

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的购物商城的数据库设计 | | |
| 课程 | Oracle数据库应用 | | |
| 学 院 | 计算机学院 | | |
| 专 业 | 软件工程 | 年级 | 2018级 |
| 学生姓名 | 干文浩 | 学号 | 201810414410 |
| 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

目录

[一. 概述 3](#_Toc74685932)

[二. 需求与功能分析 3](#_Toc74685933)

[三. 数据库设计 3](#_Toc74685934)

[1. 添加用户及权限管理 3](#_Toc74685935)

[2. 创建数据表，添加数据 4](#_Toc74685936)

[3. PL/SQL设计 13](#_Toc74685937)

[4. 设计自动备份方案或则手工备份方案 19](#_Toc74685939)

*4.1 ORACLE数据库备份与恢复详解* [19](#_Toc74685940)

[4.2从pdborcl创建可插接数据库 20](#_Toc74685942)

[4.3查询需要备份的文件 26](#_Toc74685943)

[4.4查询自己创建的表空间的大小 26](#_Toc74685945)

[4.6 更改数据，删除数据库文件模拟数据损坏 29](#_Toc74685946)

[4.7 数据库恢复 31](#_Toc74685947)

## 一. 概述

随着这些年电脑计算机的速度质的提高，成本的下降，IT互联网大众趋势的发展。我们使用电脑的高效率才处理数据信息成为可能。购物商城更是当今互联网时代最大的一个亮点之一，网上购物，让足不出户，观看、挑选、购买各种商品成为可能。购物商城，数据量复杂且庞大不仅需要细致的统计各种数据，更需要记录许多操作信息，通过这个系统，可以做到信息的规范处理，科学统计和快速的查询，从而减少管理方面的工作量。毋庸置疑，切实有效的把计算机管理引入学校教务管理中，对于促进学校管理制度，提高学校教学质量与办学水平有着显著意义。

购物商城是现在非常流行的一个商业化系统。需要统计的信息多，分类复杂，关系紧密且涉及各种其他信息，整合难度大。人工管理信息的难度也越来越大，显然是不能满足实际的需要，效率也是很低的。并且这种传统的方式存在着众多的弊端，如：保密性差.查询不便.效率低，很难维护和更新等，然而，本系统针对以上的缺点能够极大的提高学生信息管理的效率，也是科学化.正规化的管理，与世界接轨的重要条件。所以如何自动高效地管理信息是这些年来许多人所研究的。

## 二. 需求与功能分析

购物商城系统，可用于购物商城方对商城物品、用户、商家、订单等信息进行管理。查询，更新与维护，使用方便，易用性强。该系统实现的大致功能；用户登录。提供了商品的查询，分类搜索，热门推荐等功能；商品购买，记录购买商品信息，记录订单信息。管理员管理拥有最高的权限。允许添加商家和维护系统信息等。

## 三. 数据库设计

## 1. 添加用户及权限管理

oracle中的表就是一张存储数据的表。表空间是逻辑上的划分。方便管理的。

数据表空间：存放数据总是需要空间， Oracle把一个数据库按功能划分若干空间来保存数据。当然数据存放在磁盘最终是以文件形式，所以一盘一个数据表空间包含一个以上的物理文件数据表。

1.1创建表空间：

create tablespace gwh\_01

datafile './gwh\_01.dbf ' size 50M

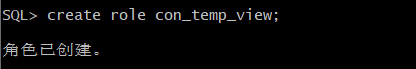
create tablespace gwh\_02

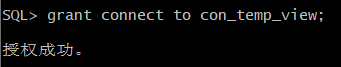
datafile './gwh\_02.dbf ' size 50M

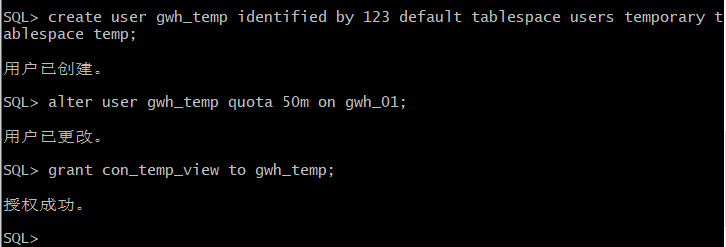
创建两个表空间，gwh\_01和gwh\_02，空间大小未50m

1.2创建角色与用户

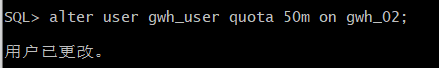
创建con\_temp\_view 角色，只拥有连接查看数据库的权限。







创建管理员角色，拥有创建表、视图、存储过程、存储函数等权限，将表空间gwh\_02赋予角色。



## 2. 创建数据表，添加数据

-- 商家表

create table "business" (

"business\_id" NUMBER(6,0),

"bus\_name" VARCHAR2(10 BYTE),

"bus\_gender" VARCHAR2(4 BYTE),

"bus\_age" NUMBER(2,0),

CONSTRAINT ORDERS\_PK PRIMARY KEY (business\_id)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

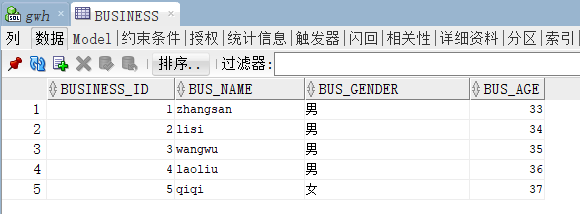
NOCOMPRESS NOLOGGING

STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "GWH\_02";



-- 商品表

create table "product" (

"product\_id" NUMBER(6,0),

"product\_name" VARCHAR2(40 BYTE),

"product\_content" VARCHAR2(255 BYTE),

"start\_date" DATE,

"business\_id" NUMBER(6,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS NOLOGGING

STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "GWH\_02";

-- 商品详情表

create table "product\_details" (

"pro\_detail\_id" NUMBER(6,0),

"pro\_type" VARCHAR2(40 BYTE),

"discount" NUMBER(8,2),

"pro\_number" NUMBER(8,0),

"price" NUMBER(8,0),

"start\_date" DATE,

"product\_id" NUMBER(6,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS NOLOGGING

STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "GWH\_02";

脚本数据插入10000条

-- 数据插入

--create sequence user\_sequence

-- minvalue 1

-- maxvalue 30000

-- increment by 1

-- start **with** 1

-- nocache;

declare

dt date;

bus\_id number;

pro\_name varchar2(40);

pro\_content varchar2(40);

p number;

pro\_type varchar2(40);

discount number;

num number;

begin

-- **delete** from order\_details;

-- **delete** from orders;

-- isbn := 1;

**for** i **in** 1..10000

loop

**if** i mod 6 =0 then

dt:=to\_date('2015-3-2','yyyy-mm-dd')+(i mod 60);

elsif i mod 6 =1 then

dt:=to\_date('2016-3-2','yyyy-mm-dd')+(i mod 60);

elsif i mod 6 =2 then

dt:=to\_date('2017-3-2','yyyy-mm-dd')+(i mod 60);

elsif i mod 6 =3 then

dt:=to\_date('2018-3-2','yyyy-mm-dd')+(i mod 60);

elsif i mod 6 =4 then

dt:=to\_date('2019-3-2','yyyy-mm-dd')+(i mod 60);

**else**

dt:=to\_date('2020-3-2','yyyy-mm-dd')+(i mod 60);

end **if**;

-- 商品归属商家id

bus\_id:=CASE i MOD 6

WHEN 0 THEN 1

WHEN 1 THEN 2

WHEN 2 THEN 3

WHEN 3 THEN 4

WHEN 4 THEN 5 END;

-- select isbn into isbn from (select \* from book order by dbms\_random.value) where rownum <2;

pro\_name := CASE i MOD 6

WHEN 0 THEN concat('电脑\_', i)

WHEN 1 THEN concat('手机\_', i)

WHEN 2 THEN concat('冰箱\_', i)

WHEN 3 THEN concat('打火机\_', i)

WHEN 4 THEN concat('耳机\_', i)

**else** concat('电视\_', i) END;

pro\_content := concat('这是', pro\_name);

insert into PRODUCT(PRODUCT\_ID, PRODUCT\_NAME, PRODUCT\_CONTENT, START\_DATE, business\_id)

values (i, pro\_name, pro\_content, dt, bus\_id);

-- 插入一个商品包括3个商品种类

**for** j **in** 1..3

loop

p:=dbms\_random.value(10000,4000);

pro\_type := pro\_name|| '\_' ||(j mod 3 + 1);

discount := dbms\_random.value(0,1);

num := dbms\_random.value(100,1000);

INSERT INTO PRODUCT\_DETAILS(PRO\_DETAIL\_ID, PRO\_TYPE ,DISCOUNT, PRO\_NUMBER ,PRICE, START\_DATE, PRODUCT\_ID)

values (user\_sequence.nextval, pro\_type, discount, num, p, dt, i);

end loop;

IF i MOD 1000 =0 THEN

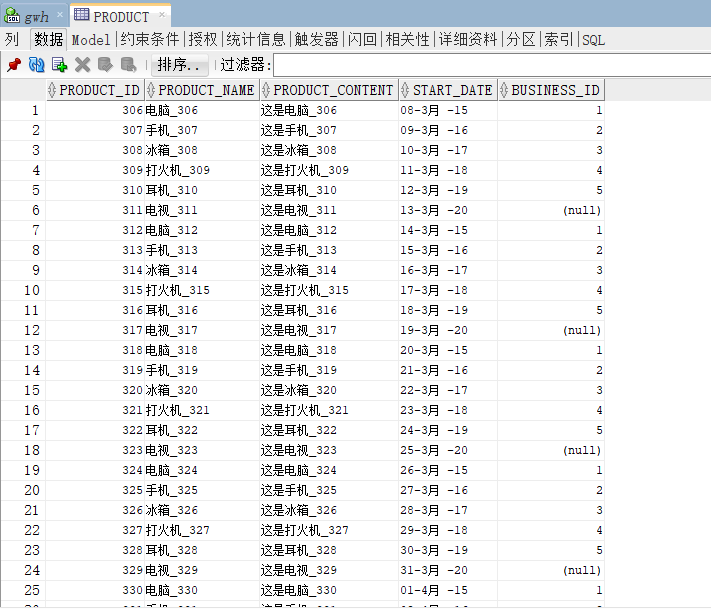
commit; --每次提交会加快插入数据的速度

END IF;

end loop;

end;

/



商品数量1万条

商品种类3万条



-- 用户表

create table "consumers" (

"consumer\_id" NUMBER(6,0),

"consumer\_name" varchar2(40 **byte**),

"consumer\_gender" varchar2(2 **byte**),

"consumer\_age" NUMBER(6,0),

"consumer\_tel" varchar2(11 **byte**)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS NOLOGGING

STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "GWH\_02";

-- 订单表

create table "pro\_order" (

"order\_id" NUMBER(6,0),

"create\_date" DATE,

"states" VARCHAR2(10 BYTE),

"customer\_address" VARCHAR2(40 BYTE),

"customer\_name" VARCHAR2(20 BYTE),

"customer\_tel" VARCHAR2(20 BYTE),

"pro\_detail\_id" NUMBER(6,0),

"user\_id" NUMBER(6,0)

