記	位置	說明	備註
@Entity @Data	在class上方	告訴 Spring 這是個 Entity，所以會對應到資料庫內的某一張資料表	1. 通常 @Data 會加在一個值可以被更新的 Object 上 2. 是使用頻率最高的 Lombok 標記，等於同時加了以下標記 ◦ @Getter/@Setter ◦ @EqualsAndHashCode ◦ @RequiredArgsConstructor ◦ @ToString (在雙向關聯模式，請改手動重新覆寫 !!!)
@Table	在class上方	對應到資料表的真實表名，也可指定要建立的 index	需明示 name `@Table(name = "ACCOUNT_ROLE_LOG")`
@IdClass	在class上方	表示有複合主鍵	需實現 Serializable，可參考 composite primary key in JPA `@IdClass(AccountRoleLogPK.class)`

@EntityListeners	在class上方	表示要使用審計功能	1.
			
			`@EnableJpaAuditing`
			
			2. Entity
			
			`@EntityListeners(AuditingEntityListener.class)`
			
			3.
			
			`@Column(name = "CREATE_DATETIME", updatable = false)`
			`@CreatedDate //`
			
			`private LocalDateTime createDateTime;`
			
			`@Column(name = "UPDATE_DATETIME")`
			`@LastModifiedDate //`
			
			`private LocalDateTime updateDateTime;`
			
			4. AuditorAware AuditorConfiguration
			
			`@Column(name = "CREATE_USER", columnDefinition = "NVARCHAR(50)", updatable = false)`
			`@CreatedBy //`
			`private String createUser;`
			
			`@Column(name = "UPDATE_USER", columnDefinition = "NVARCHAR(50)")`
			`@LastModifiedBy //`
			
			`private String updateUser;`

@Id	在基本型變數的上方	此變數為資料表的主鍵	請留意 @Id ，必須是 JPA ，而不是 Spring Data（即 org.springframework.data.annotation.Id）原生的

@GeneratedValue	在基本型變數的上方	自動產生獨一無二的主鍵	適用於MSSQL
			
			`@GeneratedValue(strategy = GenerationType.IDENTITY)`
			
			直接產生長度36的UUID
			
			`@GeneratedValue(generator = "UUID")`
			`@GenericGenerator(name = "UUID", strategy = "org.hibernate.id.UUIDGenerator")`

@Column	在基本型變數的上方	對應到資料表的真實欄位名稱	需明示 columnDefinition、nullable `@Column(name = "ACCOUNT_ROLE_ID", columnDefinition = "NVARCHAR(36)", nullable = false)`

@Version	在基本型變數的上方	此變數為本表欄位	若是透過多執行續新增資料表的資料，建議多一個欄位：樂觀鎖
			
			1. 當version=1
			2. 此時事務1提交後，該數據的版本控制字段 version=version+1=2
			3. 然後事務2提交時，version=1 < 2，所以Hibernate認為事務2提交的數據為過時數據，拋出異常
			
			=== 用法 ===
			
			`@Version`
			`private long version;`

@OneToOne @JoinColumn	在變數的上方	外鍵(Forgign Key)設定 本表會一對一關聯其他表	若是透過Entity產生實體資料表，本表的外鍵預設會一併產生關聯限制條件
@ManyToOne @JoinColumn	在變數的上方	外鍵(Forgign Key)設定 本表會多對一關聯其他表	若是透過Entity產生實體資料表，本表的外鍵預設會一併產生關聯限制條件
@OneToMany	在變數的上方	本表會一對多關聯其他表	若有搭配 @JoinColumn並透過Entity產生實體資料表，被關聯資料表的外鍵預設會一併產生關聯限制條件
@ManyToMany @JoinTable name JoinColumns inverseJoinColumns	在變數的上方	本表會多對多關聯其他表 明示中間表名稱 中間表的外鍵名稱 中間表的外鍵名稱	Entity

