

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的服装店管理系统的数据库设计 | | |
| 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 | 2017级 |
| 学生姓名 | 卢峥 | 学号 | 201710414111 |
| 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 | 10 |  |
| PL/SQL设计 | 存储过程和函数设计正确 | 25 |  |
| 备份方案 | 备份方案设计正确 | 25 |  |
| 容灾方案 | DataGuard设计正确 | 10 |  |
| **得分合计** | | |  |

2019 年 11 月 18 日

## 第1章．系统概述

随着人们生活水平的提高，品牌服装越来越受到人们的青睐。但是，很多专卖店仍然采用落后的人工管理方式。这种状况对提高整个专卖店的管理水平、市场预测和辅助决策等带来了负面影响。

本数据库应用系统是针对县区专卖店的实际需求开发设计的，利用计算机运算速度快、存储信息容量大、处理逻辑问题强、功能强大的优势，从专卖店的服装订购、进库、销售、店员管理等实际需求出发，针对性强，功能齐备，旨在通过帮助专卖店服装管理实 现信息化，以提高专卖店管理的效率。

随着人们生活水平的提高，品牌服装越来越受到人们的青睐。但是，很多专卖店仍然采用落后的人工管理方式。这种状况对 提高整个专卖店的管理水平、市场预测和辅助决策等带来了负面影响。 本数据库应用系统是针对县区专卖店的实际需求开发设计的，利用计算机运算速度快、存储信息容量大、处理逻辑问题强、功

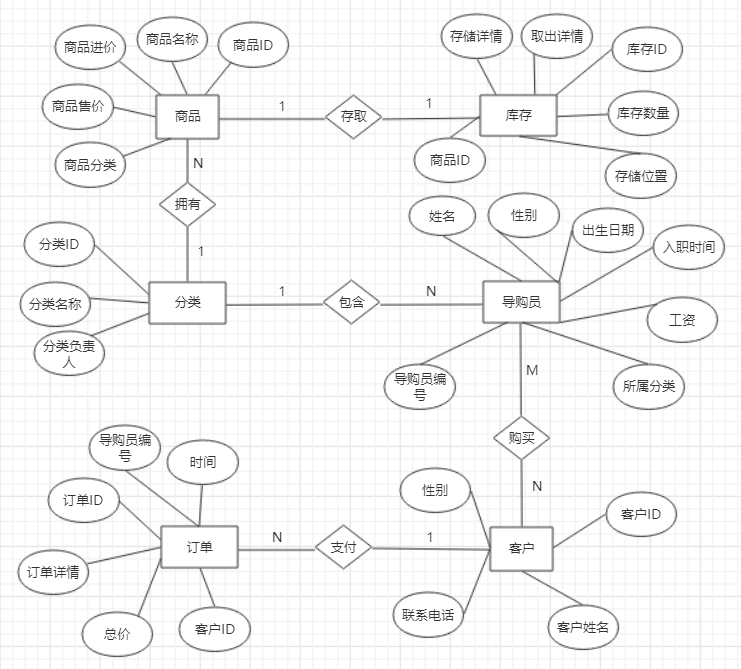
能强大的优势，从专卖店的服装订购、进库、销售、店员管理等实际需求出发，针对性强，功能齐备，旨在通过帮助专卖店服装管理实现信息化，以提高专卖店管理的效率。通过服装专卖店管理系统数据库的设计与实现的具体实践，把软件工程的原理和方法应用到数据库系统的设计中来，简单的开发了数据库。

## **业务流程**

业务流程，是为达到特定的[价值目标](https://baike.baidu.com/item/%E4%BB%B7%E5%80%BC%E7%9B%AE%E6%A0%87/5941750" \t "https://baike.baidu.com/item/%E4%B8%9A%E5%8A%A1%E6%B5%81%E7%A8%8B/_blank)而由不同的人分别共同完成的一系列活动。活动之间不仅有严格的先后顺序限定，而且活动的内容、方式、责任等也都必须有明确的安排和界定，以使不同活动在不同岗位角色之间进行转手交接成为可能。活动与活动之间在时间和空间上的转移可以有较大的跨度。而狭义的业务流程，则认为它仅仅是与[客户价值](https://baike.baidu.com/item/%E5%AE%A2%E6%88%B7%E4%BB%B7%E5%80%BC/10679191" \t "https://baike.baidu.com/item/%E4%B8%9A%E5%8A%A1%E6%B5%81%E7%A8%8B/_blank)的满足相联系的一系列活动。

在本服装实体店管理系统中，结合着实际情况以及网上考察借鉴，整个的业务流程可以大致分为三部分。仓库的商品库存管理，导购员分配管理，客户购买订单管理。

1. 仓库的商品库存管理。无论什么行业，你出售或者售卖的商品永远是重中之重，没有了商品就没有了立足之本，一切都没有意义。所以商品的库存管理也是整个服装实体店管理系统的重中之重。在购进商品时，我们会保留商品的进价以及售价，并在数据库里创建该商品的商品ID，在购进时同时将商品的分类信息保存在商品表里。在通过库存表将商品的数量，存储位置，以及最新的存取时间保存。库存表和商品表是一对一的关系。
2. 导购员分配管理。整个服装店根据商品类型分为不同的分区。每个分区有各自的负责人和若干导购员。客户进入到某个分区，就会有该分区的导购员进行讲解和引导。帮助客户修购选购商品。
3. 客户购买订单管理。客户进入店里，选择商品并支付成功后，生成本次订单，每次订单都有唯一的订单ID。

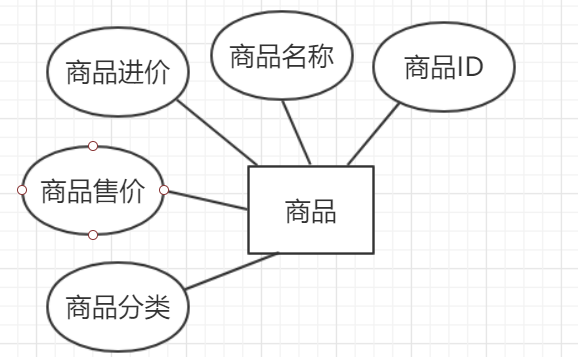


**第3章. 实体分析**

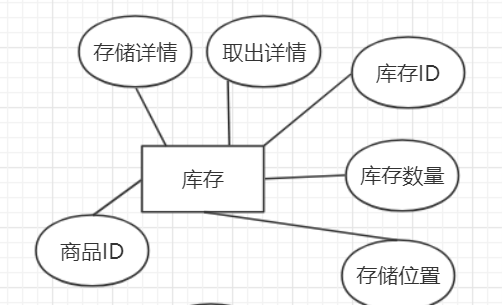
3.1**实体模型**

根据上述业务流程分析，一共有6个实体，分别为：商品，库存，分区，导购员，客户，订单。

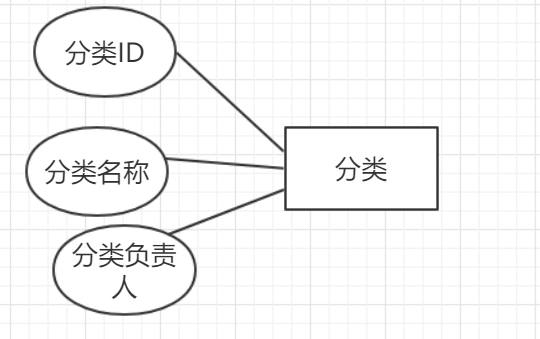
1. 商品(commodity):商品ID(commodity\_id)，商品名称(commodity\_name)，商品进价(commodity\_price)，商品售价(commodity\_sellPrice)，商品分类(commodity\_class);



