

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的逸趣行-列车售票系统的数据库设计 | | |
| 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 | 2018级 |
| 学生姓名 | 周睿锋 | 学号 | 201810414330 |
| 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**

[1.项目简介 4](#_Toc72688101)

[2.功能分析 4](#_Toc72688102)

[2.1车次管理功能 4](#_Toc72688103)

[2.2车站管理功能 4](#_Toc72688104)

[2.3用户管理功能 4](#_Toc72688105)

[2.4订单管理功能 4](#_Toc72688106)

[2.5票务查询功能 4](#_Toc72688107)

[2.6登录注册功能 5](#_Toc72688108)

[3.类图设计 5](#_Toc72688109)

[4.数据库设计 6](#_Toc72688110)

[4.1manage表（管理员表） 6](#_Toc72688111)

[4.2train\_user表（普通用户表） 7](#_Toc72688112)

[4.3station表（车站表） 7](#_Toc72688113)

[4.4train\_order表（订单表表） 7](#_Toc72688114)

[4.5train\_num表（车次表） 8](#_Toc72688115)

[4.6seat表（座位表） 8](#_Toc72688116)

[4.7ticket表（车票信息表） 8](#_Toc72688117)

[5.设计项目涉及的表及表空间使用方案，并插入总共十万条数据 9](#_Toc72688118)

[5.1设计简介 9](#_Toc72688119)

[5.2创建表空间 9](#_Toc72688120)

[5.2.1代码部分 9](#_Toc72688121)

[5.2.2截图部分 10](#_Toc72688122)

[5.3创建表 10](#_Toc72688123)

[5.3.1代码部分 10](#_Toc72688124)

[5.3.2截图部分 15](#_Toc72688125)

[5.4插入数据 18](#_Toc72688126)

[5.4.1插入数据总览 18](#_Toc72688127)

[5.4.2代码部分 18](#_Toc72688128)

[5.4.3截图部分 25](#_Toc72688129)

[6.设计权限及用户分配方案,两类角色，两个用户 26](#_Toc72688130)

[6.1设计简介 26](#_Toc72688131)

[6.2代码部分 26](#_Toc72688132)

[6.3截图部分 27](#_Toc72688133)

[7.在数据库中建立一个程序包，在包中用PL/SQL语言设计一些存储过程和函数，实现比较复杂的业务逻辑，用模拟数据进行执行计划分析 28](#_Toc72688134)

[7.1设计简介 28](#_Toc72688135)

[7.2代码与截图 29](#_Toc72688136)

[8.设计手动备份方案 31](#_Toc72688137)

[8.1设计简介 31](#_Toc72688138)

[8.2实验步骤 32](#_Toc72688139)

[8.2.1创建恢复目录 32](#_Toc72688140)

[8.2.2连接RMAN恢复目录数据库 32](#_Toc72688141)

[8.2.3通道分配 33](#_Toc72688142)

[8.2.4归档模式下备份与恢复 35](#_Toc72688143)

[8.2.5测试备份情况 37](#_Toc72688144)

[8.2.6测试恢复功能 37](#_Toc72688145)

[9.项目总结 41](#_Toc72688146)

# 1.项目简介

项目名称：逸趣行-火车售票系统

本项目是基于Oracle的逸趣行-列车售票系统数据库设计。

涉及角色/用户：管理员、普通用户

涉及表：管理员表、普通用户表、订单表、车次表、座位表、车站表、车票信息表

# 2.功能分析

## 2.1车次管理功能

车次管理功能，是由管理员进行的，主要包含车次查询、车次增加、车次修改、车次删除，主要通过操作数据库来实现这些操作。

## 2.2车站管理功能

车站管理功能，是由管理员进行的，主要包含车站查询、车站增加、车站修改、车站删除，主要通过操作数据库来实现这些操作。

## 2.3用户管理功能

用户管理功能主要指两个方面，一个是指管理员和普通用户都可以对自己的用户信息进行管理，包括查看、修改等操作，另一个是指管理员对普通用户的用户管理，包括查询用户、删除用户。

