

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的超市管理系统数据库设计 | | |
| 课程 | Oracle数据库应用 | | |
| 学 院 | 信息科学与工程学院 | | |
| 专 业 | 软件工程 | 年级 | 2018级 |
| 学生姓名 | 张逸 | 学号 | 201810414429 |
| 指导教师 | 赵卫东 | 职称 | 副教授 |

|  |  |  |  |
| --- | --- | --- | --- |
| **评分项** | **评分标准** | **满分** | **得分** |
| 文档整体 | 文档内容详实、规范，美观大方 | 10 |  |
| 表设计 | 表，表空间设计合理，数据合理 | 20 |  |
| 用户管理 | 权限及用户分配方案设计正确 | 20 |  |
| PL/SQL设计 | 存储过程和函数设计正确 | 30 |  |
| 备份方案 | 备份方案设计正确 | 20 |  |
| **得分合计** | | |  |

2021 年 6 月 1 日

## 一、实验目的：

在如今，随着社会的发展，人们消费能力日益渐增。与之而来的结果就是超市规模不断扩大，社会所需的商品数量与日俱增。超市的管理难度与日俱增。为了解决这个难题，我基于Oracle设计了这个超市管理系统数据库。

## 二、实验内容

这次设计的超市管理系统主要包含以下几个模块:

* 系统权限的设定
* 原始数据录入
* 数据的汇总及查询。
* 从而实现对货品、销售以及员工等信息实现全面、动态、及时的管理。

### 数据库设计环境：

* 数据库软件系统：oracle数据库
* 数据库名称：ZYfinal
* 开发平台和开发工具：Windows 10 与 SQL Developer

### 对象实体：

本次超市系统主要有6个实体：商品、部门、库存、促销员、顾客、订单

* 商品（goods）：商品编号(goods\_no)，商品名称（goods\_name），商品分类（goods\_class），商品进价（intake\_price）,商品单价（outtake\_price),商品品牌（goods\_brand），商品规格（goods\_info）
* 库存（stock）：商品编号(goods\_no)，库存编号(stock\_no)，库存数量(size)，库存位置(position)，入库时间(instore\_time)，出库时间(outstock\_time)，出库商品数量（outgoods\_size）
* 部门（department）：部门编号（department\_no）,部门名称（department\_name）,部门店长（department\_minister），电话（phone），商品编号（goods\_no）
* 促销员（promoter）：促销员编号（promoter\_no）,姓名（promoter\_name）,性别（sex），出生日期（birthday），所属部门编号（department\_no），入职时间（admission\_time），工资(salary)
* 订单（orders）：订单编号（orders\_no），时间（orders\_time）,订单详情（orders\_info）,订单金额（orders\_price）
* 顾客（customer）：订单编号（orders\_no），姓名（customer\_name），电话（customer\_tel），顾客id（customer\_id），

### 业务逻辑

* 商品库存业务  
  库存管理是指在上产经营中为销售和耗用而存储的各种资产。企业存货不当都会增加不必要的费用:存货过多将导致物资的积压、存储费用增加、流动资金周转不利，并且过量的库存会掩盖管理不善的问题。存货不足则影响企业的正常销售活动。库存管理的目标是需求最佳存货数量，使库存的成本总额最小化。其主要业务如下:产品入库、出库、盘库等日常处理。  
  科学合理地确定某库存物资的库存控制参数。进行严格的库存控制工作。汇总统计各类库存数据，反应和监督储备资金占用及采购资金使用情况。
* 促销业务  
  销售对一个企业的生存和发展是至关重要的，它不再是传统意义上的“卖东西”，它对外关系到企业产品、服务和企业形象等多方面因素，对内涉及到销售、库存、财务等信息。此外信息化的高速发展为企业的销售工作提出了新的挑战，封闭意味着落后，超市要抓住时机，引入现代化的销售管理模式，实施企业的信息化、智能化销售管理，才能将利益最大化。  
  本系统业务为促销员与部门的联系，进行促销业务的实施。

## 详细设计

首先，进行创建数据库

由于oracle利用命令行创建数据库较为繁琐，因此利用oracle自带的Database configuration Assistant 创建数据库zyfinal。

* 实验结果截图如下：
* 创建用户ZY并进行数据库授权

Alter session set container =zyfinal；  
Create user ZY IDENTIFIED BY abc;  
Grant dba to ZY;

实验结果截图：

* 表空间的建立

create tablespace final\_space1  
datafile 'final\_space1.dbf'   
size 150M autoextend on next 50m maxsize unlimited;  
  
create tablespace final\_space2  
datafile 'final\_space2.dbf'   
size 150M autoextend on next 50m maxsize unlimited;  
  
create tablespace final\_space3  
datafile 'final\_space3.dbf'   
size 150M autoextend on next 50m maxsize unlimited;  
create tablespace final\_space4  
  
datafile 'final\_space4.dbf'   
size 150M autoextend on next 50m maxsize unlimited;

