哈尔滨工业大学（威海）

计算机科学与技术学院暨软件学院

2017级软件工程专业

《数据库系统》实验报告

|  |  |  |
| --- | --- | --- |
| **序号** | **学号** | **姓名** |
| **1(组长)** | **171110417** | **刘晟驰** |
| **2** | **171110416** | **梁逸聪** |
| **3** | **171110434** | **朱玉鹤** |
| **4** | **171110415** | **李袁名** |
| **5** |  |  |
| **6** |  |  |
| **7** |  |  |

指导教师： 王大顺

报告日期： 2019/5/14

计算机科学与技术学院

软件工程教研室

|  |  |
| --- | --- |
| 得分 | 评价 |
|  |  |

1. **实验环节一 简述系统功能**

***要求：***

1. 在淘宝网任意寻找一家店铺，假定这个店铺的前台程序已经由你们小组设计开发完毕。现在需要设计配套的数据库，请配合店铺截图（5张以上）用500字左右描述你们分析的，和数据库相关的店铺核心功能。
2. 注意，店铺人员、顾客登录、管理等之类的都不算店铺核心功能
3. 给出店铺链接，已备我查看

1.店铺链接：

<https://item.taobao.com/item.htm?id=584962337916&ali_refid=a3_430673_1006:1102238506:N:hQTyEvXcA4FojtIUHDhywA%3D%3D:a1547551146729e87d8bd3e6b824c739&ali_trackid=1_a1547551146729e87d8bd3e6b824c739&spm=a2e15.8261149.07626516002.3>

2.店铺核心功能：

1）店铺基本信息：



2）商品图片展示：



3）该店铺本件商品分类选择：



4）购物明细选择：



5）商品的信息详情：



6）商品评价：



7）订单：



该店铺的核心功能为店家发布商品，买家购买商品。核心数据为店家发布的商品信息，和买家购买后生成的订单信息。对于图一，图一中展示了卖家与该店铺的基本信息，以供顾客浏览查看；图二为商品图片展示，由卖家在店铺上发布商品相关图片，以供买家可以对商品有个直观了解；图三为本件商品分类选择，卖家发布该件商品时，会选择有哪些属性可供买家选择，对应每个属性有几种选择。买家购买时可根据自己的实际需要选择相关属性的商品进行购买；图四为购物明细选择，卖家一般是根据支付方式以及该商品拥有的服务选择属性供买家选择（数量属性必须存在）。买家根据自己的支付方式以及所需要的服务选择相应属性值（数量属性一栏至少为1）；图五为商品信息详情，商品信息详情是对于卖家所售该类商品属性的描述，是一个通用描述，例如该手机的尺寸、摄像头、核心数等，对于销售该种商品的所有店铺的商品信息详情均相同；图六为商品评价，显示购买该店铺商品的用户对所购买商品的评价，评价信息包括：满意度、文字描述、评价时间、评价商品等；图七为订单，买家根据自己需求（图三、图四）选择一个特定属性的商品，进行购买，提交订单。

1. **实验环节二 安装Oracle数据库软件并创建数据库**

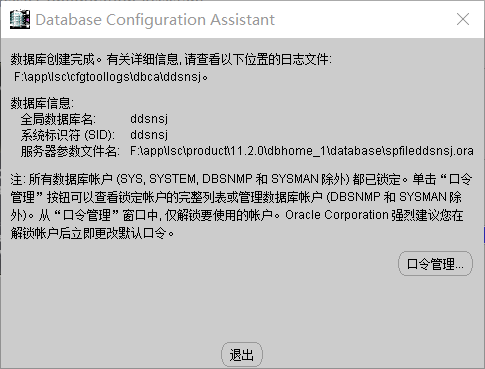
***要求：***

1. 安装oracle数据库软件，配置网络环境
2. 用命令创建一个以“淘宝店铺名称”为名字的数据库和一个以“掌柜名称”为名字的表空间（“淘宝店铺名称”和“掌柜名称”均采用首字母拼音方式，例如：“小米之家”用xmzj）

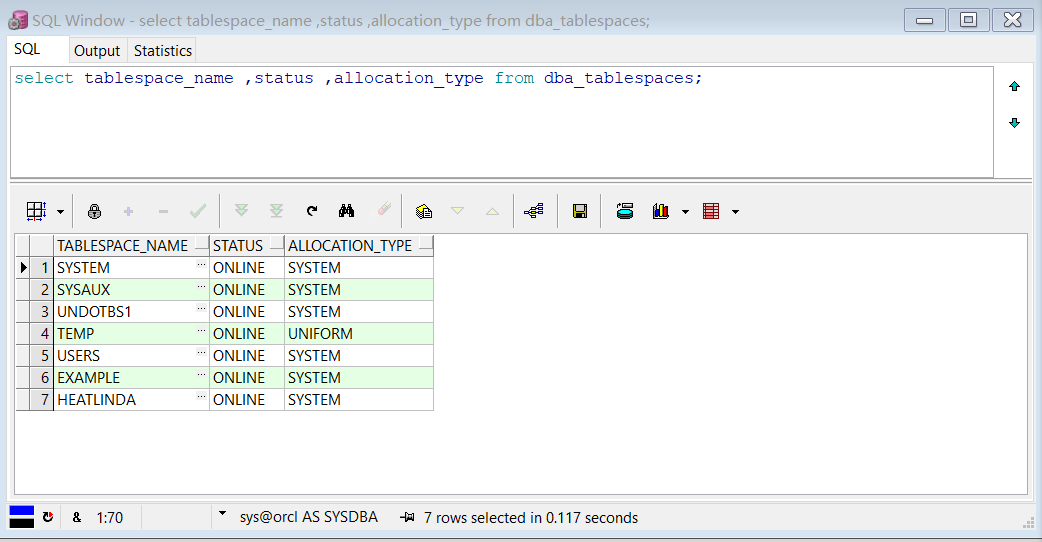
***内容：***

1. *数据库软件、环境配置等的截图*





1. *表空间、数据库的截图*



使用创建数据库时自动创建的dba\_tablespaces视图，查询当前数据库中的表空间名字、状态、支配类型，可以看到新建的’heatlinda’表空间已被创建。

1. *创建表空间的命令源代码*

见附件

1. **实验环节三 分析并创建系统需要的角色并分配相应权限**

***要求***

1. 用命令给每个组员分配角色，全组至少4种以上角色
2. 赋予不同权限，要求每个组员至少3种以上权限，并且每两个组员的权限都不相同

***内容：***

1. *简单对系统需要的角色以及每个成员需要赋予权限的分析（500字左右）*
2. *用PLSQL Developer查看角色的截图*
3. *用PLSQL Developer查看小组成员权限的截图*
4. *创建角色的命令源代码*
5. *赋予权限的命令源代码*
6. 角色及权限分析

*1）系统需要的角色有如下：总管理员，店铺管理员，商品管理员，服务计划员*

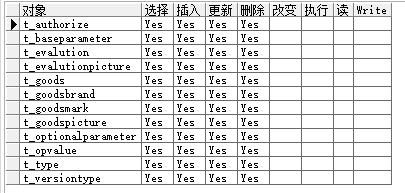
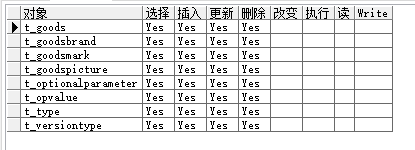
*2）总管理员：负责对整个数据库进行管理，*

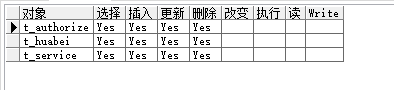
*3）店铺管理员：负责对店铺的信息进行检查管理，以及对店铺老板信息进行记录*

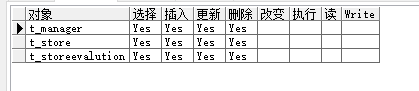
*4）商品管理员，负责对商品的信息进行管理，筛选出不合法的商品，对商品的质量进 行监测*

*5）服务计划员：对商品促销进行计划和实施，以及对售后服务进行安排和管理*

1. 角色权限截图

**

**

**

**



1. 创建角色与权限源代码

见附件

1. **实验环节四 分析并创建并描述系统需要的基本对象**

***要求：***

1. 简单分析系统需要的基本表的构成，描述全局ER图
2. 基本表数量30个以上
3. Check约束5个以上
4. 默认值约束5个以上
5. 序列5个以上
6. 游标5个以上

***内容：***

1. *分析并书写ER图、表对象、属性以及基于此形成的表（30个以上，要有说明）*
2. *用PLSQL Developer查看表、约束、游标等的截图*
3. *创建表、约束等的命令源代码*

1.ER图分析：

整个店铺销售商品系统主要由掌柜、店铺、商品、评论、订单、买家等构成。为了对商品表的表示减少冗余度并且满足三范式，我们将商品概念进一步分割，分为商品型号与商品。商品型号相较于商品为一个较高的层次（如本此实验店铺所售商品型号：Xiaomi/小米 小米9透明九se故宫版10尊享探索plus新手机骁龙855），即不考虑用户可选属性，将具有一个品牌并且同一型号的不同属性的商品集合抽象为一个商品型号。商品即由用户选择了商品具体属性后，可以下单的商品。因而，库存量与价格均与商品直接联系在一起，商品名应与商品型号联系在一起。

经过小组仔细的观察，我们分析得出订单是联系买家、商品、评价的“桥梁”，订单中记录了交易的商品、买家信息、评论信息等。一个订单只能包含一个商品，一个商品可以对应多个订单。买家可通过订单对商品进行评价，评价中只需记录该订单的id即可通过订单获得评价的商品与买家信息。订单还需记录收件人信息的ID，一个买家可能对应多个收件人信息。

为了减少数据库的存储冗余度，以及使数据库表的逻辑更加清晰，我们小组设计每个种类的商品都有一个参数集。对于商家发布新的商品型号，根据所选商品型号的种类确定的参数集（存储参数的名字），并从中选出可选参数集和基本参数集，并且同一商品型号下的商品基本参数与可选参数相同，因此他们应作为商品型号的多值属性存在。其区别在于，对于同一商品型号的商品而言，基本参数集属性值固定，而可选参数集的取值可以由用户根据选项自由选择。