## 2.4订单管理功能

订单管理能，是由管理员进行的，主要包含查看订单、退订、改签，主要通过操作数据库来实现这些操作。

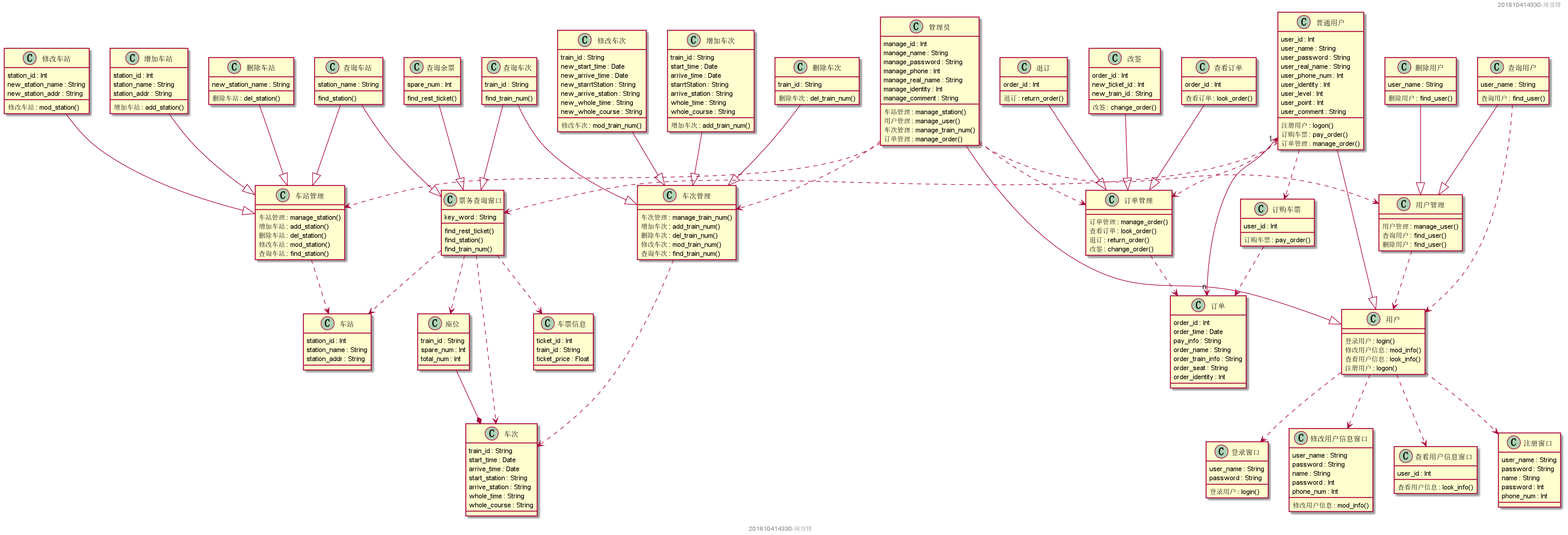
## 2.5票务查询功能

票务查询功能，是由普通用户进行的，主要包含车次查询、车站查询、余票查询，主要通过从数据库匹配关键信息来实现这些操作。

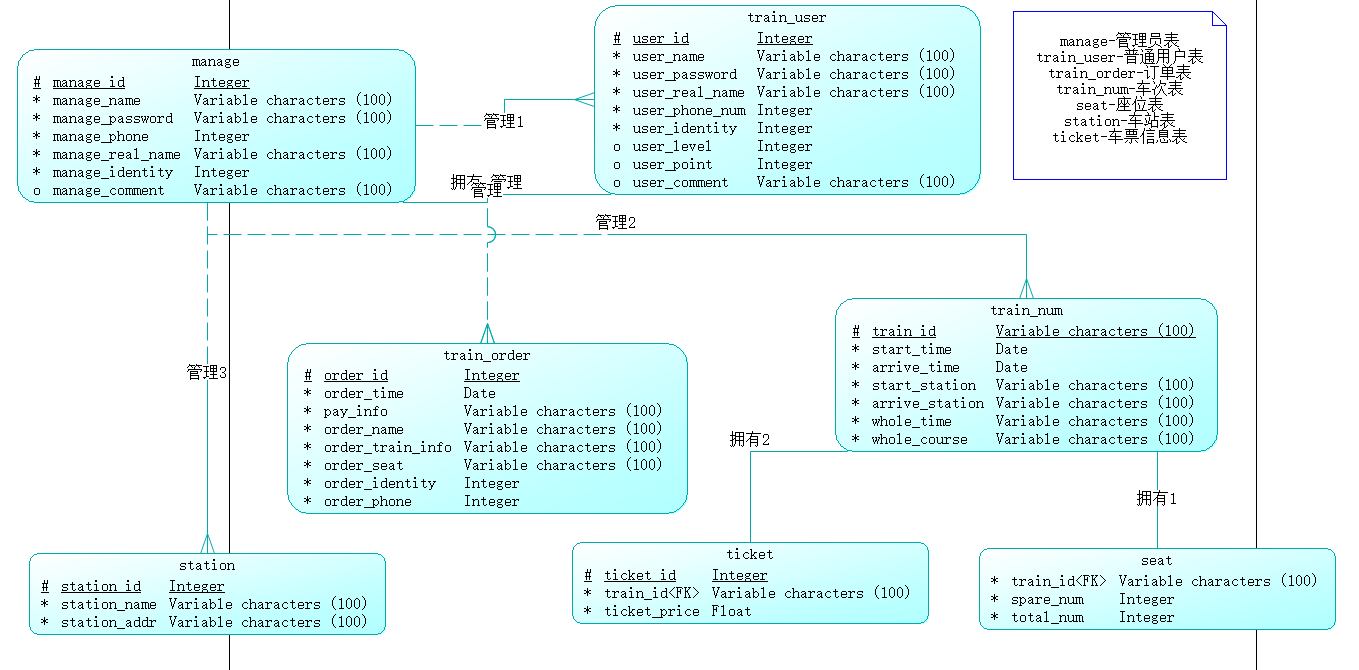
## 2.6登录注册功能

登录注册功能，为使本系统设计符合使用者需求，管理员和普通用户可以进行登录和注册的功能。

# 3.类图设计



# 4.数据库设计



## 4.1manage表（管理员表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| manage\_id | Integer（20） | 主键 | 否 |  |  | 管理员ID |
| manage\_name | VARCHAR(100 BYTE) |  | 否 |  |  | 管理员用户名 |
| manage\_password | VARCHAR(100 BYTE) |  | 否 |  |  | 管理员密码 |
| manage\_phone | Integer（20） |  | 否 |  |  | 管理员电话号码 |
| manage\_real\_name | VARCHAR(100 BYTE) |  | 否 |  |  | 管理员真实姓名 |
| manage\_identity | Integer（20） |  | 否 |  |  | 管理员身份证号码 |
| manage\_comment | VARCHAR(100 BYTE) |  | 是 | 空 |  | 管理员个性签名 |

## 4.2train\_user表（普通用户表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| user\_id | Integer（20） | 主键 | 否 |  |  | 普通用户ID |
| user\_name | VARCHAR(100 BYTE) |  | 否 |  |  | 普通用户姓名 |
| user\_password | VARCHAR(100 BYTE) |  | 否 |  |  | 普通用户密码 |
| user\_real\_name | VARCHAR(100 BYTE) |  | 否 |  |  | 普通用户真实姓名 |
| user\_phone\_num | Integer（20） |  | 否 |  |  | 普通用户手机号 |
| user\_identity | Integer（20） |  | 否 |  |  | 普通用户身份证号码 |
| user\_level | Integer（20） |  | 是 | 空 |  | 普通用户等级 |
| user\_point | Integer（20） |  | 是 | 空 |  | 普通用户积分 |
| user\_comment | VARCHAR(100 BYTE) |  | 是 | 空 |  | 普通用户个性签名 |

## 4.3station表（车站表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| station\_id | Integer（20） | 主键 | 否 |  |  | 车站ID |
| station\_name | VARCHAR(100 BYTE) |  | 否 |  |  | 车站姓名 |
| station\_addr | VARCHAR(100 BYTE) |  | 否 |  |  | 车站地址 |

## 4.4train\_order表（订单表表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| order\_id | Integer（20） | 主键 | 否 |  |  | 订单ID |
| order\_time | Date |  | 否 |  |  | 订单创建时间 |
| pay\_info | VARCHAR(100 BYTE) |  | 否 |  |  | 订单支付信息 |
| order\_name | VARCHAR(100 BYTE) |  | 否 |  |  | 订单所属人姓名 |
| order\_train\_info | VARCHAR(100 BYTE) |  | 否 |  |  | 订单列车信息 |
| order\_seat | VARCHAR(100 BYTE) |  | 否 |  |  | 订单座位 |
| order\_identity | Integer（20） |  | 否 |  |  | 订单身份证号码 |
| order\_phone | Integer（20） |  | 否 |  |  | 订单电话号码 |