) SEGMENT CREATION IMMEDIATE

PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255

NOCOMPRESS NOLOGGING

STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS 2147483645

PCTINCREASE 0 FREELISTS 1 FREELIST GROUPS 1

BUFFER\_POOL DEFAULT FLASH\_CACHE DEFAULT CELL\_FLASH\_CACHE DEFAULT)

TABLESPACE "GWH\_02";

订单表脚本数据插入

-- 数据插入

declare

create\_date date;

s varchar2(10);

address varchar2(40);

m varchar2(40);

tel varchar2(40);

detail\_id number;

consumer\_id number;

consumer\_tel varchar2(40);

consumer\_name varchar2(40);

begin

**for** i **in** 1..10000

loop

-- 商品归属商家id

s:=CASE i MOD 6

WHEN 0 THEN '已付款'

WHEN 1 THEN '已发货'

WHEN 2 THEN '已收货'

WHEN 3 THEN '已退货'

WHEN 4 THEN '待付款' END;

select pro\_detail\_id,start\_date into detail\_id,create\_date from (select \* from PRODUCT\_DETAILS order by dbms\_random.value) where rownum <2;

select consumer\_id,consumer\_tel,consumer\_name into consumer\_id,consumer\_tel,consumer\_name from (select \* from consumers order by dbms\_random.value) where rownum <2;

address := CASE i MOD 6

WHEN 0 THEN concat('四川省乐山市', i)

WHEN 1 THEN concat('四川省成都市', i)

WHEN 2 THEN concat('重庆市', i)

WHEN 3 THEN concat('北京市', i)

WHEN 4 THEN concat('河南市', i)

**else** concat('老挝', i) END;

insert into PRO\_ORDER(order\_id, create\_date, states, customer\_address, customer\_name, customer\_tel, pro\_detail\_id, user\_id)

values (i, create\_date, s, address, consumer\_name, consumer\_tel, detail\_id, consumer\_id);

IF i MOD 1000 =0 THEN

commit; --每次提交会加快插入数据的速度

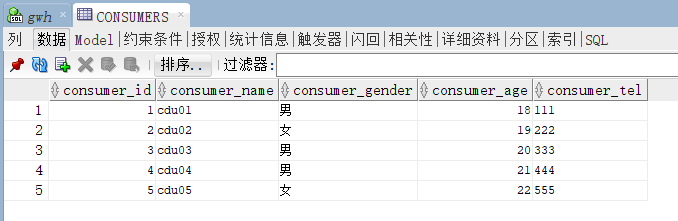
END IF;

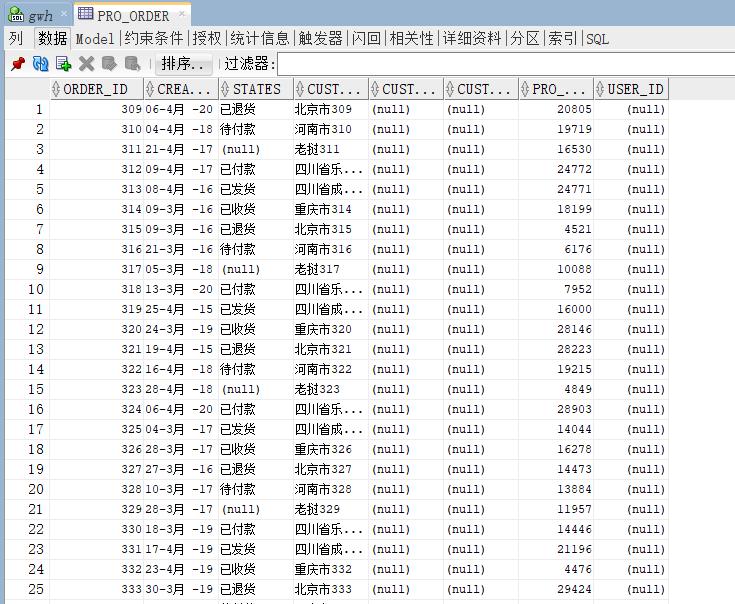
end loop;

end;

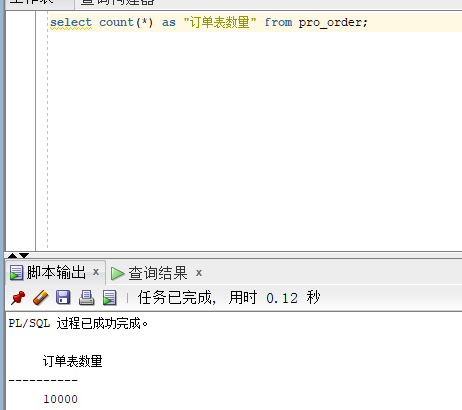
/

select count(\*) as "订单表数量" from pro\_order;





1万条订单表数据



商家表

|  |  |  |
| --- | --- | --- |
| 属性 | 字段 | 注解 |
| 编号 | business\_id | 商家编号 |
| 姓名 | bus\_name | 商家姓名 |
| 性别 | bus\_gender | 商家编号 |
| 年龄 | bus\_age | 商家年龄 |

商品表

|  |  |  |
| --- | --- | --- |
| 属性 | 字段 | 注解 |
| 商品编号 | product\_id | 商品编号 |
| 商品名称 | product\_name | 商品姓名 |
| 商品描述 | product\_content | 商品描述 |
| 创建日期  商家编号 | start\_date  business\_id | 创建日期  商家编号 |

商品详情表

|  |  |  |
| --- | --- | --- |
| 属性 | 字段 | 注解 |
| 商品详情编号 | pro\_detail\_id | 商品详情编号 |
| 商品类型 | pro\_type | 商品类型 |
| 折扣 | discount | 折扣 |
| 商品数量 | number | 商品数量 |
| 商品价格  创建时间  商品编号 | Price  start\_date  product\_id | 商品价格  创建时间  商品编号 |