在分析过程中，我们小组也发现了一些卖家对所售商品推出的附加符合以及可选支付方式，同一支付方式可以出现在多个商品中，同一商品可以有多种支付方式；同一服务可以赋予多个商品，多个商品可以同时提供多个服务。因此商品与服务、支付方式为多对多关系。

ER图：

**

2.基本表：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| t\_manager | M\_id | 掌柜id | int | 4 | NO | Primary key |
|  | M\_name | 掌柜名 | varchar(20) | 20 | NO |  |

表1 掌柜表

掌柜表中有掌柜id，掌柜名属性，店铺等与掌柜的关联属性，通过店铺表的外键与掌柜进行关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_store* | *S\_id* | *店铺id* | *int* | *4* | *NO* | *Primary key* |
|  | *S\_name* | *店铺名* | *varchar(20)* | *20* | *NO* |  |
|  | *M\_id* | *掌柜id* | *int* | *4* | *NO* | *Foreign key* |

表2店铺表

店铺表中有店铺id，店铺名和掌柜id属性，通过外键掌柜id与掌柜进行关联，对于店铺的基本信息以及发布的商品信息，均通过其他表的外键与店铺关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_storeevalution* | *SE\_id* | *店铺考评id* | *int* | *4* | *NO* | *Primary key* |
|  | *grade* | *等级* | *int* | *4* | *NO* |  |
|  | *reputation* | *信誉* | *int* | *4* | *NO* |  |
|  | *qualification* | *资质* | *varchar(20)* | *20* | *NO* |  |
|  | *SE\_description* | *描述* | *int* | *4* | *NO* |  |
|  | *serve* | *服务* | *int* | *4* | *NO* |  |
|  | *logistics* | *物流* | *int* | *4* | *NO* |  |
|  | *S\_id* | *店铺id* | *int* | *4* | *NO* | *Foreign key* |

表3 店铺考评表

店铺考评表中有店铺考评id,等级，信誉，资质描述，服务，物流和店铺id属性，通过外键店铺id与店铺关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_goods* | *G\_id* | *商品id* | *int* | *4* | *NO* | *Primary key* |
|  | *inventory* | *库存量* | *int* | *4* | *NO* |  |
|  | *price* | *价格* | *float* | *8* | *NO* |  |
|  | *GM\_id* | *商品型号id* | *int* | *4* | *NO* | *Foreign key* |
|  | *OP\_id* | *可选参数值id* | *int* | *4* | *NO* | *Foreign key* |

表4 商品表

商品表中有商品id，库存量，价格，商品型号id和可选参数值id属性，通过商品型号id和可选参数值id和商品型号，可选参数值关联，而订单则通过其外键商品id与商品表关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_goodspicture* | *GP\_id* | *商品图片id* | *int* | *4* | *NO* | *Primary key* |
|  | *GP\_information* | *商品图片信息* | *varchar(20)* | *20* | *YES* |  |
|  | *GM\_id* | *商品型号id* | *int* | *4* | *NO* | *Foreign key* |

表5 商品图片表

商品图片表中有商品图片id，商品图片信息，商品型号id属性，通过外键商品型号id与商品型号进行关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_goodsmark* | *GM\_id* | *商品型号id* | *int* | *4* | *NO* | *Primary key* |
|  | *GM\_name* | *商品型号名* | *varchar(20)* | *20* | *NO* |  |
|  | *commentquantity* | *累计评论量* | *int* | *4* | *NO* |  |
|  | *TSV* | *交易成功量* | *int* | *4* | *NO* |  |
|  | *BP\_id* | *基本参数值id* | *int* | *4* | *NO* | *Foreign key* |
|  | *GB\_id* | *品牌id* | *int* | *4* | *NO* | *Foreign key* |
|  | *T\_id* | *种类id* | *int* | *4* | *NO* | *Foreign key* |
|  | *S\_id* | *店铺id* | *int* | *4* | *NO* | *Foreign key* |

表6 商品型号表

商品型号表中含有商品型号id，商品型号名，累计评论量，交易成功量，基本参数值id，品牌id，种类id，店铺id属性，其中基本参数值id，品牌id，种类id，店铺id为外键，通过它们与基本参数值，品牌，种类，店铺四张表关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_goodsbrand* | *GB\_id* | *品牌id* | *int* | *4* | *NO* | *Primary key* |
|  | *GB\_name* | *品牌名* | *varchar(20)* | *4* | *NO* |  |

表7 品牌表

品牌表中包含品牌id，品牌名两个属性。与商品型号表通过商品型号表的外键品牌id进行关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_type* | *T\_id* | *种类id* | *int* | *4* | *NO* | *Primary key* |
|  | *T\_name* | *种类名* | *varchar(20)* | *20* | *NO* |  |

表8 种类表

种类表中包含种类id，种类名两个属性。与商品型号表通过商品型号表的外键种类id进行关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_authorize* | *GB\_id* | *品牌id* | *int* | *4* | *NO* | *Primary key* |
|  | *S\_id* | *店铺id* | *int* | *4* | *NO* | *Primary key* |
|  | *aut\_information* | *授权信息* | *varchar(200)* | *20* | *NO* |  |

表9 授权信息表

授权信息表中有品牌id，店铺id，授权信息属性，品牌id和店铺id组合作为其主键。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_parameterset* | *P\_id* | *参数id* | *int* | *4* | *NO* | *Primary key* |
|  | *P\_name* | *参数名* | *varchar(20)* | *20* | *NO* |  |
|  | *T\_id* | *种类id* | *int* | *4* | *NO* | *Foreign key* |

表10 参数表

参数表中有参数id，参数名，种类id属性，通过种类id外键建立与种类表的关联关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_optionalparameter* | *GM\_id* | *商品型号id* | *int* | *4* | *NO* | *Primary key* |
|  | *P\_id* | *参数id* | *int* | *4* | *NO* | *Primary key* |

表11 可选参数表

可选参数表中有商品型号id和参数id，此二属性共同作为此表主键。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_baseparameter* | *GM\_id* | *商品型号id* | *int* | *4* | *NO* | *Primary key* |
|  | *P\_id* | *参数id* | *int* | *4* | *NO* | *Primary key* |

表12 基本参数表

基本参数表中有商品型号id和参数id，此二属性共同作为此表主键。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_opvalue* | *OP\_id* | *可选参数值id* | *int* | *4* | *NO* | *Primary key* |
|  | *color\_id* | *机身颜色id* | *int* | *4* | *NO* | *Foreign key* |
|  | *PT\_id* | *套餐类型id* | *int* | *4* | *NO* | *Foreign key* |
|  | *MC\_id* | *存储容量id* | *int* | *4* | *NO* | *Foreign key* |
|  | *VT\_id* | *版本类型id* | *int* | *4* | *NO* | *Foreign key* |
|  | *HB\_id* | *花呗分期id* | *int* | *4* | *NO* | *Foreign key* |

表13 可选参数值表

可选参数值表中有可选参数值id,机身颜色id，套餐类型id，存储容量id，版本类型id，花呗分期id。机身颜色id，套餐类型id，存储容量id，版本类型id，花呗分期id为外键，通过它们与对应的表关联，商品通过外键可选参数值id与可选参数值表关联。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_bodycolor* | *color\_id* | *机身颜色id* | *int* | *4* | *NO* | *Primary key* |
|  | *color* | *机身颜色* | *varchar(20)* | *20* | *NO* |  |

表14 机身颜色表

机身颜色表中有机身颜色id和机身颜色，通过机身颜色id与可选参数值表建立关联关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_packagetype* | *PT\_id* | *套餐类型id* | *int* | *4* | *NO* | *Primary key* |
|  | *PT\_name* | *套餐类型* | *varchar(20)* | *20* | *NO* |  |

表15 套餐类型表

套餐类型表中有套餐类型id和套餐类型，通过套餐类型id与可选参数值表建立关联关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_memorycapacity* | *MC\_id* | *存储容量id* | *int* | *4* | *NO* | *Primary key* |
|  | *MC\_capacity* | *存储容量* | *int* | *4* | *NO* |  |

表17 存储类型表

存储容量表中有存储容量id和存储容量，通过存储容量id与可选参数值表建立关联关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *表名* | *字段名* | *含义* | *数据类型* | *大小* | *是否为空* | *备注* |
| *t\_versiontype* | *VT\_id* | *版本类型id* | *int* | *4* | *NO* | *Primary key* |
|  | *VT\_name* | *版本类型* | *varchar(20)* | *20* | *NO* |  |

表18 版本类型表

版本类型表中有版本类型id和版本类型，通过版本类型id与可选参数值表建立关联关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_huabei | HB\_id | 花呗分期id | int | 4 | NO | Primary key |
|  | HB | 花呗分期 | varchar(20) | 20 | NO |  |

表19 花呗分期表

花呗分期表中有花呗分期id和花呗分期，通过花呗分期id与可选参数值表建立关联关系

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_bpvalue | BP\_id | 基本参数值id | int | 4 | NO | Primary key |
|  | resolvingpower | 分辨率 | varchar(50) | 20 | NO |  |
|  | networktype | 网络类型 | varchar(50) | 20 | NO |  |
|  | networkmode | 网络模式 | varchar(50) | 20 | NO |  |
|  | corenumber | 核心数 | varchar(50) | 20 | NO |  |
|  | RAM | 运行内存 | varchar(50) | 20 | NO |  |
|  | rearcamera | 后置摄像头 | varchar(50) | 20 | NO |  |
|  | cameratype | 摄像头类型 | varchar(50) | 20 | NO |  |

表20 基本参数值表

基本参数值表中有基本参数值id,分辨率，网络模型，网络模式，核心数，运行内存，后置摄像头，摄像头类型属性。通过基本参数值id属性与商品型号表建立关联关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_paymentmethod | PM\_id | 支付方式id | int | 4 | NO | Primary key |
|  | PM\_explain | 支付方式说明 | varchar(20) | 20 | NO |  |