資料表關聯

關聯（Cascade）操作

> *在關聯映射中，如一對一、一對多、多對一等，都可設定 cascade*

CascadeType.PERSIST	在儲存時，一併儲存被參考的物件
CascadeType.REMOVE	在移除時，一併移除被參考的物件
CascadeType.MERGE	在修改時，一併合併修改被參考的物件
CascadeType.REFRESH	• Refresh（刷新/更新/重新整理），也就是將資料庫的資料表狀態更新到物件上 • 若物件之前已有任何改變，仍會被資料庫的狀態覆寫(overwrite)
CascadeType.ALL	允許所有關聯操作

FetchType

> *當 FetchType.EARGE 時，在查詢時，就立刻載入關聯的物件*

> *當 FetchType.LAZY 時， 除非真正要使用到該屬性的值，否則不會真正將資料從表格中載入物件*

@OneToOne	FetchType.EARGE
@ManyToOne	FetchType.EARGE
@OneToMany	FetchType.LAZY
@ManyToMany	FetchType.LAZY

請注意....EntityManager 關閉後，才想要載入該屬性值，就會發生例外錯誤 !!!

解法 1：是在 EntityManager 關閉前取得資料

解法 2：改標示為 FetchType.EARGE， 表示立即從表格取得資料

舉例說明（OneToOne）

> *本表要一對一關聯其他表(EmployeeProfile)*

> *本表要一對一關聯其他表(Employee)*

Employee

```
        package com.fubonlife.day3.common.entity;

import lombok.Data;
import lombok.EqualsAndHashCode;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToOne;
import javax.persistence.Table;


@Entity
@Table
@Data
@EqualsAndHashCode(of = "id")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    private long id;

    //@Column(name = "employee_name")
    private String employeeName;

    @Column(name = "ep_id")
    private String epId;

    @OneToOne /* SinglePK referencedColumnName =
"id", */
    @JoinColumn(name = "ep_id", insertable =
false, updatable = false)
    private EmployeeProfile employeeProfile;

 /* ....(ref: error_result_01)
    private String epId;

    @OneToOne
    @JoinColumn(name = "ep_id", insertable =
false, updatable = false)
    private EmployeeProfile employeeProfile;
*/

/*
    private String epId;

    @OneToOne
    @JoinColumn(name = "epId", insertable = false,
updatable = false)
    private EmployeeProfile employeeProfile;
*/

}
```

EmployeeProfile

```
        package com.fubonlife.day3.common.entity;

import com.fubonlife.day3.common.model.enums.
Gender;
import lombok.Data;
import lombok.EqualsAndHashCode;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToOne;
import javax.persistence.Table;
import java.time.LocalDate;


@Entity
@Table//(name = "employee_profile")
@Data @EqualsAndHashCode(of = "id")
//@Builder @NoArgsConstructor @AllArgsConstructor
@Builder
public class EmployeeProfile {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    private long id;

    @Column(name = "phone_number")
    private String phoneNumber;

    private LocalDate birthOfDate;

    @Enumerated(EnumType.STRING)
    private Gender gender;

    @OneToOne(mappedBy = "employeeProfile")
    private Employee employee;

/*  ....(EMPLOYEE_ID, FK)
    @OneToOne
    private Employee employee;
*/

/*
    @OneToOne
    @JoinColumn(name = "ID", insertable = false,
updatable = false)
    private Employee employee;
*/

}
```

ref: error_result_01

舉例說明（ManyToOne 和 OneToMany）

> *本表要多對一關聯其他表(Person)*

Address

```
        package com.fubonlife.day3.common.entity;

import lombok.Data;
import lombok.EqualsAndHashCode;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.Table;


@Entity
@Table
@Data @EqualsAndHashCode(of = "id")
public class Address {

    @Id
    private Long id;

    private String street;

    private String zipCode;

    private String personId;

    @ManyToOne
    @JoinColumn(name = "personId", //  personId
personId  @Column
            insertable = false,
            updatable = false)
    private Person person;
}
```