## 4.5train\_num表（车次表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| train\_id | VARCHAR(100 BYTE) | 主键 | 否 |  |  | 车次编号 |
| start\_time | Date |  | 否 |  |  | 起始时间 |
| arrive\_time | Date |  | 否 |  |  | 到达时间 |
| start\_station | VARCHAR(100 BYTE) |  | 否 |  |  | 起始车站 |
| arrive\_station | VARCHAR(100 BYTE) |  | 否 |  |  | 到达车站 |
| whole\_time | VARCHAR(100 BYTE) |  | 否 |  |  | 全程耗时 |
| whole\_course | VARCHAR(100 BYTE) |  | 否 |  |  | 全程距离 |

## 4.6seat表（座位表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| train\_id | VARCHAR(100 BYTE) | 外键 | 否 |  |  | 车次编号 |
| spare\_num | Integer（20） |  | 否 |  |  | 余座 |
| total\_num | Integer（20） |  | 否 |  |  | 总共座位数量 |

## 4.7ticket表（车票信息表）

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字段 | 类型 | 主键，外键 | 可以为空 | 默认值 | 约束 | 说明 |
| ticket\_id | Integer（20） | 主键 | 否 |  |  | 车次编号 |
| train\_id | VARCHAR(100 BYTE) | 外键 | 否 |  |  | 余座 |
| ticket\_price | Float（20） |  | 否 |  |  | 总共座位数量 |

# 5.设计项目涉及的表及表空间使用方案，并插入总共十万条数据

## 5.1设计简介

创建表空间：

表空间总共三个：train\_manage、train\_user、train\_public

表空间中存放的表:

train\_manage: manage

train\_user: train\_num

train\_public: station、train\_order、train\_num、seat、ticket

每个表插入数据：

manage：20000条

train\_num：20000条

station：20000条

train\_order：10000条

train\_num：10000条

seat：10000条

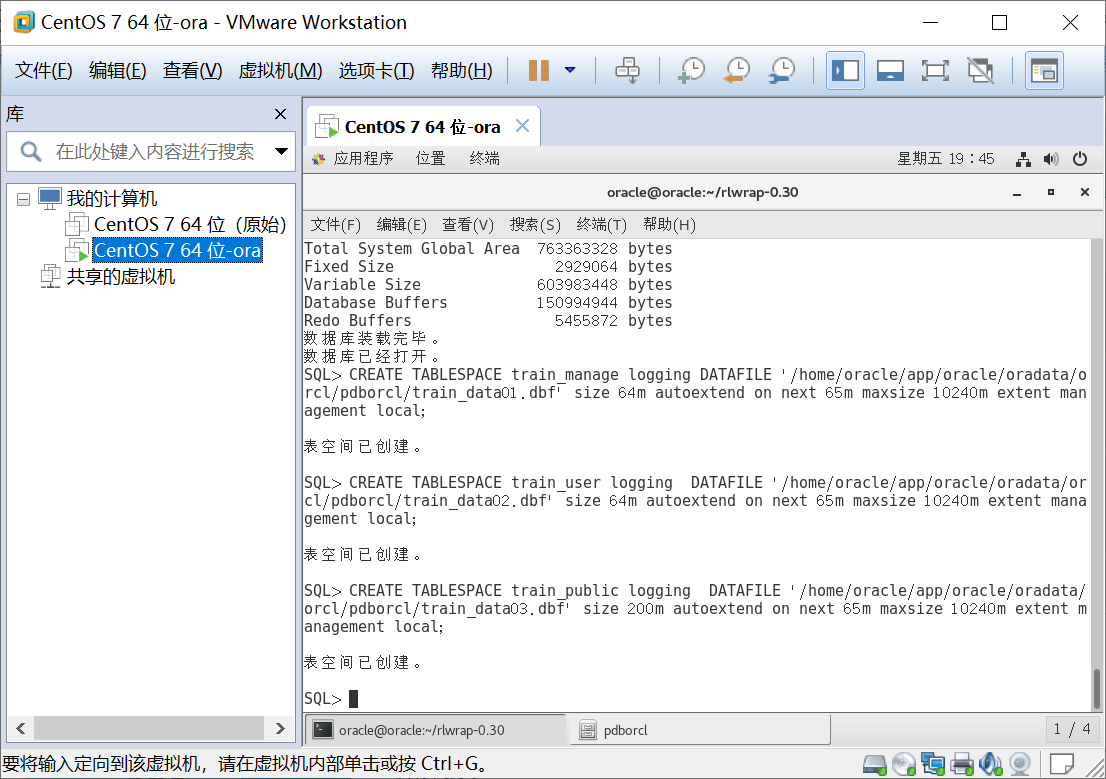
ticket：10000条

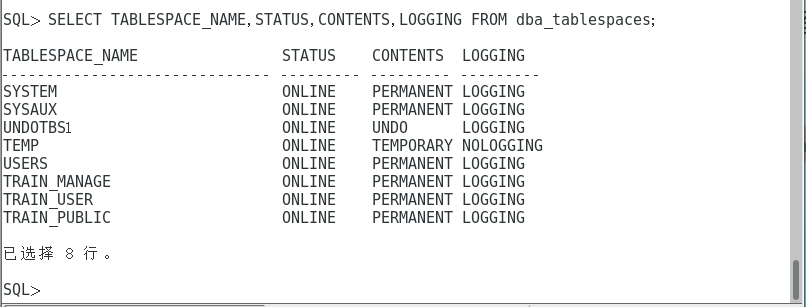
## 5.2创建表空间

### 5.2.1代码部分

-- 创建表空间  
CREATE TABLESPACE train\_manage logging DATAFILE '/home/oracle/app/oracle/oradata/orcl/pdborcl/train\_data01.dbf' size 64m autoextend on next 65m maxsize 10240m extent management local;  
CREATE TABLESPACE train\_user logging DATAFILE '/home/oracle/app/oracle/oradata/orcl/pdborcl/train\_data02.dbf' size 64m autoextend on next 65m maxsize 10240m extent management local;  
CREATE TABLESPACE train\_public logging DATAFILE '/home/oracle/app/oracle/oradata/orcl/pdborcl/train\_data03.dbf' size 200m autoextend on next 65m maxsize 10240m extent management local;

