Структура БД

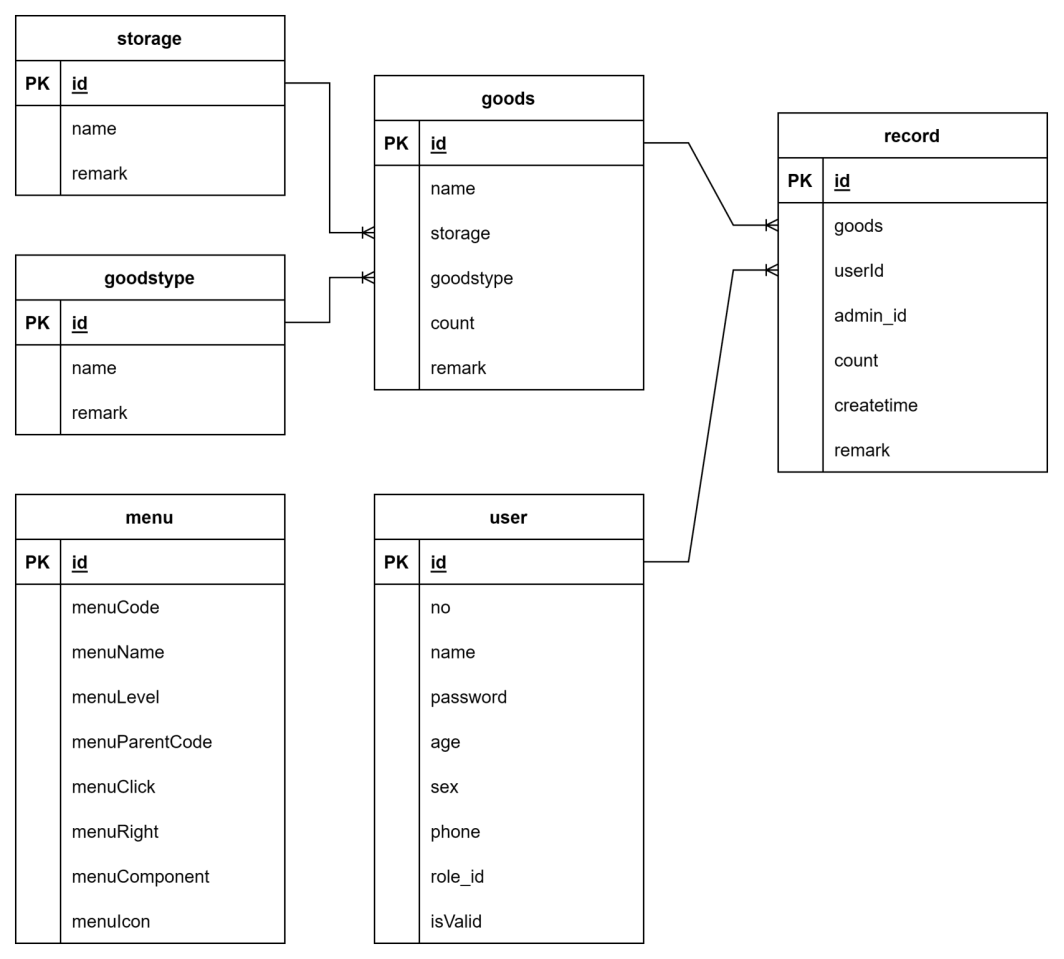
### Цель

Выделить сущности из предметной области проекта, составить ER диаграмму для описания предметной области, создание проекта и реализация модели разрабатываемого приложения.

### Задание

1. Создать ER-диаграмму, описывающую предметную область.
2. Создать проект Spring Boot приложения.
3. Описать Модель приложения.
4. Проверить структуру базы данных, сгенерированную по описанию Модели.

ER-диаграмма



В разработанной нами базе данных есть шесть таблиц, здесь мы возьмем одну таблицу в качестве примера.

На рисунке представлена ER-диаграмма, описывающая предметную область из предыдущих работ.В предметной области можно выделить 6 сущности goods, goodstype, user,storage,menu,record.

1.Сущность goods описывает товары, хранящиеся на складе, и имеет следующие

поля:опишите поля объекта товара (id, name, storage, goodsType, count, remark)

1. Сущность goodstype описывает тип товаров на складе, и имеет следующие

поля:Опишите поле объекта типа товара (id, name, remark)

3.Сущность user описывает администраторов и сотрудников, работающих на складе, и имеет следующие поля:

Опишите поле сущности пользователя (id, no, name, password, age, sex, phone, role\_id, isValid)

4.Объект хранения описывает все склады, и имеет следующие поля:

Описать поля сущности склада(id, name, remark)

5.Сущность меню описывает панель навигации ,и имеет следующие поля:

Опишите поле сущности меню (id, menuCode, menuName, menuLevel, menuParentCode, menuClick, menuRight, menuComponent, menuIcon)

6.Сущность записи описывает исходящие и входящие записи товаров на складе и имеет следующие поля:

Опишите поле объекта записи (id, goods, userId, admin\_id,count, createtime,remark)

Создание проекта Spring Boot приложения

1.Версия springboot2.6.7, используемая в нашей программе, версия JDK — 1.8.Примечание: если вам нужно использовать версию springboot3.0, вам необходимо обновить JDK до 17 или выше.

