**Oracle数据库实训任务书**

**一、实训目标**

通过项目实战，理解数据库的基本概念，熟悉Oracle的基本操作，创建用户、表、检查约束，编写SQL语句执行数据库查询、新增、修改、删除，为Java、.Net开发提供数据存储及数据操作的基础。

课程要求从两个方面的能力目标实现，素养目标要求学员在通过学习具备相对应的学习技能、素养要求；技术目标要求学员必须掌握课程产品所定义的知识与技能。

（1）素养目标

* 分析需求的能力；
* 问题解决能力、总结归纳能力。

（2）技术目标

* 操作Oracle的能力；
* SQL语句编写的能力。

**二、实训内容（见附录）**

|  |  |  |
| --- | --- | --- |
| 日期 | 课题 | 内容 |
| 2017-6-20 | Oracle数据库简介、怎样设计数据库、创建用户、创建表、修改表、添加约束、简单的查询、  导入导出 | * Oracle数据库简介以及创建用户、表； * 怎样设计数据库； * 创建表的时候添加约束； * 简单查询； * 数据及表和用户的导入导出 * 了解学生目前基础知识的掌握情况为后续做准备 |
| 2017-6-21 | Oracle数据库的查询、新增、修改、删除 | * 分组查询、模糊查询； * 连接查询、子查询； * 函数； * 增加、修改、删除操作 |
| 2017-6-22 | Oracle高级应用及数据库事务 | * PL/Sql语句； * 序列、同义词、索引、视图； * 游标 * 存储过程、函数和包 * 触发器 * 数据库事务 |
| 2017-6-23—  2017-6-24 | 项目 | * 给出本期培训项目的详细需求 * 引导学生通过分析需求，完成模块分析 * 项目数据库建立。 * 引导学生完成项目中基础数据录入，以及各种查询，修改，统计功能。 * 项目中数据库事务的使用规则 * SQL语句优化 * 修改项目中的SQL语句 * 验收项目及相关文档，指出其中不足做最后修改。 |
| 备注 | * 根据学生完成进度会加入适当的用户体验的功能作为扩展。 | |

**三、实训进度安排**

按教学计划规定，该设计总学时数为1周，其进度及时间大致分配如下：

|  |  |  |
| --- | --- | --- |
| 序号 | 设计内容 | 天数（约占比例） |
| 1 | 基于Oracle的网上商城购物系统数据库的设计 | 5 |
| 总计 |  | 5天 |

**四、考核评定**

|  |  |  |  |
| --- | --- | --- | --- |
| **考核分类** | | **考核方式** | **成绩比例** |
| 过程考核 | 课堂表现 | 学习态度、课堂提问、作业 | 30％ |
| 项目考核 | 项目考核 | 项目实战表现、自主学习 | 70％ |

青岛科技大学

**《Oracle数据库实训》**

题目 网上商城购物系统

指导教师： 徐美娇

姓 名： 刘洪泽

学 号： 1508060306

班 级： 软件153

专 业： 软件工程

地 点：青科大崂山校区弘毅楼c408

时 间： 2017.6.20 至 2017.6.24

**一、需求规格说明**

1 文档介绍

* 1. 文档目的

本需求分析规格说明书的目的是为软件开发和客户之间就“网上购物平台”项目提供需求范围和需求内容的规定。

通过此文档向客户展示他们对项目的理解，并且当规格说明书得到用户确认后，它将成为开发此软件在进行项目设计、实现、测试和实施过程中唯一的需求规范。

1.2 文档范围

文档包括，对产品的介绍，产品面向的用户群体，产品遵循的规范，产品中的角色，产品范围，业务流程，各功能模块的划分，以及各模块包含的子功能，产品的功能性需求和非功能性需求。

1.3 读者对象

需求的合法读者：支持本项目的徐美娇老师，开发成员，项目负责人，系统设计人员，数据库设计人员，编码人员，软件测试人员，客户。

1.4 参考文档

1.5 术语与缩写解释

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
| 模块 | 功能的集合。 |
| 功能 | 处理任务的最小单元。 |
| SRS | Software Requirement Specification |
| … |  |

2 产品介绍

在社会竞争日益激烈的今天，我们面临越来越多的工作和生活压力，随之而来的就是自身的休息时间逐渐减少，从而有很多的大众没有时间或者宁愿节省时间去工作、加班，受到生活环境和时间的限制，有很多人没有时间去购物或者宁愿珍惜自己的休息时间去好好休整。为了改变这种状况，越来越多的人选择在网上购物，节省大量的时间。

随着互联网的发展，人们对互联网购物的要求也在不断提高，这就要求商城网站不单只做单纯的买卖平台，还要提供更多的，更方便的功能来满足人们的需求。

本应用系统应达到的目标概述如下：

为用户们提供一个拥有大量选择的平台，使那些需要节省大量时间的用户以最少的时间选择自己最需要的物品，并物有所值。

3 产品面向的用户群体

本产品最终面向的用户群是以全龄，以网络购物为目的的用户。主要给那些不愿出门购物或没有时间购物为主的人提供一个节省购物时间的平台，在这里用户可以找到各类物品，易于比较并节省大量时间。

本产品不适用于年龄太小、不以网络购物为主要目的的其它网民。

4产品应当遵循的标准或规范

本软件的开发完全按照企业标准开发。数据命名的规则遵循《语言编程规约》中相关的规定，软件开发遵循国际通用CMMI标准。

5 产品范围

本产品最终面向的用户群是以网络购物为主要目的的用户，通过本网络使用户通过搜索、选择产品、购入、评价等方式实现网络购物的目的。

6 产品中的角色

|  |  |
| --- | --- |
| 角色名称 | 职责描述 |
| 网站管理员 | 本网站维护人员，负责本网站的日常维护工作。 |
| 普通用户 | 本网站的注册人员。 |

7 产品的功能性需求

7.1 业务整体规划

7.1.1 整体业务用例图

**系统**

自我管理局

卖家

管理员

用户

搜查商品

购买商品

评价商品

投诉管理

划账管理

订单管理

商品管理

<<uses>>

收藏商品

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

<<uses>>

7.1.2 整体功能划分

在本系统中，主要包括用户、卖家和管理员三个角色。本系统初期重点突出用户的管理的权限，具体用户的权限如下：

1. 用户自我管理。主要包括新用户的注册、登录和退出登录以及用户基本信息的管理三个部分，实现用户的相关自我的一些基本操作。
2. 查看商品。实现用户对商品的搜索查看等功能。
3. 收藏夹。实现用户对商品的收藏。
4. 购买商品。实现用户对商品的购买。
5. 评价商品。实现用户收货后对商品的评价。

管理员权限如下：

1. 商品管理。体现管理员对各卖家的商品进行管理，如发现不合格商品，马上下架。
2. 支付宝。体现用户和卖家之间的桥梁，用户收到货物后管理员将用户打到支付宝的钱转到卖家卡上。
3. 举报管理。实现用户对卖家的投诉，核实实情，解决用户与卖家的矛盾，如果卖家货物不合格，则将卖家账号整顿，合格后继续上架。

7.2 用户自我管理

**系统**

登陆界面功能

基本信息管理

注册

基本信息查看

基本信息修改

用户

用户登陆

用户退出登陆

用户重新登陆

用户找回密码

<<uses>>

<<uses>>

<<uses>>

<<include>>

<<include>>

<<include>>

<<include>>

<<include>>

<<include>>

**用例描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户注册 | | |
| 用例编号 | SRS-01 | | |
| 用例简介 | 注册一个新用户 | | |
| 优先级 | High | | |
| 前置条件 | 无 | | |
| 后置条件 | 显示注册成功提示，并跳到登录页。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 输入个人信息。 |
| 2 | 系统 | 检测个人信息是否合法。 |
| 3 | 系统 | 将个人信息录入数据库。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 注册用户名要合法，并检测用户名是否为空或已存在。 | | |
| 输入及约束 | 用户名(name) ：字符串，长度0-10，直接输入，必填。  密码(password)：字符串，长度0-30，直接输入，必填。  确认密码(passwordCheck)：字符串，长度0-30，直接输入，必填。  省份(province)：整型，必填。  城市(city)：整型，必填。  邮箱(email)：字符串，长度0-30，直接输入。  手机号(telephone)：字符串，长度11，直接输入。  验证码(validateCode)：字符串，长度0-5，用户输入。必填。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户登录 | | |
| 用例编号 | SRS-02 | | |
| 用例简介 | 用户登录 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经注册 | | |
| 后置条件 | 转到用户管理主页。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 输入登录信息，勾选是否记住用户。 |
| 2 | 系统 | 检测是否为空。 |
| 3 | 系统 | 若勾选，则写入Cookie，访问数据库，检测登陆信息是否正确。若正确，改写Session，进入个人管理首页；否则，请用户重新登录，到用例SRS-05。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 检测输入的用户名是否合法，并检测用户名是否为空。 | | |
| 输入及约束 | 用户名(name)：字符串，长度0-10，直接输入，必填。  密码(password)：字符串，长度0-30，直接输入，必填。  是否记住用户(isRemember)：布尔型，可选输入。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户退出登录 | | |
| 用例编号 | SRS-03 | | |
| 用例简介 | 用户退出登录 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录 | | |
| 后置条件 | 转到用户登录页。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 单击退出链接。 |
| 2 | 系统 | 置Session为Null。转到用户登录页面。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 无。 | | |
| 输入及约束 |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户找回密码 | | |
| 用例编号 | SRS-04 | | |
| 用例简介 | 用户找回密码 | | |
| 优先级 | High | | |
| 前置条件 |  | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 单击找回密码链接，输入想要找回账号的登录名。 |
| 2 | 系统 | 访问数据库，检测邮箱是否存在。 |
| 3 | 用户 | 若存在，发送密码到邮箱；否则，提示用户错误信息。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 无。 | | |
| 输入及约束 | 用户名(name)：字符串，长度0-10，直接输入，必填。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户重新登录 | | |
| 用例编号 | SRS-05 | | |
| 用例简介 | 用户重新登录 | | |
| 优先级 | High | | |
| 前置条件 | 用户登录失败 | | |
| 后置条件 | 转到用户管理主页。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 输入登录信息，勾选是否记住用户。 |
| 2 | 系统 | 检测是否为空。 |
| 3 | 系统 | 若勾选，则写入Cookie，访问数据库，检测登陆信息是否正确。若正确，改写Session，进入个人管理首页；否则，重复本用例。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 无。 | | |
| 输入及约束 | 用户名(name)：字符串，长度0-10，直接输入，必填。  密码(password)：字符串，长度0-30，直接输入，必填。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户基本信息查看 | | |
| 用例编号 | SRS-06 | | |
| 用例简介 | 用户分类别的查看自己或者他人的详细信息。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 进入详细信息显示页。 |
| 2 | 系统 | 按标签显示用户各类信息（基本信息、联系方式、个人爱好），初始显示基本信息。 |
| 3 | 用户 | 单击其它未展开选项卡。 |
| 4 | 系统 | 展开相应选项卡，显示相应信息。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 无。 | | |
| 输入及约束 |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户基本信息修改 | | |
| 用例编号 | SRS-07 | | |
| 用例简介 | 用户修改自己的详细信息。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录 | | |
| 后置条件 | 进入用户详细信息显示页 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 打开自己的详细信息显示页，并进入修改页面。 |
| 2 | 系统 | 分类显示用户的基本信息，要求以文本框的形式显示。 |
| 3 | 用户 | 修改自己的基本信息(姓名、昵称、姓名可见度、性别、年龄、生日、所在省份、所在城市、头像、个性签名)。  修改自己的联系方式(联系电话、QQ、MSN)。  修改自己的个人爱好(爱好书籍、爱好音乐、爱好电影、爱好运动、爱好游戏)。 |
| 4 | 用户 | 单击确定按钮。 |
| 5 | 系统 | 将用户修改的信息提交到数据库，并返回修改结果。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 无。 | | |
| 输入及约束 | 性别(sex)：布尔型，选择输入。  年龄(age)：整型，选择输入。  账户余额(money):money型，默认为0。  生日(birthday)：日期类型，选择输入。  所在省份(provinceID)：外键(Province表)，长度0-255，列表框选择输入。  所在城市(cityID)：外键(City表)，长度0-255，列表框选择输入。  头像(photo)：字符串，长度0-500，记录文件保存在网站文件夹的位置，用户上传。  个性签名(dsp)：字符串，长度0-200，直接输入。  邮箱(email)：字符串，长度0-30，直接输入。  手机号(telephone)：字符串，长度11，直接输入。  QQ号码(qq)：字符串，长度0-15，只要求数字，直接输入。  MSN(msn)：字符串，长度0-50，邮箱格式，直接输入。  爱好书籍(loveBook)：字符串，长度0-200，直接输入。  爱好音乐(loveMusic)：字符串，长度0-200，直接输入。  爱好电影(loveMovie)：字符串，长度0-200，直接输入。  爱好运动(loveSport)：字符串，长度0-200，直接输入。  爱好游戏(loveGame)：字符串，长度0-200，直接输入。 | | |