### 5.2.2截图部分



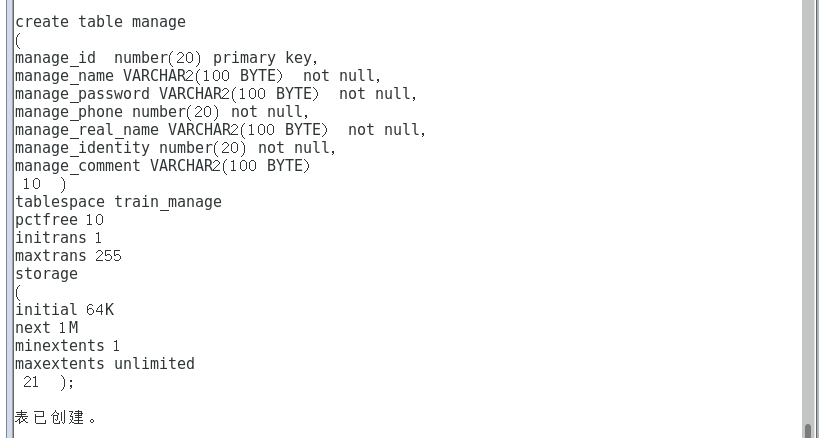


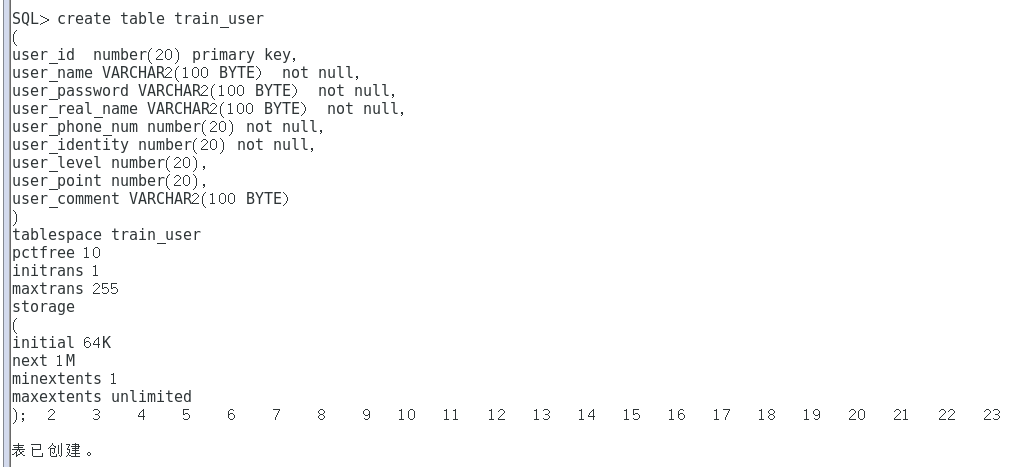
## 5.3创建表

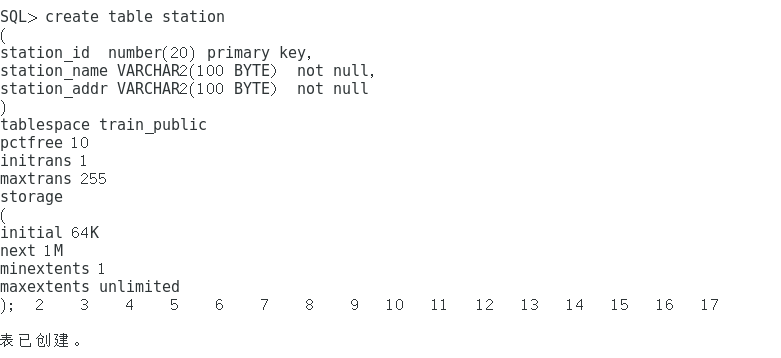
### 5.3.1代码部分

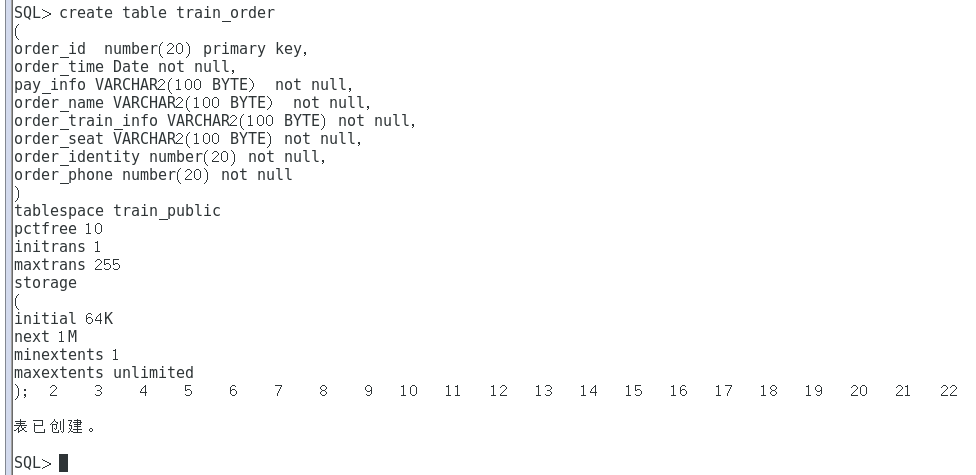
-- manage表的创建  
create table manage  
(  
manage\_id number(20) primary key,  
manage\_name VARCHAR2(100 BYTE） not null,  
manage\_password VARCHAR2(100 BYTE） not null,  
manage\_phone number(20) not null,  
manage\_real\_name VARCHAR2(100 BYTE） not null,  
manage\_identity number(20) not null,  
manage\_comment VARCHAR2(100 BYTE）  
)  
tablespace train\_manage   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);  
-- train\_user表的创建  
create table train\_user  
(  
user\_id number(20) primary key,  
user\_name VARCHAR2(100 BYTE） not null,  
user\_password VARCHAR2(100 BYTE） not null,  
user\_real\_name VARCHAR2(100 BYTE） not null,  
user\_phone\_num number(20) not null,   
user\_identity number(20) not null,  
user\_level number(20),  
user\_point number(20),   
user\_comment VARCHAR2(100 BYTE）  
)  
tablespace train\_user   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);  
   