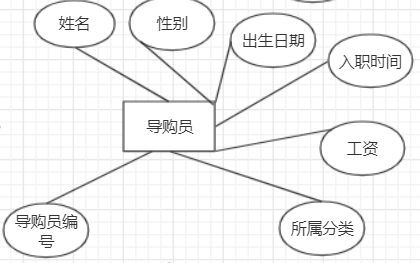
（2）库存（stock）：商品编号（commodity\_id），库存ID(stock\_id)，商品数量(size)，存储位置(position)，存储详情(store\_info)，取出详情(take\_info)，



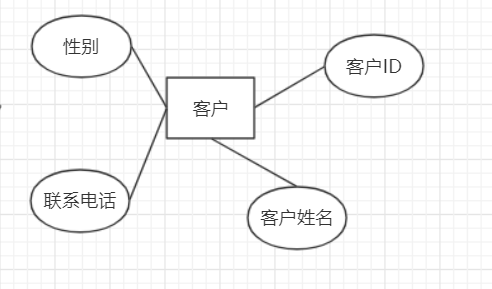
（3）分类（class）：分类ID(class\_id)，分类名称(class\_name)，分类负责人(class\_headPerson)。



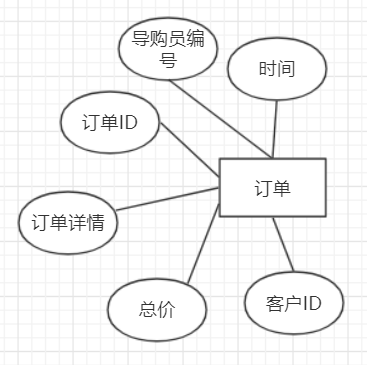
（4）导购员(shopGuide)：导购员编号(shopGuide\_id)，导购员姓名(shopGuide\_name)，导购员性别(shopGuide\_sex)，导购员出生日期(shopGuide\_brithday)。导购员入职时间(shopGuide\_entryTime)，导购员所属分类(class\_id)，工资(wages)。



（5）客户（customer）：客户ID(customer\_id)，客户性别(customer\_sex)，联系电话(customer\_phone)，客户姓名(customer\_name)，订单编号(order\_id)，导购员编号（shopGuide\_id）。



（6）订单(order)：时间(order\_time)，订单ID(order\_id)，订单详情(order\_info)，订单总价(order\_price)，客户ID(customer\_id)，导购员编号（shopGuide\_id）。



3.2 实体联系模型

（1）商品品需要在经过库存里面进行存储或取出。



1. 一个分类拥有多个商品



1. 一个分类拥有多个导购员



1. 导购员与客户应是多对多的关系



1. 一个客户应该拥有多个订单



# 第4章 数据表的设计

（1）商品表

商品表（commodity）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 是否为空 | 主键/外键 | 取值范围 |
| commodity\_id | char(5) | 否 | 主键 |  |
| commodity\_name | char(20) | 否 |  |  |
| class\_id | char(8) | 否 | 外键 |  |
| commodity\_price | float | 否 |  | 大于零 |
| commodity\_sellprice | float | 否 |  | 大于零 |

（2）商品库存表

商品库存表（stock）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 是否为空 | 主键/外键 | 取值范围 |
| stock\_id | char(5) | 否 | 主键 |  |
| commodity\_id | char(5) | 否 | 外键 |  |
| size | char(10) |  |  |  |
| position | char(20) |  |  |  |
| store\_info | char(10) |  |  |  |
| take\_info | char(10) |  |  |  |

（3）商品分类表

商品分类表（class）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 是否为空 | 主键/外键 | 取值范围 |
| class\_id | char(5) | 否 | 主键 |  |
| class\_name | char(10) | 否 |  |  |
| class\_headperson | char(5) | 否 |  |  |

（4）导购员表

导购员表（shopGuide）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 是否为空 | 主键/外键 | 取值范围 |
| shopGuide\_id | char(10) | 否 | 主键 |  |
| shopGuide\_name | char(10) | 否 |  |  |
| shopGuide\_sex | char(2) | 否 |  | 男，女 |
| shopGuide\_brithday | char(5) |  |  |  |
| shopGuide\_entryTime | char(10) | 否 |  |  |
| class\_id | char(5) | 否 | 外键 |  |
| wages | float |  |  |  |

（5）客户表

客户表（customer）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 是否为空 | 主键/外键 | 取值范围 |
| customer\_id | char(10) | 否 | 主键 |  |
| customer\_name | char(15) | 否 |  |  |
| customer\_sex | char(2) |  |  |  |
| customer\_phone | char(15) |  |  |  |

（6）订单表

订单表（order）

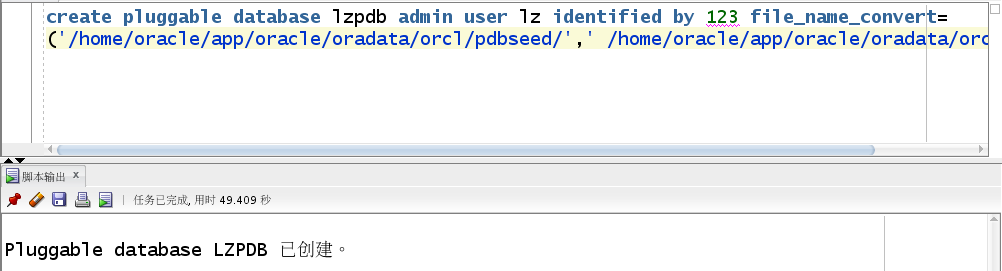
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 是否为空 | 主键/外键 | 取值范围 |
| order\_id | char(10) | 否 | 主键 |  |
| order\_time | char(10) | 否 |  |  |
| order\_price | char(2) | 否 |  |  |
| order\_info | char(8) |  |  |  |
| cusomer\_id | char(10) |  | 外键 |  |
| shopGuide\_id | char(10) |  | 外键 |  |

## 创建pdb数据库，表空间，表，以及用户

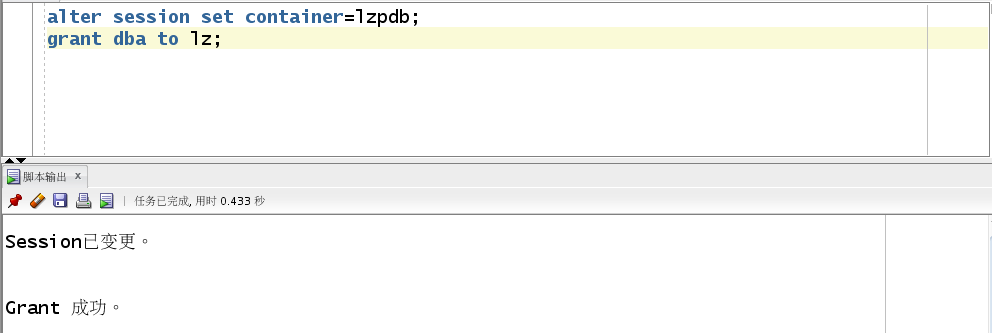
5.1 创建pdb数据库以及管理员

创建了一个名叫lzpdb的数据库，指定了它的存储位置，以及创建了数据库lzpdb的管理员，具体sql如下

|  |
| --- |
| create pluggable database lzpdb admin user lz identified by 123 file\_name\_convert=  ('/home/oracle/app/oracle/oradata/orcl/pdbseed/',' /home/oracle/app/oracle/oradata/orcl/lzpdb'); |