7.3搜查商品

**系统**

<<uses>>

<<uses>>

查看商品

搜索商品

用户

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 搜索商品 | | |
| 用例编号 | SRS-08 | | |
| 用例简介 | 用户输入关键字来搜索相应的所有商品。 | | |
| 优先级 | High | | |
| 前置条件 |  | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 进入搜索页面，输入搜索商品的关键字，可选择一项搜索关键字的条件（如商品类别、品牌、价格等）来搜索商品。 |
| 2 | 系统 | 根据用户搜索的关键字查询数据库，若找到符合要求的商品则显示出来，否则显示没有相应商品信息存在。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 输入搜索商品关键字不为空。 | | |
| 输入及约束 | 关键字(searchKey)：字符串，长度0-20，直接输入，必填。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 查看商品 | | |
| 用例编号 | SRS-09 | | |
| 用例简介 | 用户查看相应商品的详细信息。 | | |
| 优先级 | High | | |
| 前置条件 |  | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 点击已搜索到的想要查看其信息的商品。 |
| 2 | 系统 | 根据用户点击的商品查询数据库，若找到符合要求的商品则将该商品的详细信息显示出来，否则显示没有相应商品信息存在。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 |  | | |

7.4收藏商品

**系统**

<<uses>>

收藏商品

用户

<<uses>>

删除收藏记录

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 收藏商品 | | |
| 用例编号 | SRS-10 | | |
| 用例简介 | 用户收藏看中的相应商品。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 点击收藏商品。 |
| 2 | 系统 | 将相应收藏的商品收藏到用户的收藏商品中。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 删除收藏记录 | | |
| 用例编号 | SRS-11 | | |
| 用例简介 | 用户删除收藏商品的记录。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 选中收藏的商品，点击删除收藏。 |
| 2 | 系统 | 将相应收藏商品的记录删除。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 |  | | |

7.5购买商品

**系统**

<<uses>>

下订单

用户

支付费用

<<uses>>

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 下订单 | | |
| 用例编号 | SRS-12 | | |
| 用例简介 | 用户将想要购买的商品订单下好，生成新的订单。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 将订单的具体信息填写完整。 |
| 2 | 系统 | 将这份订单的信息存入数据库，生成一份新订单，没有付款则订单的付款项为：未付款，订单只下发给卖家但卖家不发货。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 商品信息不为空、商品库存足够订单要求、用户账户余额足够、一个订单号只给一次订单使用，不管订单完成与否订单号只能使用一次。 | | |
| 输入及约束 | 买家用户名：从买家信息表中调出该信息，存入订单表中。  卖家用户名：从卖家信息表中调出该信息，存入订单表中。  商品信息：从商品表中调出该商品的各类信息。  购买数量：整型，直接输入，必填。  送货地址：字符串，长度0-100，必填。  买家手机号：字符串，长度11，必填。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 支付费用 | | |
| 用例编号 | SRS-13 | | |
| 用例简介 | 用户支付费用到商城（作为中间人），并在订单中显示已付款，卖家可发货。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已经登录，并下订单。 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 选择支付此订单。 |
| 2 | 系统 | 将订单中显示为已付款，提醒卖家发货。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 买家支付宝中余额足够。 | | |
| 输入及约束 |  | | |
| 备注 | 将订单中支付状态改为已支付。 | | |

7.6评价商品

**系统**

<<uses>>

<<uses>>

发表评论

用户

回复评论

删除评论

<<uses>>

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 发表、回复评论 | | |
| 用例编号 | SRS-14 | | |
| 用例简介 | 发表一条新的留言，或者对已有的留言进行回复。 | | |
| 优先级 | High | | |
| 前置条件 | 用户是已注册成员，且已收到商品。 | | |
| 后置条件 | 跳转到该商品或商家的评论区。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 进入留言面板。 |
| 2 | 系统 | 显示相关有效评论。 |
| 3 | 用户 | 单击回复按钮。 |
| 4 | 系统 | 在新建留言文本框里面显示“回复XX：”。 |
| 5 | 用户 | 在新建留言文本框里输入留言内容。 |
| 6 | 用户 | 单击发表按钮。 |
| 7 | 系统 | 进入数据库，进行相关操作。若成功，刷新留言列表；若失败，通过提示框提示错误信息。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 无。 | | |
| 输入及约束 | 评论用户名：依赖于买家用户名 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 删除评论 | | |
| 用例编号 | SRS-15 | | |
| 用例简介 | 对已发表的评论进行删除。 | | |
| 优先级 | High | | |
| 前置条件 | 用户是已注册成员，且已发表评论。 | | |
| 后置条件 | 跳转到该商品或商家的评论区。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 进入留言面板。 |
| 2 | 系统 | 显示相关有效评论。 |
| 3 | 用户 | 单击删除按钮。 |
| 4 | 系统 | 删除评论。 |
| 7 | 系统 | 进入数据库，进行相关操作。若成功，刷新留言列表；若失败，通过提示框提示错误信息。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 | 卖家不可以删除买家评论。 | | |
| 输入及约束 | 评论用户名：依赖于买家用户名 | | |

7.7订单管理

**系统**

<<include>>

<<uses>>

修改订单状态

完成

用户

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 用户修改订单状态 | | |
| 用例编号 | SRS-16 | | |
| 用例简介 | 用户收到货物后对订单状态改为完成，管理员可进行转账。 | | |
| 优先级 | High | | |
| 前置条件 | 有用户已对卖家下订单，并且订单状态为发货状态。 | | |
| 后置条件 | 显示订单状态修改为：XX，后跳转到该订单信息页面。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 收到货后用户打开订单。 |
| 2 | 系统 | 从数据库调出订单信息。 |
| 3 | 用户 | 用户将订单状态改为完成。 |
| 4 | 系统 | 数据库记录修改后的订单状态（完成）。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 | 商品信息（从订单所对应的外键商品表中调取信息）。  卖家用户名（从订单表中调出信息）。  用户信息（从数据库调出发货地址、手机号等）。  修改订单状态：完成。 | | |

7.8订单管理

**系统**

撤销

<<uses>>

<<uses>>

修改订单状态

卖家

付款

修改订单信息

<<include>>

<<include>>

<<include>>

延后

发货

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 卖家修改订单状态 | | |
| 用例编号 | SRS-17 | | |
| 用例简介 | 卖家对订单状态进行修改。 | | |
| 优先级 | High | | |
| 前置条件 | 有用户已对卖家下订单，卖家可修改订单状态。 | | |
| 后置条件 | 显示订单状态修改为：XX，后跳转到该订单信息页面。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 卖家 | 查询用户对自己所下的订单。 |
| 2 | 系统 | 从数据库调出订单信息。 |
| 3 | 卖家 | 根据货物库存对订单状态进行修改。 |
| 4 | 系统 | 数据库记录修改后的订单状态（撤销、付款、发货、延后）。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 | 商品信息（从订单所对应的外键商品表中调取信息）。  卖家用户名（从订单表中调出信息）。  用户信息（从数据库调出发货地址、手机号等）。  修改订单状态：选择输入，分为4种状态（撤销、付款、发货、延后）。 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 卖家修改订单信息 | | |
| 用例编号 | SRS-18 | | |
| 用例简介 | 卖家对订单信息进行修改。 | | |
| 优先级 | High | | |
| 前置条件 | 该订单商品为此用户所卖才能修改。 | | |
| 后置条件 | 跳转到该订单信息页面。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 卖家 | 查询用户对自己所下的订单。 |
| 2 | 系统 | 从数据库调出订单信息。 |
| 3 | 卖家 | 将想要修改的订单信息进行修改。 |
| 4 | 系统 | 调取数据库记录，判断修改后订单是否合理（货物库存是否足够，且商品类型不能修改，数量只能减少不能增加，若要增加只能卖家与买家再次进行一次订单交易），修改后保存。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 | 商品信息（从订单所对应的外键商品表中调取信息）。  卖家用户名（从订单表中调出信息）。  用户信息（从数据库调出发货地址、手机号等）。 | | |

7.9商品管理

**系统**

<<uses>>

<<uses>>

商品上架

修改商品信息

卖家

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 商品上架 | | |
| 用例编号 | SRS-19 | | |
| 用例简介 | 卖家发布要出售的商品的信息。 | | |
| 优先级 | High | | |
| 前置条件 | 用户是已注册成员。 | | |
| 后置条件 | 跳转到该商品的信息页面。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 卖家 | 发布商品各项信息。 |
| 2 | 系统 | 进入数据库，添加商品信息。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 | 买家用户名：  商品信息： | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 修改商品信息 | | |
| 用例编号 | SRS-20 | | |
| 用例简介 | 修改卖家发布的商品信息。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已登陆，且修改的是自己发布的商品。 | | |
| 后置条件 | 跳转到该商品的信息页面。 | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 卖家 | 修改商品信息。 |
| 2 | 系统 | 进入数据库，修改商品信息。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 | 买家用户名:  商品信息： | | |