-- station表的创建  
create table station  
(  
station\_id number(20) primary key,  
station\_name VARCHAR2(100 BYTE） not null,  
station\_addr VARCHAR2(100 BYTE） not null  
)  
tablespace train\_public   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);  
   
-- train\_order表的创建  
create table train\_order  
(  
order\_id number(20) primary key,  
order\_time Date not null,  
pay\_info VARCHAR2(100 BYTE） not null,  
order\_name VARCHAR2(100 BYTE） not null,  
order\_train\_info VARCHAR2(100 BYTE) not null,   
order\_seat VARCHAR2(100 BYTE) not null,  
order\_identity number(20) not null,  
order\_phone number(20) not null   
)  
tablespace train\_public   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);  
   
-- train\_num表的创建  
create table train\_num  
(  
train\_id VARCHAR2(100 BYTE） primary key,  
start\_time Date not null,  
arrive\_time Date not null,  
start\_station VARCHAR2(100 BYTE） not null,  
arrive\_station VARCHAR2(100 BYTE） not null,  
whole\_time VARCHAR2(100 BYTE） not null,   
whole\_course VARCHAR2(100 BYTE） not null   
)  
tablespace train\_public   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);   
  
-- seat表的创建  
create table seat  
(  
train\_id VARCHAR2(100 BYTE） not null references train\_num(train\_id),  
spare\_num number(20) not null,  
total\_num number(20) not null   
)  
tablespace train\_public   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);   
  
-- ticket表的创建  
create table ticket  
(  
ticket\_id number(20) primary key,  
train\_id VARCHAR2(100 BYTE） not null references train\_num(train\_id),  
ticket\_price number(20,2) not null   
)  
tablespace train\_public   
pctfree 10   
initrans 1   
maxtrans 255   
storage   
(  
initial 64K   
next 1M  
minextents 1   
maxextents unlimited   
);   
 -- 查询表空间的数据库   
select TABLE\_NAME from dba\_tables where TABLESPACE\_NAME='TRAIN\_MANAGE';  
select TABLE\_NAME from dba\_tables where TABLESPACE\_NAME='TRAIN\_USER';  
select TABLE\_NAME from dba\_tables where TABLESPACE\_NAME='TRAIN\_PUBLIC';