<parent>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-parent</artifactId>  
 <version>2.6.7</version>  
 <relativePath/> </parent>

2.Инструмент для подключения к базе данных в нашей программе — mysql5.1.44.Примечание: если ваша база данных выше 8.0, вам необходимо добавить CJ в файл конфигурации драйвера базы данных, например, это имя класса драйвера: com.mysql.cj .jdbc .Водитель. Наша среда интеграции Springboot — SSM, в которой мы используем Mybatis-puls для работы с базой данных.Эта структура похожа на SpringDataJpa и может помочь нам сэкономить много утомительных операторов SQL.Поскольку версия JDK нашего компьютера — 1.8, она не поддерживает многие методы и аннотации SpringDataJpa., поэтому мы выбрали Mybatis-Plus.

<**dependency**>

<**groupId**>mysql</**groupId**>

<**artifactId**>mysql-connector-java</**artifactId**>

<**version**>5.1.44</**version**>

<**scope**>runtime</**scope**>

</**dependency**>

<dependency>  
 <groupId>com.baomidou</groupId>  
 <artifactId>mybatis-plus-boot-starter</artifactId>  
 <version>3.4.1</version>  
</dependency>

3.Мы используем Freemarker в качестве внешнего шаблонизатора HTML.

<dependency>  
 <groupId>org.freemarker</groupId>  
 <artifactId>freemarker</artifactId>  
 <version>2.3.30</version>  
</dependency>

Также проверьте файл src/main/java/resources/application.properties . Этот файл содержит настройки приложения, укажите в нем свои параметры подключения к СУБД

**server**:

**port**: 8090

**spring**:

**datasource**:

**url**: jdbc:mysql://localhost:3306/wms02?useUnicode=true&characterEncoding=utf-8&useSSL=false&serverTimezone=GMT%2B8

**driver-class-name**: com.mysql.jdbc.Driver

**username**: root

**password**: 123456

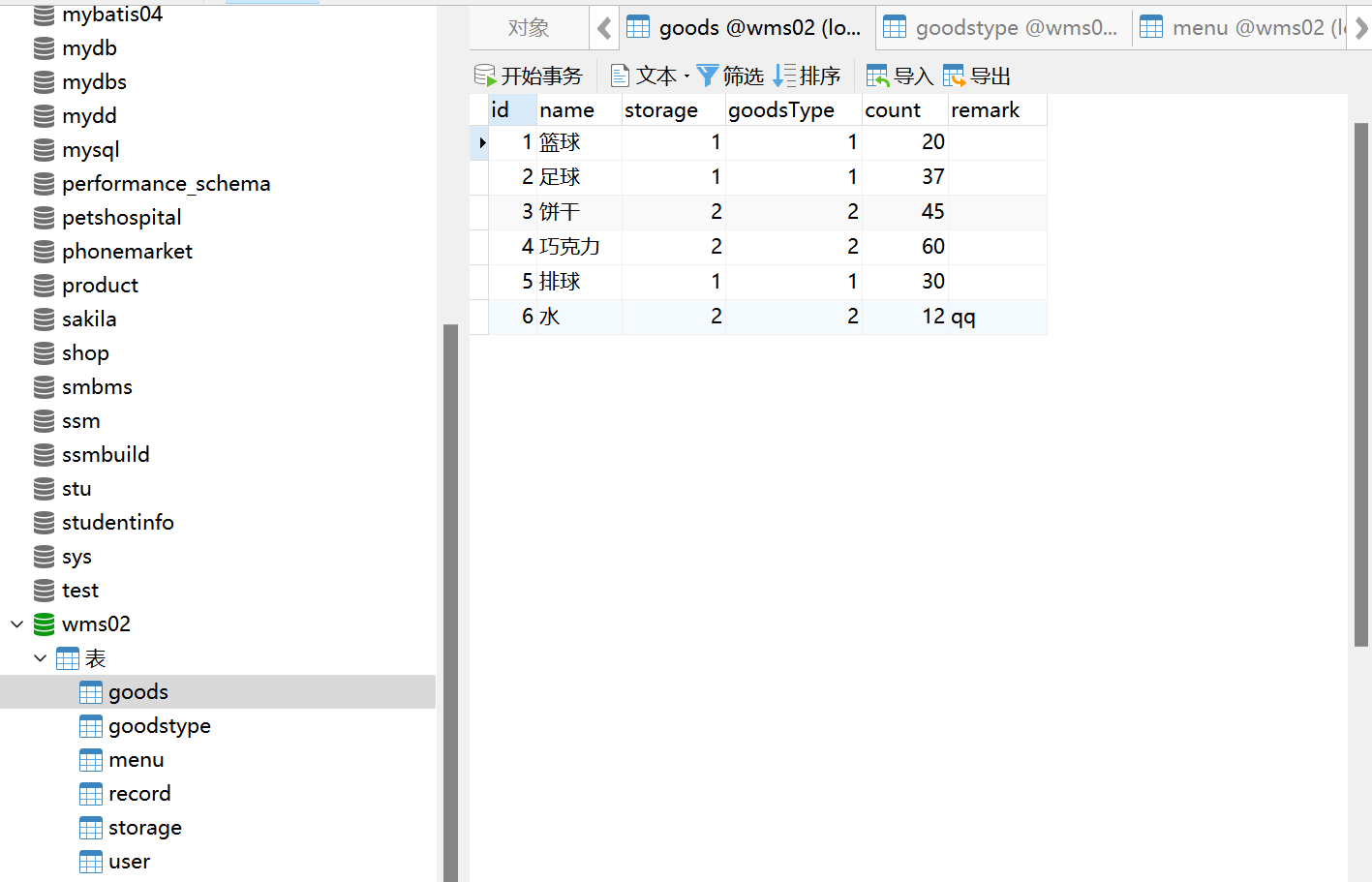
**Logging**:

**level**:

**com.wms**: debug

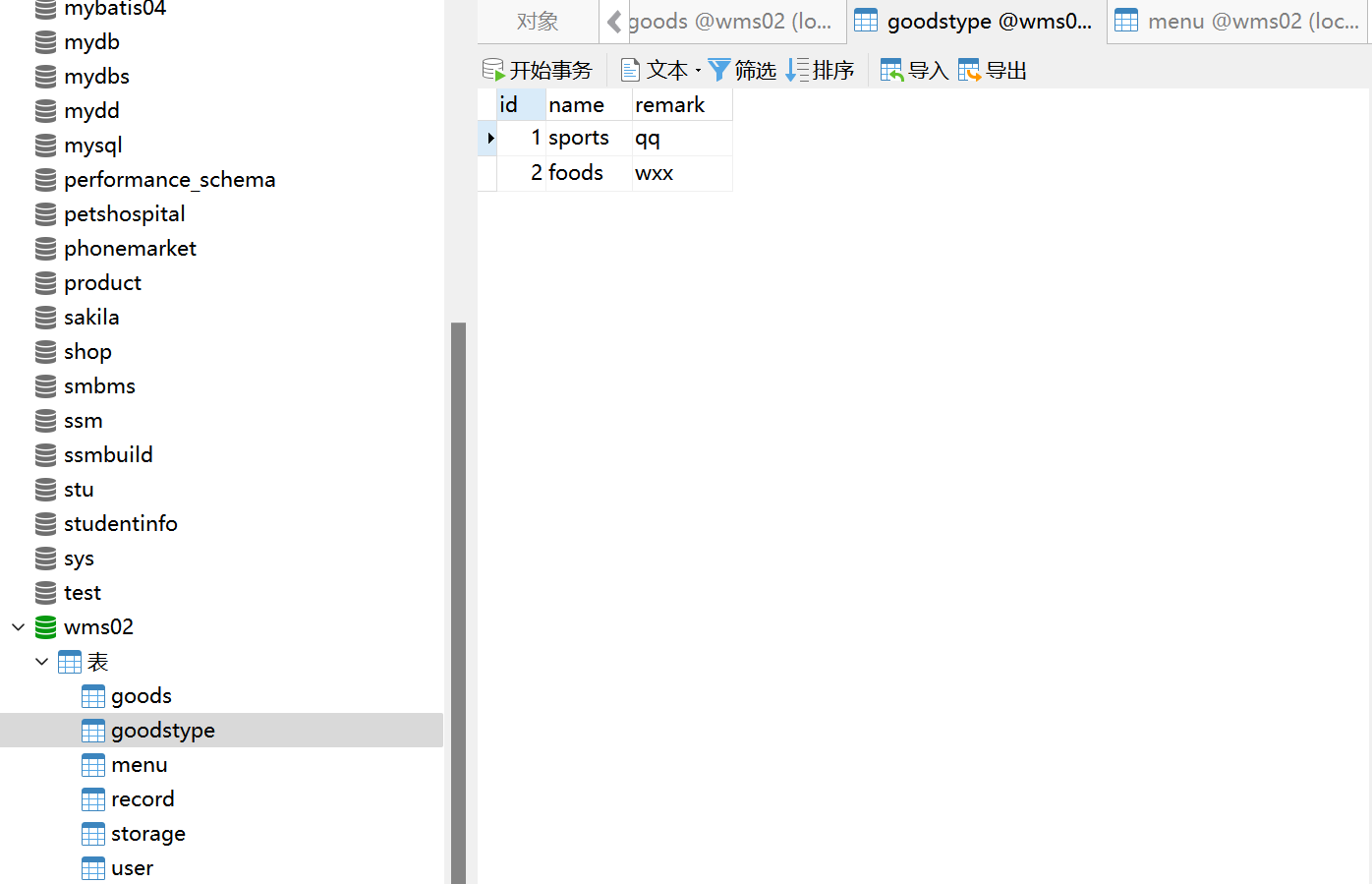
### Описание Модели приложения

Далее идет наша таблица базы данных и класс сущностей, я приведу все таблицы и классы сущностей нашей базы данных и возьму одну из них в качестве примера, чтобы объяснить код и аннотации внутри.

Good:

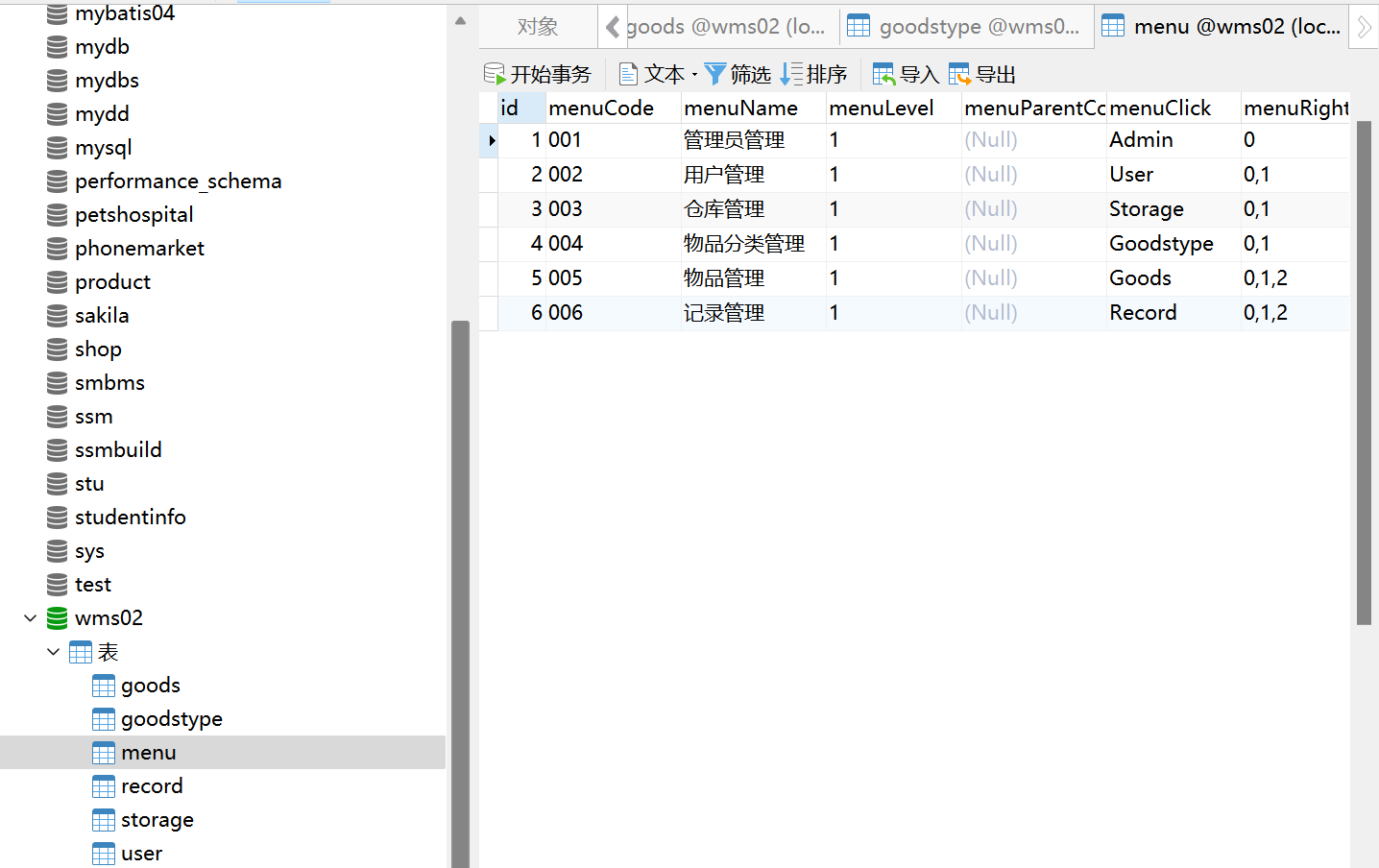
@Data  
@EqualsAndHashCode(callSuper = false)  
@ApiModel(value="Goods对象", description="")  
public class Goods implements Serializable {  
 private static final long *serialVersionUID* = 1L;  
 @ApiModelProperty(value = "主键")  
 @TableId(value = "id", type = IdType.*AUTO*)  
 private Integer id;  
 @ApiModelProperty(value = "货名")  
 private String name;  
 @ApiModelProperty(value = "仓库")  
 private Integer storage;  
 @ApiModelProperty(value = "分类")  
 @TableField("goodsType")  
 private Integer goodstype;  
  