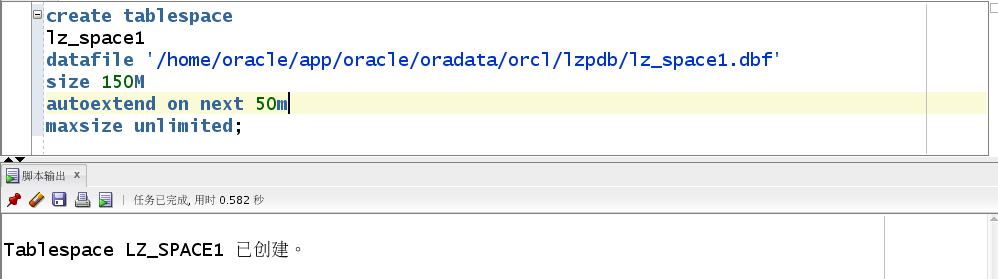
|  |
| --- |
| alter session set container=lzpdb;  grant dba to lz; |



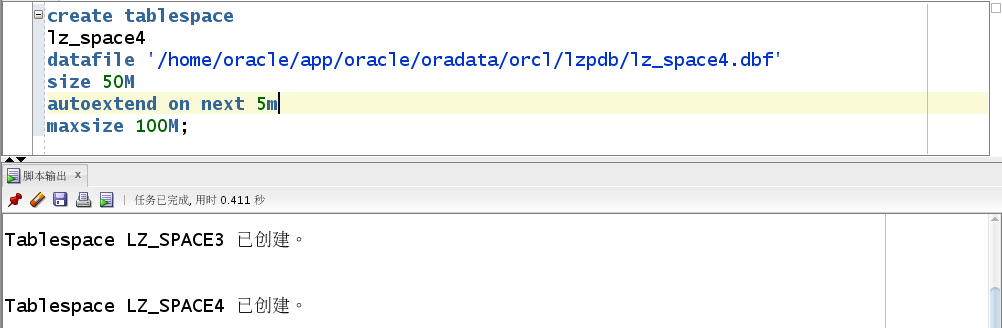
5.2 创建表空间

在lzpdb数据库中创建了表空间lz\_space1，表空间lz\_space2,表空间lz\_space3,表空间lz\_space4

|  |
| --- |
| create tablespace  lz\_space1  datafile '/home/oracle/app/oracle/oradata/orcl/lzpdb/lz\_space1.dbf'  size 150M  autoextend on next 50m  maxsize unlimited; |



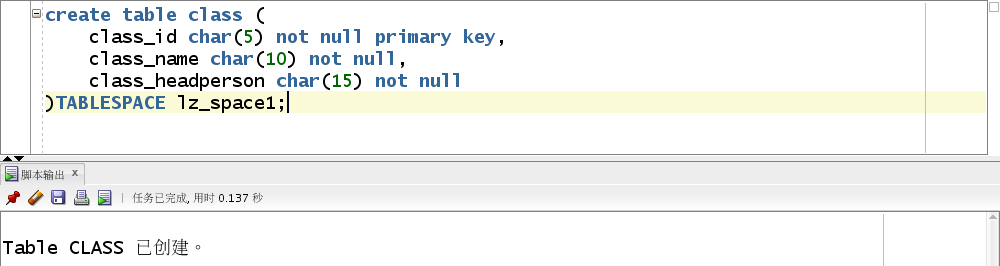
|  |
| --- |
| create tablespace lz\_space2 DATAFILE '/home/oracle/app/oracle/oradata/orcl/lzpdb/lz\_space2.bdf' SIZE 50M AUTOEXTEND ON NEXT 5M MAXSIZE 100M;  create tablespace lz\_space3 DATAFILE '/home/oracle/app/oracle/oradata/orcl/lzpdb/lz\_space3.bdf' SIZE 50M AUTOEXTEND ON NEXT 5M MAXSIZE 100M;  create tablespace lz\_space4 DATAFILE '/home/oracle/app/oracle/oradata/orcl/lzpdb/lz\_space4.bdf' SIZE 50M AUTOEXTEND ON NEXT 5M MAXSIZE 100M; |



### 5.3 创建表以及将orders表进行分区

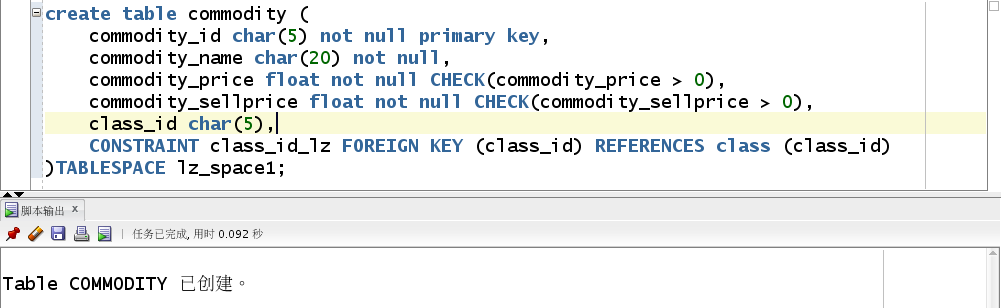
（1）创建分类表（class）

|  |
| --- |
| create table class (  class\_id char(5) not null primary key,  class\_name char(10) not null,  class\_headperson char(15) not null  )TABLESPACE lz\_space1; |



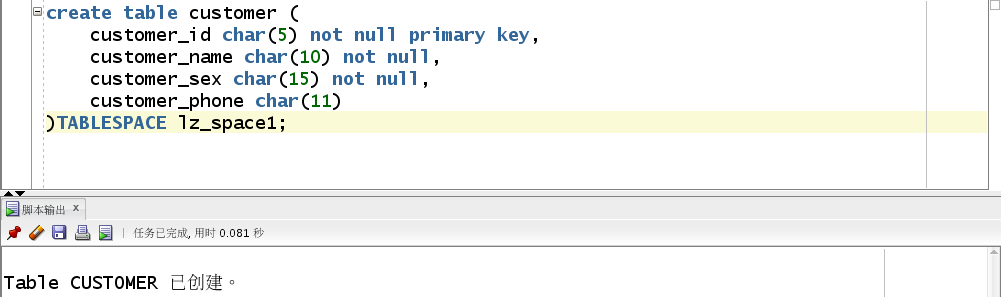
（2）创建商品表（commodity ）

|  |
| --- |
| create table commodity (  commodity\_id char(5) not null primary key,  commodity\_name char(20) not null,  commodity\_price float not null CHECK(commodity\_price > 0),  commodity\_sellprice float not null CHECK(commodity\_sellprice > 0),  class\_id char(5),  CONSTRAINT class\_id\_lz FOREIGN KEY (class\_id) REFERENCES class (class\_id)  )TABLESPACE lz\_space1; |



