Hibernate表的继承关系

Java继承：

继承关系：什么是一种什么（猫是动物）

作用：1.代码的复用 2.子类的扩展性

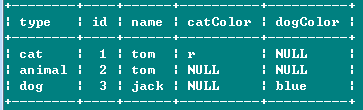
Hibernate 的继承：

为什么表之间的需要继承？

在hibernate中将面向关系类型的数据库转换为面向对象的编程，建表也是一种面向对象的编程，从而继承关系自然出现

注解版：

1. 第一种表的结构：



父类表|Animal

@Entity

**@Inheritance(strategy=InheritanceType.*SINGLE\_TABLE*)**

**@DiscriminatorColumn(name="type",discriminatorType=DiscriminatorType.*STRING*)**

**@DiscriminatorValue("animal")**

@Table(name="t\_animal")

**public** **class** Animal {

**private** **int** id;

**private** String name;

@Id

@GeneratedValue

Get与set方法略

}

子类：Dog

@Entity

**@DiscriminatorColumn(discriminatorType=DiscriminatorType.*STRING*)**

**@DiscriminatorValue("dog")**

**public** **class** Dog **extends** Animal{

**private** String dogColor;

}

子类：cat

@Entity

@DiscriminatorColumn(discriminatorType=DiscriminatorType.*STRING*)

@DiscriminatorValue("cat")

**public** **class** Cat **extends** Animal{

**private** String catColor;}

解释下：

@Inheritance(strategy = InheritanceType.SINGLE\_TABLE)确定采用单表策略

通过@DiscriminatorColumn确定了标志值的字段和类型，**@DiscriminatorColumn(name="type",discriminatorType=DiscriminatorType.*STRING*)**

***@DiscriminatorValue("")这个可以存在，也可以不存在***

***就是当插入子类dog，当存在是@DiscriminatorValue("dog1")则表示type字段是dog1；***

***当不存在时，则表示插入子类Dog（就是子类名）***

**创建表：**

create table t\_animal (

type varchar(31) not null,

id integer not null auto\_increment,

name varchar(255),

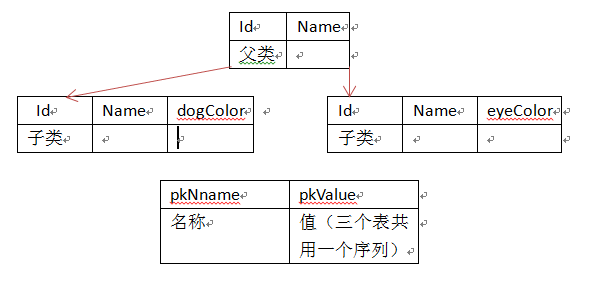
catColor varchar(255),

dogColor varchar(255),

primary key (id)

)

1. 第二种（每个类生成一张表）



[@Inheritance(Strategy=InheritanceType.TABLE\_PER\_CLASS](mailto:@Inheritance(Strategy=InheritanceType.TABLE_PER_CLASS))

注意：@TableGenerator(

name=”自定义序列的名称”, //（**必写，对应于父类id**）

table=””, //**不写就自动产生hibernate的表名**

pkColumnName=””, //**产生的序列表中字段，区别那张表的插入**

valueColumnName=””, // **插入相应表的名称值**

pkColumnvalue=””. // **产生序列表中字段，显示值**

initialValue=1,初始值

allocationSize=1步长

)

在父类ID上添加

@GeneratedValue(generator=”自定义序列名称”,strategy=GenerationType.TABLE)

子类只需要@Entity

示例：父类：

@Entity

@Inheritance(strategy=InheritanceType.*TABLE\_PER\_CLASS*)

@TableGenerator(name="pk\_animal",table="t\_pk\_animal",pkColumnName="pkName",valueColumnName="pkValue",initialValue=1,allocationSize=1)

@Table(name="t\_animal")

**public** **class** Animal {

**private** **int** id;

**private** String name;

@Id

@GeneratedValue(generator="pk\_animal",strategy=GenerationType.*TABLE*)

**public** **int** getId() {

**return** id;

}

子类：

@Entity

@Inheritance(strategy=InheritanceType.*TABLE\_PER\_CLASS*)

@TableGenerator(name="pk\_cat",table="t\_pk\_animal",pkColumnName="pkName",pkColumnValue="cat",valueColumnName="pkValue",initialValue=1,allocationSize=1)**//可写可不写没作用**

@Table(name="t\_cat")

**public** **class** Cat **extends** Animal{

**private** String catColor;

生成表结构：

create table t\_animal (

id integer not null,

name varchar(255),

primary key (id)

)

create table t\_cat (

id integer not null,

name varchar(255),

catColor varchar(255),

primary key (id)

)

**create table t\_pk\_animal (**

**pkName varchar(255),**

**pkValue integer**

**)**

**第三种：每个子类表存储独有字段，子类表中主键于父类表中主键外键关联**

父类中@Inheritance(Strategy=InheritanceType.JOIN)

Id自动生成，子类中什么都不用写

**生成表：**

create table Cat (

catColor varchar(255),

id integer not null,

primary key (id)

)

create table t\_animal (

id integer not null auto\_increment,

name varchar(255),

primary key (id)

)

alter table Cat

add index FK107B649ED234F (id),

add constraint FK107B649ED234F

foreign key (id)

references t\_animal (id)

配置版：