 @ApiModelProperty(value = "数量")  
 private Integer count;  
 @ApiModelProperty(value = "备注")  
 private String remark;  
}

Goodstype:



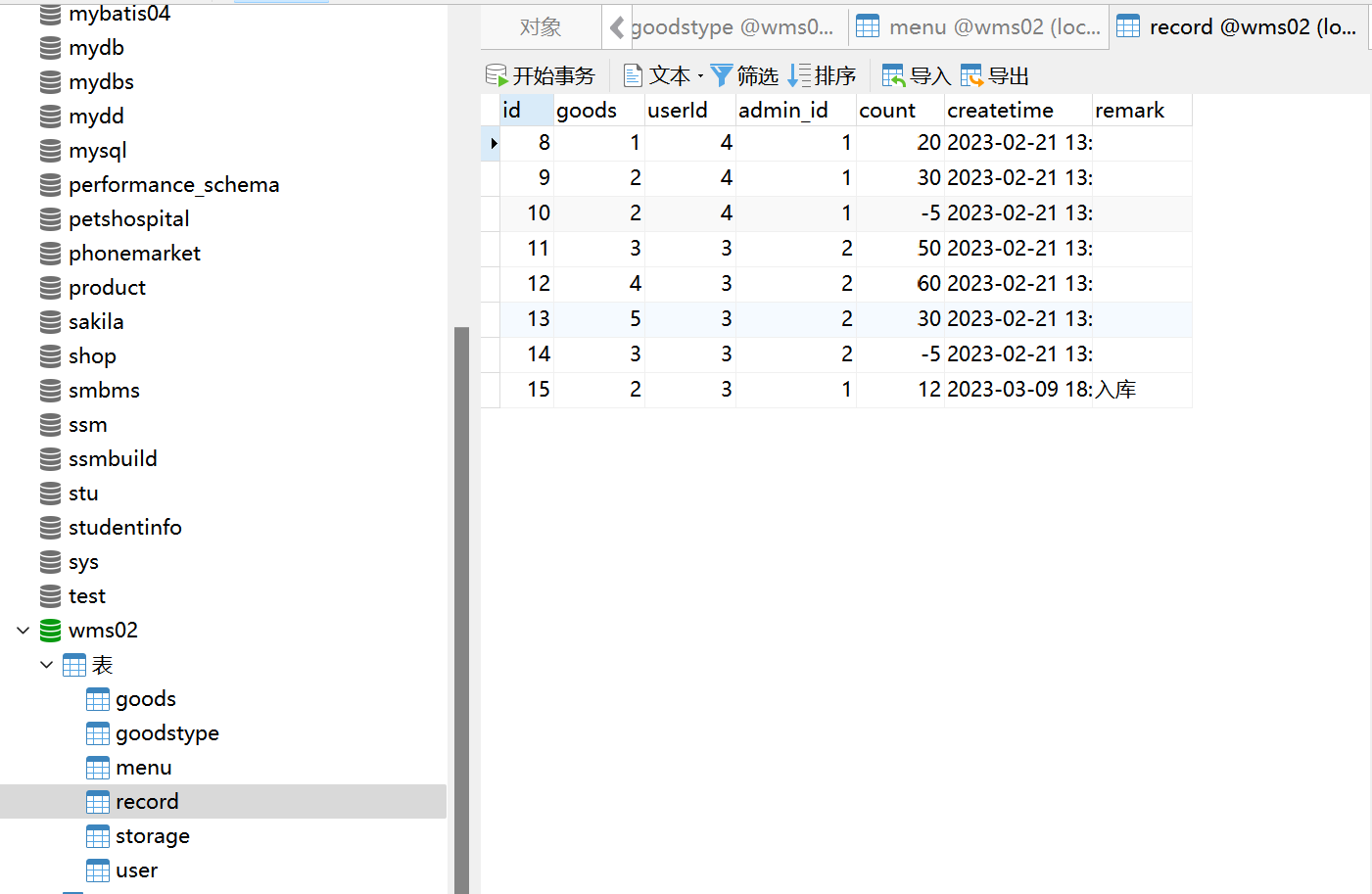
@Data  
@EqualsAndHashCode(callSuper = false)  
@TableName("goodsType")  
@ApiModel(value="Goodstype对象", description="")  
public class Goodstype implements Serializable {  
 private static final long *serialVersionUID* = 1L;  
 @ApiModelProperty(value = "主键")  
 @TableId(value = "id", type = IdType.*AUTO*)  
 private Integer id;  
 @ApiModelProperty(value = "分类名")  
 private String name;  
 @ApiModelProperty(value = "备注")  
 private String remark;  
}