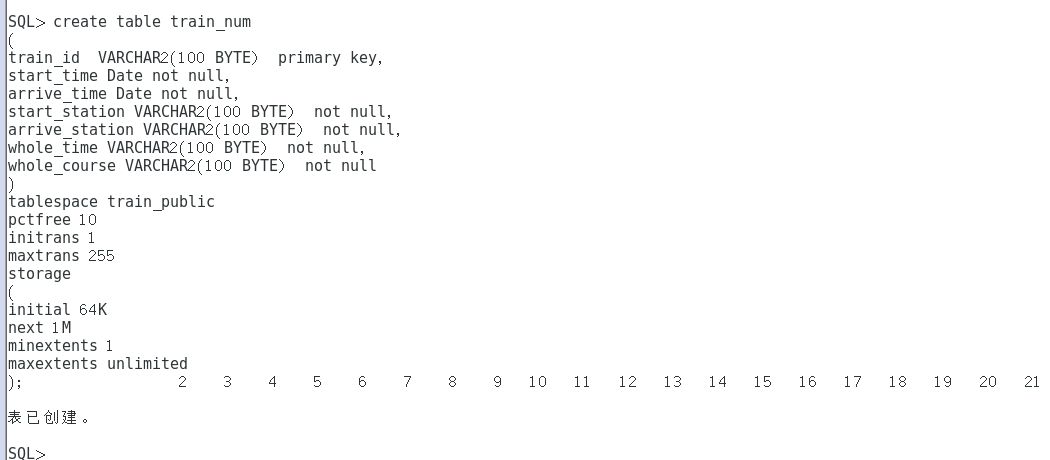
### 5.3.2截图部分

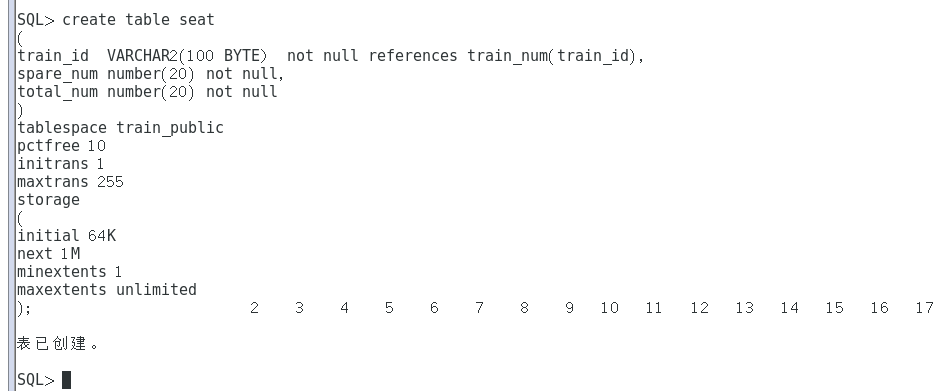


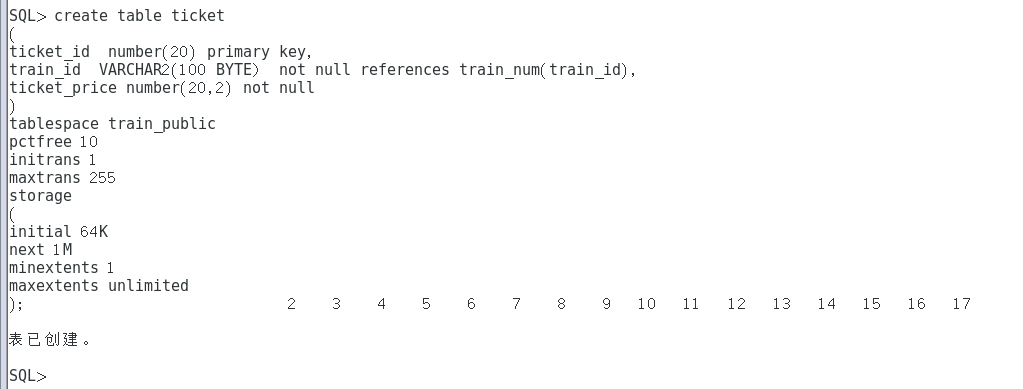




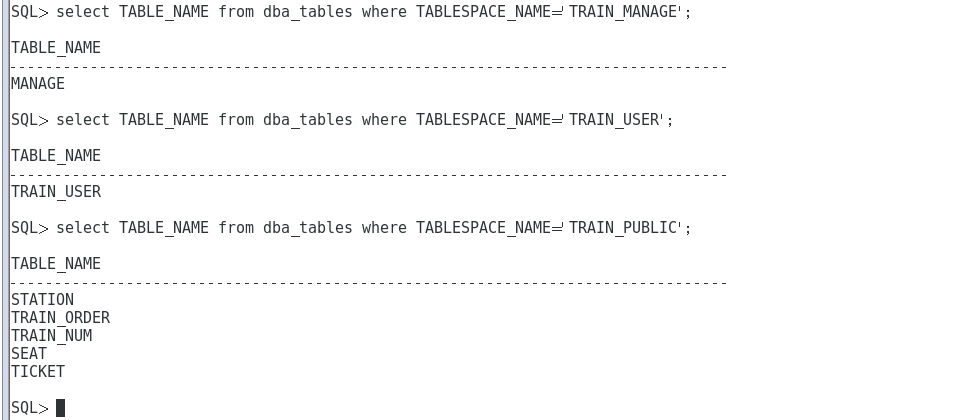








验证表是否创建在对应的表空间：



## 5.4插入数据

### 5.4.1插入数据总览

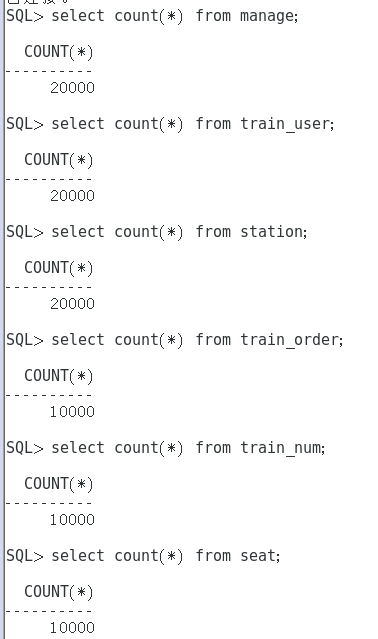
总共插入数据：100000条  
manage：20000条  
train\_num：20000条  
station：20000条  
train\_order：10000条  
train\_num：10000条  
seat：10000条  
ticket：10000条

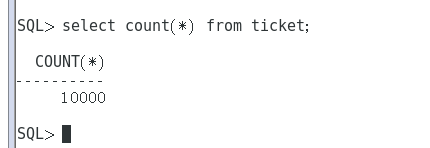
### 5.4.2代码部分

-- manage表的数据插入20000条  
declare   
i int;  
manage\_id number(20);  
manage\_name VARCHAR2(100);  
manage\_password VARCHAR2(100);  
manage\_phone number(20);  
manage\_real\_name VARCHAR2(100);  
manage\_identity number(20);  
manage\_comment VARCHAR2(100);  
begin  
i:=1;  
while i<=20000   
loop  
manage\_id:=i;  
manage\_name:= 'manage'|| i;  
manage\_password := '123'|| i;  
manage\_phone := '180810'|| i;  
manage\_real\_name := '周睿锋'||i;  
manage\_identity := '51323120'||i;  
manage\_comment := '个人签名'||i;  
insert into manage(MANAGE\_ID,MANAGE\_NAME,MANAGE\_PASSWORD,MANAGE\_PHONE,MANAGE\_REAL\_NAME,MANAGE\_IDENTITY,MANAGE\_COMMENT) values (manage\_id,manage\_name,manage\_password,manage\_phone,manage\_real\_name,manage\_identity,manage\_comment);  
i:=i+1;  
end loop;  
commit;  
end;  
/  
  
  
-- train\_user表数据插入20000  
declare   
i int;  
user\_id number(20);  
user\_name VARCHAR2(100);  
user\_password VARCHAR2(100);  
user\_phone\_num number(20);  
user\_real\_name VARCHAR2(100);  
user\_identity number(20);  
user\_level number(20);  
user\_point number(20);  
user\_comment VARCHAR2(100);  
begin  
i:=1;  
while i<=20000   
loop  
user\_id:=i;  
user\_name:= 'user'|| i;  
user\_password := '123'|| i;  
user\_phone\_num := '180810'|| i;  
user\_real\_name := 'zrf'||i;  
user\_identity := '51323120'||i;  
user\_level := '1'||i;  
user\_point := '1'||i;  
user\_comment := '个人签名'||i;  
insert into train\_user(user\_id,user\_name,user\_password,user\_real\_name,user\_phone\_num,user\_identity,user\_level,user\_point,user\_comment) values (user\_id,user\_name,user\_password,user\_real\_name,user\_phone\_num,user\_identity,user\_level,user\_point,user\_comment);  
i:=i+1;  
end loop;  
commit;  
end;  
/  
  
  
   
-- station表的数据插入20000条  
declare   
i int;  
station\_id number(20);  
station\_name VARCHAR2(100);  
station\_addr VARCHAR2(100);  
begin  
i:=1;  
while i<=20000   
loop  
station\_id:=i;  
station\_name:= '地点'|| i;  
station\_addr := '地址'|| i;  
insert into station(station\_id,station\_name,station\_addr) values (station\_id,station\_name,station\_addr);  
i:=i+1;  
end loop;  
commit;  
end;  
/  
  