7.10划账管理

管理员

**系统**

<<uses>>

<<uses>>

款项返还买家

款项划给卖家

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 管理员将款项返还买家 | | |
| 用例编号 | SRS-21 | | |
| 用例简介 | 用户已付款给中介管理员，但因退货或货物丢失所导致的订单交易失败，用户可申请退款请求，管理员将用户转账资金返还买家。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已登陆，并且订单付款项为已付款、订单状态为撤销则为退货或货物未到，款项返还买家。 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 订单付款项为已付款且订单状态被撤销时用户可点击请求退款。 |
| 2 | 管理员 | 核实交易情况，若属实则点击同意请求。 |
| 3 | 系统 | 系统将用户在此订单所转账冻结的资金返还给用户的账户上，此项交易完成。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 |  | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 管理员将款项划给卖家 | | |
| 用例编号 | SRS-22 | | |
| 用例简介 | 交易成功，管理员将买家已付款的款项划给卖家。 | | |
| 优先级 | High | | |
| 前置条件 | 订单付款项为已付款、订单状态为完成则视为交易成功，管理员将款项划给卖家。 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 系统 | 系统检测到某一订单付款项为已付款，订单状态为完成，则视为交易成功，提醒管理员转账。 |
| 2 | 管理员 | 核实交易情况，若属实则点击同意转账。 |
| 3 | 系统 | 系统将用户在此订单所转账冻结的资金划给该订单的卖家，此项交易完成。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 |  | | |

7.11投诉管理

**系统**

<<uses>>

受理投诉

管理员

|  |  |  |  |
| --- | --- | --- | --- |
| 用例名称 | 受理投诉 | | |
| 用例编号 | SRS-23 | | |
| 用例简介 | 受理用户的投诉。 | | |
| 优先级 | High | | |
| 前置条件 | 用户已登陆，且投诉的交易为下过的订单且订单状态为已完成。 | | |
| 后置条件 |  | | |
| 操作流程 | *步骤* | *触发者* | *描述* |
| 1 | 用户 | 收到货物不满意，针对此订单发起投诉消息。 |
| 2 | 管理员 | 受理、查看投诉。 |
| 替代流程 |  | | |
| 例外流程 |  | | |
| 约束条件 |  | | |
| 输入及约束 |  | | |

8 软硬件环境需求

|  |  |
| --- | --- |
| **需求名称** | **详细要求** |
| 数据库 | Oracle |
| 操作系统 | Windows2000、XP |
| 浏览器 | Windows系统下的IE5.0及以上版本或FireFox |
| 处理器 | 推荐：Intel服务器CPU Xeon 5355 2.66G |
| 内存 | 推荐：1G以上 |

9 产品质量需求

|  |  |
| --- | --- |
| **主要质量属性** | **详细要求** |
| 正确性 | 在系统设计和开发过程中，要充分考虑网站当前和将来可能承受的工作量，使系统的处理能力和响应时间能够满足企业对信息处理的需求。 |
| 健壮性 |  |
| 可靠性 |  |
| 性能，效率 | 网站的相应速度应该满足及时与用户进行交互的基本要求。占用客户端的资源也不应太多。 |
| 易用性 | 本网站面对广大用户，包括计算机水平较低的用户。因此，应该力求操作的简洁与已于接受，而且界面也应力求较好的交互性和友好性。 |
| 清晰性 |  |
| 安全性 | 本网站涉及到较多的用户个人隐私信息，因此，较高的安全性应当是建站的一个基础。所有的工作的展开都应保证是在安全性的基础上来展开的。 |
| 可扩展性 | 在网站运行的过程中，网站应该随着用户的需求的不断改变而改变，要求网站具有良好的可扩展性以适应广大用户的需求。 |
| 兼容性 |  |
| 可移植性 |  |

**二、数据库设计**

0. 文档介绍

0.1 文档目的

本文档是SNS交友社区项目的第二阶段，本阶段主要解决系统用到的数据库的设计问题，数据库设计的根据是前一阶段形成的《系统需求规格说明书》，根据《系统规格说明书》中对系统的描述，确定系统涉及到的数据库表。

本系统设计的数据库表是进行系统开发的基础，在数据库设计的基础上可以进行系统的详细设计。在以后的软件测试以及软件维护阶段也可参考此说明书，以便于了解在数据库设计阶段中设计的各个表的信息，或在下一阶段修改时找出在本阶段设计的不足或错误。

0.2 文档范围

文档包括：数据库环境说明，数据库命名规则，逻辑设计，物理设计，安全设计，数据库优化，数据库管理和维护说明等信息。

0.3 读者对象

数据库设计报告书的合法读者系统设计人员，数据库设计人员，编码人员，软件测试人员，客户。

0.5 术语与缩写解释

|  |  |
| --- | --- |
| **缩写、术语** | **解 释** |
|  |  |
|  |  |
|  |  |
|  |  |
| … |  |

1. 数据库环境说明

本系统采用Oracle数据库系统

2. 数据库的命名规则

（1）、表名以tb\_开头，当表名中含有两个以上的英文单词时，单词间不加空格，但是从第二个单词开始的单词首字母要大写。

（2）、表中的字段名首字母小写，字段名中含有两个以上的英文单词时，单词间不加空格，但是从第二个单词开始的单词首字母要大写。

3. 逻辑设计

详细设计略。

4. 物理设计

4.0 表汇总

|  |  |
| --- | --- |
| 表名 | 功能说明 |
| tb\_user | 用户表 |
| tb\_product | 产品表（商品表） |
| tb\_productType | 产品类型 |
| tb\_order | 订单表 |
| tb\_userAccount | 用户资金流水表（只允许新增记录，不允许修改、删除） |
| tb\_review | 评论表 |
| tb\_city | 城市表 |
| tb\_province | 省份表 |
| tb\_productSize | 产品型号表 |
| tb\_productColor | 产品颜色 |

4.1 tb\_user表(用户信息表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_user | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| userID | 表id | number | 否 | 主键，自动增量(1) |
| name | 用户名 | varchar2 | 否 | 唯一约束 |
| password | 登录密码 | varchar2(30) | 否 |  |
| email | 邮箱 | varchar2(30) | 是 |  |
| provinceID | 所在省id | number | 是 |  |
| cityID | 所在城市id | number | 是 | 外键，tb\_city表cityID项 |
| telephone | 手机号 | char(11) | 是 |  |
| sex | 性别 | char(3) | 是 | 默认约束“男” |
| age | 年龄 | number | 是 |  |
| birthday | 生日 | date | 是 |  |
| money | 账户余额 | number | 是 | 默认约束“0” |
| photo | 头像路径 | varchar2(500) | 是 |  |
| dsp | 个性签名 | varchar2(200) | 是 |  |
| qq | QQ号码 | varchar2(15) | 是 |  |
| msn | msn号码 | varchar2(50) | 是 |  |
| loveBook | 爱好书籍 | varchar2(200) | 是 |  |
| loveMusic | 爱好音乐 | varchar2(200) | 是 |  |
| loveMovie | 爱好电影 | varchar2(200) | 是 |  |
| loveSport | 爱好运动 | varchar2(200) | 是 |  |
| loveGame | 爱好游戏 | varchar2(200) | 是 |  |
| 补充说明 |  | | | |

4.2 tb\_product表(商品信息表)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_product | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| productID | 商品id | number | 否 | 主键，自动增加(1) |
| userID | 卖家id | number | 否 | 外键，tb\_user表userID项 |
| productName | 商品名 | varchar2(50) | 否 |  |
| typeID | 商品类型id | number | 否 | 外键，tb\_productType表productTypeID项 |
| price | 商品价格 | number | 否 |  |
| photo | 商品图片路径 | varchar2(500) | 是 |  |
| information | 商品简介 | varchar2(500) | 是 |  |
| 补充说明 |  | | | |

**4.3** tb\_productType**表（商品类型表）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_productType | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| productTypeID | 商品类型id | number | 否 | 主键，自动增加(1) |
| productTypeName | 商品名称 | varchar2(100) | 否 |  |
| 补充说明 |  | | | |

**4.4 tb\_order表（订单表）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_order | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| orderID | 订单id | number | 否 | 主键，自动增加(1) |
| toID | 买家id | number | 否 | 外键，tb\_user表userID项 |
| colorID | 颜色id | number | 否 | 外键，tb\_ productColor表productColorID项 |
| productID | 商品id | number | 否 | 外键，tb\_ product表productID项 |
| address | 送货地址 | varchar2(100) | 否 |  |
| telephone | 买家手机 | char(11) | 否 |  |
| orderState | 订单状态 | number | 否 |  |
| count | 商品数量 | number | 否 |  |
| price | 总金额 | number | 否 |  |
| orderDate | 订单日期 | date | 否 |  |
| consignmentDate | 发货日期 | date | 否 |  |
| 补充说明 |  | | | |

**4.5** tb\_userAccount**表（用户资金流水表）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_userAccount | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| userAccountID | 资金流水表id | number | 否 | 主键，自动递增(1) |
| userID | 用户id | number | 否 | 外键，tb\_user表userID项 |
| orderID | 订单id | number | 否 | 外键，tb\_order表orderID项 |
| type | 类型 | number | 否 |  |
| time | 时间 | date | 否 |  |
| 补充说明 |  | | | |

**4.6** tb\_review**表（评论表）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_review | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| reviewID | 评论id | number | 否 | 主键，自动递增(1) |
| review | 评论内容 | varchar2(250) | 否 |  |
| isReply | 是否回复 | boolean | 否 | 默认值(false) |
| isDel | 是否删除 | boolean | 否 | 默认为(false) |
| senderID | 发表人id | number | 否 | 外键，tb\_user表userID项 |
| productID | 留言的商品id | number | 否 | 外键，tb\_product表productID项 |
| addTime | 发表时间 | date | 否 |  |
| replyID | 回复人id | number | 是 |  |
| 补充说明 | 是否回复：回复为true，未回复为false，默认为false。  是否删除：删除为true，未删除为false，默认为false。 | | | |

**4.7** tb\_province**表（省份数据字典）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_province | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| provinceID | 省份id | number | 否 | 主键，自动递增(1) |
| provinceName | 省份名称 | varchar2(30) | 否 |  |
| 补充说明 |  | | | |

**4.8 t**b\_city**表（城市数据字典）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_city | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| cityID | 城市id | number | 否 | 主键，自动递增(1) |
| provinceID | 省份id | number | 否 | 外键，tb\_province表provinceID项 |
| cityName | 城市名称 | varchar2(30) | 否 |  |
| 补充说明 |  | | | |

**4.9 tb\_productSize表（商品型号表）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_productSize | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| productSizeID | 商品型号id | number | 否 | 主键，自动递增(1) |
| productSize | 商品型号 | varchar2(20) | 否 |  |
| productID | 商品id | number | 否 | 外键，tb\_product表productID项 |
| 补充说明 |  | | | |