实验结果截图：

## 创建表以及实现表分区操作

表进行分区后，逻辑上表仍然是一张完整的表，只是将表中的数据在物理上存放到多个表空间(物理文件上)，这样查询数据时，不至于每次都扫描整张表。所以本次使用分为三张表的目的便是如此。用NOCOMPRESS NO INMEMORY语句来缓解了内存资源的压力，这就将有限的资源分配给更需要的用户，增加了数据库资源的利用率。

* 商品表（goods）

create table goods (  
 goods\_no char(5) not null primary key,  
 goods\_name char(20) not null,  
 goods\_class char(8),  
 goods\_info char(10),  
 goods\_brand char(10),  
 intake\_price float not null CHECK(intake\_price > 0),  
 outtake\_price float not null CHECK(outtake\_price > 0)  
)TABLESPACE final\_space1;

实验结果截图：

* 库存表（stock）

create table stock (  
 goods\_no char(5) not null,  
 stock\_no char(5) not null,  
 size\_ char(10),  
 position\_ char(20),  
 instore\_time date not null,  
 outstock\_time date not null,  
 outgoods\_size char(10),  
 CONSTRAINT stock\_goods\_key foreign key (goods\_no) references goods(goods\_no)  
 )TABLESPACE final\_space1;

实验结果截图：

* 3.部门表（department）

create table department (  
 department\_no char(5) not null primary key,  
 department\_name char(10) not null,  
 department\_minister char(5) not null,  
 phone char(11),  
 goods\_no char(5),  
 CONSTRAINT department\_goods\_key FOREIGN KEY (goods\_no) REFERENCES goods (goods\_no)  
)TABLESPACE final\_space1;

实验结果截图：

* 促销员表（promoter）

create table promoter (  
 promoter\_no char(5) not null primary key,  
 promoter\_name char(10) not null,  
 sex char(2) not null CHECK(sex IN ('男','女')),  
 department\_no char(5) not null,  
 birthday date ,  
 admission\_time date not null,  
 salary float not null,  
 CONSTRAINT promoter\_department\_key FOREIGN KEY (department\_no) REFERENCES department (department\_no))

实验结果截图：

* 订单表（orders）

CREATE TABLE orders   
(  
 orders\_no CHAR(5 BYTE) NOT NULL primary key   
, orders\_info CHAR(15 BYTE) NOT NULL   
, orders\_price FLOAT(126)   
, orders\_time DATE NOT NULL   
) partition by range (orders\_time)  
 (  
partition p1 values LESS THAN (TO\_DATE(' 2018-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN')) tablespace final\_space1,  
  
 partition p2 values LESS THAN (TO\_DATE(' 2019-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN')) tablespace final\_space2,  
  
 partition p3 values LESS THAN (TO\_DATE(' 2020-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN')) tablespace final\_space3,  
   
 partition p4 values LESS THAN (MAXVALUE) tablespace final\_space4)

实验结果截图：

* 顾客表(customer)

create table customer(  
 customer\_id char(5) not null primary key,  
 customer\_name char(10) not null,  
 customer\_tel char(11),  
 orders\_no char(5) not null,  
 CONSTRAINT customer\_orders\_key FOREIGN KEY (orders\_no) REFERENCES orders (orders\_no)  
)TABLESPACE final\_space1;

实验结果截图：

## 创建用户，并授权

分别创建俩个用户user1和user2，并且授权他们只能使用space1表空间，同时创建两个角色role1，role2，并分别赋予role1，role2读任何表和读、修改任何表的权利。同时还要用grant create session to user1语句授予不同用户进入pdb的权利。

create user user1 IDENTIFIED by 123;  
create user user2 IDENTIFIED by 123;  
alter user user1 quota unlimited on wlj\_space1;  
alter user user2 quota unlimited on wlj\_space1;  
grant create session to user1;  
grant create session to user2;  
  
create role role1;  
create role role2;  
grant select any table to role1;  
grant select any table to role2;  
grant update any table to role2;  
grant role1 to user1;  
grant role2 to user2;