用户表

|  |  |  |
| --- | --- | --- |
| 属性 | 字段 | 注解 |
| 编号 | business\_id | 用户编号 |
| 姓名 | bus\_name | 用户姓名 |
| 性别 | bus\_gender | 用户编号 |
| 年龄 | bus\_age | 用户年龄 |
| 电话 | Bus\_tel | 电话 |

订单表

|  |  |  |
| --- | --- | --- |
| 属性 | 字段 | 注解 |
| 订单编号 | order\_id | 订单编号 |
| 创建时间  状态 | create\_date  states | 创建时间  状态 |
| 收获地址 | customer\_address | 收获地址 |
| 收货人 | customer\_name | 收货人 |
| 收获手机号  商品详情编号  用户编号 | customer\_tel  pro\_detail\_id  user\_id | 收获手机号  商品详情编号  用户编号 |

## 

## 3. PL/SQL设计

过程和函数由以下4部分：

* 签名或头
* 关键字IS或AS
* 局部声明（可选）
* BEGIN和END之间的过程体（包括异常处理程序）

几点说明：

1、参数没有指定长度，当有实际数据传递进来的时候，参数的长度才被确定。

2、局部声明为：actual\_line varchar2(150);

3、使用命令SQL> show errors在SQLPLUS里面查看错误。

**3.1创建程序包Exam**

create or replace PACKAGE Exam IS

PROCEDURE bus\_pros;

**function** hot RETURN SYS\_REFCURSOR;

FUNCTION recommend RETURN SYS\_REFCURSOR;

PROCEDURE Expenditure\_ranking(startDate date,endDate date);

FUNCTION popular\_product(startDate date,endDate date)

RETURN SYS\_REFCURSOR;

END Exam;

/

**3.2 程序包体**

**3.2.1、商家拥有总商品数排名，储存过程。**

计算每个商家拥有的各种的商品总数，进行从大到小的排名。

create or replace PACKAGE BODY Exam IS

PROCEDURE bus\_pros

AS

LEFTSPACE VARCHAR(200);

begin

LEFTSPACE:=' ';

DBMS\_OUTPUT.PUT\_LINE('商家 商品总数');

DBMS\_OUTPUT.PUT\_LINE('=======================');

**for** v **in**

(select b.bus\_name,count(\*) as num from business b left join

product p on p.business\_id = b.business\_id

left join product\_details pd on p.product\_id = pd.product\_id

group by b.bus\_name order by count(\*) desc)

LOOP

DBMS\_OUTPUT.PUT\_LINE(LPAD(LEFTSPACE,v.bus\_name||' '||v.num));

END LOOP;

END;

END Exam;

/

## 

**3.2.2、最热门商品 、最推荐商品展示，储存函数**

根据订单表数据，统计用户下单购买的商品（status不为3、4，表示用户已付款购买），展示为最热门的商品，对用户收藏最多的商品（status为0），展示为最推荐的商品。

create or replace PACKAGE BODY Exam IS

FUNCTION hot

RETURN SYS\_REFCURSOR is

type\_cur SYS\_REFCURSOR;

BEGIN

open type\_cur **for** select p.product\_name,pd.pro\_type,o."次数" from

(select pro\_detail\_id,count(\*) as "次数" from pro\_order

where status != '已退款' and status != '未付款' group by pro\_detail\_id

order by count(\*) desc ) o left join product\_details pd

on pd.pro\_detail\_id = o.pro\_detail\_id

left join product p on p.product\_id = pd.product\_id

where rownum < 2;

**return** type\_cur;

END;

FUNCTION recommend

RETURN SYS\_REFCURSOR is

type\_cur SYS\_REFCURSOR;

AS

N NUMBER(20,2);

BEGIN

open type\_cur **for** select p.product\_name,pd.pro\_type,o."次数" from

(select pro\_detail\_id,count(\*) as "次数" from pro\_order

where status = '待付款' group by pro\_detail\_id

order by count(\*) desc ) o left join product\_details pd

on pd.pro\_detail\_id = o.pro\_detail\_id

left join product p on p.product\_id = pd.product\_id

where rownum < 2;

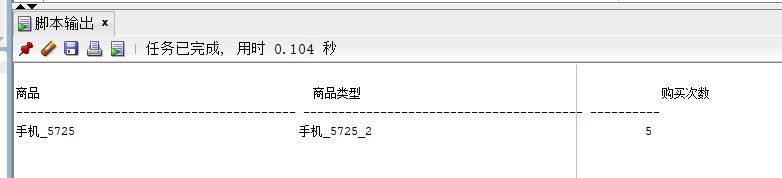
**return** type\_cur;

END;

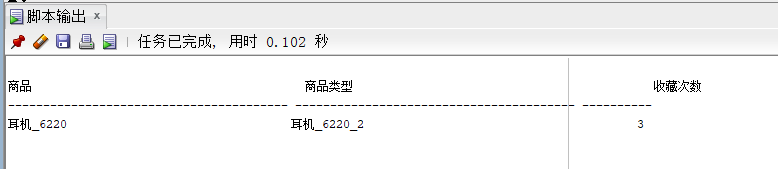
END Exam;

/

热门商品



推荐商品



**3.2.3、一段时间内用户购买商品花销排行，储存过程**

创建存储过程，在存储过程中使用游标，根据传入参数startDate和endDate，开始和结束时间。统计每个用户在该段时间内购买商品的花销，进行排名。信息用dbms\_output包中的put\_line函数输出。

create or replace PACKAGE BODY Exam IS

PROCEDURE Expenditure\_ranking(startDate date,endDate date)

AS

LEFTSPACE VARCHAR(2000);

begin

LEFTSPACE:=' ';