表21 支付方式表

支付方式表中含有支付方式和支付方式说明两个属性，其他表如商品支付方式访问支付方式id来访问此表。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_goodspaymentmethod | G\_id | 商品id | int | 4 | NO | Primary key |
|  | PM\_id | 支付方式id | int | 4 | NO | Primary key |

表22 商品支付方式表

商品支付方式中有商品id和支付方式id，两个属性共同作为主键，体现商品与支付方式多对多的数量关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_service | Ser\_id | 服务id | int | 4 | NO | Primary key |
|  | Ser\_content | 服务内容 | varchar(20) | 20 | NO |  |

表23 服务表

服务表中含有服务id和服务内容，通过服务id属性与商品服务表建立关联关系

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_goodsservice | G\_id | 商品id | int | 4 | NO | Primary key |
|  | Ser\_id | 服务id | int | 4 | NO | Primary key |
|  | GS\_exclusive | 是否专享 | varchar(2) | 2 | NO | Primary key |

表24 商品服务表

商品服务表中有商品id，服务id，是否专享三个属性，它们共同作为此表的主键。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_evalution | E\_id | 评论id | int | 4 | NO | Primary key |
|  | E\_content | 评论内容 | varchar(20) | 20 | YES |  |
|  | descriptivescore | 描述评分 | int | 4 | NO |  |
|  | goodsscore | 商品评分 | int | 4 | NO |  |
|  | logisticssocre | 物流评分 | int | 4 | NO |  |
|  | time | 时间 | varchar(20) | 10 | NO |  |
|  | O\_id | 订单id | int | 4 | NO | Foreign key |
|  | useful | 有用 | int | 4 | NO |  |

表25 评论表

评论表中有以下属性：评论id，评论内容，描述评分，商品评分，物流评分，时间，订单id，是否有用。其中通过外键订单id与订单建立关联关系，而又以评论id属性与评论图片表建立关系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_evalutionpicture | EP\_id | 评论图片id | int | 4 | NO | Primary key |
|  | EP\_information | 评论图片信息 | varchar(50) | 50 | NO |  |
|  | E\_id | 评论id | int | 4 | NO | Foreign key |

表26 评论图片表

评论图表属性有评论图片id，评论图片信息，评论id，可通过外键评论id访问评论表中的具体内容。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_order | O\_id | 订单id | int | 4 | NO | Primary key |
|  | O\_number | 商品数量 | int | 4 | NO |  |
|  | RI\_id | 收件信息id | int | 4 | NO | Foreign key |
|  | B\_id | 买家id | int | 4 | NO | Foreign key |
|  | G\_id | 商品id | int | 4 | NO | Foreign key |

表27 订单表

订单表有如下属性：订单id，商品数量，收件信息id，买家id，商品id。收件信息id，买家id，商品id为外键，与收件信息，买家，商品建立关联，而其他表亦可通过访问订单id来得到订单表中数据。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_shoppingcarrecord | G\_id | 商品id | int | 4 | NO | Primary key |
|  | B\_id | 买家id | int | 4 | NO | Primary key |
|  | G\_number | 购物车中数量 | int | 4 | NO |  |

表28 购物车记录表

购物车记录表中含有的属性为商品id，买家id和购物车中数量。其中商品id和买家id 同时作为表的主键。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_buyer | B\_id | 买家id | int | 4 | NO | Primary key |
|  | B\_name | 买家姓名 | varchar(10) | 10 | NO |  |
|  | B\_nickname | 买家昵称 | varchar(10) | 10 | NO |  |
|  | b\_level | 买家等级 | int | 4 | NO |  |

表29 买家表

买家表中的属性为买家id，买家姓名，买家昵称和买家等级，通过买家id与购物车就，商品订单建立联系。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_receinformation | RI\_id | 收件信息id | int | 4 | NO | Primary key |
|  | RI\_address | 收件地址 | varchar(20) | 20 | NO |  |
|  | RI\_receiver | 收件人 | varchar(10) | 10 | NO |  |
|  | RI\_contactinformation | 联系方式 | varchar(20) | 20 | NO |  |
|  | B\_id | 买家id | int | 4 | NO | Foreign key |

表30 收件信息表

收件信息表中的属性有收件信息id，收件地址，收件人，联系方式，买家id。其中买家id为外键，用其建立与买家表的关联关系，收件信息id为主键，订单表可通过访问收件信息id得到收件信息表中的数据。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 表名 | 字段名 | 含义 | 数据类型 | 大小 | 是否为空 | 备注 |
| t\_headportrait | H\_id | 头像id | int | 4 | NO | Primary key |
|  | G\_pictureinformation | 图片信息 | varchar(50) | 50 | NO |  |
|  | B\_id | 买家id | int | 4 | NO | Foreign key |

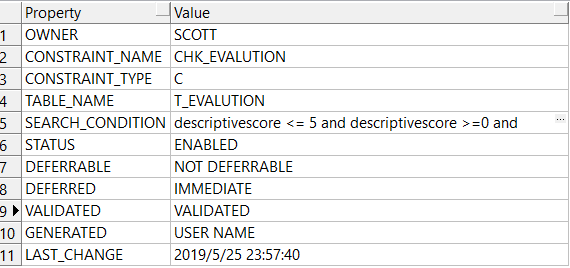
表31 头像表

头像表中的属性为头像id，图片信息，买家id。其中买家id为外键，用其建立与买家表的关联关系。

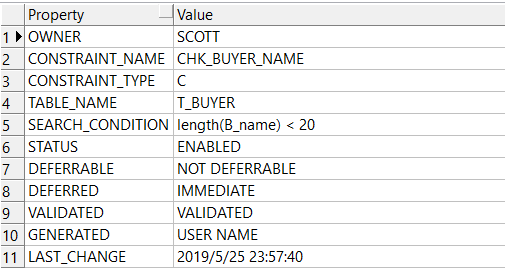
3.check约束

1）对表t\_evalution的check约束：描述评分、商品评分、物流评分值均大于0

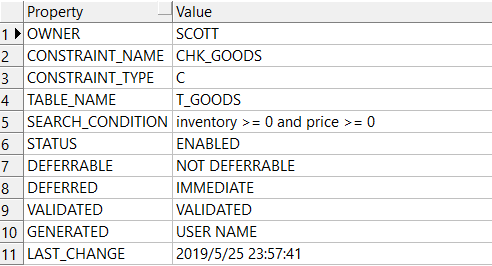
小于等于5.



2）对表t\_buyer的check约束：买家的昵称长度不能超过20

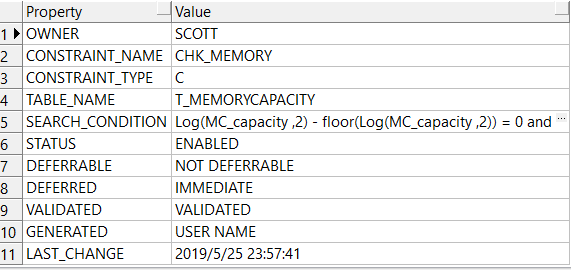


3）对表t\_goods的check约束：库存量与价格均要大于等于0

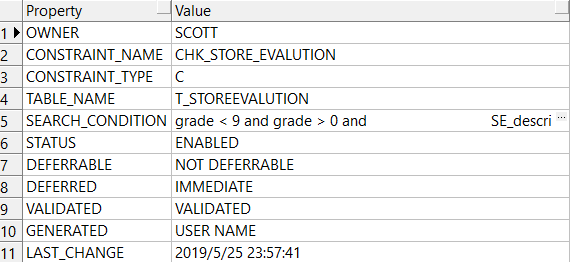


1. 对表t\_memorycapacity的check约束：要求内存大小必须为2的幂次方，并且

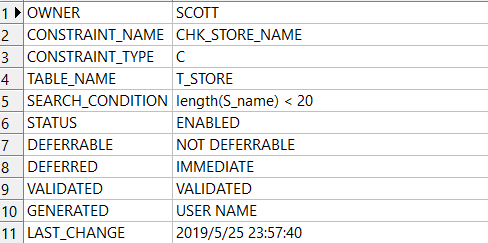
内存大小大于16小于512。



5）对表t\_storeevalution的check约束：要求等级大于0小于等于9，描述、服务、物流评分均要大于0小于等于5.



6)对表t\_store的check约束，要求店铺名字长度小于20



3.默认值约束

1）对商品表中属性invertory(库存量)的默认值约束，默认值为0

注：data\_default显示恒为<long>，点击右侧文本按钮，从Text中可以查看其默认值，下同

IMG_256

2）对商品型号属性commentquantity（评论量）和TSV（交易成功量）默认值约束为0

IMG_256

IMG_256

3）对评论表中属性E\_content（评论内容）设定默认值为‘此用户很懒什么也没有留下’，属性userful（有用）默认值为0。

IMG_256

IMG_256

4）对买家表中属性b\_level（等级）设定默认值为

IMG_256

5）对考评表中属性grade(等级)默认值为1，SE\_\_description（描述）、serve(服务)、logistics(物流)默认值为0.