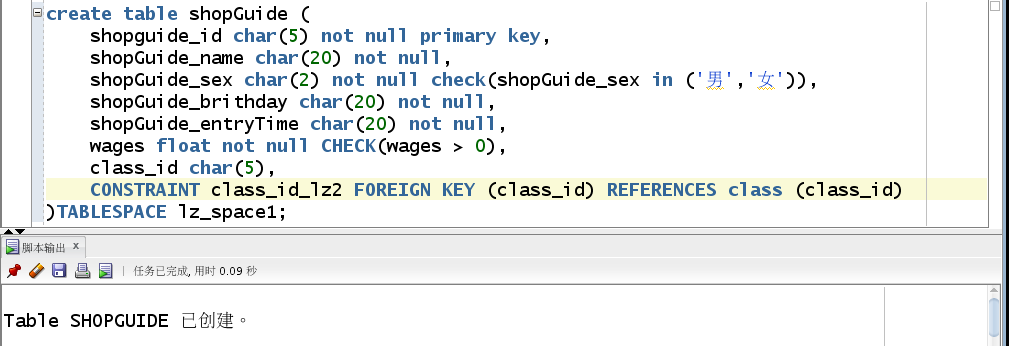
（3）创建商品表（customer）

|  |
| --- |
| create table customer (  customer\_id char(5) not null primary key,  customer\_name char(10) not null,  customer\_sex char(15) not null,  customer\_phone char(11)  )TABLESPACE lz\_space1; |



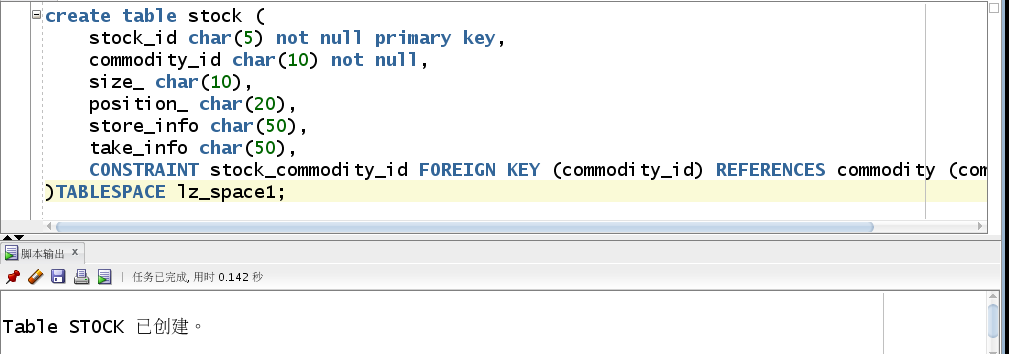
（4）创建导购员表（customer）

|  |
| --- |
| create table shopGuide (  shopguide\_id char(5) not null primary key,  shopGuide\_name char(20) not null,  shopGuide\_sex char(2) not null check(shopGuide\_sex in ('男','女')),  shopGuide\_brithday char(20) not null,  shopGuide\_entryTime char(20) not null,  wages float not null CHECK(wages > 0),  class\_id char(5),  CONSTRAINT class\_id\_lz2 FOREIGN KEY (class\_id) REFERENCES class (class\_id)  )TABLESPACE lz\_space1; |



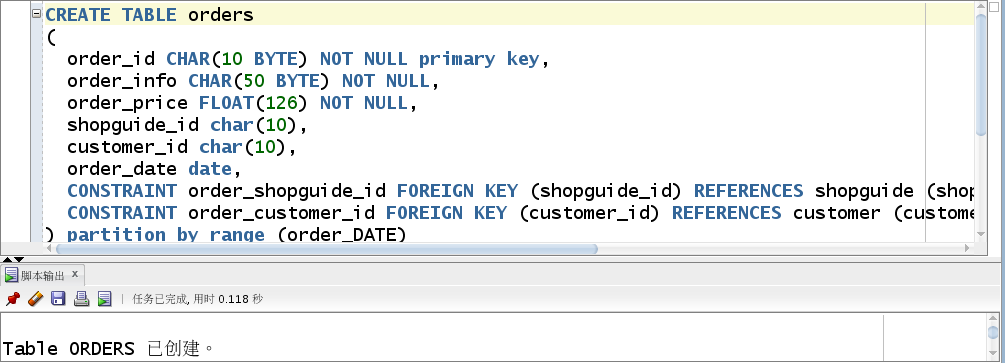
（5）创建库存表（stock）

|  |
| --- |
| create table stock (  stock\_id char(5) not null primary key,  commodity\_id char(10) not null,  size\_ char(10),  position\_ char(20),  store\_info char(50),  take\_info char(50),  CONSTRAINT stock\_commodity\_id FOREIGN KEY (commodity\_id) REFERENCES commodity (commodity\_id)  )TABLESPACE lz\_space1; |



（6）创建订单表（stock），将订单的时间（order\_date），进行分区，分别将2018年之前的数据放在表空间lz\_space1, 将2019 到 2018年的数据放在了表空间lz\_space2，将2020年 到 2019年的数据放在了表空间lz\_space3,最后将大于2020年的数据放在了lz\_spcae4里面。

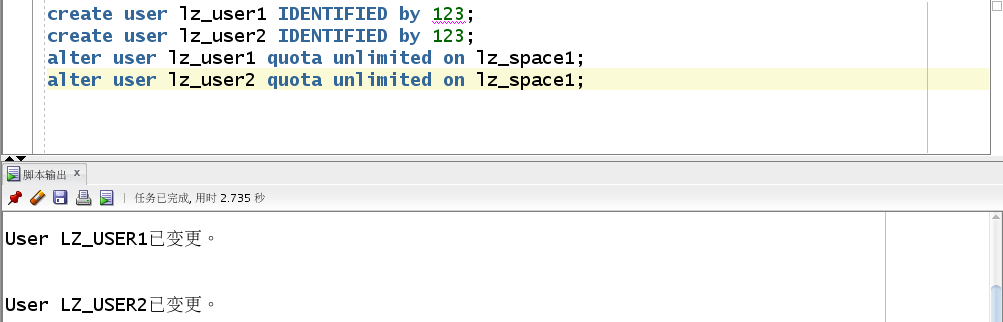
|  |
| --- |
| CREATE TABLE orders  (  order\_id CHAR(10 BYTE) NOT NULL primary key,  order\_info CHAR(50 BYTE) NOT NULL,  order\_price FLOAT(126) NOT NULL,  shopguide\_id char(10),  customer\_id char(10),  order\_date date,  CONSTRAINT order\_shopguide\_id FOREIGN KEY (shopguide\_id) REFERENCES shopguide (shopguide\_id),  CONSTRAINT order\_customer\_id FOREIGN KEY (customer\_id) REFERENCES customer (customer\_id)  ) partition by range (order\_DATE)  (  partition p1 values LESS THAN (TO\_DATE(' 2018-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN')) tablespace lz\_space1,  partition p2 values LESS THAN (TO\_DATE(' 2019-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN')) tablespace lz\_space2,  partition p3 values LESS THAN (TO\_DATE(' 2020-01-01 00:00:00', 'SYYYY-MM-DD HH24:MI:SS', 'NLS\_CALENDAR=GREGORIAN')) tablespace lz\_space3,    partition p4 values LESS THAN (MAXVALUE) tablespace lz\_space4) |