**4.10** tb\_productColor**表（商品颜色表）**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 表名 | tb\_productColor | | | |
| 列名 | 描述 | 数据类型（精度范围） | 允许空 | 约束条件 |
| productColorID | 商品颜色id | number | 否 | 主键，自动增量(1) |
| productColor | 商品颜色 | varchar2(20) | 否 |  |
| stockpile | 商品库存 | number | 否 |  |
| productSizeID | 商品型号id | number | 否 | 外键，tb\_productSize表productSizeID项 |
| 补充说明 |  | | | |

5. 安全性设计

5.1 防止用户直接操作数据库的方法

用户只能用帐号登陆到应用软件，通过相应的功能访问数据库，而没有其他途径操作数据库。

5.2 角色与权限

***提示：****确定每个角色对数据库表的操作权限，如创建、检索、更新、删除等。每个角色拥有刚好能够完成任务的权限，不多也不少。在应用时再为用户分配角色，则每个用户的权限等于他所兼角色的权限之和。*

|  |  |  |
| --- | --- | --- |
| 角色 | 可以访问的表与列 | 操作权限 |
| 普通用户 |  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| 管理员 |  |  |
|  |  |
|  |  |

6. 优化

***提示：****分析并优化数据库的“时－空”效率，尽可能地“提高处理速度”并且“降低数据占用空间”。*

*（1）分析“时－空”效率的瓶颈，找出优化对象（目标），并确定优先级。*

*（2）当优化对象（目标）之间存在对抗时，给出折衷方案。*

*（3）给出优化的具体措施，例如优化数据库环境参数，对表格进行反规范化处理等。*

|  |  |  |
| --- | --- | --- |
| 优先级 | 优化对象（目标） | 措施 |
|  |  |  |
|  |  |  |
|  |  |  |

7. 数据库管理与维护说明

***提示：****在设计数据库的时候，及时给出管理与维护本数据库的方法，有助于将来撰写出正确完备的用户手册。*

**三、附录**

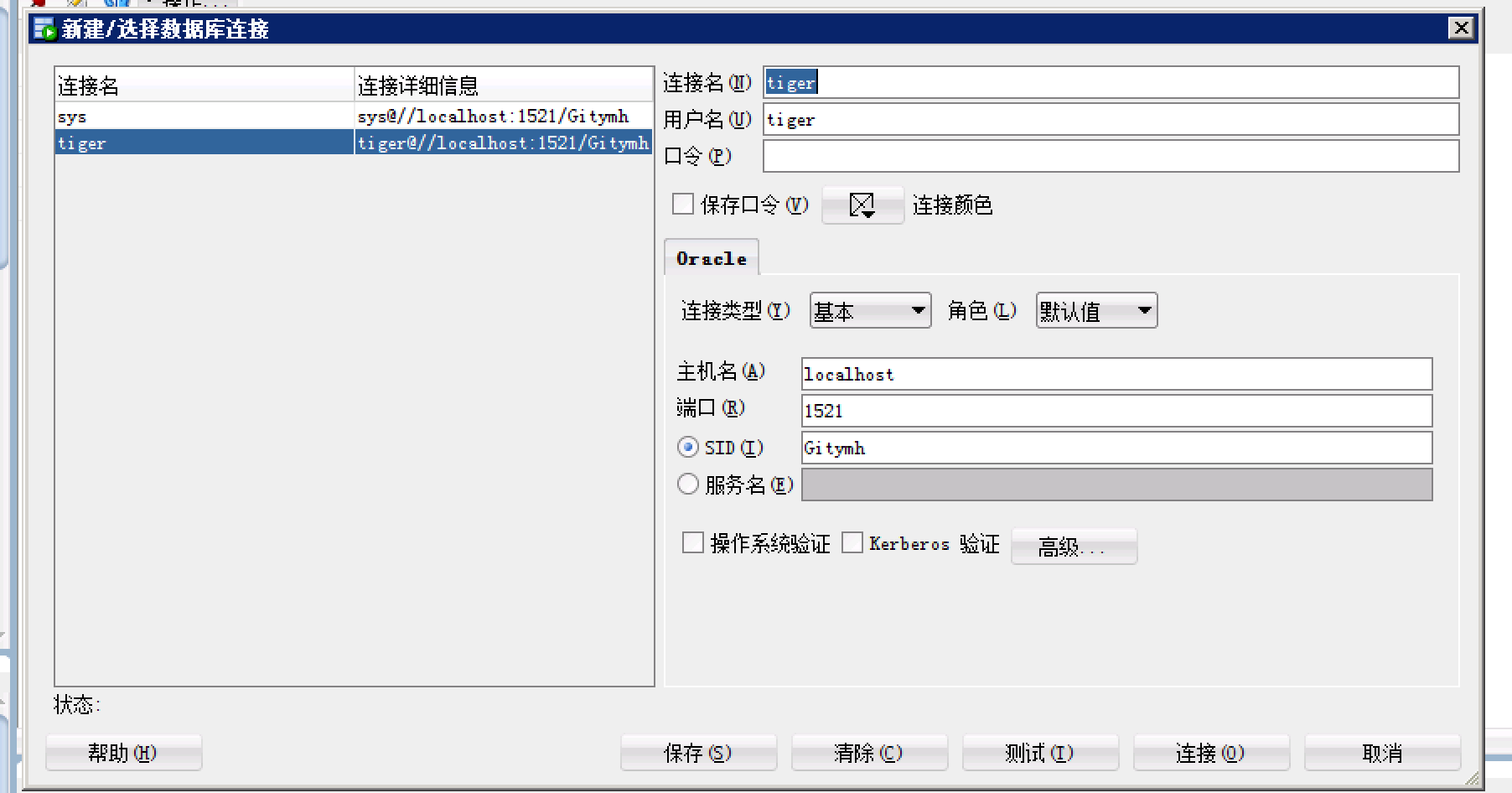
**注意:代码下附上图示**

**一、创建数据库**

/\*创建用户\*/

CREATE USER Tiger IDENTIFIED BY 666;

GRANT CONNECT,RESOURCE TO Tiger



**二、创建表**

/\*创建tb\_user表\*/

CREATE TABLE B\_USER

(

USERID NUMBER NOT NULL

, NAME VARCHAR2(20 BYTE) NOT NULL

, COLUMN3 VARCHAR2(30 BYTE) NOT NULL

, COLUMN4 VARCHAR2(30 BYTE)

, COLUMNSID NUMBER

, COLUMN6 NUMBER

, COLUMN7 CHAR(11 BYTE)

, COLUMN8 CHAR(3 BYTE) DEFAULT '男'

, COLUMN9 NUMBER

, COLUMN10 DATE

, COLUMN11 NUMBER DEFAULT 0

, COLUMN12 VARCHAR2(500 BYTE)

, COLUMN13 VARCHAR2(200 BYTE)

, COLUMN14 VARCHAR2(15 BYTE)

, COLUMN15 VARCHAR2(50 BYTE)

, COLUMN16 VARCHAR2(200 BYTE)

, LOVEMUSIC VARCHAR2(200 BYTE)

, COLUMN18 VARCHAR2(200 BYTE)

, COLUMN19 VARCHAR2(200 BYTE)

, COLUMN5 VARCHAR2(200 BYTE)

, CONSTRAINT TB\_USER\_PK PRIMARY KEY

(

USERID

)

USING INDEX

(

CREATE UNIQUE INDEX TB\_USER\_PK ON TB\_USER (USERID ASC)

LOGGING

TABLESPACE USERS

PCTFREE 10

INITRANS 2

STORAGE

(

BUFFER\_POOL DEFAULT

)

NOPARALLEL

)

ENABLE

)

LOGGING

TABLESPACE USERS

PCTFREE 10

INITRANS 1

STORAGE

(

BUFFER\_POOL DEFAULT

)

NOCOMPRESS

NOPARALLEL;

--创建sequence

create sequence identify\_tb\_user

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

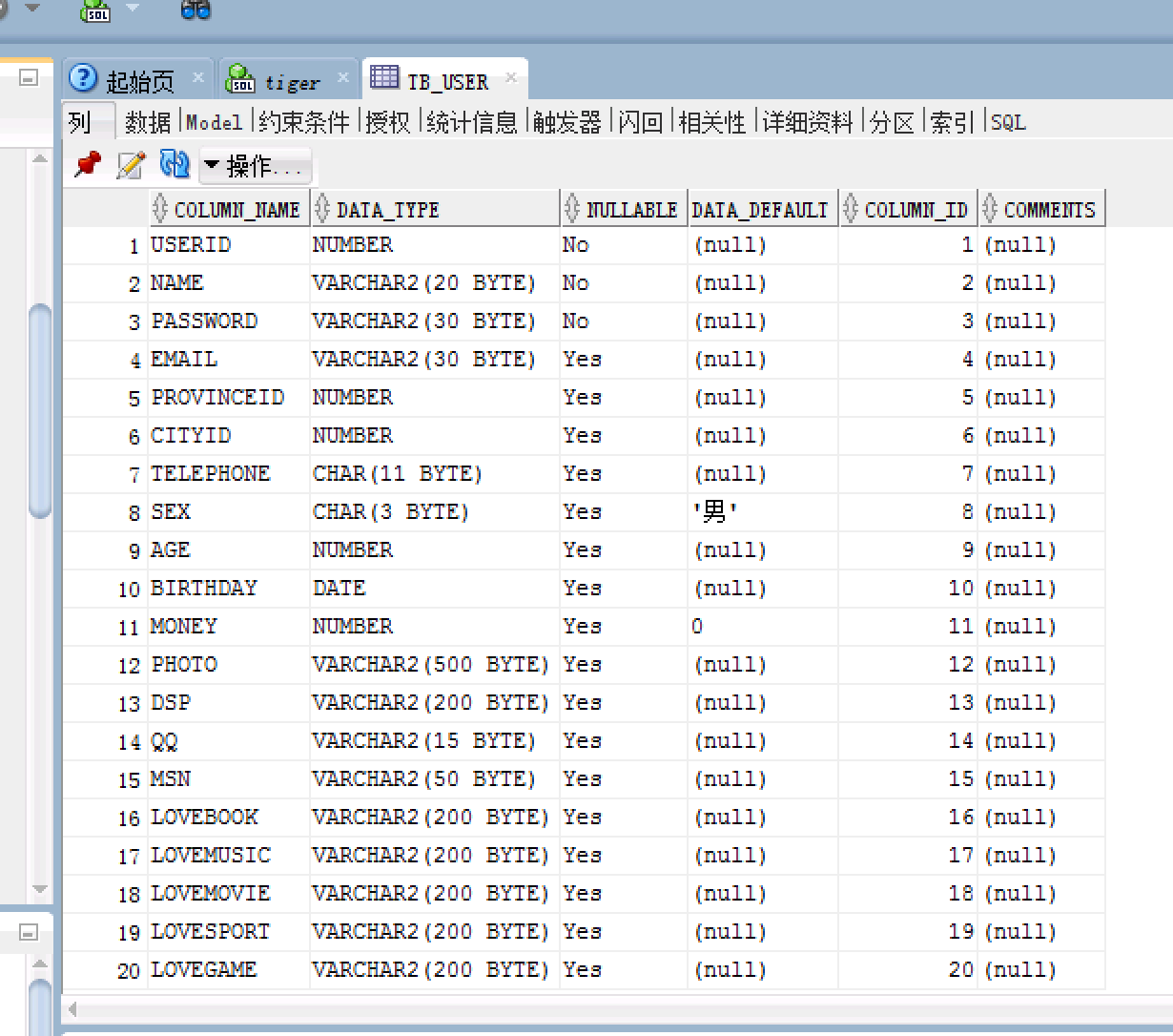
CREATE OR REPLACE TRIGGER selfplus\_tb\_user