IMG_256

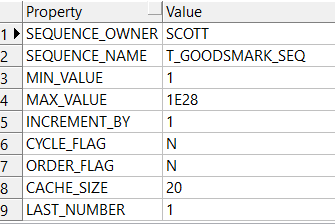
IMG_256

IMG_256

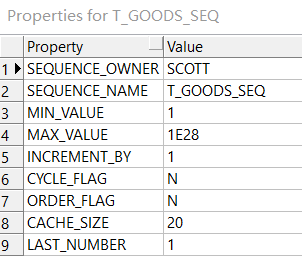
IMG_256

4.序列

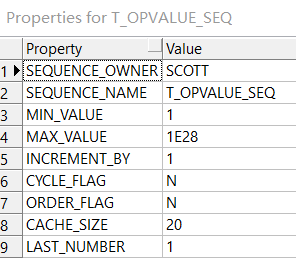
1）商品型号表（t\_goodsmark）序列



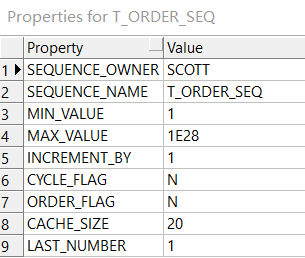
2）商品表（t\_goods）序列



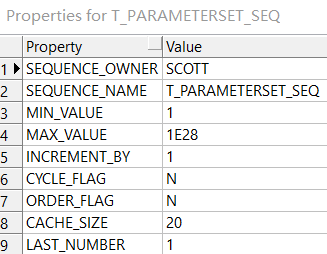
3）可选参数值表（t\_opvalue）序列



4）订单表（t\_order）序列

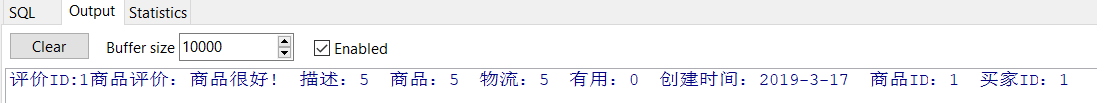


5）参数集表（t\_parameterset）序列



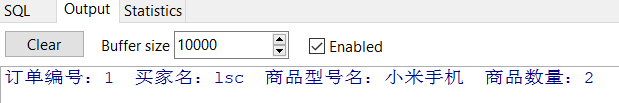
5.游标

1）查看某件商品的全部评论，游标需要传入参数：商品型号ID



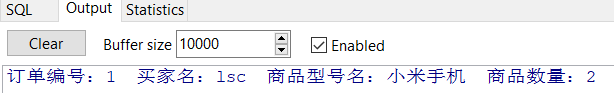


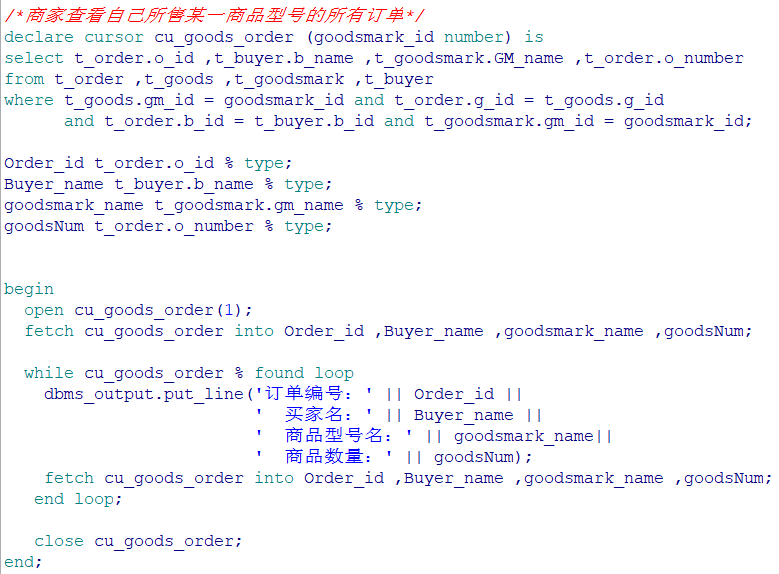
2）查看某位买家的购物车记录，传入参数：买家购物车ID



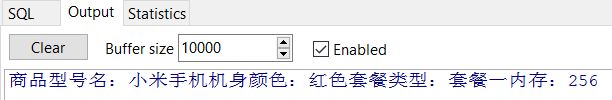


3）订单查询，卖家可以将与自己店铺售卖商品相关的订单全部获取出来，游标传入参数：店铺ID



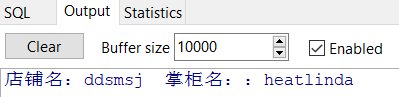


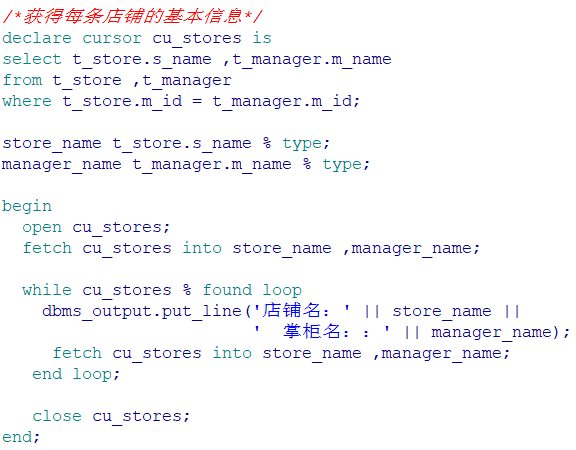
4）店家可以获得店铺所售的所有商品，需要传入参数：店铺ID





1. 通过游标获取每个店铺的基本信息





1. **实验环节五 分析系统需要的视图（7个以上）**

***要求：***

1. 结合系统需要，提取基本表数据形成视图

***内容：***

1. *简要说明视图使用的场景（300-500字左右）*
2. *用PLSQL Developer查看视图的截图*
3. *创建视图的命令源代码*

1.视图使用场景

1）v\_goods视图:几乎所有用户都有查看商品的详细信息的功能，在数据库的实际

存储中，一个商品表被拆分为多张表，因此需要通过视图将商品的完整属性显示出来，以供其他功能方便调用。

2）v\_evalution视图：在网店中，用户可以通过销售商品界面查看一个商品下的评论，如果我们提供一个评论视图，将与评论相关的信息通过视图聚合在一起。系统就可以通过视图得到有关某件商品的某条评论的一些信息。

3）v\_chooseingblock视图：对于卖家来说，需要对店铺内销的售商品型号进行管理，其中某一商品型号的可选参数进行增删改查是一项基本功能。商品型号与商品可选参数分辨位于不同的表中，我们通过视图，将多个表的关键属性聚合在一起。

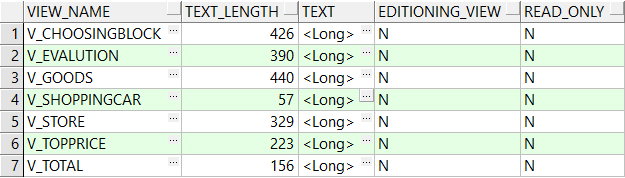
4）v\_store视图：通过淘宝网页可以，用户可以看到一些店铺的信息，这新信息分散数据库中的多张表中，我们可以通过视图，将数据库中与店铺相关的多张表中的属性聚合起来，以便其他与店铺信息有关的功能获取店铺信息。

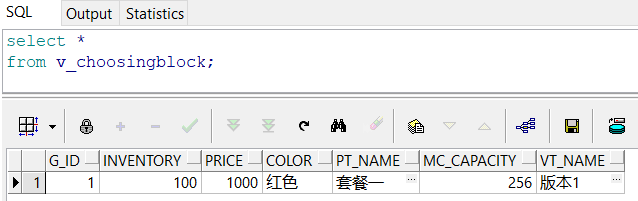
5）v\_topPrice视图：进入一个商品销售界面，我们可以看到该商品型号下商品的基本信息，例如价格区间、累计评论量、交易成功量、店铺名。

6）v\_total视图：该视图是对整个店铺所售商品型号、商品的一个总览，可以从该视图中快速得到一些关键信息。

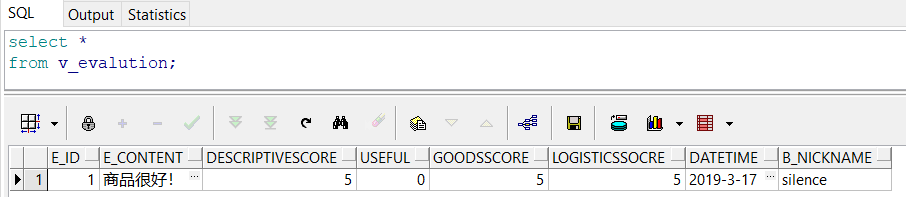
7）v\_shoppingcar视图：买家有查看自己购物车的功能，购物车记录的属性由多个表联合构成；属性组成：商品名、数量、买家名构成。

2.视图截图

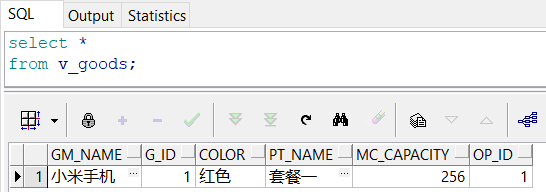




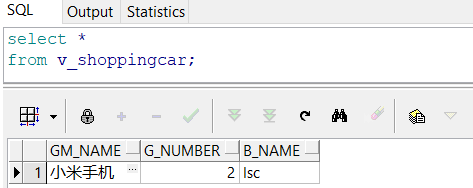
v\_choosingblock视图结果



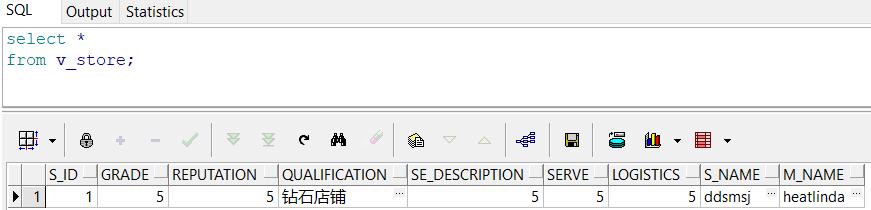
v\_evalution视图截图



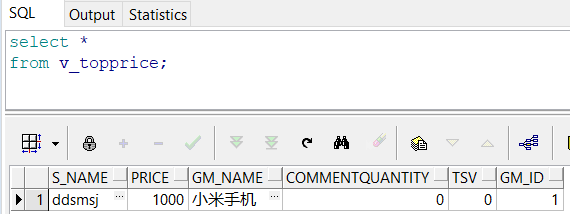
v\_goods视图截图



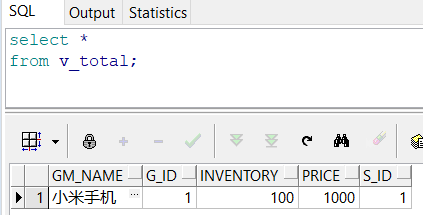
v\_shoppingcar视图截图



v\_store视图截图



v\_topprice视图截图



v\_total视图截图

3.视图命令源代码

见附件

1. **实验环节六 分析系统可能需要的触发器（10以上）**

***要求：***

1. 结合系统需要，设计可能需要的触发器
2. 同类型的触发器数量记为1个，即例如记录日志的触发器设计100个，也按1个计算

***内容：***

1. *分析并描述触发器使用的场景、触发条件、触发类型等（300-500字左右）*
2. *用PLSQL Developer查看触发器的截图*
3. *创建触发器的命令源代码*