DBMS\_OUTPUT.PUT\_LINE(' 起始时间：' || startDate || ',结束时间' || endDate);

DBMS\_OUTPUT.PUT\_LINE('用户 花费金额');

DECLARE cursor cur\_info is

select consumer\_name,n.total from consumers u left join

(select user\_id,count(price\*discount) as total from pro\_order po left join

product\_details pd on po.pro\_detail\_id = pd.pro\_detail\_id

where po.create\_date >= startDate and po.create\_date <= endDate

group by user\_id) n

on u.consumer\_id = n.user\_id;

v\_info cur\_info%ROWTYPE;

begin open cur\_info;

loop

fetch cur\_info into v\_info;

exit when cur\_info %notfound;

dbms\_output.put\_line(LPAD(' ',(v\_info.user\_name)||' '||v\_info.total));

end loop;

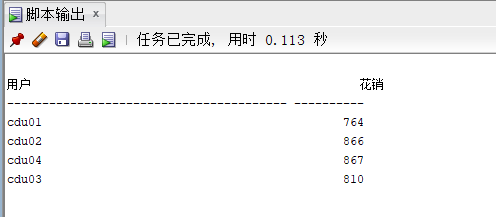
close cur\_info;

end;

END;

END Exam;

/



**3.2.4、一段时间内最受欢迎的商品，储存函数**

创建存储函数，查询最受欢迎的商品，根据传入参数startDate和endDate，开始和结束时间。统计并输出在该段时间内被购买次数最最多的商品及其被购买次数。

create or replace PACKAGE BODY Exam IS

FUNCTION popular\_product(startDate date,endDate date)

RETURN SYS\_REFCURSOR is

type\_cur SYS\_REFCURSOR;

AS

N NUMBER(20,2);

BEGIN

select p.product\_name,pd.pro\_type,o."次数" from

(select pro\_detail\_id,count(\*) as "次数" from pro\_order

where status != '已退款' and status != '未付款'

and create\_date >= startDate and create\_date <= endDate

group by pro\_detail\_id

order by count(\*) desc ) o left join product\_details pd

on pd.pro\_detail\_id = o.pro\_detail\_id

left join product p on p.product\_id = pd.product\_id

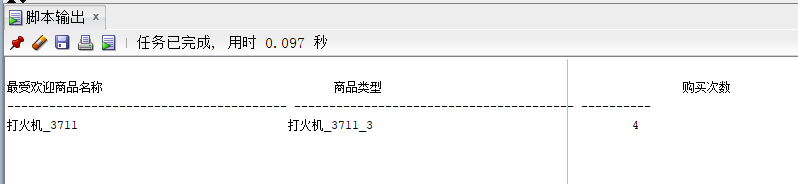
where rownum < 2;

**return** type\_cur;

END;

END Exam;

/



## 4. 设计自动备份方案或则手工备份方案

**4.1** **ORACLE数据库备份与恢复详解**

Oracle的备份与恢复有三种标准的模式，大致分为两 大类，备份恢复(物理上的)以及导入导出(逻辑上的)，而备份恢复又可以根据数据库的工作模式分为非归档模式(Nonarchivelog-style) 和归档模式(Archivelog-style),通常，我们把非归档模式称为冷备份，而相应的把归档模式称为热备份。

### 热备份和冷备份优缺点

#### 热备份的优点是：

1．可在表空间或数据文件级备份，备份时间短。

2．备份时数据库仍可使用。

3．可达到秒级恢复（恢复到某一时间点上）。

4．可对几乎所有数据库实体作恢复。

5．恢复是快速的，在大多数情况下在数据库仍工作时恢复。

#### 热备份的不足是：

1．不能出错，否则后果严重。

2．若热备份不成功，所得结果不可用于时间点的恢复。

3．因难维护，所以要特别仔细小心，不允许“以失败而告终”。

#### 冷备份的优点是：

1．是非常快速的备份方法（只需拷贝文件）

2．容易归档（简单拷贝即可）

3．容易恢复到某个时间点上（只需将文件再拷贝回去）

4．能与归档方法相结合，作数据库“最新状态”的恢复。

5．低度维护，高度安全。

#### 冷备份不足是：

1．单独使用时，只能提供到“某一时间点上”的恢复。

2．在实施备份的全过程中，数据库必须要作备份而不能作其它工作。也就是说，数据库必须是关闭状态。

3．若磁盘空间有限，只能拷贝到磁带等其它外部存储设备上，速度会很慢。

4．不能按表或按用户恢复。

### 物理备份之冷备份：

当数据库可以暂时处于关闭状态时，我们需要将它在这一稳定时刻的数据相关文件转移到安全的区域，当数据库遭到破坏，再从安全区域将备份的数据库相关文件拷 贝回原来的位置，这样，就完成了一次快捷安全等数据转移。由于是在数据库不提供服务的关闭状态，所以称为冷备份。冷备份具有很多优良特性，比如上面图中我 们提到的，快速，方便，以及高效。一次完整的冷备份步骤应该是：

1，首先关闭数据库（shutdown normal）

2，拷贝相关文件到安全区域（利用操作系统命令拷贝数据库的所有的数据文件、日志文件、控制文件、参数文件、口令文件等（包括路径））

3，重新启动数据库（startup）

这样，我们就完成了一次冷备份，请确定你对这些相应的目录（包括写入的目标文件夹）有相应的权限。

恢复的时候，相对比较简单了，我们停掉数据库，将文件拷贝回相应位置，重启数据库就可以了，当然也可以用脚本来完成。

### 4.2从pdborcl创建可插接数据库

$ sqlplus / as sysdba

替代命令：

$ sqlplus sys/123@202.115.82.8/orcl as sysdba

查看创建前的数据库：

show pdbs

--1.关闭pdborcl

ALTER PLUGGABLE DATABASE pdborcl CLOSE immediate;