Menu:



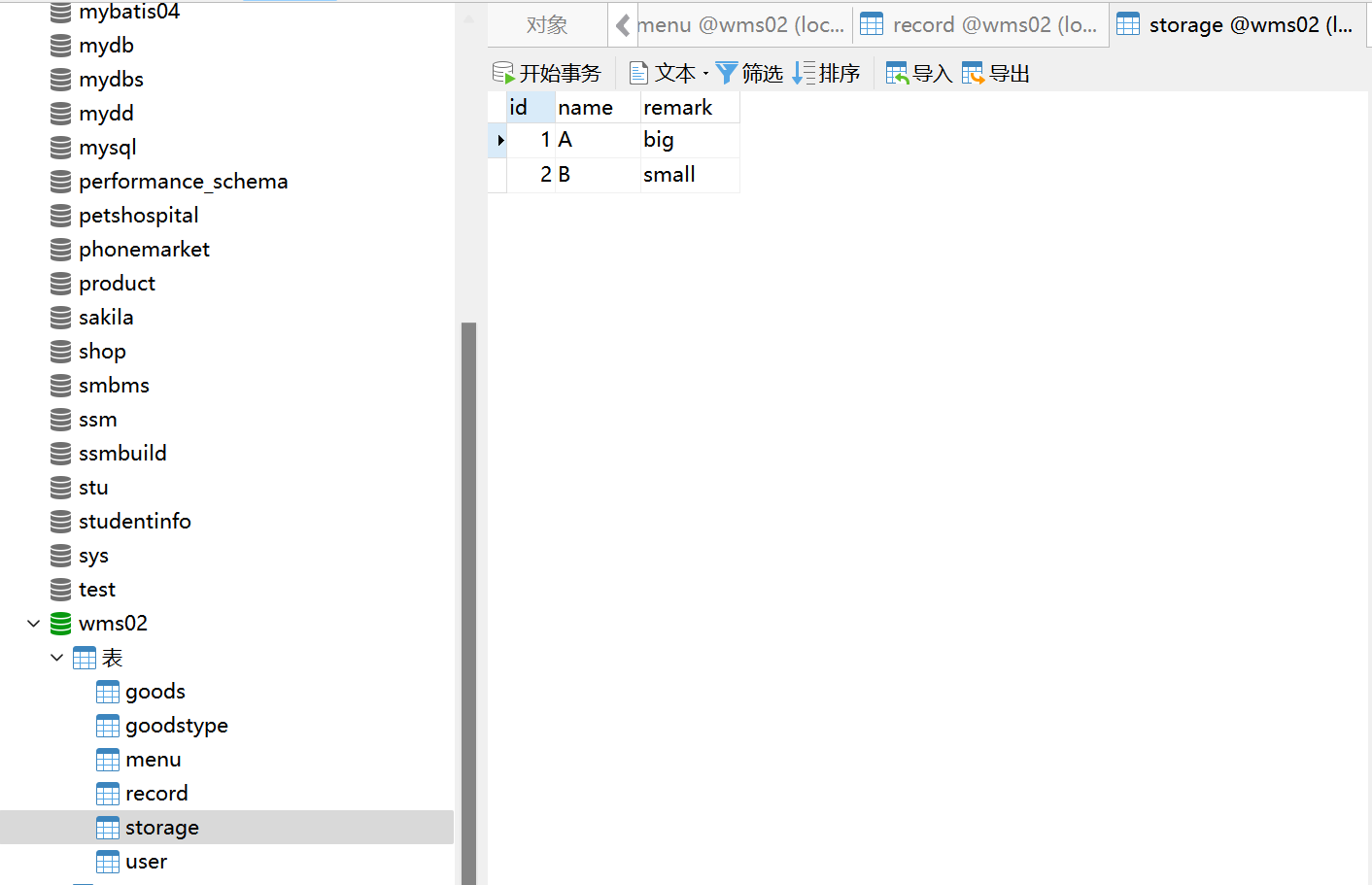
@Data  
@EqualsAndHashCode(callSuper = false)  
@ApiModel(value="Menu对象", description="")  
public class Menu implements Serializable {  
 private static final long *serialVersionUID* = 1L;  
 private Integer id;  
 @ApiModelProperty(value = "菜单编码")  
 @TableField("menuCode")  
 private String menucode;  
 @ApiModelProperty(value = "菜单名字")  
 @TableField("menuName")  
 private String menuname;  
 @ApiModelProperty(value = "菜单级别")  
 @TableField("menuLevel")  
 private String menulevel;  
 @ApiModelProperty(value = "菜单的父code")  
 @TableField("menuParentCode")  
 private String menuparentcode;  
 @ApiModelProperty(value = "点击触发的函数")  
 @TableField("menuClick")  
 private String menuclick;  
 @ApiModelProperty(value = "权限 0超级管理员，1表示管理员，2表示普通用户，可以用逗号组合使用")  
 @TableField("menuRight")  
 private String menuright;  
 @TableField("menuComponent")  
 private String menucomponent;  
 @TableField("menuIcon")  
 private String menuicon;  
}