1.触发器分析：

1）tr\_adddate\_t\_goods触发器：

对于淘宝等许多购物平台，有时会举行一些活动，对于一些商品可能会限制其发布的时间。因此我们利用行触发器，在卖家发布商品前（向数据库表插入数据前）判断系统当前时间，如果不在允许时间段内，卖家不能发布商品（无法插入）。

1. tr\_autoid\_t\_goods触发器：

对于许多表，主键是通过一个ID标识的，多数情况下ID的选取无特定规则，因此采用递增序可以满足多数条件，通过行触发器，在insert过程前，获取当前表最大ID，并自增一即可实现该功能。

1. tr\_orderdiary\_t\_order触发器：

用户的很多操作比较重要，需要记录在数据库中。例如用户下一个订单，系统需要将该操作记录到数据库中。利用行触发器，在订单插入数据库之后（after）将相关信息记录到日志中。

1. tr\_evalution\_t\_evalution触发器：

店铺的考评表的分值会随着用户对商品的评论的评分而改变，对于每一个评论插入操作，在插入后将评论的分与考评当前分和累计评论量进行加权，得出最终结果。该触发器可以用行触发器实现，执行时机为after。

1. tr\_viewdelete\_t\_goods触发器：

在数据库的实际使用中，我们更多的是对视图进行操作，因此删除视图中一行数据时，我们也要将数据库中对应的若干个表的数据进行删除。采用instead of触发器，当删除v\_goods视图中行数据时，执行删除对应数据库表中数据的操作。

1. tr\_viewinsert\_t\_goods触发器：

当向视图添加数据时，我们想在数据库表中也有相应的插入数据操作。采用instead of触发器，向v\_goods视图添加数据时，向表中也添加数据。

7）tr\_commentquantity\_t\_evalution触发器：

当用户对某一商品评论时，将改商品对应商品型号的评论量+1。采用行触发器，执行时机为after。

1. tr\_tsv\_t\_order触发器：

当用户下订单后（insert订单操作），将该订单中的商品对应的商品型号中交易成功量属性+1。采用行触发器，触发时机为after。

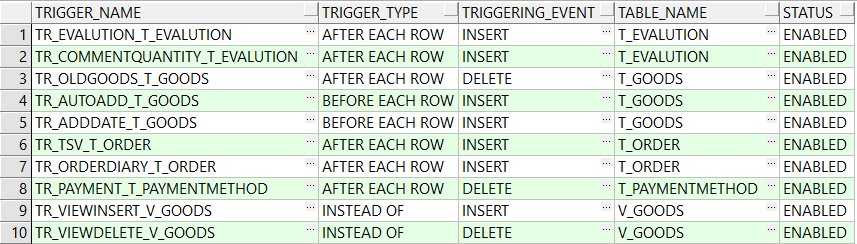
1. tr\_oldgoods\_t\_goods触发器：

当商家删除一个商品时，系统往往不会立刻移除该商品全部信息，触发器会将该商品信息转移到下架商品表中。采用行触发器，触发时机为删除一个商品之后（after）。

1. tr\_payment\_panymentmethod触发器：

当商家统一删除一种支付方式时，会将所有对应此支付方式的商品的支付方式变为支付宝支付。采用行触发器，在对支付方式表进行删除操作之后（after）。

2.触发器截图：



3.触发器代码：

见附件

1. **实验环节七 分析系统可能需要的函数和存储过程（各5个以上）**

***要求：***

1. 结合系统需要，设计可能需要的函数和存储过程

***内容：***

1. *分析并描述函数和存储过程使用的场景等（300-500字左右）*
2. *用PLSQL Developer查看函数和存储过程的截图*
3. *创建函数和存储过程的命令源代码*
4. 函数与存储过程分析

函数分析：

1. f\_goodsnamesearch：

在淘宝等网上商城中，用户需要检索商品。因此我们设计一个函数，输入商品

名，返回检索到的所有同名商品。

1. f\_markinventory:

有时候用户需要看到某一型号商品的库存量，当我们输入商品型号时，通过遍历

商品表中的同型号商品，计算其库存量之和后返回，就可得到某商品型号的库存量。

3）f\_minmarkprice：

用户在电商网站中需要对同型号商品的价格进行对比，通过输入商品型号，返回该型号商品的最便宜价格，来实现此功能。

4）f\_storeordernum:

卖家有时需要知道所拥有的店铺里已经卖出多少订单，通过输入店铺id，返回该店铺下所有订单的数目来实现。

5）f\_maxmarkprice:

在对比价格时，用户有时需要知道某一型号商品的最高端价格，通过输入商品型号，返回该型号下商品的最昂贵价格，来实现此功能。

6）f\_autoinfo:

经常在店铺首页可以看到店铺的授权信息，这是通过输入店铺id，返回该店铺的授权信息来实现的。

存储过程分析：

1. pr\_modifyprice:

商家经常要做的一件事是对商品的价格进行调整，这是一个数据库表中更新操

作，即对输入的商品id所对应的商品价格进行更新。

2）pr\_modifyaddress:

买家有时需要变更收货地址，如出差等。这时候我们需要对买家的收货地址进行更新，通过输入买家信息以及新的收获地址，对数据库表进行更新操作。

3）pr\_pubnewgoods:

卖家发布新商品是一个很常见的功能，这涉及到数据库中7张表的插入操作，卖家的输入信息是文字的，通过插入各对应表中，找到主键id，再将各主键id作为外键写入商品表中来实现。

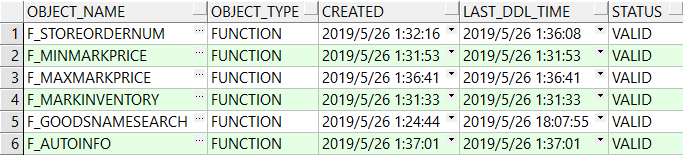
4）pr\_clearshoppingcart:

买家有时候登陆之后，会发现购物车中的商品已经不再需要购买，这时候需要执行清空购物车操作，这是通过对相应卖家的购物车表中对应记录进行删除操作来实现的。

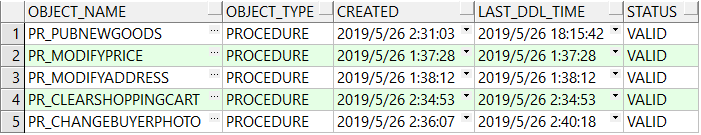
5）pr\_changebuyphoto:

买家可能希望更换一个更酷炫的头像，这时候只需要输入新头像图品在服务器中的url地址字符串以及买家的id。涉及的操作是对头像表中添加一行记录后再更新买家表中相应的关联域。

2.函数截图



3.存储过程截图：



**附录**

一．实验二数据库表空间创建代码：

create tablespace heatlinda datafile 'F:\app\lsc\oradata\orcl\heatlinda.dbf' size 5M;

二．实验三角色与权限源代码：

-- Create the role

create role R\_BOSS;

-- Grant/Revoke object privileges

grant select, insert, update, delete on T\_AUTHORIZE to R\_BOSS;

grant select, insert, update, delete on T\_BASEPARAMETER to R\_BOSS;

grant select, insert, update, delete on T\_EVALUTION to R\_BOSS;

grant select, insert, update, delete on T\_EVALUTIONPICTURE to R\_BOSS;

grant select, insert, update, delete on T\_GOODS to R\_BOSS;

grant select, insert, update, delete on T\_GOODSBRAND to R\_BOSS;

grant select, insert, update, delete on T\_GOODSMARK to R\_BOSS;

grant select, insert, update, delete on T\_GOODSPICTURE to R\_BOSS;

grant select, insert, update, delete on T\_OPTIONALPARAMETER to R\_BOSS;

grant select, insert, update, delete on T\_OPVALUE to R\_BOSS;

grant select, insert, update, delete on T\_TYPE to R\_BOSS;

grant select, insert, update, delete on T\_VERSIONTYPE to R\_BOSS;

-- Create the role

create role R\_GOODSMANAGER;

-- Grant/Revoke object privileges

grant select, insert, update, delete on T\_GOODS to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_GOODSBRAND to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_GOODSMARK to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_GOODSPICTURE to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_OPTIONALPARAMETER to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_OPVALUE to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_TYPE to R\_GOODSMANAGER;

grant select, insert, update, delete on T\_VERSIONTYPE to R\_GOODSMANAGER;

-- Create the role

create role R\_STOREMANAGER;

-- Grant/Revoke object privileges

grant select, insert, update, delete on T\_MANAGER to R\_STOREMANAGER;

grant select, insert, update, delete on T\_STORE to R\_STOREMANAGER;

grant select, insert, update, delete on T\_STOREEVALUTION to R\_STOREMANAGER;

-- Create the role

create role R\_SERVICE;

-- Grant/Revoke object privileges

grant select, insert, update, delete on T\_AUTHORIZE to R\_SERVICE;

grant select, insert, update, delete on T\_HUABEI to R\_SERVICE;

grant select, insert, update, delete on T\_SERVICE to R\_SERVICE;

-- Create the user

create user zhuyuhe

identified by "1234"

default tablespace SYSTEM

temporary tablespace TEMP

profile DEFAULT;

-- Grant/Revoke role privileges

grant r\_goodsmanager to zhuyuhe with admin option;

-- Create the user

create user liushengchi

identified by "1234"

default tablespace SYSTEM

temporary tablespace TEMP

profile DEFAULT;

-- Grant/Revoke role privileges

grant r\_boss to liushengchi with admin option;

-- Create the user

create user liyuanming

identified by "1234"

default tablespace SYSTEM

temporary tablespace TEMP

profile DEFAULT;

-- Grant/Revoke role privileges

grant r\_service to liyuanming;