   
-- train\_order表的数据插入10000条  
declare   
i int;  
order\_id number(20);  
order\_time Date ;  
pay\_info VARCHAR2(100);  
order\_name VARCHAR2(100);  
order\_train\_info VARCHAR2(100) ;   
order\_identity number(20);  
order\_seat VARCHAR2(100);  
order\_phone number(20);  
begin  
i:=1;  
while i<=10000   
loop  
order\_id:=i;  
if i mod 6 =0 then  
 order\_time:=to\_date('2015-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =1 then  
 order\_time:=to\_date('2016-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =2 then  
 order\_time:=to\_date('2017-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =3 then  
 order\_time:=to\_date('2018-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =4 then  
 order\_time:=to\_date('2019-3-2','yyyy-mm-dd')+(i mod 60);  
else  
 order\_time:=to\_date('2020-3-2','yyyy-mm-dd')+(i mod 60);  
end if;  
pay\_info:= '已支付';  
order\_name := 'zrf'||i;  
order\_train\_info :=i;  
order\_identity := '51323120'||i;  
order\_phone := '180810'||i;  
order\_seat :='B'||i mod 60;  
insert into train\_order(order\_id,order\_time,pay\_info,order\_name,order\_train\_info,order\_identity,order\_seat,order\_phone) values (order\_id,order\_time,pay\_info,order\_name,order\_train\_info,order\_identity,order\_seat,order\_phone);  
i:=i+1;  
end loop;  
commit;  
end;  
/  
  
   
-- train\_num表的数据插入10000条  
declare   
i int;  
train\_id VARCHAR2(100);  
start\_time Date;  
arrive\_time Date;  
start\_station VARCHAR2(100);  
arrive\_station VARCHAR2(100);  
whole\_time VARCHAR2(100);   
whole\_course VARCHAR2(100);  
begin  
i:=1;  
while i<=10000   
loop  
train\_id:=i;  
if i mod 6 =0 then  
 start\_time:=to\_date('2015-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =1 then  
 start\_time:=to\_date('2016-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =2 then  
 start\_time:=to\_date('2017-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =3 then  
 start\_time:=to\_date('2018-3-2','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =4 then  
 start\_time:=to\_date('2019-3-2','yyyy-mm-dd')+(i mod 60);  
else  
 start\_time:=to\_date('2020-3-2','yyyy-mm-dd')+(i mod 60);  
end if;  
if i mod 6 =0 then  
 arrive\_time:=to\_date('2015-3-4','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =1 then  
 arrive\_time:=to\_date('2016-3-4','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =2 then  
 arrive\_time:=to\_date('2017-3-4','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =3 then  
 arrive\_time:=to\_date('2018-3-4','yyyy-mm-dd')+(i mod 60);  
elsif i mod 6 =4 then  
 arrive\_time:=to\_date('2019-3-4','yyyy-mm-dd')+(i mod 60);  
else  
 arrive\_time:=to\_date('2020-3-4','yyyy-mm-dd')+(i mod 60);  
end if;  
start\_station:= '起始站'||i;  
arrive\_station := '终点站'|| i;  
whole\_time := (TO\_NUMBER(arrive\_time - start\_time));  
whole\_course := '200'||i mod 6;  
insert into train\_num(train\_id,start\_time,arrive\_time,start\_station,arrive\_station,whole\_time,whole\_course) values (train\_id,start\_time,arrive\_time,start\_station,arrive\_station,whole\_time,whole\_course);  
i:=i+1;  
end loop;  
commit;  
end;  
/  
   
-- seat表的数据的插入10000条  
declare   
i int;  
train\_id VARCHAR2(100);  
spare\_num number(20);  
total\_num number(20);  
begin  
i:=1;  
while i<=10000   
loop  
train\_id:=i;  
spare\_num:= i mod 60;  
total\_num := 60;  
insert into seat(train\_id,spare\_num,total\_num) values (train\_id,spare\_num,total\_num);  
i:=i+1;  
end loop;  
commit;  
end;  
/  
  
-- ticket表的数据插入10000条  
declare   
i int;  
ticket\_id number(20);  
train\_id VARCHAR2(100);  
ticket\_price number(20,2);  
begin  
i:=1;  
while i<=10000   
loop  
ticket\_id:=i;  
train\_id:=i;  
ticket\_price:=dbms\_random.value(400,100);  
insert into ticket(ticket\_id,train\_id,ticket\_price) values (ticket\_id,train\_id,ticket\_price);  
i:=i+1;  
end loop;  
commit;  
end;  
/   
  
-- 查询是否插入成功  
select count(\*) from manage;  
select count(\*) from train\_user;  
select count(\*) from station;  
select count(\*) from train\_order;  
select count(\*) from train\_num;  
select count(\*) from seat;  
select count(\*) from ticket;  
select count(\*) from train\_order where pay\_info = '已支付';