--或者先切换到pdborcl：

alter session set container=pdborcl;

shutdown immediate

--切换数据库，回到CDB

alter session set container=cdb$root;

--2.只读方式打开pdborcl

ALTER PLUGGABLE DATABASE pdborcl OPEN READ ONLY;

--3.创建数据库clonedb

CREATE PLUGGABLE DATABASE clonedb FROM pdborcl file\_name\_convert=('/home/oracle/app/oracle/oradata/orcl/pdborcl','/home/student/pdb/clonedb');

如果pdborcl创建失败，可以试一下其他pdb:

CREATE PLUGGABLE DATABASE mydb2 FROM rsj file\_name\_convert=(

'/home/student/pdb/RSJ/','home/student/pdb/mydb2/')

创建时，如果出现:

ORA-65005: 文件缺少文件名模式或其文件名模式无效 - /home/oracle/app/oracle/oradata/orcl/pdborcl/temp01.dbf，说明源pdb中的目录比较复杂，不能自动映射，可以：

先通过：select name,ts#,con\_id from v$datafile 查询文件名，手工一一指定文件

CREATE PLUGGABLE DATABASE mydb2 FROM pdborcl file\_name\_convert=(

'/home/oracle/app/oracle/oradata/orcl/pdborcl/temp01.dbf','/home/student/pdb/db3/temp01.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/system01.dbf','/home/student/pdb/db3/system01.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/sysaux01.dbf','/home/student/pdb/db3/sysaux01.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/undotbs01.dbf','/home/student/pdb/db3/undotbs01.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/users01.dbf','/home/student/pdb/db3/users01.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_user02\_1.dbf','/home/student/pdb/db3/pdbtest\_user02\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_user02\_2.dbf','/home/student/pdb/db3/pdbtest\_user02\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_user03\_1.dbf','/home/student/pdb/db3/pdbtest\_user03\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_user03\_2.dbf','/home/student/pdb/db3/pdbtest\_user03\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_users02\_1.dbf','/home/student/pdb/db3/pdbtest\_users02\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_users02\_2.dbf','/home/student/pdb/db3/pdbtest\_users02\_2.dbf',

'/home/student/rsj/pdbtest\_users02\_1.dbf','/home/student/pdb/db6/pdbtest\_users02\_1.dbf',

'/home/student/rsj/pdbtest\_users02\_2.dbf','/home/student/pdb/db6/pdbtest\_users02\_2.dbf',

'/home/student/cyy/pdbtest\_users02\_1.dbf','/home/student/pdb/db7/pdbtest\_users02\_1.dbf',

'/home/student/cyy/pdbtest\_users02\_2.dbf','/home/student/pdb/db7/pdbtest\_users02\_2.dbf',

'/home/student/lds/pdbtest\_users02\_1.dbf','/home/student/pdb/db8/pdbtest\_users02\_1.dbf',

'/home/student/lds/pdbtest\_users02\_2.dbf','/home/student/pdb/db8/pdbtest\_users02\_2.dbf',

'/home/student/wish/pdbtest\_users02\_1.dbf','/home/student/pdb/db9/pdbtest\_users02\_1.dbf',

'/home/student/wish/pdbtest\_users02\_2.dbf','/home/student/pdb/db9/pdbtest\_users02\_2.dbf',

'/home/student/lqx/pdbtest\_users02\_1.dbf','/home/student/pdb/db10/pdbtest\_users02\_1.dbf',

'/home/student/lqx/pdbtest\_users02\_2.dbf','/home/student/pdb/db10/pdbtest\_users02\_2.dbf',

'/home/student/zzy/pdbtest\_users02\_1.dbf','/home/student/pdb/db11/pdbtest\_users02\_1.dbf',

'/home/student/zzy/pdbtest\_users02\_2.dbf','/home/student/pdb/db11/pdbtest\_users02\_2.dbf',

'/home/student/cx/pdbtest\_users02\_1.dbf','/home/student/pdb/db12/pdbtest\_users02\_1.dbf',

'/home/student/cx/pdbtest\_users02\_2.dbf','/home/student/pdb/db12/pdbtest\_users02\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_users1\_1.dbf','/home/student/pdb/db3/pdbtest\_users1\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_users1\_2.dbf','/home/student/pdb/db3/pdbtest\_users1\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_users2\_1.dbf','/home/student/pdb/db3/pdbtest\_users2\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_users2\_2.dbf','/home/student/pdb/db3/pdbtest\_users2\_2.dbf',

'/home/student/kyrene/pdbtest\_users02\_1.dbf','/home/student/pdb/db13/pdbtest\_users02\_1.dbf',

'/home/student/kyrene/pdbtest\_users02\_2.dbf','/home/student/pdb/db13/pdbtest\_users02\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/ZXQ\_user1.dbf','/home/student/pdb/db3/ZXQ\_user1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/ZXQ\_user2.dbf','/home/student/pdb/db3/ZXQ\_user2.dbf',