BEFORE INSERT ON TB\_USER FOR EACH ROW

begin

select identify\_tb\_user.nextval into:new.USERID from dual;

end;



/\*创建tb\_product表\*/

CREATE TABLE TB\_PRODUCT

(

PRODUCTID NUMBER NOT NULL

, USERID NUMBER NOT NULL

, PRODUCTNAME VARCHAR2(50) NOT NULL

, TYPEID NUMBER NOT NULL

, PRICE NUMBER NOT NULL

, PHOTO VARCHAR2(500)

, INFORMATION VARCHAR2(500)

, CONSTRAINT TB\_PRODUCT\_PK PRIMARY KEY

(

PRODUCTID

)

ENABLE

)

LOGGING

TABLESPACE USERS

PCTFREE 10

INITRANS 1

STORAGE

(

BUFFER\_POOL DEFAULT

)

NOCOMPRESS

NOPARALLEL;

--创建sequence

create sequence identify\_tb\_product

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

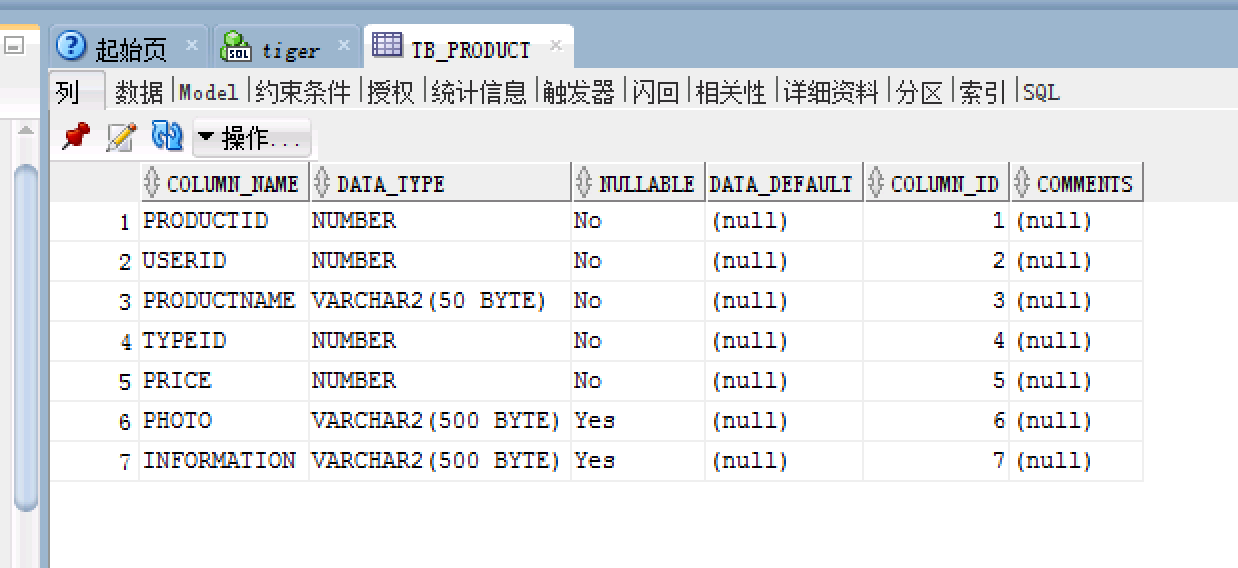
CREATE OR REPLACE TRIGGER selfplus\_tb\_product

BEFORE INSERT ON tb\_product FOR EACH ROW

begin

select identify\_tb\_product.nextval into:new.productID from dual;

end;



/\*创建tb\_productType表\*/

CREATE TABLE TB\_PRODUCTTYPE

(

PRODUCTTYPEID NUMBER NOT NULL

, COLUMN1 VARCHAR2(100) NOT NULL

, CONSTRAINT TB\_PRODUCTTYPE\_PK PRIMARY KEY

(

PRODUCTTYPEID

)

ENABLE

);

--创建sequence

create sequence identify\_tb\_productType

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

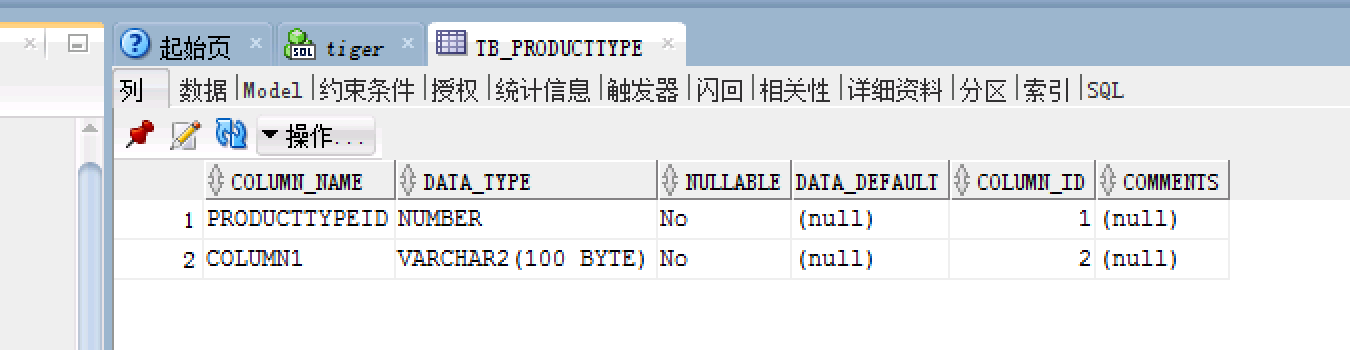
CREATE OR REPLACE TRIGGER selfplus\_tb\_productType

BEFORE INSERT ON tb\_productType FOR EACH ROW

begin

select identify\_tb\_productType.nextval into:new.productTypeID from dual;

end;



/\*创建tb\_order表\*/

(orderState,orderID)

CREATE TABLE TB\_ORDER

(

ORDERID NUMBER NOT NULL

, TOID NUMBER NOT NULL

, COLORID NUMBER NOT NULL

, PRODUCTID NUMBER NOT NULL

, COLUMN1 VARCHAR2(100) NOT NULL

, TELEPHONE CHAR(11) NOT NULL

, ORDERSTATE NUMBER NOT NULL

, COUNT NUMBER NOT NULL

, PRICE NUMBER NOT NULL

, ORDERDATE DATE NOT NULL

, CONSIGNMENTDATE DATE NOT NULL

, CONSTRAINT TB\_ORDER\_PK PRIMARY KEY

(

ORDERID

)

ENABLE

)

LOGGING

TABLESPACE USERS

PCTFREE 10

INITRANS 1

STORAGE

(

BUFFER\_POOL DEFAULT

)

NOCOMPRESS

NOPARALLEL;

--创建sequence

create sequence identify\_tb\_order

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

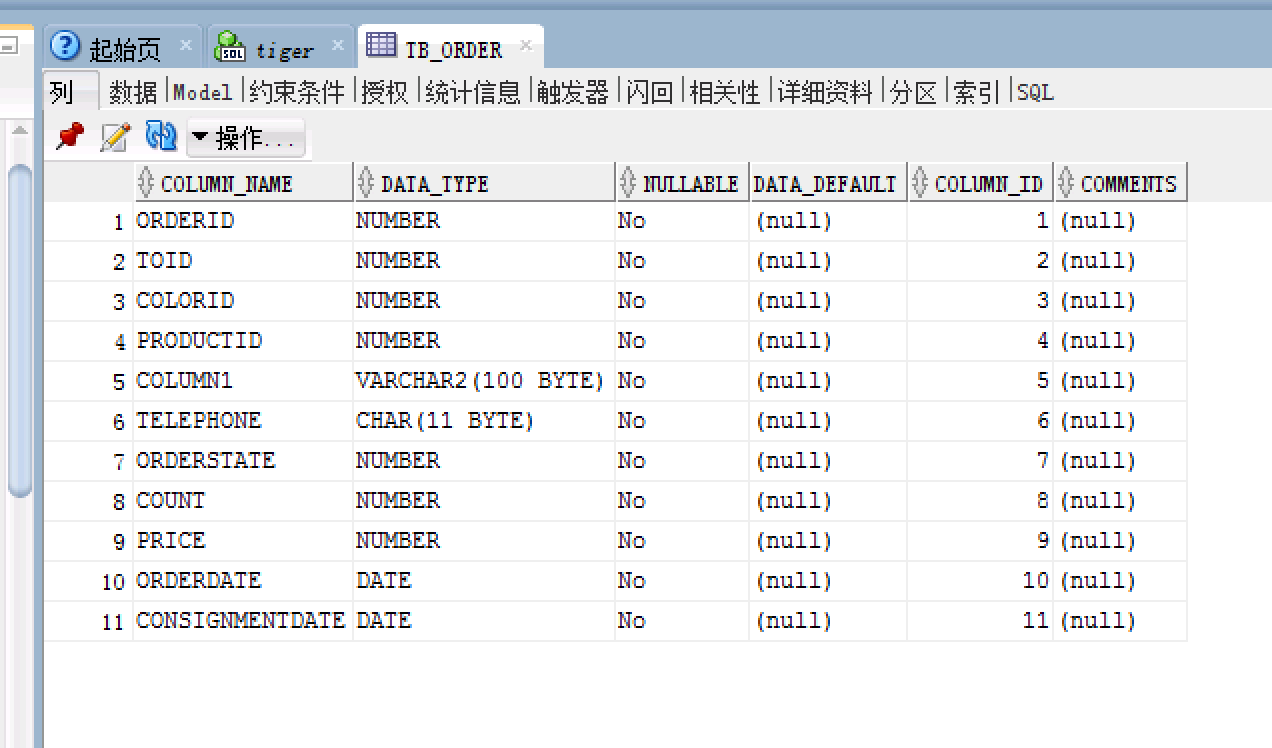
CREATE OR REPLACE TRIGGER selfplus\_tb\_order

BEFORE INSERT ON tb\_order FOR EACH ROW

begin

select identify\_tb\_order.nextval into:new.orderID from dual;

end;