### 5.3 创建用户,授权,插入数据，以及对分区，授权的验证

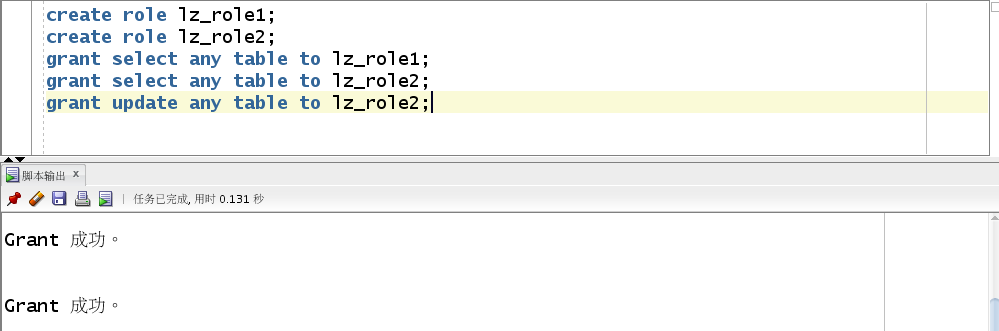
（1）分别创建两个用户lz\_user1和lz\_user2,并允许他们是使用lz\_space1表空间

|  |
| --- |
| create user lz\_user1 IDENTIFIED by 123;  create user lz\_user2 IDENTIFIED by 123;  alter user lz\_user1 quota unlimited on lz\_space1;  alter user lz\_user2 quota unlimited on lz\_space1; |



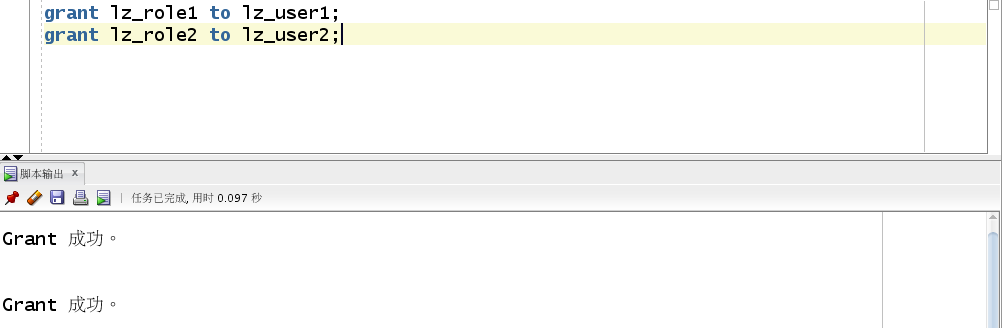
1. 创建两个角色lz\_role1，lz\_role2，并分别赋予lz\_role1读任何表和读，lz\_role2读任何表和读、修改任何表的权利；

|  |
| --- |
| create role lz\_role1;  create role lz\_role2;  grant select any table to lz\_role1;  grant select any table to lz\_role2;  grant update any table to lz\_role2; |



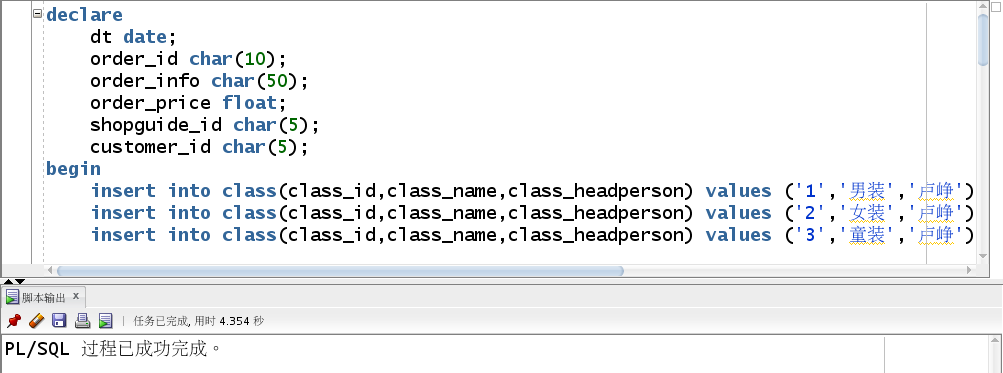
（3）将lz\_role1，lz\_role2分别赋予给用户lz\_user1，lz\_user2

|  |
| --- |
| grant lz\_role1 to lz\_user1;  grant lz\_role2 to lz\_user2; |



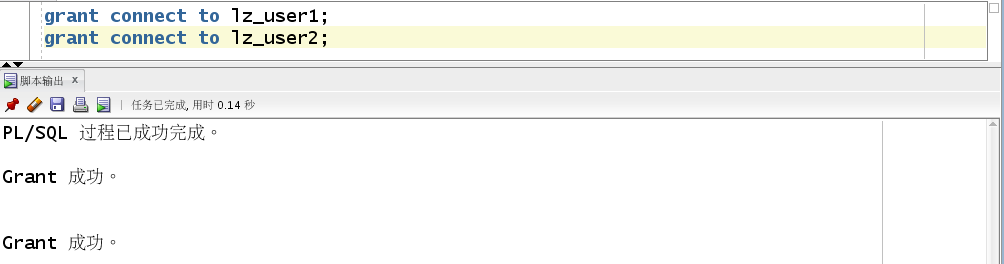
（4）向表中插入数据：

|  |
| --- |
| declare  dt date;  order\_id char(10);  order\_info char(50);  order\_price float;  shopguide\_id char(5);  customer\_id char(5);  begin  insert into class(class\_id,class\_name,class\_headperson) values ('1','男装','卢峥');  insert into class(class\_id,class\_name,class\_headperson) values ('2','女装','卢峥');  insert into class(class\_id,class\_name,class\_headperson) values ('3','童装','卢峥');    Insert into commodity(commodity\_id,commodity\_name,commodity\_price,commodity\_sellprice,class\_id) values ('101','男上衣',20,30,'1');  insert into commodity(commodity\_id,commodity\_name,commodity\_price,commodity\_sellprice,class\_id) values ('102','男裤子',15,25,'1');  insert into commodity(commodity\_id,commodity\_name,commodity\_price,commodity\_sellprice,class\_id) values ('103','女上衣',15,25,'2');  insert into commodity(commodity\_id,commodity\_name,commodity\_price,commodity\_sellprice,class\_id) values ('104','女裤子',25,35,'2');  insert into commodity(commodity\_id,commodity\_name,commodity\_price,commodity\_sellprice,class\_id) values ('105','儿童上衣',10,20,'3');  insert into commodity(commodity\_id,commodity\_name,commodity\_price,commodity\_sellprice,class\_id) values ('106','儿童裤子',10,20,'3');    insert into shopguide(shopguide\_id,shopguide\_name,shopguide\_brithday,shopguide\_sex,shopguide\_entrytime,wages,class\_id)  values ('1','卢峥','0505','男','0101',1000,'1');    insert into customer(customer\_id,customer\_name,customer\_sex,customer\_phone) values ('1','卢峥','男','15516102919');    for i in 1..50000  loop  if i mod 3=0 then  dt := to\_date('2018-01-01','yyyy-mm-dd')+(i mod 60);  --PARTITION\_2018  elsif i mod 6 = 1 then  dt := to\_date('2019-01-01','yyyy-mm-dd')+(i mod 60);  --PARTITON\_2019  elsif i mod 6 = 2 then  dt := to\_date('2020-01-01','yyyy-mm-dd')+(i mod 60);  --PARTITION\_2020  end if;  --开始向order表里插入数据  order\_id := i;  customer\_id := '1';  shopguide\_id := '1';  order\_info := '订单详情';  order\_price := dbms\_random.value(5,30);    insert into orders (order\_id,order\_info,order\_price,customer\_id,shopguide\_id)  values (order\_id,order\_info,order\_price,customer\_id,shopguide\_id);    end loop;  end; |