实验结果截图：

## 向数据库中批量的插入数据

在已经创建好的数据库中批量插入数据用循环进行数据插入。

declare  
dt date;  
orders\_no char(5);  
orders\_info char(5);  
orders\_price float;  
BEGIN  
insert into goods (goods\_no,goods\_name,goods\_class,goods\_info,goods\_brand,intake\_price,outtake\_price) values ('1','金龙鱼','日用','食用','金龙鱼',59.9,75.6);  
insert into goods (goods\_no,goods\_name,goods\_class,goods\_info,goods\_brand,intake\_price,outtake\_price) values ('2','冰红茶','饮料','食用','康师傅',2.3,3.0);  
insert into goods (goods\_no,goods\_name,goods\_class,goods\_info,goods\_brand,intake\_price,outtake\_price) values ('3','巧克力','休闲','食用','德夫',25.5,35.6);  
insert into goods (goods\_no,goods\_name,goods\_class,goods\_info,goods\_brand,intake\_price,outtake\_price) values ('4','旺仔牛奶','牛奶','食用','旺仔',32.3,53.4);  
insert into goods (goods\_no,goods\_name,goods\_class,goods\_info,goods\_brand,intake\_price,outtake\_price) values ('5','徐福记','休闲','食用','徐福记',16.5,55.4);  
insert into department (department\_no,department\_name,department\_minister,phone,goods\_no) values ('501','日用','zs','110','1');  
insert into department (department\_no,department\_name,department\_minister,phone,goods\_no) values ('502','酒水','ls','120','2');  
insert into department (department\_no,department\_name,department\_minister,phone,goods\_no) values ('503','休闲','ww','119','3');  
insert into department (department\_no,department\_name,department\_minister,phone,goods\_no) values ('504','牛奶','cl','10000','4');

实验结果截图：

批量插入测试数据：

insert into orders   
(orders\_no,orders\_info,orders\_price,orders\_time)  
select rownum as orders\_no,  
rownum as orders\_info,  
trunc(dbms\_random.value(0, 100)) as orders\_price ,  
TO\_DATE(' 2018-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN') as orders\_time  
from dual  
connect by level <= 50000;  
select \* from orders;

## 切换角色验证分区和授权

* 在user1中无法修改数据，而在user2中可以修改数据，证明权限分配成功，
* 在orders表单中根据时间段不同将其放在了不同的表空间中，通过查询语句也可以说明其分区的正确性！

update wlj.orders set orders\_info='wlj' where orders\_no=2130;  
select \* from wlj.orders;  
select count(\*) from wlj.orders partition(p2);  
select count(\*) from wlj.orders partition(p3);  
update wlj.orders set orders\_info='wlj';  
select \* from wlj.orders;

## 程序包的创建

在包中建立一个FUNCTION和一个PROCEDURE,功能是：统计orders在一段时间中的订单收入金额，另一个功能是通过输入商品信息查找对应商品并进行分类统计。

create or replace PACKAGE MyPack IS  
 FUNCTION Get\_TOTAL(dt1 char,dt2 char) RETURN NUMBER;  
 PROCEDURE get\_goods(dt1 char);  
END MyPack;  
create or replace PACKAGE BODY MyPack IS  
FUNCTION Get\_TOTAL(dt1 char,dt2 char) RETURN NUMBER  
 AS  
 N NUMBER;  
 BEGIN  
 select sum(orders\_price) into N from orders where ORDERS\_TIME >= to\_date(dt1,'yyyy-mm-dd hh24:mi:ss')  
 and ORDERS\_TIME <= to\_date(dt2,'yyyy-mm-dd hh24:mi:ss');  
 RETURN N;  
 END;  
PROCEDURE get\_goods(dt1 char)  
 AS  
 a1 NUMBER;  
 b1 NUMBER;  
 c1 NUMBER;  
 d1 NUMBER;  
 cursor cur is  
 select \* from goods where GOODS\_INFO=dt1;  
 begin  
 a1 := 0;  
 b1 := 0;  
 c1 := 0;  
 d1 := 0;  
 --使用游标  
 for v in cur   
 loop  
 if v.goods\_class = '日用'  
 then a1 := a1 + 1;  
 elsif v.goods\_class = '饮料'  
 then b1 := b1 + 1;  
 elsif v.goods\_class = '休闲'  
 then c1 := c1 + 1;  
 elsif v.goods\_class = '牛奶'  
 then d1 := d1 + 1;  
 end if;  
 END LOOP;  
 DBMS\_OUTPUT.PUT\_LINE('日用商品种类：' || a1);  
 DBMS\_OUTPUT.PUT\_LINE('饮料商品种类：' || b1);  
 DBMS\_OUTPUT.PUT\_LINE('休闲商品种类：' || c1);  
 DBMS\_OUTPUT.PUT\_LINE('牛奶商品种类：' || d1);  
 end;  
END MyPack;

* 由图可见程序包创建成功，在编译完成之后，通过sql语句进行程序包的验证

## 数据库备份

通过rman\_level0.sh和rman\_level1.sh脚本对数据库进行全备份和全恢复，在数据库出现异常时候，不损失任何数据！

* 全备份

[oracle@oracle-pc ~]$ cat rman\_level0.sh  
[oracle@oracle-pc ~]$ ./rman\_level0.sh

## 总结

本次基于oracle数据库超市管理系统，通过linux操作系统在sql developer的开发工具，创建了六个实体模型，同时完成了他们之间相互关联的逻辑关系，同时设计了自动备份方案，一个相对完整的管理系统，在安全性方面还有所欠缺，在后期也会日益完善。