'/home/oracle/app/oracle/oradata/orcl/users02\_1.dbf','/home/student/pdb/db4/users02\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/users02\_2.dbf','/home/student/pdb/db4/users02\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/users03\_1.dbf','/home/student/pdb/db4/users03\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/users03\_2.dbf','/home/student/pdb/db4/users03\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/users04\_1.dbf','/home/student/pdb/db4/users04\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/users04\_2.dbf','/home/student/pdb/db4/users04\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/users05\_1.dbf','/home/student/pdb/db4/users05\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/users05\_2.dbf','/home/student/pdb/db4/users05\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user1\_1.dbf','/home/student/pdb/db4/hzl\_user1\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user1\_2.dbf','/home/student/pdb/db4/hzl\_user1\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user2\_1.dbf','/home/student/pdb/db4/hzl\_user2\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user2\_2.dbf','/home/student/pdb/db4/hzl\_user2\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user3\_1.dbf','/home/student/pdb/db4/hzl\_user3\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user3\_2.dbf','/home/student/pdb/db4/hzl\_user3\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user4\_1.dbf','/home/student/pdb/db4/hzl\_user4\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/hzl\_user4\_2.dbf','/home/student/pdb/db4/hzl\_user4\_2.dbf',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/zhengshenwen\_ data.dbf','/home/student/pdb/db5/zhengshenwen\_ data.dbf',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/qhl001\_1.dbf','/home/student/pdb/db5/qhl001\_1.dbf',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/qhl001\_2.dbf','/home/student/pdb/db5/qhl001\_2.dbf',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/qhl002\_1.dbf','/home/student/pdb/db5/qhl002\_1.dbf',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/qhl002\_2.dbf','/home/student/pdb/db5/qhl002\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_MAQIAO1.dbf','/home/student/pdb/db3/pdbtest\_MAQIAO1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_MAQIAO2.dbf','/home/student/pdb/db3/pdbtest\_MAQIAO2.dbf',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/gwh\_01.dbf ','/home/student/pdb/db5/gwh\_01.dbf ',

'/home/oracle/app/oracle/product/12.2.0/dbhome\_1/dbs/gwh\_02.dbf ','/home/student/pdb/db5/gwh\_02.dbf ',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_U123\_1.dbf','/home/student/pdb/db3/pdbtest\_U123\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_U123\_2.dbf','/home/student/pdb/db3/pdbtest\_U123\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_ly001\_1.dbf','/home/student/pdb/db3/pdbtest\_ly001\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_ly001\_2.dbf','/home/student/pdb/db3/pdbtest\_ly001\_2.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_ly002\_1.dbf','/home/student/pdb/db3/pdbtest\_ly002\_1.dbf',

'/home/oracle/app/oracle/oradata/orcl/pdborcl/pdbtest\_ly002\_2.dbf','/home/student/pdb/db3/pdbtest\_ly002\_2.dbf'

);

--4.打开新数据库

ALTER PLUGGABLE DATABASE clonedb OPEN;

--查看新数据库

show pdbs;

--5.创建成功后，重新打开pdborcl为读写状态

ALTER PLUGGABLE DATABASE pdborcl CLOSE immediate;

ALTER PLUGGABLE DATABASE pdborcl OPEN;

--6.测试

--创建成功后,退出sys用户，以hr登录到新数据库,测试一下

$ sqlplus hr/123@202.115.82.8/clonedb

--查看数据库相关文件

$ ls /home/student/pdb/clonedb

--7.删除新数据库（可选）

--重新sys登录,删除新增的数据库

DROP PLUGGABLE DATABASE clonedb INCLUDING DATAFILES;

### 4.3查询需要备份的文件

SELECT NAME FROM v$datafile

UNION ALL

SELECT MEMBER AS NAME FROM v$logfile

UNION ALL

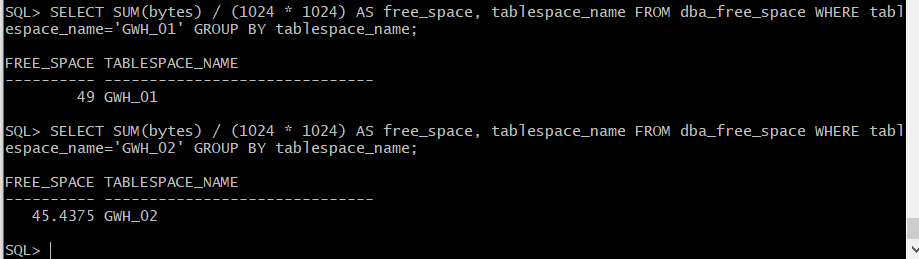
SELECT NAME FROM v$controlfile;

### 

### 4.4查询自己创建的表空间的大小

SELECT SUM(bytes) / (1024 \* 1024) AS free\_space, tablespace\_name FROM dba\_free\_space WHERE tablespace\_name='GWH\_01' GROUP BY tablespace\_name;

SELECT SUM(bytes) / (1024 \* 1024) AS free\_space, tablespace\_name FROM dba\_free\_space WHERE tablespace\_name='GWH\_02' GROUP BY tablespace\_name;



**4.5手工文件备份**

**全库0级备份(只作一次)**

run{

configure retention policy to redundancy 1;

configure controlfile autobackup on;

configure controlfile autobackup format **for** device type disk to '/home/student/rman\_backup/%F';

configure **default** device type to disk;

crosscheck backup;

crosscheck archivelog all;

allocate channel c1 device type disk;

allocate channel c2 device type disk;

allocate channel c3 device type disk;

backup incremental level 0 database format '/home/student/rman\_backup/level0\_%d\_%T\_%U.bak';

report obsolete;

**delete** noprompt obsolete;

**delete** noprompt expired backup;

**delete** noprompt expired archivelog all;

release channel c1;

release channel c2;

release channel c3;

}

**全库1级增量备份**

run{

configure retention policy to redundancy 1;

configure controlfile autobackup on;

configure controlfile autobackup format **for** device type disk to '/home/student/rman\_backup/%F';

configure **default** device type to disk;

crosscheck backup;

crosscheck archivelog all;

allocate channel c1 device type disk;

allocate channel c2 device type disk;

allocate channel c3 device type disk;

backup incremental level 1 database format '/home/student/rman\_backup/level1\_%d\_%T\_%U.bak';

report obsolete;