三．实验四源代码：

1.创建表：

create table t\_manager (

M\_id number not null primary key,

M\_name varchar(20) not null) ;

create table t\_store (

S\_id number not null primary key ,

S\_name varchar(20) not null ,

M\_id number not null);

create table t\_storeevalution (

SE\_id number not null primary key ,

grade number not null ,

reputation number not null,

qualification varchar(20) not null,

SE\_description number not null,

serve number not null,

logistics number not null,

S\_id number not null);

create table t\_goods (

G\_id number not null primary key,

inventory number not null,

price float not null,

GM\_id number not null,

OP\_id number not null);

create table t\_goodspicture (

GP\_id number not null primary key,

GP\_information varchar(50) not null,

GM\_id number not null);

create table t\_goodsmark (

GM\_id number not null primary key,

GM\_name varchar(20) not null,

commentquantity number not null,

TSV number not null,

BP\_id number not null,

GB\_id number not null,

T\_id number not null,

S\_id number not null);

create table t\_goodsbrand (

GB\_id number not null primary key,

GB\_name varchar(20) not null);

create table t\_type (

T\_id number not null primary key,

T\_name varchar(20) not null);

create table t\_authorize (

GB\_id number not null,

S\_id number not null,

aut\_information varchar(200) not null,

constraint pk\_tauthorize primary key(GB\_id,S\_id));

create table t\_parameterset (

P\_id number not null primary key,

P\_name varchar(20) not null,

T\_id number not null);

create table t\_optionalparameter (

GM\_id number not null,

P\_id number not null,

constraint pk\_t\_optionalparameter primary key(GM\_id,P\_id));

create table t\_baseparameter (

GM\_id number not null,

P\_id number not null,

constraint pk\_t\_baseparameter primary key(GM\_id,P\_id));

create table t\_opvalue (

OP\_id number not null primary key,

color\_id number not null,

PT\_id number not null,

MC\_id number not null,

VT\_id number not null,

HB\_id number not null);

create table t\_bodycolor (

color\_id number not null primary key,

color varchar(15) not null);

create table t\_packagetype(

PT\_id number not null primary key,

PT\_name varchar(20) not null);

create table t\_memorycapacity (

MC\_id number not null primary key,

MC\_capacity number not null);

create table t\_versiontype (

VT\_id number not null primary key,

VT\_name varchar(20) not null);

create table t\_huabei (

HB\_id number not null primary key,

HB varchar(20) not null);

create table t\_bpvalue (

BP\_id number not null primary key,

resolvingpower varchar(20) not null,

networktype varchar(20) not null,

networkmode varchar(20) not null,

corenumber varchar(20) not null,

RAM varchar(20) not null,

rearcamera varchar(20) not null,

cameratype varchar(20) not null);

create table t\_paymentmethod (

PM\_id number not null primary key,

PM\_explain varchar(20) not null);

create table t\_goodspaymentmethod(

G\_id number not null ,

PM\_id number not null,

constraint pk\_t\_goodspaymentmethod primary key(G\_id,PM\_id));

create table t\_service (

Ser\_id number not null primary key,

Ser\_content varchar(20) not null);

create table t\_goodsservice (

G\_id number not null ,

Ser\_id number not null,

GS\_exclusive varchar(2) not null,

constraint pk\_t\_goodsservice primary key(G\_id,Ser\_id,GS\_exclusive));

create table t\_evalution (

E\_id number not null primary key,

E\_content varchar(20) ,

descriptivescore number not null,

goodsscore number not null ,

logisticssocre number not null,

dateTime varchar(20) not null,

useful number not null,

O\_id number not null);

create table t\_evalutionpicture (

EP\_id number not null primary key,

EP\_information varchar(50) not null,

E\_id number not null);

create table t\_order (

O\_id number not null primary key,

O\_number number not null,

RI\_id number not null,

B\_id number not null,

G\_id number not null);

create table t\_shoppingcarrecord(

B\_id number not null,

G\_id number not null,

G\_number number not null,

constraint pk\_t\_shoppingcarrecord primary key(B\_id,G\_id));

create table t\_buyer(

B\_id number not null primary key,

B\_name varchar(10) not null,

B\_nickname varchar(10) not null,

b\_level number not null);

create table t\_receinformation(

RI\_id number not null primary key,

RI\_address varchar(100) not null,

RI\_receiver varchar(20) not null,

RI\_contactinformation varchar(20) not null,

B\_id number not null );

create table t\_headportrait(

H\_id number not null primary key,

G\_pictureinformation varchar(50) not null,

B\_id number not null);

添加外键约束：

alter table t\_store add constraint fk\_tstore\_tmanager foreign key (M\_id) references t\_manager(M\_id);

alter table t\_storeevalution add constraint fk\_tstoreevalution\_tstore foreign key (S\_id) references t\_store(S\_id);

alter table t\_goods add constraint fk\_t\_goods\_t\_goodsmark foreign key(GM\_id) references t\_goodsmark(GM\_id);

alter table t\_goods add constraint fk\_t\_goods\_t\_opvalue foreign key(OP\_id) references t\_opvalue(OP\_id);

alter table t\_goodspicture add constraint fk\_tgoodspicture\_t\_goodsmark foreign key(GM\_id) references t\_goodsmark(GM\_id);

alter table t\_goodsmark add constraint fk\_t\_goodsmark\_t\_bpvalue foreign key(BP\_id) references t\_bpvalue(BP\_id);

alter table t\_goodsmark add constraint fk\_t\_goodsmark\_t\_goodsbrand foreign key(GB\_id) references t\_goodsbrand(GB\_id);

alter table t\_goodsmark add constraint fk\_t\_goodsmark\_t\_type foreign key(T\_id) references t\_type(T\_id);

alter table t\_goodsmark add constraint fk\_t\_goodsmark\_t\_store foreign key(S\_id) references t\_store(S\_id);

alter table t\_parameterset add constraint fk\_t\_parameterset\_t\_type foreign key(T\_id) references t\_type(T\_id);

alter table t\_opvalue add constraint fk\_t\_opvalue\_t\_bodycolor foreign key(color\_id) references t\_bodycolor(color\_id);

alter table t\_opvalue add constraint fk\_t\_opvalue\_t\_packagetype foreign key(PT\_id) references t\_packagetype(PT\_id);

alter table t\_opvalue add constraint fk\_t\_opvalue\_t\_memorycapacity foreign key(MC\_id) references t\_memorycapacity(MC\_id);

alter table t\_opvalue add constraint fk\_t\_opvalue\_t\_versiontype foreign key(VT\_id) references t\_versiontype(VT\_id);

alter table t\_opvalue add constraint fk\_t\_opvalue\_t\_huabei foreign key(HB\_id) references t\_huabei(HB\_id);

alter table t\_evalution add constraint fk\_t\_evalution\_t\_order foreign key(O\_id) references t\_order(O\_id);

alter table t\_evalutionpicture add constraint fk\_t\_ep\_t\_evalution foreign key(E\_id) references t\_evalution(E\_id);

alter table t\_order add constraint fk\_t\_order\_t\_receinformation foreign key(RI\_id) references t\_receinformation(RI\_id);

alter table t\_order add constraint fk\_t\_order\_t\_buyer foreign key(B\_id) references t\_buyer(B\_id);

alter table t\_order add constraint fk\_t\_order\_t\_goods foreign key(G\_id) references t\_goods(G\_id);

alter table t\_receinformation add constraint fk\_t\_receinformation\_t\_buyer foreign key(B\_id) references t\_buyer(B\_id);

alter table t\_headportrait add constraint fk\_t\_headportrait\_t\_buyer foreign key(B\_id) references t\_buyer(B\_id);

1. 创建check约束

alter table t\_evalution add constraint chk\_evalution check

(descriptivescore <= 5 and descriptivescore >=0 and

goodsscore <= 5 and goodsscore >= 0 and

logisticssocre <= 5 and logisticssocre >= 0);

alter table t\_buyer add constraint chk\_buyer\_name check (length(B\_name) < 20);

alter table t\_store add constraint chk\_store\_name check (length(S\_name) < 20);

alter table t\_storeevalution add constraint chk\_store\_evalution check

(grade < 9 and grade > 0 and

SE\_description <= 5 and SE\_description >0 and

serve <= 5 and serve > 0 and

logistics <= 5 and logistics > 0);

alter table t\_memorycapacity add constraint chk\_memory check

(Log(MC\_capacity ,2) - floor(Log(MC\_capacity ,2)) = 0 and MC\_capacity <= 512 and MC\_capacity >= 16);

alter table t\_goods add constraint chk\_goods check (inventory >= 0 and price >= 0);

1. 默认值约束

/\*商品表：库存量默认为0\*/

alter table t\_goods modify inventory number default 0;

/\*商品型号表：累计评论量、交易成功量默认为0\*/

alter table t\_goodsmark modify commentquantity number default 0;

alter table t\_goodsmark modify TSV number default 0;

/\*评论表：评论内容默认为"此用户很懒什么也没有留下",有用默认为0\*/

alter table t\_evalution modify E\_content varchar(200) default '此用户很懒什么也没有留下';

alter table t\_evalution modify useful number default 0;

/\*考评表：等级默认为1，描述、服务、物流默认为0\*/

alter table t\_storeevalution modify grade number default 1;

alter table t\_storeevalution modify SE\_description number default 0;

alter table t\_storeevalution modify serve number default 0;

alter table t\_storeevalution modify logistics number default 0;

/\*买家表：等级默认为1\*/

alter table t\_buyer modify b\_level number default 1;

/\*商品服务表：是否专享默认为'否'\*/

alter table t\_goodsservice modify GS\_exclusive varchar(2) default '否';

1. 创建序列

/\*可选参数值表主键OP\_id建立一个序列,实现自增\*/

create sequence t\_opvalue\_seq;

/\*订单表主键O\_id建立一个序列\*/

create sequence t\_order\_seq;

/\*参数值表主键P\_id建立一个序列\*/

create sequence t\_parameterset\_seq;

/\*为商品表主键G\_id建立一个序列\*/

create sequence t\_goods\_seq;

/\*为商品型号主键GM\_id建立一个序列\*/

create sequence t\_goodsmark\_seq;