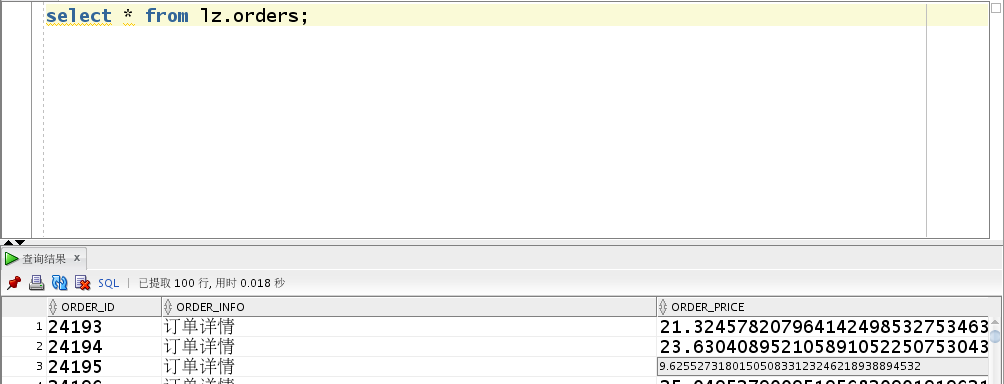


（5）之前创建了两个用户，一个是lz\_user1,一个是lz\_user2，并且给lz\_user1和lz\_user2分别授予了查看表，查看表以及更新表的权限，接下来进行验证：

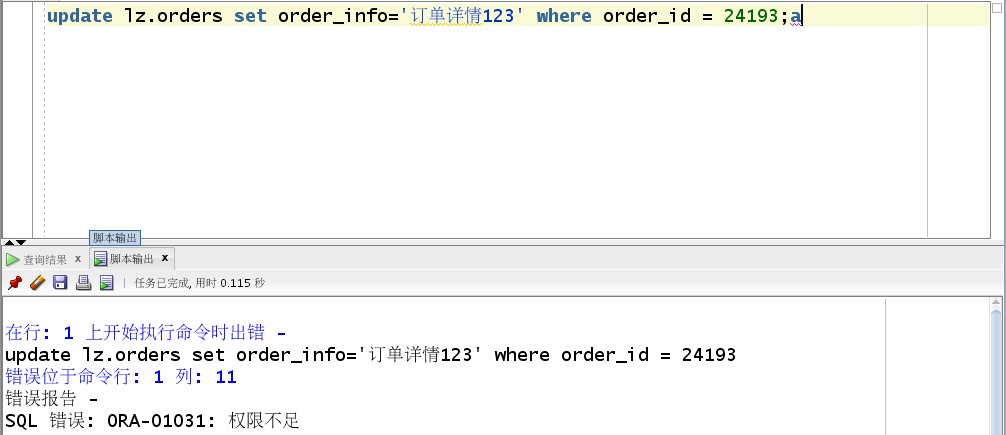
|  |
| --- |
| grant connect to lz\_user1;  grant connect to lz\_user2; |



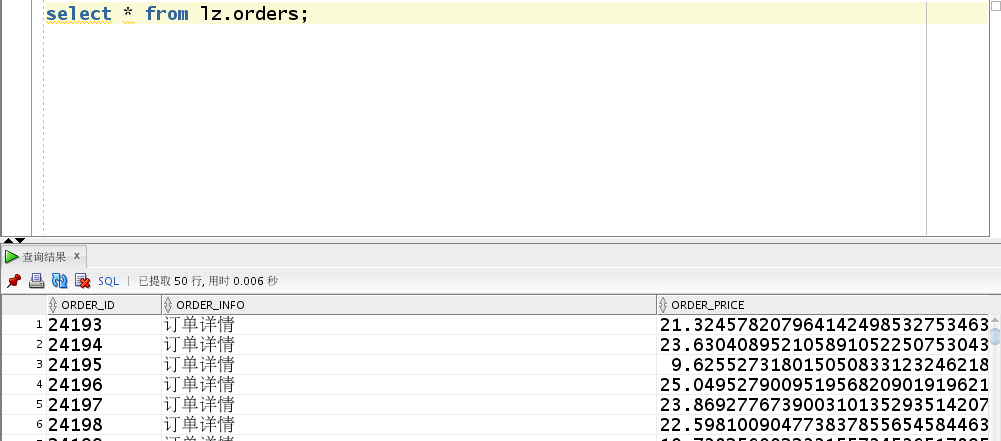
首先，让lz\_user1连接数据库lzpdb,让其查看lz的ORDER表，如下图所示，是可以看见的。

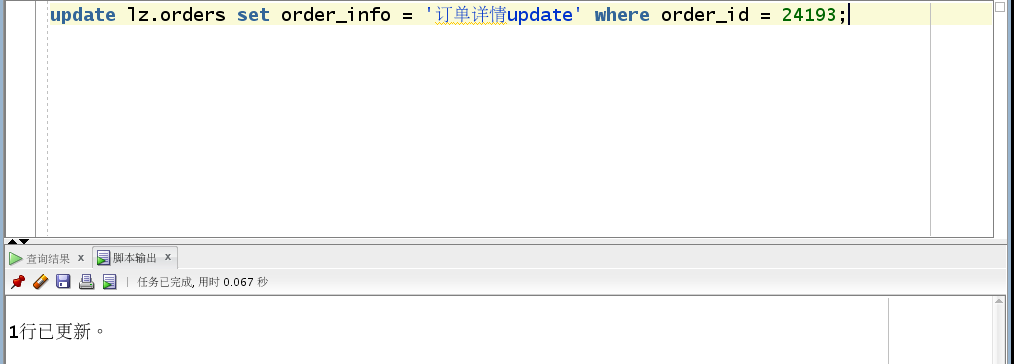


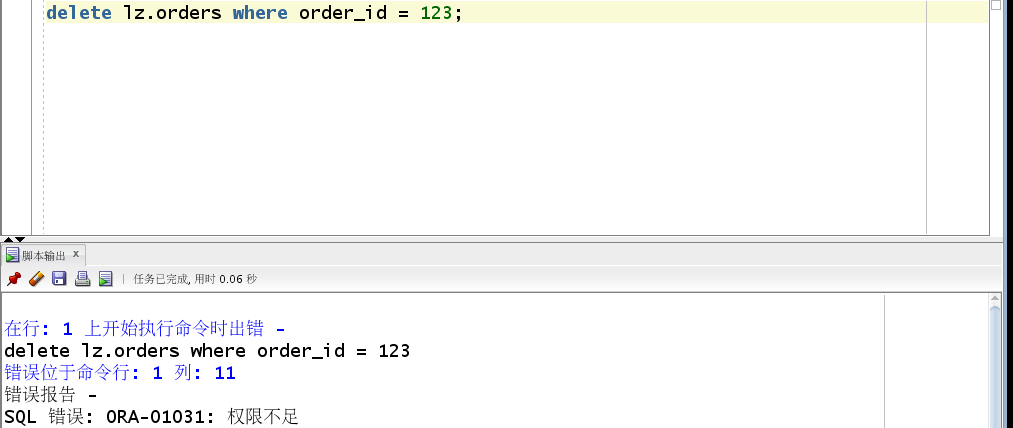
然后，让lz\_user1更新一条数据，会发现其权限不足，由此可以证明，授权成功。