**delete** noprompt obsolete;

**delete** noprompt expired backup;

**delete** noprompt expired archivelog all;

release channel c1;

release channel c2;

release channel c3;

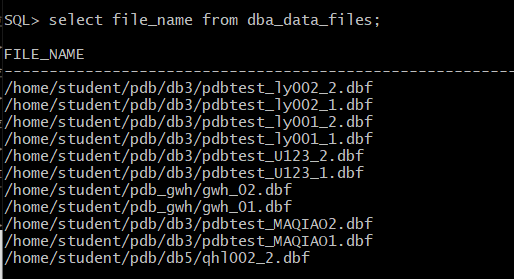
}

### 4.6 更改数据，删除数据库文件模拟数据损坏

- system登录到gwh，查看gwh的数据文件

$ sqlplus system/123@202.115.82.8/gwh

SQL> select file\_name from dba\_data\_files;



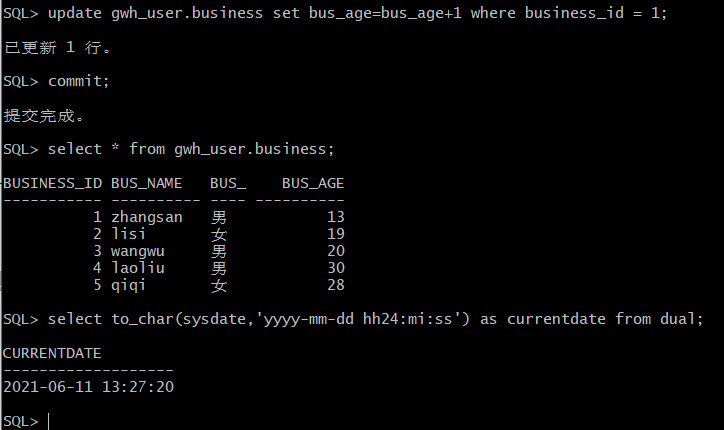
查看表数据

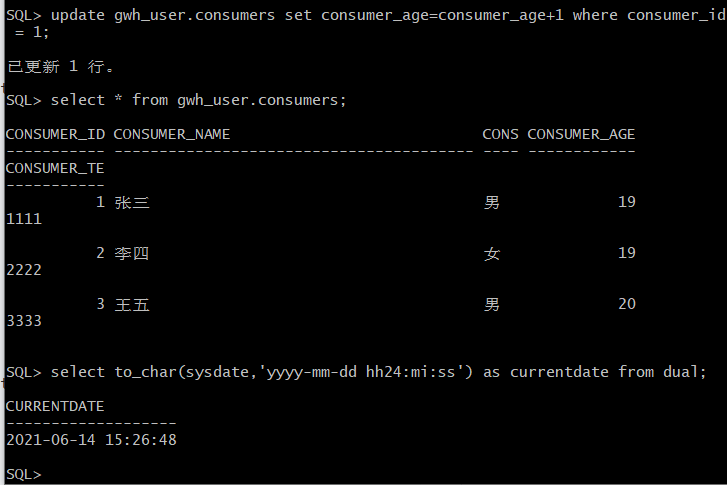
select \* from gwh\_user.consumers;

select to\_char(sysdate,'yyyy-mm-dd hh24:mi:ss') as currentdate from dual;

更新数据

update gwh\_user.consumers set consumer\_age=consumer\_age+1 where consumer\_id = 1;





- 关闭gwh数据库

$rman target sys/123@202.115.82.8/orcl:dedicated

RMAN> alter pluggable database gwh close;

RMAN> exit;

或者

$sqlplus sys/123@202.115.82.8/orcl:dedicated as sysdba;

SQL>SELECT server FROM v$session WHERE SID=(SELECT DISTINCT SID FROM v$mystat);

SQL>alter session set container=gwh;

SQL>shutdown immediate; 或者 shutdown abort;

- 数据文件改名，模拟文件损失

$mv -f /home/student/pdb\_gwh/GWH\_02.dbf /home/student/pdb\_ gwh / GWH\_02\_01.dbf2

### 4.7 数据库恢复

1、**单库完全恢复**

$rman target sys/123@202.115.82.8/orcl:dedicated

RMAN> restore pluggable database gwh;

RMAN> recover pluggable database gwh;

RMAN> alter pluggable database gwh open;

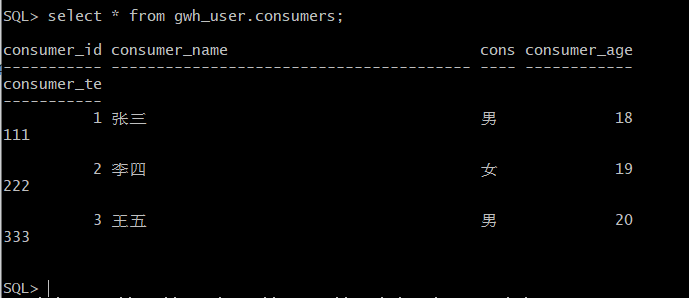
RMAN> exit;

- 完全恢复成功后，system用户登录gwh，

$ sqlplus system/123@202.115.82.8/gwh

SQL> select \* from consumers;

可见，完全恢复成功，无损失。



2、**单库不完全恢复,恢复到update语句之前的状态**

$rman target sys/123@202.115.82.8/orcl:dedicated

RMAN> restore pluggable database gwh;

RMAN> recover pluggable database gwh until time "to\_date('2021-04-27 08:02:24','yyyy-mm-dd hh24:mi:ss')" AUXILIARY DESTINATION '/home/student/zwd';

已处理语句

RMAN>exit

- 不完全恢复成功后，hr用户登录ly，

$ sqlplus system/123@202.115.82.8/gwh

SQL> select \* from consumers;

可见，不完全恢复成功，数据回到了修改前的状态