1. 创建游标

/\*查看某件商品的全部评论

t\_evalution ,t\_order 两表联合\*/

declare cursor cu\_evalution (goods\_id in number) is

select t\_evalution.E\_id ,t\_evalution.E\_content ,t\_evalution.descriptivescore ,

t\_evalution.goodsscore ,t\_evalution.logisticssocre ,t\_evalution.datetime ,

t\_evalution.useful ,t\_order.g\_id ,t\_order.b\_id

from t\_evalution ,t\_order

where t\_order.g\_id = goods\_id and t\_evalution.o\_id = t\_order.o\_id;

E\_id t\_evalution.e\_id % type;

E\_content t\_evalution.e\_content % type;

E\_description t\_evalution.descriptivescore % type;

E\_goodsscore t\_evalution.goodsscore % type;

E\_logistic t\_evalution.logisticssocre % type;

E\_useful t\_evalution.useful % type;

E\_time t\_evalution.dateTime % type;

E\_goods t\_order.g\_id % type;

E\_buyer t\_order.b\_id % type;

begin

open cu\_evalution(1);

fetch cu\_evalution into E\_id ,E\_content ,E\_description ,E\_goodsscore ,

E\_logistic ,E\_useful ,E\_time ,E\_goods ,E\_buyer;

while cu\_evalution % found loop

dbms\_output.put\_line('nihao');

dbms\_output.put\_line('评价ID:' ||E\_id ||CHR(10) ||

' 商品评价：' ||E\_content || CHR(10) ||

' 描述：' ||E\_description || CHR(10) ||

' 商品：' ||E\_goodsscore || CHR(10) ||

' 物流：'||E\_logistic || CHR(10) ||

' 有用：'||E\_useful || CHR(10) ||

' 创建时间：'||E\_time || CHR(10) ||

' 商品ID：' ||E\_goods || CHR(10) ||

' 买家ID：' ||E\_buyer);

fetch cu\_evalution into E\_id ,E\_content ,E\_description ,E\_goodsscore ,

E\_logistic ,E\_useful ,E\_time ,E\_goods ,E\_buyer;

end loop;

close cu\_evalution;

end;

/\*某位买家查看自己的购物车\*/

declare cursor cu\_shoppingcar is

select t\_goodsmark.gm\_name ,t\_shoppingcarrecord.g\_number

from t\_shoppingcarrecord ,t\_goods ,t\_goodsmark

where t\_shoppingcarrecord.b\_id = 1 and t\_shoppingcarrecord.g\_id = t\_goods.g\_id

and t\_goods.gm\_id = t\_goodsmark.gm\_id;

goodsName varchar(20);

goodsNum number;

begin

open cu\_shoppingcar;

fetch cu\_shoppingcar into goodsName ,goodsNum;

while cu\_shoppingcar % found loop

dbms\_output.put\_line('商品型号名：' || goodsName ||

' 商品数量：' || goodsNum);

fetch cu\_shoppingcar into goodsName ,goodsNum;

end loop;

close cu\_shoppingcar;

end;

/\*商家查看自己所售某一商品型号的所有订单\*/

declare cursor cu\_goods\_order (goodsmark\_id number) is

select t\_order.o\_id ,t\_buyer.b\_name ,t\_goodsmark.GM\_name ,t\_order.o\_number

from t\_order ,t\_goods ,t\_goodsmark ,t\_buyer

where t\_goods.gm\_id = goodsmark\_id and t\_order.g\_id = t\_goods.g\_id

and t\_order.b\_id = t\_buyer.b\_id and t\_goodsmark.gm\_id = goodsmark\_id;

Order\_id t\_order.o\_id % type;

Buyer\_name t\_buyer.b\_name % type;

goodsmark\_name t\_goodsmark.gm\_name % type;

goodsNum t\_order.o\_number % type;

begin

open cu\_goods\_order(1);

fetch cu\_goods\_order into Order\_id ,Buyer\_name ,goodsmark\_name ,goodsNum;

while cu\_goods\_order % found loop

dbms\_output.put\_line('订单编号：' || Order\_id ||

' 买家名：' || Buyer\_name ||

' 商品型号名：' || goodsmark\_name||

' 商品数量：' || goodsNum);

fetch cu\_goods\_order into Order\_id ,Buyer\_name ,goodsmark\_name ,goodsNum;

end loop;

close cu\_goods\_order;

end;

/\*商家查看自己所售的所有商品\*/

declare cursor cu\_goods (store\_id number) is

Select t\_goodsmark.gm\_name ,t\_bodycolor.color ,

t\_packagetype.pt\_name ,t\_memorycapacity.mc\_capacity

from t\_goodsmark ,t\_goods ,t\_opvalue ,t\_bodycolor ,t\_packagetype ,t\_memorycapacity

where t\_goodsmark.S\_id = store\_id and t\_goods.GM\_id = t\_goodsmark.gm\_id and

t\_goods.op\_id = t\_opvalue.op\_id and t\_opvalue.color\_id = t\_bodycolor.color\_id and

t\_opvalue.pt\_id = t\_packagetype.pt\_id and t\_opvalue.mc\_id = t\_memorycapacity.mc\_id;

goodsmark\_name t\_goodsmark.gm\_name % type;

bodycolor t\_bodycolor.color % type;

packagetype t\_packagetype.pt\_name % type;

memorycapacity t\_memorycapacity.mc\_capacity % type;

begin

open cu\_goods(1);

fetch cu\_goods into goodsmark\_name ,bodycolor ,packagetype ,memorycapacity;

while cu\_goods % found loop

dbms\_output.put\_line('商品型号名：' || goodsmark\_name ||

' 机身颜色：' || bodycolor ||

' 套餐类型：' || packagetype ||

' 内存：' || memorycapacity);

fetch cu\_goods into goodsmark\_name ,bodycolor ,packagetype ,memorycapacity;

end loop;

close cu\_goods;

end;

/\*获得每条店铺的基本信息\*/

declare cursor cu\_stores is

select t\_store.s\_name ,t\_manager.m\_name

from t\_store ,t\_manager

where t\_store.m\_id = t\_manager.m\_id;

store\_name t\_store.s\_name % type;

manager\_name t\_manager.m\_name % type;

begin

open cu\_stores;

fetch cu\_stores into store\_name ,manager\_name;

while cu\_stores % found loop

dbms\_output.put\_line('店铺名：' || store\_name ||

' 掌柜名：：' || manager\_name);

fetch cu\_stores into store\_name ,manager\_name;

end loop;

close cu\_stores;

end;

四．实验五视图代码

1. v\_goods

create or replace view v\_goods as

select t\_goodsmark.gm\_name ,t\_goods.g\_id ,t\_bodycolor.color ,t\_packagetype.pt\_name ,

t\_memorycapacity.mc\_capacity ,t\_goods.op\_id

from t\_goodsmark ,t\_goods ,t\_opvalue ,t\_bodycolor ,t\_packagetype ,t\_memorycapacity

where t\_goods.op\_id = t\_opvalue.op\_id and t\_opvalue.color\_id = t\_bodycolor.color\_id and

t\_opvalue.pt\_id = t\_packagetype.pt\_id and t\_opvalue.mc\_id =t\_memorycapacity.mc\_id and t\_goodsmark.gm\_id = t\_goods.gm\_id;

2.v\_evalution

create or replace view v\_evalution as

select t\_evalution.e\_id,t\_evalution.e\_content,t\_evalution.descriptivescore,t\_evalution.useful,

t\_evalution.goodsscore,t\_evalution.logisticssocre,t\_evalution.datetime,

t\_buyer.b\_nickname,t\_headportrait.g\_pictureinformation

from t\_evalution,t\_headportrait,t\_buyer ,t\_order

where t\_evalution.o\_id = t\_order.o\_id and t\_order.b\_id = t\_buyer.b\_id and

t\_buyer.b\_id=t\_headportrait.b\_id;

3.v\_chooseingblock

create or replace view v\_choosingblock as

select t\_goods.g\_id,t\_goods.inventory,t\_goods.price,t\_bodycolor.color,t\_packagetype.pt\_name,

t\_memorycapacity.mc\_capacity,t\_versiontype.vt\_name

from t\_goods,t\_bodycolor,t\_memorycapacity,t\_packagetype,t\_versiontype,t\_opvalue

where t\_goods.op\_id=t\_opvalue.op\_id and t\_opvalue.color\_id=t\_bodycolor.color\_id and t\_opvalue.pt\_id=t\_packagetype.pt\_id andt\_opvalue.mc\_id=t\_memorycapacity.mc\_id

and t\_opvalue.vt\_id=t\_versiontype.vt\_id;

4.v\_store

create or replace view v\_store as

select t\_store.s\_id,t\_storeevalution.grade,t\_storeevalution.reputation,t\_storeevalution.qualificati on,t\_storeevalution.se\_description,t\_storeevalution.serve,t\_storeevalution.logistics,

t\_store.S\_name,t\_manager.M\_name

from t\_store,t\_manager,t\_storeevalution

where t\_store.m\_id=t\_manager.m\_id and t\_store.s\_id=t\_storeevalution.s\_id;

5.v\_topPrice

create or replace view v\_topprice as

select t\_store.s\_name,t\_goods.price,t\_goodsmark.gm\_name,t\_goodsmark.commentquantity,

t\_goodsmark.tsv,t\_goodsmark.gm\_id

from t\_store,t\_goods,t\_goodsmark

where t\_goods.gm\_id=t\_goodsmark.gm\_id and t\_store.s\_id=t\_goodsmark.s\_id;

6.v\_total

create or replace view v\_total as

select t\_goods.g\_id,t\_store.s\_id,t\_goodsmark.gm\_id

from t\_goods,t\_goodsmark,t\_store

where t\_goods.gm\_id=t\_goodsmark.gm\_id and t\_goodsmark.s\_id=t\_store.s\_id;

7.v\_shoppingcar

create or replace view v\_shoppingcar as

select t\_goodsmark.gm\_name ,t\_shoppingcarrecord.g\_number ,t\_buyer.b\_name

from t\_shoppingcarrecord ,t\_goods ,t\_goodsmark ,t\_buyer

where t\_shoppingcarrecord.b\_id = t\_buyer.b\_id and

t\_shoppingcarrecord.g\_id = t\_goods.g\_id and

t\_goods.gm\_id = t\_goods.gm\_id;

