操作记录

1.关联说明

1.一对一（单向）

1.关键实体

```
People -> t_people

idCard -> t_idcard 维护外键
@OneToOne(cascade = {CascadeType.PERSIST})
@JoinColumn(name = "t_people_id")//外键id
private People people;
```

2.自动生成的表结构

```sql
CREATE TABLE `t_person` (
  `t_id` int(11) NOT NULL,
  `t_address` varchar(255) DEFAULT NULL,
  `t_age` int(11) DEFAULT NULL,
  `t_birthday` datetime(6) DEFAULT NULL,
  `t_name` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`t_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;


CREATE TABLE `t_idcard` (
  `t_id` int(11) NOT NULL AUTO_INCREMENT,
  `t_number` varchar(255) DEFAULT NULL,
  `t_people_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`t_id`),
  KEY `FK7gdvrysil6gxmt806ysqr8atn` (`t_people_id`),
  CONSTRAINT `FK7gdvrysil6gxmt806ysqr8atn` FOREIGN KEY (`t_people_id`)
REFERENCES `t_people` (`t_id`)
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
```

2.一对一（双向）

1.关键实体

```
1  People2 -> t_people2
2  @OneToOne(mappedBy = "people2")//负责维护外键的表对应的实体中，持有的本类类型属性的
   属性名字
3      private IdCard2 idCard2;
4
5
6  idCard2 -> t_idcard2 维护外键
7
8      @OneToOne(cascade = {CascadeType.PERSIST})
9      @JoinColumn(name = "t_people_id")//外键id
10     private People2 people2;
```

2.自动生成的表结构

```
1  CREATE TABLE `t_people2` (
2    `t_id` int(11) NOT NULL AUTO_INCREMENT,
3    `t_age` int(11) DEFAULT NULL,
4    `t_name` varchar(255) DEFAULT NULL,
5    PRIMARY KEY (`t_id`)
6  ) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8;
7
8  CREATE TABLE `t_idcard2` (
9    `t_id` int(11) NOT NULL AUTO_INCREMENT,
10   `t_number` varchar(255) DEFAULT NULL,
11   `t_people_id` int(11) DEFAULT NULL,
12   PRIMARY KEY (`t_id`),
13   KEY `FKrfj903v724ch0q2ex5u6m2min` (`t_people_id`),
14   CONSTRAINT `FKrfj903v724ch0q2ex5u6m2min` FOREIGN KEY (`t_people_id`)
   REFERENCES `t_people2` (`t_id`)
15 ) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8;
```

3.一对多（单向）

1.主要实体

```
1  Student -> t_student
2  @Id
3      @Column(name = "t_id")
4      @GeneratedValue(strategy = GenerationType.IDENTITY)
5      private Integer id;
6  --------------------------------
7  Project -> t_project
8
9  @ManyToOne()
10     @JoinColumn(name = "t_student_id")
11     private Student student;
12 #### 1
13 加了级联
14 测试一个student
15 多个project ,设置同一个student
16 会报错
17
```

```
18  解决方法
19  不用级联
20  先保存student，再保存project
21
22  #### 2
23  尝试加级联看看怎么才能可以同时插入多条project(student为同一个
24  解决方案
25  需要在test方法上加上下面连个注解
26  作用是使得代码在同一个事务中，同时自动提交
27  没有@Commit ,则不会提交会回滚
28  @Transactional
29  @Commit
```

2.自动生成的表

```sql
1   CREATE TABLE `t_student` (
2     `t_id` int(11) NOT NULL AUTO_INCREMENT,
3     `t_name` varchar(255) DEFAULT NULL,
4     PRIMARY KEY (`t_id`)
5   ) ENGINE=InnoDB AUTO_INCREMENT=2 DEFAULT CHARSET=utf8;
6
7   CREATE TABLE `t_project` (
8     `t_id` int(11) NOT NULL AUTO_INCREMENT,
9     `t_sub_name` varchar(255) DEFAULT NULL,
10    `t_student_id` int(11) DEFAULT NULL,
11    PRIMARY KEY (`t_id`),
12    KEY `FKjbiOsj2aqjxfmj2tkiOwv33xd` (`t_student_id`),
13    CONSTRAINT `FKjbiOsj2aqjxfmj2tkiOwv33xd` FOREIGN KEY (`t_student_id`)
    REFERENCES `t_student` (`t_id`)
14  ) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8;
```

4，一对多（双向）

1.主要实体

```java
1   Student2 -> t_student2
2   @Id
3       @Column(name = "t_id")
4       @GeneratedValue(strategy = GenerationType.IDENTITY)
5       private Integer id;
6
7       @Column(name = "t_name")
8       private String name;
9
10      @OneToMany(mappedBy = "student2")//维护外键的一方对应的实体类中的本类类型的属性
    字段名称
11      private List<Project2> project2;
12
13  Project2 -> t_project2
14
```

```java
15  @Id
16      @GeneratedValue(strategy = GenerationType.IDENTITY)
17      @Column(name = "t_id")
18      private Integer id;
19
20      @Column(name = "t_sub_name")
21      private String name;
22
23      @ManyToOne()
24      @JoinColumn(name = "t_student_id")
25      private Student2 student2;
26
27
```