Record:



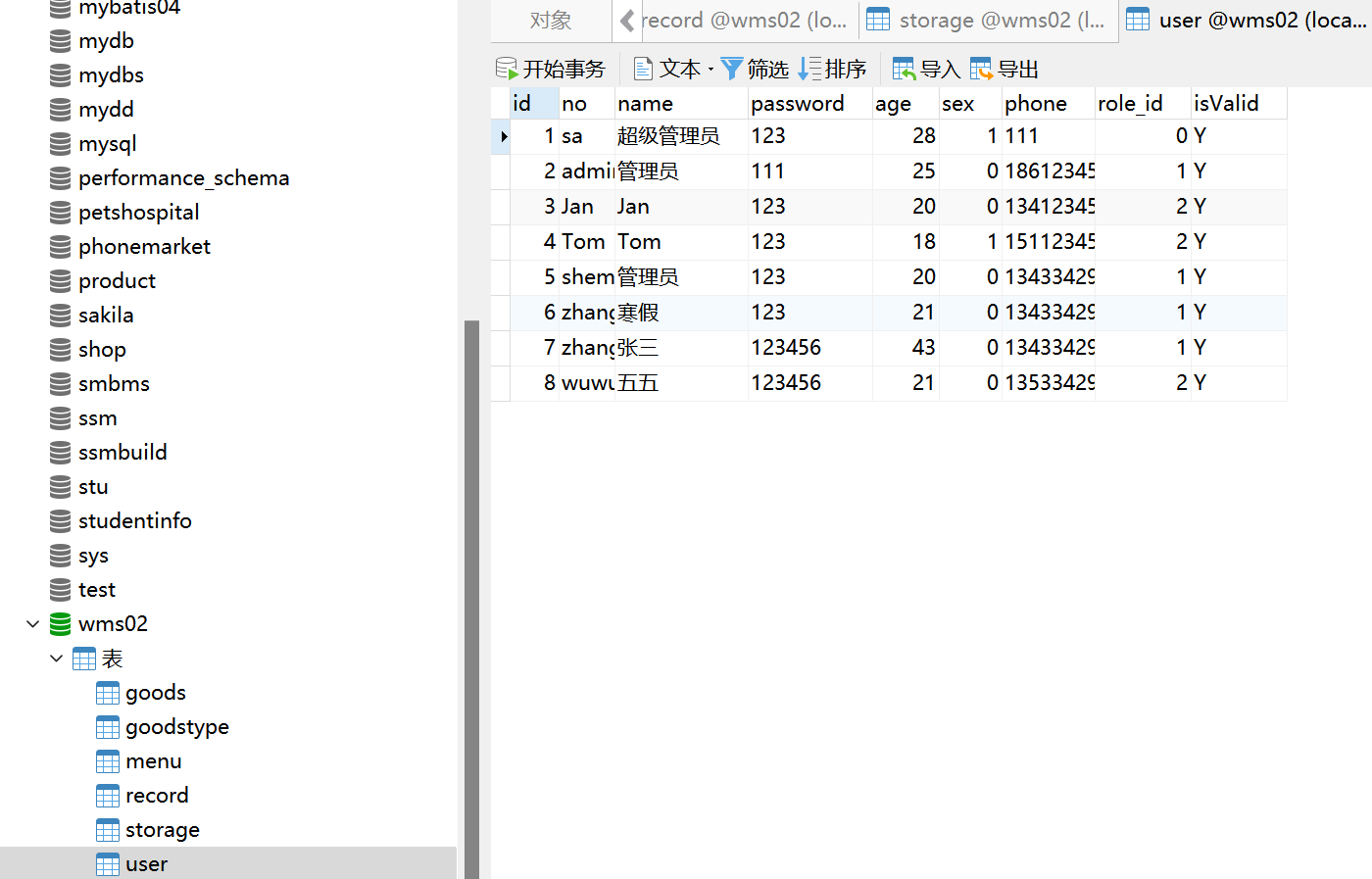
@Data  
@EqualsAndHashCode(callSuper = false)  
@ApiModel(value="Record对象", description="")  
public class Record implements Serializable {  
  
 private static final long *serialVersionUID* = 1L;  
  