/\*创建tb\_userAccount表\*/

CREATE TABLE TB\_USERACCOUNT

(

USERACCOUNTID NUMBER NOT NULL

, USERID NUMBER NOT NULL

, ORDERID NUMBER NOT NULL

, TYPE NUMBER NOT NULL

, TIME DATE NOT NULL

, CONSTRAINT TB\_USERACCOUNT\_PK PRIMARY KEY

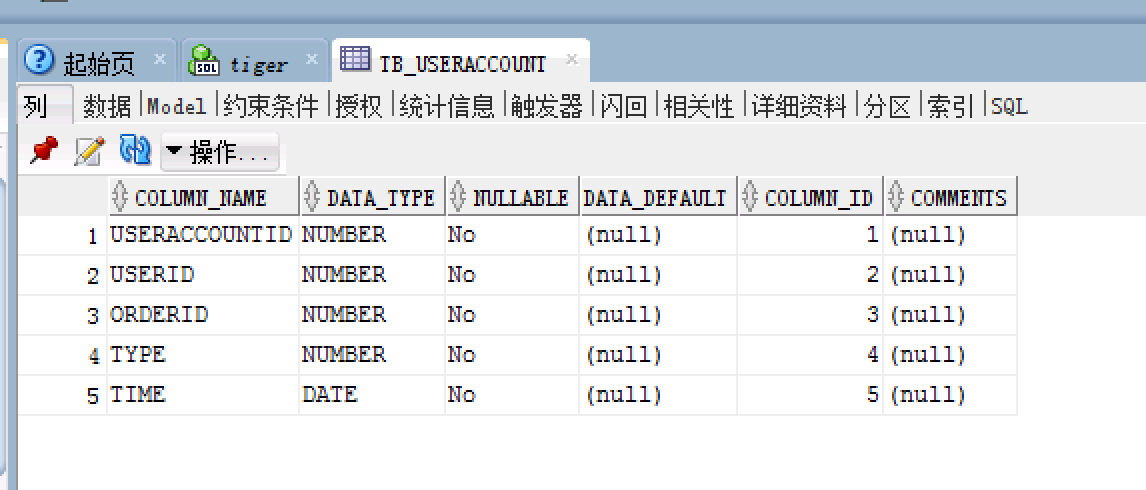
(

USERACCOUNTID

)

ENABLE

);



/\*创建tb\_review表\*/

CREATE TABLE tb\_review(

reviewID number NOT NULL,

review varchar2(250) NOT NULL,

isReply CHAR(1) CONSTRAINT C\_BOOLEAN1 CHECK (ISREPLY='Y' OR ISREPLY='N') NOT NULL,

isDel CHAR(1) CONSTRAINT C\_BOOLEAN2 CHECK (isDel='Y' OR isDel='N') NOT NULL,

senderID number NOT NULL,

productID number NOT NULL,

addTime date NOT NULL,

replyID number

);

--创建sequence

create sequence identify\_tb\_review

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

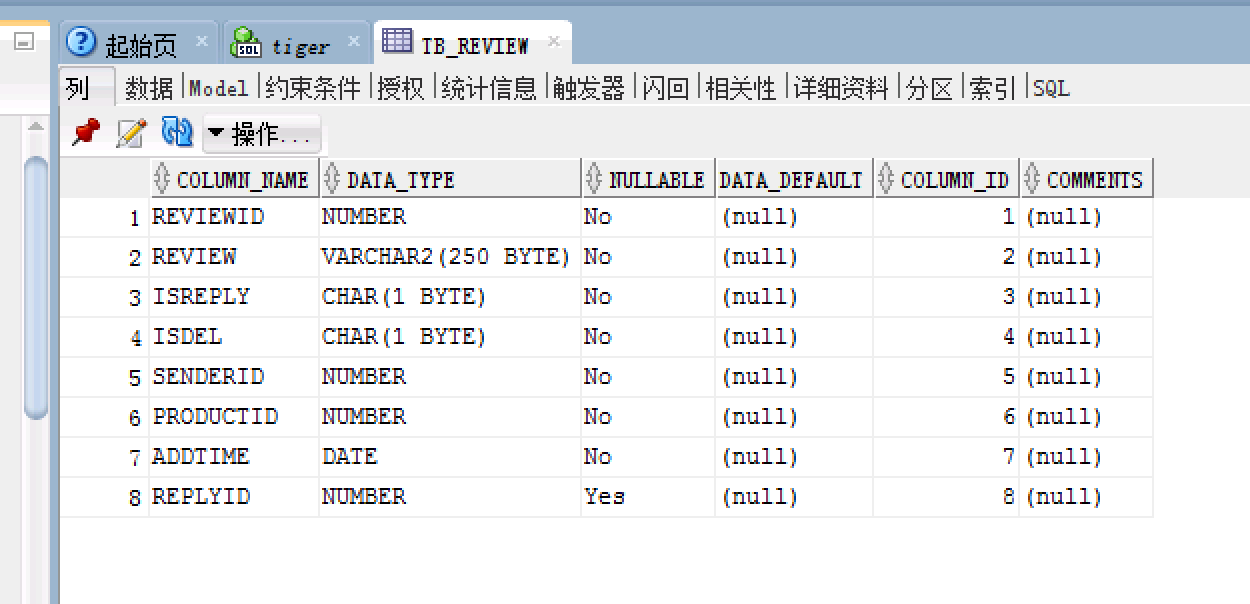
CREATE OR REPLACE TRIGGER selfplus\_tb\_review

BEFORE INSERT ON tb\_review FOR EACH ROW

begin

select identify\_tb\_review.nextval into:new.reviewID from dual;

end;



/\*创建tb\_province表\*/

CREATE TABLE tb\_province(

provinceID number NOT NULL,

provinceName varchar2(30) NOT NULL

)

--创建sequence

create sequence identify\_tb\_province

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

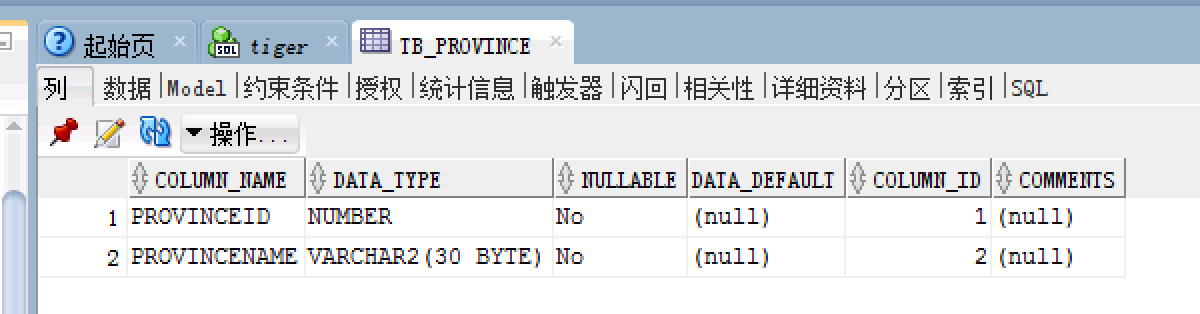
CREATE OR REPLACE TRIGGER selfplus\_tb\_province

BEFORE INSERT ON tb\_province FOR EACH ROW

begin

select identify\_tb\_province.nextval into:new.provinceID from dual;

end;



/\*创建tb\_city表\*/

CREATE TABLE tb\_city(

cityID number NOT NULL,

provinceID number NOT NULL,

cityName varchar2(30) NOT NULL

)

--创建sequence

create sequence identify\_tb\_city

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

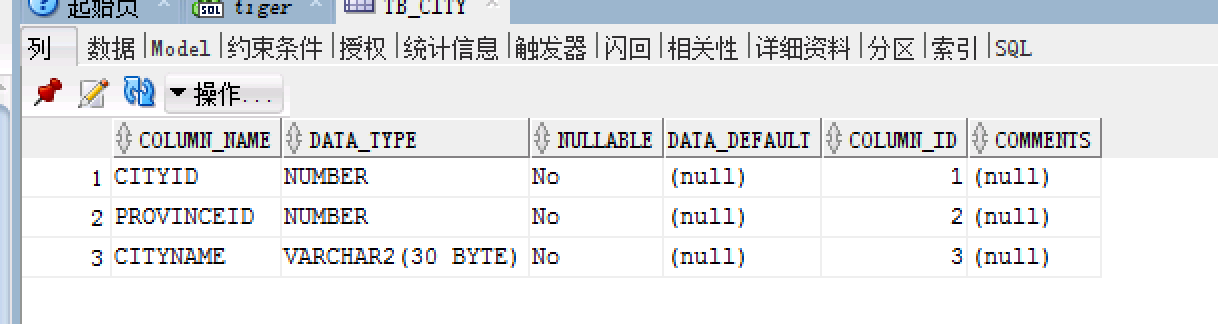
CREATE OR REPLACE TRIGGER selfplus\_tb\_city

BEFORE INSERT ON tb\_city FOR EACH ROW

begin

select identify\_tb\_city.nextval into:new.cityID from dual;

end;



/\*创建tb\_productSize表\*/

CREATE TABLE tb\_productSize(

productSizeID number NOT NULL,

productSize varchar2(20) NOT NULL,

productID number NOT NULL

)

--创建sequence

create sequence identify\_tb\_productSize

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

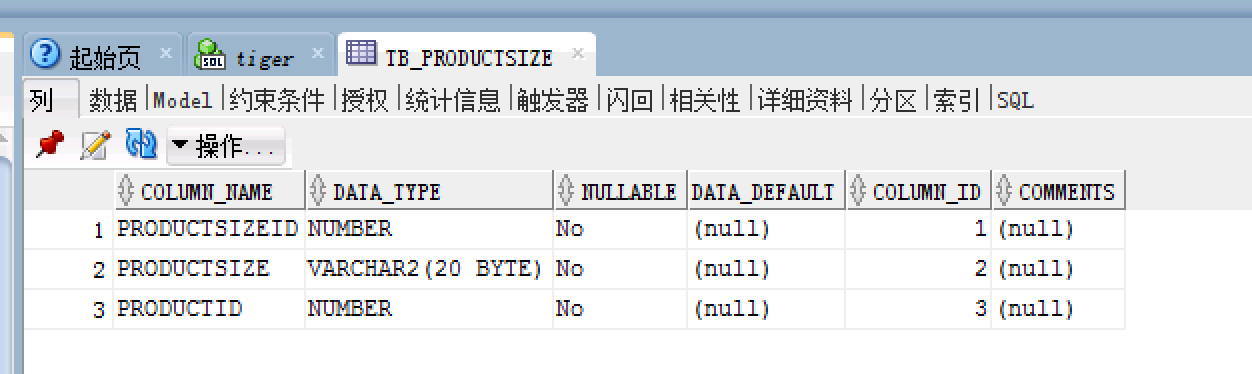
CREATE OR REPLACE TRIGGER selfplus\_tb\_productSize

BEFORE INSERT ON tb\_productSize FOR EACH ROW

begin

select identify\_tb\_productSize.nextval into:new.productSizeID from dual;

end;



/\*创建tb\_productColor表\*/

CREATE TABLE tb\_productColor(

productColorID number NOT NULL,

productColor varchar2(20) NOT NULL,

stockpile number NOT NULL,

productSizeID number

)

--创建sequence

create sequence identify\_tb\_productColor

increment by 1

start with 1

nomaxvalue

nocycle

nocache;