1. 实验六触发器代码
2. tr\_addDate\_t\_goods

create or replace trigger tr\_addDate\_t\_goods

before insert

on t\_goods

for each row

declare

-- local variables here

begin

IF(to\_char(sysdate,'DY')!='星期一') THEN

RAISE\_APPLICATION\_ERROR(-20600,'只能在周一上架商品');

END IF;

end tr\_addDate;

1. tr\_autoid\_t\_goods

create or replace trigger tr\_autoadd\_t\_goods

before insert

on t\_goods

for each row

declare

-- local variables here

NEXT\_ID NUMBER;

begin

SELECT t\_goods\_seq.NEXTVAL INTO NEXT\_ID FROM DUAL;

:NEW.G\_id := NEXT\_ID; --:NEW表示新插入的那条记录

end tr\_autoadd;

1. tr\_orderdiary\_t\_order

create or replace trigger tr\_orderdiary\_t\_order

after insert

on t\_order

for each row

declare

-- local variables here

begin

INSERT INTO ORDER\_DIARY

VALUES

(:new.b\_id ,'下订单', TO\_CHAR(SYSDATE, 'yyyy-mm-dd hh24:mi:ss')); --USER表示当前用户名

end tr\_orderdiary;

1. tr\_evalution\_t\_evalution

create or replace trigger tr\_evalution\_t\_evalution

after insert

on t\_evalution

FOR EACH ROW--对表的每一行触发器执行一次

declare

-- local variables here

mys\_id number;

begin

select t5.s\_id

into mys\_id

from t\_evalution t1,t\_order t3,t\_goods t4,t\_goodsmark t5

where t1.o\_id=t3.o\_id and

t3.g\_id=t4.g\_id and

t4.gm\_id=t5.gm\_id and

t1.o\_id=:new.o\_id;

update T\_STOREEVALUTION

set (se\_description,serve,logistics)=(

select avg(t1.descriptivescore),avg(t1.goodsscore),avg(t1.logisticssocre)

from t\_evalution t1,t\_storeevalution t2,t\_order t3,t\_goods t4,t\_goodsmark t5

where t1.o\_id=t3.o\_id and

t3.g\_id=t4.g\_id and

t4.gm\_id=t5.gm\_id and

t5.s\_id=t2.s\_id and

t2.s\_id=mys\_id

)

where s\_id=mys\_id;

end tr\_evalution;

1. tr\_viewdelete\_v\_goods

create or replace trigger tr\_viewdelete\_v\_goods

instead of delete

on v\_goods

for each row

declare

-- local variables here

begin

delete from t\_goods t1

where t1.g\_id=:old.g\_id;

delete from t\_opvalue t2

where t2.op\_id=:old.op\_id;

end tr\_viewdelete;

1. TR\_viewinsert\_v\_goods

create or replace trigger TR\_viewinsert\_v\_goods

instead of insert

on v\_goods

for each row

declare

-- local variables here

begin

insert into t\_goods(g\_id,op\_id)

values(:new.g\_id,:new.op\_id);

insert into t\_bodycolor(color)

values(:new.color);

insert into t\_packagetype(pt\_name)

values(:new.pt\_name);

insert into t\_memorycapacity(mc\_capacity)

values(:new.mc\_capacity);

end TR\_viewinsert;

1. tr\_commentquantity\_t\_evalution

create or replace trigger tr\_commentquantity\_t\_evalution

after insert

on t\_evalution

for each row

declare

-- local variables here

mygm\_id number;

begin

select t4.gm\_id

into mygm\_id

from t\_order t3,t\_goods t4

where t3.o\_id=:new.o\_id and

t3.g\_id=t4.g\_id ;

update t\_goodsmark t1

set t1.commentquantity = t1.commentquantity+1

where t1.gm\_id=mygm\_id;

end tr\_commentquantity;

1. tr\_tsv\_t\_order

create or replace trigger tr\_tsv\_t\_order

after insert

on t\_order

for each row

declare

-- local variables here

begin

update t\_goodsmark

set tsv = tsv+1

where gm\_id = (select gm\_id from t\_goods t1

where t1.g\_id=:new.g\_id);

end tr\_tsv;

1. tr\_oldgoods\_t\_goods

create or replace trigger tr\_oldgoods\_t\_goods

after delete

on t\_goods

for each row

declare

-- local variables here

begin

INSERT INTO T\_OLDGOODS

VALUES

(:old.g\_id,:old.inventory,:old.price,:old.GM\_id,:old.OP\_id);

end tr\_oldgoods;

1. TR\_PAYMENT\_t\_paymentmethod

create or replace trigger TR\_PAYMENT\_t\_paymentmethod

after delete

on t\_paymentmethod

for each row

declare

-- local variables here

begin

UPDATE t\_goodspaymentmethod

SET pm\_id = 1

WHERE pm\_id=:old.pm\_id;

end TR\_PAYMENT;

1. 实验七函数与存储过程代码

函数：

1. f\_goodsnamesearch

create or replace function f\_goodsnamesearch(g\_name in varchar2)

return t\_goods%rowtype

is

FunctionResult t\_goods%rowtype;

begin

select t1.g\_id,t1.inventory,t1.price,t1.gm\_id,t1.op\_id into FunctionResult

from t\_goods t1,t\_goodsmark t2

where t2.gm\_name=g\_name and t1.gm\_id=t2.gm\_id;

return(FunctionResult);

end f\_goodsnamesearch;

1. f\_markinventory

create or replace function f\_markinventory(mygm\_id in number)

return number

is

FunctionResult number;

begin

select count(\*) into FunctionResult

from t\_goods

where gm\_id=mygm\_id

group by gm\_id;

return(FunctionResult);

end f\_markInventory;

1. f\_minmarkprice

create or replace function f\_minmarkprice(mygm\_id in number) return float is

FunctionResult float;

begin

select min(price) into FunctionResult

from t\_goods

where gm\_id=mygm\_id

group by gm\_id;

return(FunctionResult);

end f\_minmarkprice;

1. f\_storeordernum

create or replace function f\_storeordernum(mys\_id in number) return number is

FunctionResult number:=0;

begin

select count(\*) into FunctionResult

from t\_order t1,t\_goods t2,t\_goodsmark t3

where t1.g\_id=t2.g\_id and t2.gm\_id=t3.gm\_id and t3.s\_id=mys\_id;

return(FunctionResult);

end f\_storeordernum;

1. f\_maxmarkprice

create or replace function f\_maxmarkprice(mygm\_id in number) return float is

FunctionResult float;

begin

select max(price) into FunctionResult

from t\_goods

where gm\_id=mygm\_id

group by gm\_id;

return(FunctionResult);

end f\_maxmarkprice;

1. f\_autoinfo

create or replace function f\_autoinfo(mys\_id in number)

return t\_authorize.aut\_information%type

is

FunctionResult t\_authorize.aut\_information%type;

begin

select aut\_information into FunctionResult

from t\_authorize

where s\_id=mys\_id;

return(FunctionResult);

end f\_autoinfo;

存储过程：

1. pr\_modifyprice

create or replace procedure pr\_modifyprice(myg\_id in number,myprice in float) is

begin

update t\_goods

set price = myprice

where g\_id=myg\_id;

end pr\_modifyprice;

1. pr\_modifyAddress

create or replace procedure pr\_modifyAddress(myb\_id in number,myri\_address in varchar2) is

begin

update t\_receinformation

set ri\_address = myri\_address

where b\_id=myb\_id;

end pr\_modifyAddress;

1. pr\_pubnewgoods

create or replace procedure pr\_pubnewgoods(myinventory in number, myprice in float,

mygm\_name in varchar2, mycolor in varchar2,

mypt\_name in varchar2, mymc\_capacity in number,

myvt\_name in varchar2, myhb in varchar2) is

mycolor\_id number;

mypt\_id number;

mymc\_id number;

myvt\_id number;

myhb\_id number;

mygm\_id number;

myop\_id number;

begin

insert into t\_bodycolor(color)

values(mycolor);

select color\_id into mycolor\_id

from t\_bodycolor

where color=mycolor;

insert into t\_packagetype(pt\_name)

values(mypt\_name);

select pt\_id into mypt\_id

from t\_packagetype

where pt\_name=mypt\_name;

insert into t\_memorycapacity(mc\_capacity)

values(mymc\_capacity);

select mc\_id into mymc\_id

from t\_memorycapacity

where mc\_capacity=mymc\_capacity;

insert into t\_huabei(hb)

values(myhb);

select hb\_id into myhb\_id

from t\_huabei

where hb=myhb;

insert into t\_versiontype(vt\_name)

values(myvt\_name);

select vt\_id into myvt\_id

from t\_versiontype

where vt\_name=myvt\_name;

Insert into t\_opvalue(color\_id,PT\_id,MC\_id,VT\_id,HB\_id)

values(mycolor\_id,myPT\_id,myMC\_id,myVT\_id,myHB\_id);

select op\_id into myop\_id

from t\_opvalue

where color\_id=mycolor\_id and

PT\_id=myPT\_id and

MC\_id=myMC\_id and

VT\_id=myVT\_id and

HB\_id=myHB\_id ;

select gm\_id into mygm\_id

from t\_goodsmark

where gm\_name=mygm\_name;

insert into t\_goods(inventory,price,gm\_id,op\_id)

values(myinventory,myprice,mygm\_id,myop\_id);

end pr\_pubnewgoods;

1. pr\_clearshoppingcart

create or replace procedure pr\_clearshoppingcart(myb\_id in number) is

begin

delete from t\_shoppingcarrecord

where b\_id=myb\_id;

end pr\_clearshoppingcart;

5.pr\_changebuyerphoto

create or replace procedure pr\_changebuyerphoto(myb\_id in number,myurl in varchar2) is

begin

update t\_headportrait

set G\_pictureinformation=myurl

where b\_id=myb\_id;

end pr\_changebuyerphoto;