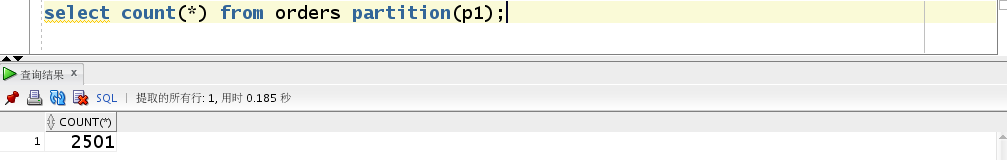
接着对lz\_user2(之前赋予了更新和查看权限）用户查看和更新lz的ORDERS表，发现均成功，但让其删除一条记录是会显示权限不足，由此可证明，授权无误。

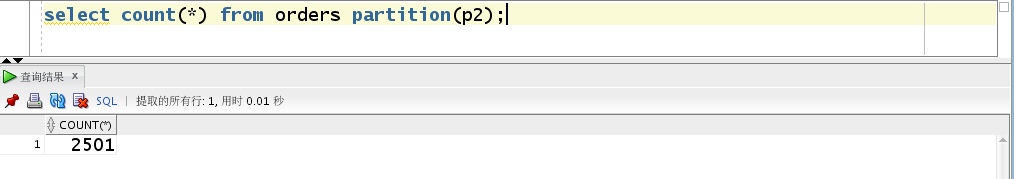


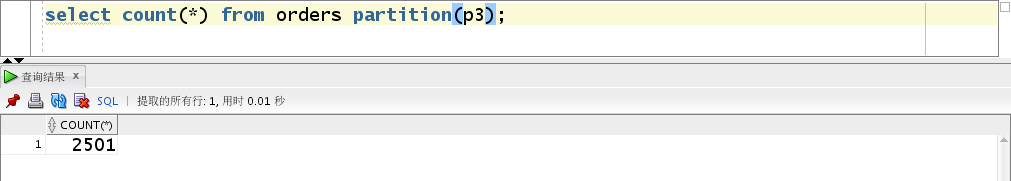


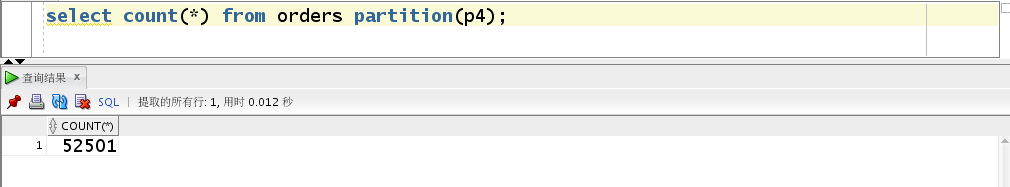


1. 根据REGISTRATION\_FORM表中的REGISTRATION\_DATE字段根据时间段的不同，将其放在了不同的表空间里面，由下图可知，每个表空间里面均有REGISTRATION\_FORM的数据，也可证明分区成功。





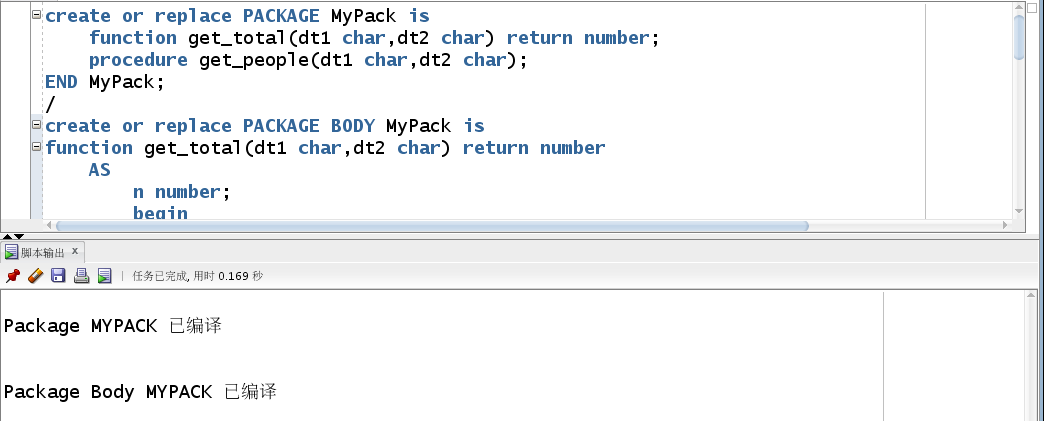




## 程序包的建立

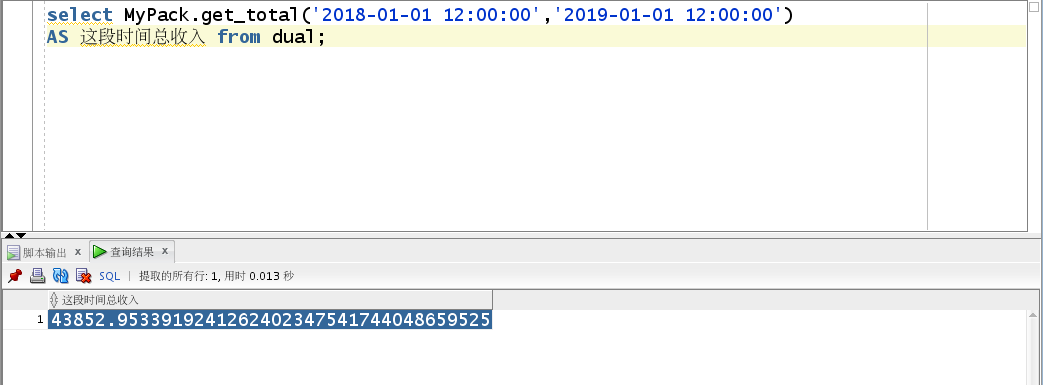
建立一个程序包，里面有一个函数和一个存储过程，其功能分别为：统计一段时间内的订单总收入；统计一段时间内，各个导购员的接待数。具体过程如下：

|  |
| --- |
| create or replace PACKAGE MyPack is  function get\_total(dt1 char,dt2 char) return number;  procedure get\_people(dt1 char,dt2 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(order\_price) into n from orders where order\_date >=to\_date(dt1,'yyyy-mm-dd hh24:mi:ss')  and order\_date<=to\_date(dt2,'yyyy-mm-dd hh24:mi:ss');  return n;  end;  procedure get\_people(dt1 char,dt2 char)  AS  a number;  b number;  cursor people is  select \* from orders where order\_date>= to\_date(dt1,'yyyy-mm-dd hh24:mi:ss')  and order\_date<= to\_date(dt2,'yyyy-mm-dd hh24:mi:ss');  begin  a:=0;  b:=0;  for v in people  loop  if v.shopguide\_id = '1'  then a:= a+1;  elsif v.shopguide\_id = '2'  then b:= b+1;  end if;  end loop;  dbms\_output.put\_line('导购员卢峥接待客户数为:'|| a);  dbms\_output.put\_line('导购员pp接待客户数为:'||b);  end;  END MyPack; |