--创建自增触发器

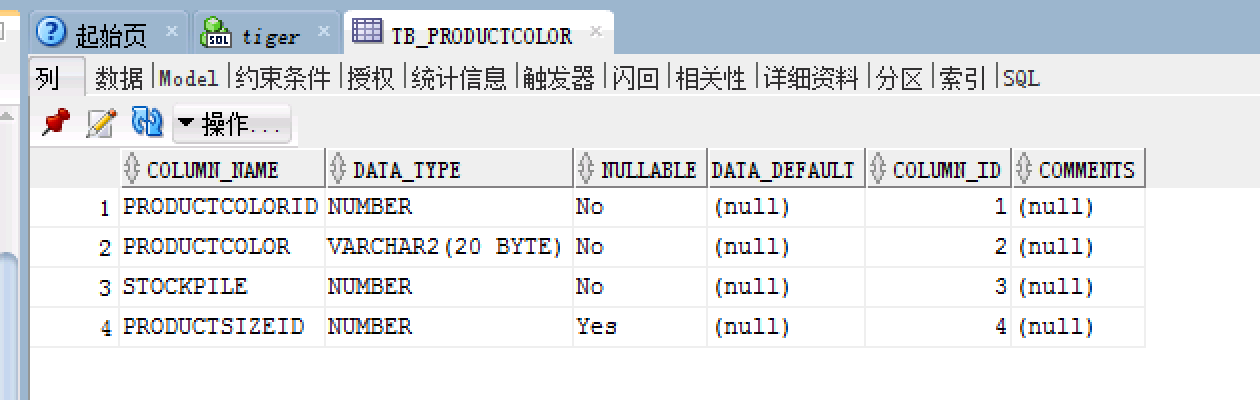
CREATE OR REPLACE TRIGGER selfplus\_tb\_productColor

BEFORE INSERT ON tb\_productColor FOR EACH ROW

begin

select identify\_tb\_productColor.nextval into:new.productColorID from dual;

end;



**三、添加约束**

/\*给每个表添加约束\*/

--主键约束

alter table tb\_user add constraint PK\_userID primary key(userID);

alter table tb\_product add constraint PK\_productID primary key(productID);

alter table tb\_productType add constraint PK\_productTypeID primary key(productTypeID);

alter table tb\_order add constraint PK\_orderID primary key(orderID);

alter table tb\_userAccount add constraint PK\_userAccountID primary key(userAccountID);

alter table tb\_review add constraint PK\_reviewID primary key(reviewID);

alter table tb\_province add constraint PK\_provinceID primary key(provinceID);

alter table tb\_city add constraint PK\_cityID primary key(cityID);

alter table tb\_productSize add constraint PK\_productSizeID primary key(productSizeID);

alter table tb\_productColor add constraint PK\_productColorID primary key(productColorID);

--唯一约束

alter table tb\_user add constraint UNI\_userName unique(name);

--外键约束

alter table tb\_user add constraint FK\_user\_cityID foreign key(cityID) references tb\_city(cityID);

alter table tb\_product add constraint FK\_product\_userID foreign key(userID) references tb\_user(userID);

alter table tb\_product add constraint FK\_product\_typeID foreign key(typeID) references tb\_productType(productTypeID);

alter table tb\_order add constraint FK\_order\_toID foreign key(toID) references tb\_user(userID);

alter table tb\_order add constraint FK\_order\_colorID foreign key(colorID) references tb\_productColor(productColorID);

alter table tb\_order add constraint FK\_order\_productID foreign key(productID) references tb\_product(productID);

alter table tb\_userAccount add constraint FK\_userAccount\_userID foreign key(userID) references tb\_user(userID);

alter table tb\_userAccount add constraint KF\_userAccount\_orderID foreign key(orderID) references tb\_order(orderID);

alter table tb\_review add constraint FK\_review\_senderID foreign key(senderID) references tb\_user(userID);

alter table tb\_review add constraint FK\_review\_productID foreign key(productID) references tb\_product(productID);

alter table tb\_city add constraint FK\_city\_provinceID foreign key(provinceID) references tb\_province(provinceID);

alter table tb\_productSize add constraint FK\_productSize\_productID foreign key(productID) references tb\_product(productID);

alter table tb\_productColor add constraint FK\_productColor\_productSizeID foreign key(productSizeID) references tb\_productSize(productSizeID);

**四、基础数据(商品类型+省份+城市)添加**

/\*基础数据(商品类型+省份+城市)添加\*/

/\*tb\_productType表添加商品类型\*/

insert into tb\_productType values (seq\_productType.nextval,'书籍');

insert into tb\_productType values (seq\_productType.nextval,'手机');

insert into tb\_productType values (seq\_productType.nextval,'平板');

insert into tb\_productType values (seq\_productType.nextval,'笔记本电脑');

/\*tb\_province表添加省份名称\*/

insert into tb\_province values (seq\_province.nextval,'北京');

insert into tb\_province values (seq\_province.nextval,'广东');

insert into tb\_province values (seq\_province.nextval,'山东');

/\*tb\_city表给山东省添加城市名称\*/

insert into tb\_city values (seq\_city.nextval,1,'北京');

insert into tb\_city values (seq\_city.nextval,2,'珠海');

insert into tb\_city values (seq\_city.nextval,3,'济南');

insert into tb\_city values (seq\_city.nextval,3,'青岛');

insert into tb\_city values (seq\_city.nextval,3,'烟台');

insert into tb\_city values (seq\_city.nextval,3,'威海');

**五、用户数据(注册+信息完善)添加**

/\*用户数据(注册+信息完善)添加\*/

/\*tb\_user表添加用户信息\*/

insert into tb\_user (userID,name,password) values (seq\_user.nextval,'雷军','123456');

insert into tb\_user (userID,name,password) values (seq\_user.nextval,'黄章','111111');

update tb\_user set email='leijun@xiaomi.com',provinceid=1,cityid=1,age=47,birthday=to\_date('1969-12-16','yyyy-mm-dd') where userid=1001;

update tb\_user set email='huangzhang@meizu.com',provinceid=2,cityid=2,age=40,birthday=to\_date('1976-2-13','yyyy-mm-dd') where userid=1002;

**六、商品信息(商品信息+型号+颜色)添加**

/\*商品信息(商品信息+型号+颜色)添加\*/

/\*tb\_product表添加商品信息\*/

insert into tb\_product(productID,userID,productName,typeID,price) values (seq\_product.nextval,1001,'索尼WALKMAN',1,1000);

insert into tb\_product(productID,userID,productName,typeID,price) values (seq\_product.nextval,1001,'小米5',2,2299);

insert into tb\_product(productID,userID,productName,typeID,price) values (seq\_product.nextval,1001,'小米4s',2,1799);

insert into tb\_product(productID,userID,productName,typeID,price) values (seq\_product.nextval,1002,'魅族PRO6',2,2499);

insert into tb\_product(productID,userID,productName,typeID,price) values (seq\_product.nextval,1002,'魅族MX5',2,1799);

/\*tb\_productSize表添加商品型号\*/

insert into tb\_productSize values (seq\_productSize.nextval,'标准版',2);

insert into tb\_productSize values (seq\_productSize.nextval,'高配版',2);

insert into tb\_productSize values (seq\_productSize.nextval,'至尊版',2);

insert into tb\_productSize values (seq\_productSize.nextval,'标准版16GB',1);

insert into tb\_productSize values (seq\_productSize.nextval,'标准版64GB',1);

insert into tb\_productSize values (seq\_productSize.nextval,'标准版128GB',1);

/\*tb\_productColor表添加商品型号\*/

insert into tb\_productSize values (seq\_productSize.nextval,'标准版',2);

insert into tb\_productSize values (seq\_productSize.nextval,'高配版',2);

insert into tb\_productSize values (seq\_productSize.nextval,'至尊版',2);

insert into tb\_productSize values (seq\_productSize.nextval,'标准版16GB',1);

insert into tb\_productSize values (seq\_productSize.nextval,'标准版64GB',1);

insert into tb\_productSize values (seq\_productSize.nextval,'标准版128GB',1);

**七、删除用户信息**

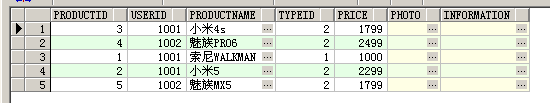
/\*删除用户\*\*\*\*/

delete from tb\_user where userid=1001;

**八、商品信息查询**

/\*商品信息查询\*/

select \* from tb\_product;



/\*单表查询\*/

/\*查询tb\_product表中商品id为的商品信息\*/

select \* from tb\_product where productid=1;



/\*查询tb\_productSize表中商品id为的型号信息\*/

select \* from tb\_productsize where productid=1;



/\*查询tb\_productColor表中商品型号id为、、的颜色与库存信息\*/

select \* from tb\_productcolor where productsizeid=1;



/\*多表查询\*/

/\*用户在商城以商品类型查询一件商品的名称\*/

select productname from tb\_producttype join tb\_product on tb\_producttype.producttypeid=tb\_product.typeid where producttypename='手机';



/\*用户在商城以商品名称查询一件商品的所有信息\*/

select \* from tb\_producttype join tb\_product on tb\_producttype.producttypeid=tb\_product.typeid where productname='小米5';



**九、用户信息查询**

/\*用户信息查询\*/

/\*用户在商城可以查询已知用户(或卖家)的基本信息\*/

select userid,name,email,provincename,cityname,telephone,sex,age,birthday,photo,dsp,qq,msn,lovebook,lovemusic,lovemovie,lovesport,lovegame from tb\_user join tb\_province on tb\_user.provinceid=tb\_province.provinceid join tb\_city on tb\_province.provinceid=tb\_city.provinceid;



**十、用户下订单并确认付款（应用）**

/\*用户下订单并确认付款(应用事务)\*/

/\*用户\*\*\*想要购买卖家\*\*\*的产品颜色-大小-产品名\*/

0-未付款 1-已付款 2-发货 4-撤回 5-完成

/\*定义变量,用于存储本次订单的物品单价\*/

declare

p number;

begin

select tb\_product.price into p from tb\_order join tb\_product on tb\_order.productid=tb\_product.productid where orderid = 1;

end;

/\*定义变量,用于存储本次订单的物品ID\*/

declare

gId number;

begin

select productId into gId from tb\_order where orderid = 1;

end;

/\*定义变量,用于存储本次订单的颜色ID\*/

declare

cId number;

begin

select colorId into cId from tb\_order where orderid = 1;

end;

/\*定义变量,用于存储本次订单的买家ID\*/

declare

tId number;

begin

select toId into tId from tb\_order where orderid = 1;

end;

/\*定义变量,用于存储流水表的用户ID\*/

declare uid number;

begin

select userId into uid from tb\_useraccount where orderid = 1;

dbms\_output.put\_line(uid);

end;