2.自动生成的表结构

```sql
1   CREATE TABLE `t_student2` (
2     `t_id` int(11) NOT NULL AUTO_INCREMENT,
3     `t_name` varchar(255) DEFAULT NULL,
4     PRIMARY KEY (`t_id`)
5   ) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
6
7   CREATE TABLE `t_project2` (
8     `t_id` int(11) NOT NULL AUTO_INCREMENT,
9     `t_sub_name` varchar(255) DEFAULT NULL,
10    `t_student_id` int(11) DEFAULT NULL,
11    PRIMARY KEY (`t_id`),
12    KEY `FKl48vus0ax7ywxcex420rik40` (`t_student_id`),
13    CONSTRAINT `FKl48vus0ax7ywxcex420rik40` FOREIGN KEY (`t_student_id`)
    REFERENCES `t_student2` (`t_id`)
14  ) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
```

5.多对多（单向）

```
1   @ManyToMany
2   作用：用于映射多对多关系
3   属性：
4   cascade：配置级联操作。
5   fetch：配置是否采用延迟加载。
6   targetEntity：配置目标的实体类。映射多对多的时候不用写。
7
8   @JoinTable
9   作用：针对中间表的配置
10  属性：
11  nam：配置中间表的名称
12  joinColumns：中间表的外键字段关联当前实体类所对应表的主键字段
13  inverseJoinColumn：中间表的外键字段关联对方表的主键字段
14
15  @JoinColumn
16  作用：用于定义主键字段和外键字段的对应关系。
17  属性：
18  name：指定外键字段的名称
```

```
19   referencedColumnName：指定引用主表的主键字段名称
20   unique：是否唯一。默认值不唯一
21   nullable：是否允许为空。默认值允许。
22   insertable：是否允许插入。默认值允许。
23   updatable：是否允许更新。默认值允许。
24   columnDefinition：列的定义信息。
```

1.主要实体

```
1    @Id
2        @Column(name = "t_id")
3        @GeneratedValue(strategy = GenerationType.IDENTITY)
4        private Integer id;
5
6        @Column(name = "t_name")
7        private String name;
8    ---------------------------------------------------------------
9
10   @Id
11       @Column(name = "t_id")
12       @GeneratedValue(strategy = GenerationType.IDENTITY)
13       private Integer id;
14
15       @Column(name = "t_name")
16       private String name;
17
18       @ManyToMany(cascade = CascadeType.PERSIST)
19       @JoinTable(name="user_role_rel",//中间表的名称
20               //中间表user_role_rel字段关联sys_role表的主键字段role_id
21               joinColumns=
     {@JoinColumn(name="role_id",referencedColumnName="t_id")},
22               //中间表user_role_rel的字段关联sys_user表的主键user_id
23               inverseJoinColumns=
     {@JoinColumn(name="user_id",referencedColumnName="t_id")}
24       )
25       private List<SysUser> sysUserList;
26
27   ###
```

2.自动生成的表结构

```
1    CREATE TABLE `t_sys_user` (
2      `t_id` int(11) NOT NULL AUTO_INCREMENT,
3      `t_name` varchar(255) DEFAULT NULL,
4      PRIMARY KEY (`t_id`)
5    ) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8;
6
7
8    CREATE TABLE `t_sys_role` (
9      `t_id` int(11) NOT NULL AUTO_INCREMENT,
10     `t_name` varchar(255) DEFAULT NULL,
11     PRIMARY KEY (`t_id`)
```

```
12   ) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
13
14
15   CREATE TABLE `user_role_rel` (
16     `role_id` int(11) NOT NULL,
17     `user_id` int(11) NOT NULL,
18     KEY `FKm28g58dhcs5u9asuuww8ui43w` (`user_id`),
19     KEY `FKi2omtqgkldjbgukc3ry5hsdf` (`role_id`),
20     CONSTRAINT `FKi2omtqgkldjbgukc3ry5hsdf` FOREIGN KEY (`role_id`)
     REFERENCES `t_sys_role` (`t_id`),
21     CONSTRAINT `FKm28g58dhcs5u9asuuww8ui43w` FOREIGN KEY (`user_id`)
     REFERENCES `t_sys_user` (`t_id`)
22   ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
23
24
```

6.多对多（多向）

1.主要的实体类

```
1   @Id
2       @Column(name = "t_id")
3       @GeneratedValue(strategy = GenerationType.IDENTITY)
4       private Integer id;
5
6       @Column(name = "t_name")
7       private String name;
8
9       @ManyToMany(mappedBy = "sysUserList2")//维护外键的表的对应实体中的属性字段名
10      private List<SysRole2> sysRole2;
11
12
13  @Id
14      @Column(name = "t_id")
15      @GeneratedValue(strategy = GenerationType.IDENTITY)
16      private Integer id;
17
18      @Column(name = "t_name")
19      private String name;
20
21      @ManyToMany(cascade = CascadeType.PERSIST)
22      @JoinTable(name="user_role_rel2",//中间表的名称
23              //中间表user_role_rel字段关联sys_role表的主键字段role_id
24              joinColumns=
     {@JoinColumn(name="role_id",referencedColumnName="t_id")},
25              //中间表user_role_rel的字段关联sys_user表的主键user_id
26              inverseJoinColumns=
     {@JoinColumn(name="user_id",referencedColumnName="t_id")}
27      )
28      private List<SysUser2> sysUserList2;
29
30
```