> *本表要一對多關聯其他表(Address)*

Person

```
        package com.fubonlife.day3.common.entity;

import lombok.Data;
import lombok.EqualsAndHashCode;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import java.util.List;

@Entity
@Table
@Data
@EqualsAndHashCode(of = "id")
public class Person {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    @Column(name = "person_id")
    private Long id;


    // mappedBy Table Addressperson_idEntity
Address
    @OneToMany(mappedBy = "person", cascade =
{CascadeType.PERSIST, CascadeType.MERGE})
    private List<Address> addresses;

/*  : OneToManyJoinColumnFKTable Addressperson_id
Entity Person

    @OneToMany(cascade = CascadeType.ALL)
    @JoinColumn(name = "person_id")
    private List<Address> addresses;
*/

}
```

> *本表不要設定關聯其他表*

> *本表要一對多關聯其他表(Comment)*

Posts

Comment

```
        @Entity
@Table
@Data @EqualsAndHashCode(of = "id")
public class Comment {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    private long id;

    private String text;
}
```

```
        package com.fubonlife.day3.common.entity;

import lombok.Data;
import lombok.EqualsAndHashCode;
import org.springframework.data.annotation.
CreatedDate;
import org.springframework.data.annotation.
LastModifiedDate;
import org.springframework.data.jpa.domain.support.
AuditingEntityListener;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EntityListeners;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

@Entity
@Table
@Data  @EqualsAndHashCode(of = "id")
@EntityListeners(AuditingEntityListener.class) //
@EnableJpaAuditing @CreatedDate@LastModifiedDate
public class Posts {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    @Column(name = "id2")
    private long id;

    @Column(name = "title")
    private String title;

    @Column(name = "description")
    private String description;

    @Column(name = "content")
    private String content;

    @Column(name = "posted_at")
    @CreatedDate
    private LocalDateTime postedAt;

    @Column(name = "last_updated_at")
    @LastModifiedDate
    private LocalDateTime lastUpdatedAt;


    @OneToMany(cascade = CascadeType.ALL)
    @JoinColumn(name = "pc_fid",
referencedColumnName = "id2") // Table Comment
pc_fidEntity Posts
    private List<Comment> comments = new
ArrayList<>();
}
```

舉例說明（ManyToMany）

> *本表會多對多關聯其他表(Tags)*

> *本表會多對多關聯其他表(Posts)*

Posts

Tags

```
        package com.fubonlife.day3.common.entity;

import lombok.Data;
import lombok.EqualsAndHashCode;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.ManyToMany;
import javax.persistence.Table;
import java.util.HashSet;
import java.util.Set;

@Entity
@Table
@Data  @EqualsAndHashCode(of = "id")
public class Tags {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    private long id;

    private String name;

    @ManyToMany(cascade = CascadeType.ALL,
mappedBy = "tags")
    private Set<Posts> posts = new HashSet<>();
}
```

```
        package com.fubonlife.day3.common.entity;

import lombok.Data;
import lombok.EqualsAndHashCode;
import org.springframework.data.annotation.
CreatedDate;
import org.springframework.data.annotation.
LastModifiedDate;
import org.springframework.data.jpa.domain.support.
AuditingEntityListener;

import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EntityListeners;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.JoinTable;
import javax.persistence.ManyToMany;
import javax.persistence.OneToMany;
import javax.persistence.Table;
import java.time.LocalDateTime;
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

@Entity
@Table
@Data  @EqualsAndHashCode(of = "id")
@EntityListeners(AuditingEntityListener.class) //
@EnableJpaAuditing @CreatedDate@LastModifiedDate
public class Posts {

    @Id
    @GeneratedValue(strategy = GenerationType.
IDENTITY)
    @Column(name = "id2")
    private long id;

    @Column(name = "title")
    private String title;

    @Column(name = "description")
    private String description;

    @Column(name = "content")
    private String content;

    @Column(name = "posted_at")
    @CreatedDate
    private LocalDateTime postedAt;

    @Column(name = "last_updated_at")
    @LastModifiedDate
    private LocalDateTime lastUpdatedAt;


    @ManyToMany(cascade = CascadeType.ALL)
    @JoinTable(name = "posts_tags",
            joinColumns = { @JoinColumn(name =
"post_id2", referencedColumnName = "id2")},
            inverseJoinColumns = { @JoinColumn
(name = "tag_id2", referencedColumnName = "id")})
    private Set<Tags> tags = new HashSet<>();
}
```