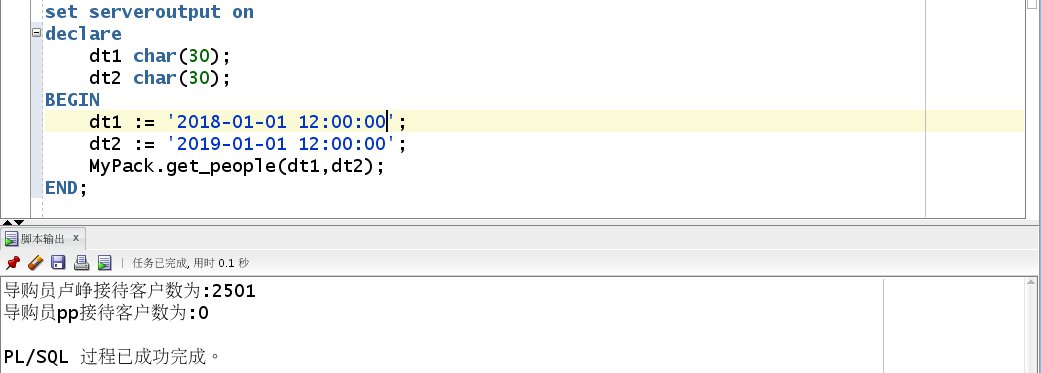


接下来是效果展示：

函数效果展示



存储过程效果展示

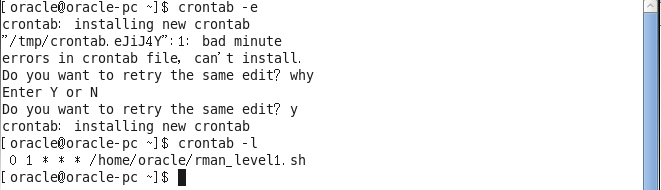


## 系统定时自动备份

（1）编写rman增量备份脚本

|  |
| --- |
| #rman\_level1.sh  #!/bin/sh  export NLS\_LANG='SIMPLIFIED CHINESE\_CHINA.AL32UTF8'  export ORACLE\_HOME=/home/oracle/app/oracle/product/12.1.0/dbhome\_1  export ORACLE\_SID=orcl  export PATH=$ORACLE\_HOME/bin:$PATH  rman target / nocatalog msglog=/home/oracle/rman\_backup/lv1\_`date +%Y%m%d-%H%M%S`\_L0.log << EOF  run{  configure retention policy to redundancy 1;  configure controlfile autobackup on;  configure controlfile autobackup format for device type disk to '/home/oracle/rman\_backup/%F';  configure default device type to disk;  crosscheck backup;  crosscheck archivelog all;  allocate channel c1 device type disk;  backup as compressed backupset incremental level 1 database format '/home/oracle/rman\_backup/dblv1\_%d\_%T\_%U.bak'  plus archivelog format '/home/oracle/rman\_backup/arclv1\_%d\_%T\_%U.bak';  report obsolete;  delete noprompt obsolete;  delete noprompt expired backup;  delete noprompt expired archivelog all;  release channel c1;  }  EOF  exit |

（2）启动linux的crontab定时任务，每天的凌晨一点自动进行备份



1. 演示备份与恢复：

（3.1）执行rman\_level1.sh脚本，进行数据库备份



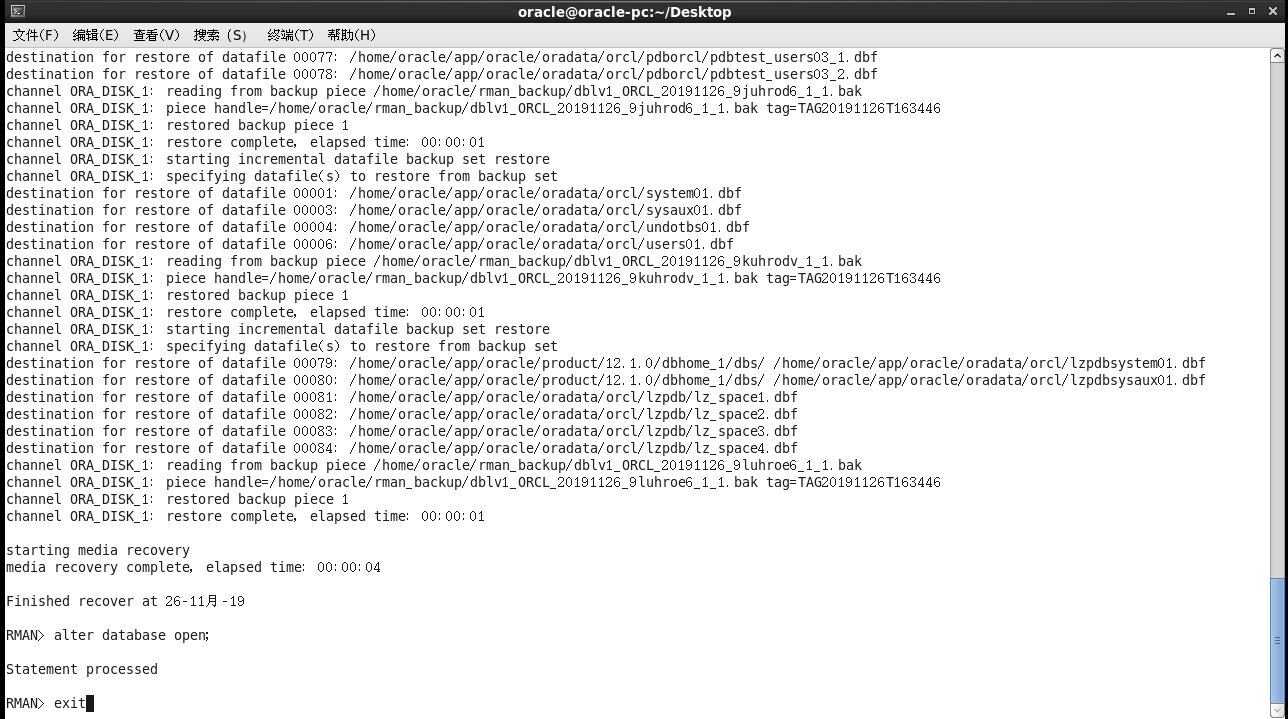
（3.2）破坏数据库





（3.3）启用rman，进行数据库恢复





（3.4）查看删除的是否恢复



（3.5）恢复成功