/\*用户\*\*\*\*下订单\*/

/\*用户\*\*\*\*将订单信息填入到tb\_order表中\*/

--orderState订单状态为用户选择填入下订单(买家付款后自动转为已付款,买家付款后卖家可选择修改为延后,卖家发货后选择修改为发货,买家未付款时卖家可选择修改为撤销,买家收到货物后可选择修改为完成)

declare

toID number;

colorID number;

pID number;

address varchar2(100);

p varchar2(50);

c varchar2(50);

tel char(11);

ostate number;

cou number;

price number;

begin

toID:=1001;

cou:=1;

select productid into pID from tb\_product where productname='小米4s';

select productcolorid into colorid from tb\_productcolor where productsizeid=pID and productcolor='黑色';

select provincename into p from tb\_province join tb\_user on tb\_user.provinceid=tb\_province.provinceid where userid=toID;

select cityname into c from (tb\_city join tb\_user on tb\_user.cityid=tb\_city.cityid) join tb\_province on tb\_province.provinceid=tb\_city.provinceid where userid=toID;

select price into price from tb\_product where productid = pID;

select telephone into tel from tb\_user where userid=toid;

price:=price \* cou;

if (p=c) then

address:=p;

else

address:=p||c;

end if;

ostate:=0;

dbms\_output.put\_line(tel);

insert into tb\_order values (3,toID,colorID,pID,address,tel,ostate,cou,price,sysdate(),sysdate());

end;

/\*下订单后减少商品库存\*/

declare

cou number;

stock number;

colorID number;

begin

cou:=1;

colorID:=2;

select stockpile into stock from tb\_productcolor where productcolorid=colorID;

stock:=stock-cou;

dbms\_output.put\_line(stock);

update tb\_productcolor set stockpile = stock where productcolorid=colorID;

end;

/\*确认付款后减少用户余额\*/

declare

pID number;

toID number;

pri number;

mon number;

begin

pID:=3;

toID:=1001;

select price into pri from tb\_product where productid=pID;

select money into mon from tb\_user where userid=toid;

mon:=mon-pri;

dbms\_output.put\_line(mon);

end;

/\*将用户付款的资金冻结后订单状态改为已付款\*/

update tb\_order set orderstate=1 where orderid=3;

**十一、买家对订单信息进行修改（包括发货后的订单状态修改及修改商品颜色时应用事务）**

/\*卖家对订单信息进行修改\*/

/\*假设买家\*\*\*想修改商品的颜色，则需要应用事务，对相应颜色的库存进行修改\*/

/\*修改信息是各项信息均为可选择项，如不想修改某项信息，则选择以前的选项，但是会将数据重复的修改一次，即用相同的数据来覆盖之前的数据\*/

/\*除此之外卖家发货后可将订单状态改为发货\*/

/\*定义变量,用于存储本次订单的物品单价\*/

declare

p number;

begin

select tb\_product.price into p from tb\_product join tb\_order on tb\_product.productid=tb\_order.productid where orderid=3;

dbms\_output.put\_line(p);

end;

/\*修改tb\_order表中的送货地址\*/

update tb\_order set address='XXXXXX' where orderid=3;

/\*修改tb\_order表中的联系方式\*/

update tb\_order set telephone='10010' where orderid=3;

/\*修改tb\_order表中的状态\*/

--若卖家因故不能发货，但已付款则卖家可修改订单状态为'延后'

update tb\_order set orderstate=3 where orderid=3;

/\*修改tb\_order表中的商品颜色\*/

update tb\_order set colorid=3 where orderid=3;

/\*修改tb\_order表中的商品总价\*/

--若和买家商议后买家同意降价，但商品信息上还是不修改的，就可以在订单总价上修改，付款后卖家修改总价，差价打回买家账户余额内

update tb\_order set price=1999 where orderid=3;

/\*将与之前的差价加回买家账户余额内\*/

declare

tid number;

cj number;

pris number;

pri number;

mon number;

begin

pris:=1999;

select price into pri from tb\_order where orderid=3;

select toid into tid from tb\_order where orderid=3;

cj:=pri-pris;

select money into mon from tb\_user where userid=tid;

mon:=mon+cj;

update tb\_user set money=mon where userid=tid;

end;

/\*修改tb\_user表中买家的账户余额\*/

declare

tid number;

cj number;

pris number;

pri number;

mon number;

begin

pris:=1999;

select price into pri from tb\_order where orderid=3;

select userid into tid from tb\_user join tb\_order on tb\_order.toid=tb\_user.userid where orderid=3;

cj:=pri-pris;

select money into mon from tb\_user where userid=tid;

mon:=mon+cj;

update tb\_user set money=mon where userid=tid;

end;

/\*修改颜色后要将之前选择的颜色库存加一再讲修改后的颜色库存减一\*/

declare

ocid number;

ncid number;

s1 number;

s2 number;

begin

ncid:=3;

select colorid into ocid from tb\_order where orderid=3;

select stockpile into s1 from tb\_productcolor where productcolorid=ocid;

select stockpile into s2 from tb\_productcolor where productcolorid=ncid;

s1:=s1+1;

s2:=s2-1;

update tb\_productcolor set stockpile=s1 where productcolorid=ocid;

update tb\_productcolor set stockpile=s2 where productcolorid=ncid;

end;

**十二、订单完成进行转账**

/\*订单完成进行转账\*/

/\*定义变量,用于存储流水表的用户ID\*/

declare uid number;

begin

select userId into uid from tb\_useraccount where orderid = 1;

end;

/\*定义变量,用于存储本次订单的物品单价\*/

declare

p number;

begin

select price into p from tb\_order where orderid = 1;

end;

/\*买家收到商品后，修改tb\_order表中的状态\*/

update tb\_order set orderstate=3 where orderid=4;

/\*订单状态为'完成'后建立用户资金流水表一条卖家出售商品收入记录\*/

假设orderstate=5为完成。

CREATE OR REPLACE TRIGGER tb\_trigger

after update of orderstate

ON tb\_order

referencing old as old\_value

new as new\_value

FOR EACH ROW

when (new\_value.orderstate=5)

DECLARE

tid number;

orid number;

typ number;

BEGIN

tid:=1001;

orid:=3;

typ:=1;

insert into tb\_useraccount values (seq\_useraccount.nextval,tid,orid,typ,to\_date('2016-07-03','yyyy-mm-dd'));

END;

/\*交易成功后卖家账户余额增加\*/

假设orderstate=5为完成。

declare

tid number;

orid number;

pri number;

mon number;

begin

orid:=3;

select price into pri from tb\_order where orderid=orid;

select userid into tid from tb\_user join tb\_order on tb\_user.userid=tb\_order.toid where orderid=orid;

select money into mon from tb\_user join tb\_order on tb\_user.userid=tb\_order.toid where orderid=orid;

mon:= pri + mon;

update tb\_user set money=mon where userid=tid;

end;

**十三、货物丢件进行退款**

/\*货物丢件进行退款\*/

/\*前提条件:订单为发货状态，但物流出现丢件情况，由物流赔偿卖家，卖家确认丢件情况发生后，可进行退款操作\*/

/\*定义变量,用于存储流水表的用户ID\*/

declare uid number;

begin

select userId into uid from tb\_useraccount where orderid = 1;

dbms\_output.put\_line(uid);

end;

/\*卖家确认商品丢件后，修改tb\_order表中的状态\*/

update tb\_order set orderstate=4 where orderid=3;

/\*订单状态为'撤销'后建立用户资金流水表一条买家商品退款收入记录\*/

CREATE OR REPLACE TRIGGER tb\_trigger3

after update of orderstate

ON tb\_order

referencing old as old\_value

new as new\_value

FOR EACH ROW

when (new\_value.orderstate=4)

DECLARE

tid number;

orid number;

typ number;

BEGIN

tid:=1001;

orid:=3;

typ:=1;

insert into tb\_useraccount values (8,tid,orid,typ,to\_date('2016-07-04','yyyy-mm-dd'));

END;

/\*撤销成功后买家账户余额恢复\*/

declare

tid number;

pri number;

mon number;

begin

select price into pri from tb\_order where orderid=3;

select money into mon from tb\_user join tb\_order on tb\_order.toid=tb\_user.userid where orderid=3;

select userid into tid from tb\_user join tb\_order on tb\_order.toid=tb\_user.userid where orderid=3;

mon:=mon+pri;

dbms\_output.put\_line(tid);

update tb\_user set money=mon where userid=tid;

end;

**十四、视图**

--/\*多表查询\*/

--/\*用户在商城以商品类型查询一件商品的名称\*/

create or replace view view1

AS

select productName

from tb\_product join tb\_producttype on tb\_product.typeid=tb\_producttype.producttypeid

where productTypeName='MP3';

--/\*用户在商城以商品名称查询一件商品的所有信息\*/

create or replace view view2

AS

select tb\_product.productid,tb\_product.information,tb\_product.photo,tb\_product.price,tb\_product.productname,tb\_product.typeid,tb\_product.userid,tb\_productcolor.productcolor,tb\_productcolor.productcolorid

,tb\_productcolor.productsizeid,tb\_productcolor.stockpile,tb\_productsize.productsize,tb\_producttype.producttypeid,tb\_producttype.producttypename

from tb\_producttype join tb\_product on tb\_producttype.productTypeID=tb\_product.typeid

join tb\_productsize on tb\_productsize.productid=tb\_product.productid

join tb\_productcolor on tb\_productcolor.productsizeid=tb\_productsize.productsizeid

where tb\_product.productName='小米5';

--/\*用户信息查询\*/

--/\*用户在商城可以查询已知用户(或卖家)的基本信息\*/

create or replace view view3

AS

select tb\_user.name,tb\_user.age,tb\_user.birthday,tb\_user.dsp,tb\_user.email,tb\_user.lovebook,tb\_user.lovegame,tb\_user.lovemovie,tb\_user.lovemusic,tb\_user.lovesport,tb\_user.msn

,tb\_user.qq,tb\_user.sex,tb\_user.telephone,tb\_city.cityname,tb\_province.provincename

from tb\_user join tb\_city on tb\_user.cityid=tb\_city.cityid

join tb\_province on tb\_province.provinceid=tb\_city.provinceid;

**十五、用户登录(选做) 例如:sql server 格式:**

/\*用户登录(应用存储过程)\*/

create procedure proc\_land

@resultName int = 0, --定义为表示用户名错误

@resultPassWord int =0, --定义为表示密码错误

@userName nvarchar(10) = '\*\*\*',

@PassWord nvarchar(30) = 'jhc123'

as

select @resultName = COUNT(1) from tb\_user where name=@userName --只要表中有输入的用户名则为表示用户名存在

select @resultPassWord = COUNT(1) from tb\_user where name=@userName and [password]=@PassWord --若存在的这个用户的信息里的密码等与输入的密码则为表示密码核对无误

begin

if @resultName = 1 and @resultPassWord = 1

print '用户登录成功！'

else if @resultName = 0

print '用户名错误！'

else if @resultPassWord = 0

print '密码错误！'

end

create or replace procedure signin

(un varchar2,up varchar2)

as

n number;

begin

select count(\*) into n from tb\_user where name=un and password=up;

if n=0 then

dbms\_output.put\_line('登录失败');

else

dbms\_output.put\_line('登录成功');

end if;

end;