2.自动生成的表结构

```sql
CREATE TABLE `t_sys_user2` (
  `t_id` int(11) NOT NULL AUTO_INCREMENT,
  `t_name` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`t_id`)
) ENGINE=InnoDB AUTO_INCREMENT=7 DEFAULT CHARSET=utf8;


CREATE TABLE `t_sys_role2` (
  `t_id` int(11) NOT NULL AUTO_INCREMENT,
  `t_name` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`t_id`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8;


CREATE TABLE `user_role_rel2` (
  `role_id` int(11) NOT NULL,
  `user_id` int(11) NOT NULL,
  KEY `FK5o1if0v99f02hwcnvmitah1j7` (`user_id`),
  KEY `FKkjme7hdhxcgkbka7kox6r9ul` (`role_id`),
  CONSTRAINT `FK5o1if0v99f02hwcnvmitah1j7` FOREIGN KEY (`user_id`)
REFERENCES `t_sys_user2` (`t_id`),
  CONSTRAINT `FKkjme7hdhxcgkbka7kox6r9ul` FOREIGN KEY (`role_id`)
REFERENCES `t_sys_role2` (`t_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

2.联合主键

1.方式一

```
1.通过 主键实体打上注解@Embeddabl
表实体中主键属性打上注解 @EmbeddedId

2.主要的实体
@Data
@AllArgsConstructor
@NoArgsConstructor
@Builder
@Embeddable
public class ComputerPK implements Serializable {

    @Column(name = "t_ip")
    private String ip;

    @Column(name = "t_owner_id")
    private String ownerId;


}


@Data
@AllArgsConstructor
@NoArgsConstructor
@Builder
@Entity
```

```
27  @Table(name = "t_computer")
28  public class Computer {
29
30      @EmbeddedId
31      private ComputerPK computerPK;
32
33      @Column(name="t_brand_name")
34      private String brandName;
35
36  }
37  3.自动生成的表结构
38  CREATE TABLE `t_computer` (
39    `t_ip` varchar(255) NOT NULL,
40    `t_owner_id` varchar(255) NOT NULL,
41    `t_brand_name` varchar(255) DEFAULT NULL,
42    PRIMARY KEY (`t_ip`,`t_owner_id`)
43  ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

2.方式二

```
1   1.@IdClass 配合 @Id   和上面的效果差不多，可能根据方法名字操作方便点
2
3   2.主要的实体
4   @Data
5   @AllArgsConstructor
6   @NoArgsConstructor
7   @Builder
8   @Entity
9   @Table(name = "t_computer2")
10  @IdClass(Computer2PK.class)
11  public class Computer2 {
12
13      @Id
14      @Column(name = "t_ip")
15      private String ip;
16
17      @Id
18      @Column(name = "t_owner_id")
19      private String ownerId;
20
21      @Column(name="t_brand_name")
22      private String brandName;
23
24  }
25  --------------------------------------------------
26  @Data
27  @AllArgsConstructor
28  @NoArgsConstructor
29  @Builder
30  public class Computer2PK implements Serializable {
31
32      private String ip;
33
34      private String ownerId;
35
36
37  }
```

```
38
39
40
41    3.自动生成的表结构
42
43    CREATE TABLE `t_computer2` (
44      `t_ip` varchar(255) NOT NULL,
45      `t_owner_id` varchar(255) NOT NULL,
46      `t_brand_name` varchar(255) DEFAULT NULL,
47      PRIMARY KEY (`t_ip`,`t_owner_id`)
48    ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

3.审计

1.添加注解

```
1    1.启动类
2
3    @EnableJpaAuditing
4    2.注解类
5    @EntityListeners({AuditingEntityListener.class})
```

2.自定义实现接口AuditorAware的类

```
1    @Component
2    public class AuditConfig implements AuditorAware {
3        /**
4         * Returns the current auditor of the application.
5         *
6         * @return the current auditor
7         */
8        @Override
9        public Optional getCurrentAuditor() {
10           return Optional.of("allen");
11       }
12   }
13
14
```

3.在实体类字段加上注解

```
1    @CreatedBy ：由谁创建这条记录
2    @LastModifiedBy：是谁最后更新了这条记录
3    @CreatedDate：创建时间
4    @LastModifiedDate：最后更新时间
```

t_computer2 @bilibili_dat... ×   t_my_audit @bilibili_data... ×

文件   编辑   查看   窗口   帮助

导入向导   导出向导   筛选向导 | 网格查看   表单查看 | 备注   十六进制   图像 | 升序排序   »

t_id	cre_date	cre_user	modi_date	modi_user	t_name
1	(Null)	(Null)	2022-11-14 23:37:52.785000	allen	jack