 @ApiModelProperty(value = "主键")  
 @TableId(value = "id", type = IdType.*AUTO*)  
 private Integer id;  
 @ApiModelProperty(value = "货品id")  
 private Integer goods;  
 @ApiModelProperty(value = "取货人/补货人")  
 @TableField("userId")  
 private Integer userid;  
 @ApiModelProperty(value = "操作人id")  
 private Integer adminId;  
 @ApiModelProperty(value = "数量")  
 private Integer count;  
 @ApiModelProperty(value = "操作时间")  
 @JsonFormat(pattern = "yyyy-MM-dd HH:mm:ss", timezone="GMT+8")  
 @TableField(fill = FieldFill.*INSERT*)  
 private LocalDateTime createtime;  
 @ApiModelProperty(value = "备注")  
 private String remark;  
 @TableField(exist = false)  
 private String action;  
}

Storage:



@Data  
@EqualsAndHashCode(callSuper = false)  
@ApiModel(value="Storage对象", description="")  
public class Storage implements Serializable {  
 private static final long *serialVersionUID* = 1L;  
 @ApiModelProperty(value = "主键")  
 @TableId(value = "id", type = IdType.*AUTO*)  
 private Integer id;  
 @ApiModelProperty(value = "仓库名")  
 private String name;  
 @ApiModelProperty(value = "备注")  
 private String remark;  
}

User:



@Data  
@EqualsAndHashCode(callSuper = false)  
@ApiModel(value="User对象", description="")  
public class User implements Serializable {  
 private static final long *serialVersionUID* = 1L;  
 @ApiModelProperty(value = "主键")  
 @TableId(value = "id", type = IdType.*AUTO*)  
 private Integer id;  
 @ApiModelProperty(value = "账号")  
 private String no;  
 @ApiModelProperty(value = "名字")  
 private String name;  
 @ApiModelProperty(value = "密码")  
 private String password;  
 private Integer age;  
 @ApiModelProperty(value = "性别")  
 private Integer sex;  
 @ApiModelProperty(value = "电话")  
 private String phone;  
 @ApiModelProperty(value = "角色 0超级管理员，1管理员，2普通账号")  
 private Integer roleId;  
 @ApiModelProperty(value = "是否有效，Y有效，其他无效")  
 @TableField("isValid")  
 private String isvalid;  
}

**Далее я возьму класс сущностей Goods в качестве примера, чтобы объяснить значение каждой аннотации в нем:**

@Data: эта аннотация предоставляется библиотекой Lombok и генерирует шаблонный код для методов getter, setter, toString, equals и hashCode для всех полей в классе.

@EqualsAndHashCode(callSuper = false): эта аннотация также предоставляется Lombok и генерирует метод equals и hashCode на основе полей в классе. Параметр callSuper = false указывает, что сгенерированный метод не должен включать в себя методы equals и hashCode суперкласса.

@ApiModel(value="Goods对象", description=""): эта аннотация предоставляется библиотекой Swagger и используется для описания модели, используемой для веб-служб RESTful. Параметр value предоставляет краткое описание модели, а параметр description предоставляет более подробное описание.

@TableId(value = "id", type = IdType.AUTO): эта аннотация предоставляется MyBatis-Plus и используется для маркировки поля первичного ключа в таблице базы данных. Параметр value указывает имя столбца первичного ключа в таблице, а параметр type указывает тип первичного ключа.

@ApiModelProperty: эта аннотация предоставляется Swagger и используется для описания свойств запроса или ответа веб-службы RESTful. Параметр value предоставляет краткое описание свойства, в то время как другие параметры, такие как required и example, могут использоваться для предоставления дополнительной информации.

@TableField("goodsType"): эта аннотация предоставляется MyBatis-Plus и используется для указания имени столбца базы данных, соответствующего полю в классе. В этом случае поле goodsType в классе соответствует столбцу с именем goodsType в таблице базы данных.

private static final long serialVersionUID = 1L;: Это стандартное поле в сериализуемых классах, которое используется для предоставления номера версии сериализованного объекта. Он используется для обеспечения правильной десериализации объекта, даже если определение класса изменяется между сериализацией и десериализацией.

Остальные аннотации (частный целочисленный идентификатор, имя частной строки, личное хранилище целых чисел, частный тип товара Integer, количество частных целых чисел и примечание частной строки) определяют поля класса Goods, которые представляют атрибуты продукта. или пункт.