### 5.4.3截图部分

验证数据是否插入成功：





# 6.设计权限及用户分配方案,两类角色，两个用户

## 6.1设计简介

用户：

c##train\_manage

拥有角色：c##t\_manage

分配表空间：train\_manage、train\_user、train\_public

c##train\_user

拥有角色：c##t\_user

分配表空间：train\_user、train\_public

角色：

c##t\_manage

拥有角色：connect、resource、dba

c##t\_user

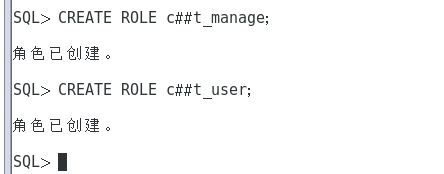
拥有角色：connect、resource

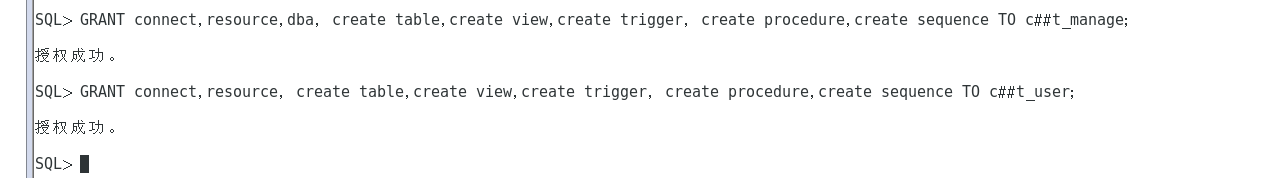
## 6.2代码部分

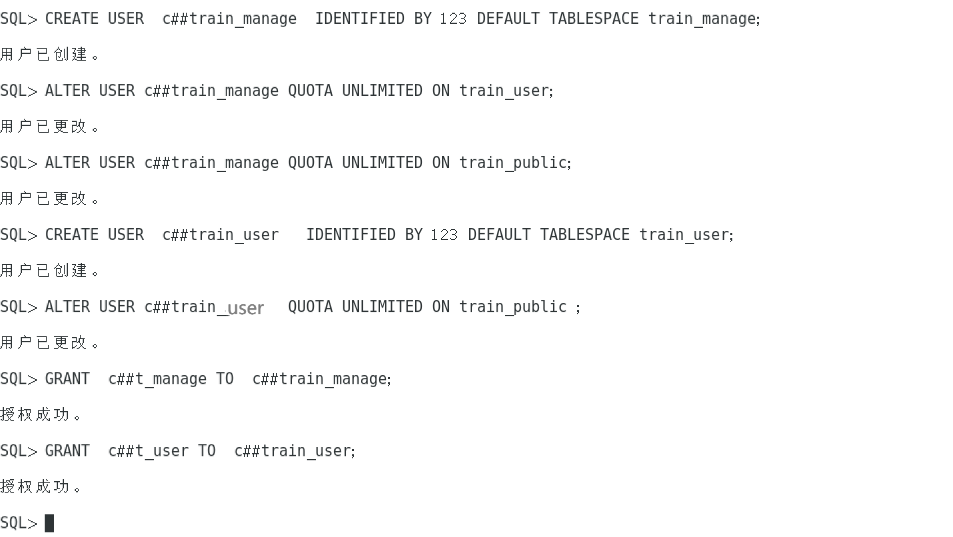
-- 创建角色   
CREATE ROLE c##t\_manage;   
CREATE ROLE c##t\_user;   
-- 授权角色   
GRANT connect,resource,dba, create table,create view,create trigger, create procedure,create sequence TO c##t\_manage;   
GRANT connect,resource, create table,create view,create trigger, create procedure,create sequence TO c##t\_user;   
-- 创建用户   
CREATE USER c##train\_manage IDENTIFIED BY 123 DEFAULT TABLESPACE train\_manage;   
-- 指定用户额外表空间   
ALTER USER c##train\_manage QUOTA UNLIMITED ON train\_user;   
ALTER USER c##train\_manage QUOTA UNLIMITED ON train\_public;   
-- 创建用户   
CREATE USER c##train\_user IDENTIFIED BY 123 DEFAULT TABLESPACE train\_user;   
-- 指定用户额外表空间   
ALTER USER c##train\_user QUOTA UNLIMITED ON train\_public ;   
-- 分配角色给用户   
GRANT c##t\_manage TO c##train\_manage;   
GRANT c##t\_user TO c##train\_user;

## 6.3截图部分

成功截图如下：







# 7.在数据库中建立一个程序包，在包中用PL/SQL语言设计一些存储过程和函数，实现比较复杂的业务逻辑，用模拟数据进行执行计划分析

## 7.1设计简介

建立的包：

包名： TrainPack

函数：

函数名：Get\_count(order\_id\_t NUMBER)

功能：

输入订单ID，查询订单表和车次表，返回与该订单相关的车次对应的起始车站

过程：

过程名：Get\_orders(train\_id\_t VARCHAR2)

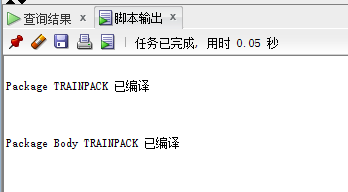
功能：

输入订单id，输出与该订单相关的车次的起始车站，以及统计同一时间段的车次数量并输出与订单相关的车票价格

## 7.2代码与截图

-- 创建包  
create or replace PACKAGE TrainPack IS  
 FUNCTION Get\_count(order\_id\_t NUMBER) RETURN VARCHAR2;  
 PROCEDURE Get\_orders(train\_id\_t VARCHAR2);  
END TrainPack;  
-- 创建函数和过程  
create or replace PACKAGE BODY TrainPack IS  
 FUNCTION Get\_count(order\_id\_t NUMBER) RETURN VARCHAR2  
 AS  
 M VARCHAR2(100);  
 N VARCHAR2(100);  
 BEGIN  
 select order\_train\_info into N from train\_order where order\_id=order\_id\_t;  
 select start\_station into M from train\_num where train\_id=N;  
 RETURN M;  
 END;  
 PROCEDURE Get\_orders(train\_id\_t VARCHAR2)  
 AS  
 N NUMBER(20);  
 L date;  
 S VARCHAR2(20);  
 R VARCHAR2(100);  
 begin  
 --使用游标  
 DBMS\_OUTPUT.PUT\_LINE('输出起始站：');  
 select start\_station into R from train\_num where train\_id = train\_id\_t;  
 DBMS\_OUTPUT.PUT\_LINE(R);  
 select start\_time into L from train\_num where train\_id = train\_id\_t;  
 select count(\*) into N from train\_num where start\_time = L;  
 DBMS\_OUTPUT.PUT\_LINE('输出车次数量：');  
 DBMS\_OUTPUT.PUT\_LINE(N);  
 select ticket\_price into S from ticket where train\_id = train\_id\_t;  
 DBMS\_OUTPUT.PUT\_LINE('输出票价：');  
 DBMS\_OUTPUT.PUT\_LINE(S);  
 END;  
END TrainPack;  
/

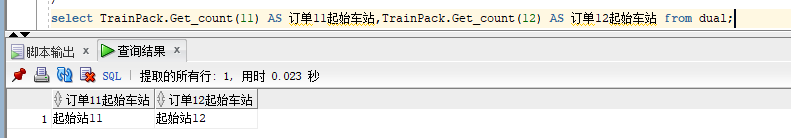
创建成功：

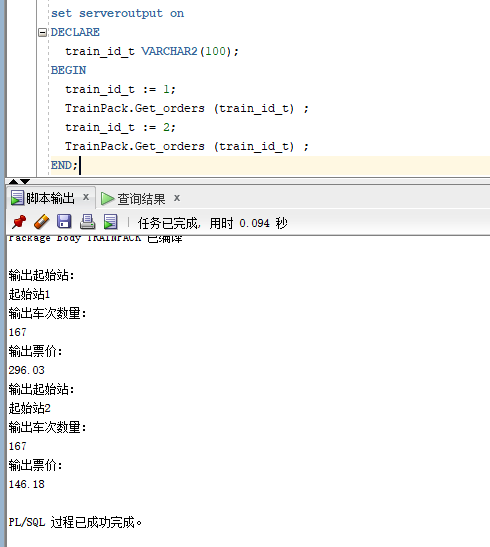


测试：

-- 测试函数  
select TrainPack.Get\_count(11) AS 订单11起始车站,TrainPack.Get\_count(12) AS 订单12起始车站 from dual;  
-- 测试过程  
DECLARE  
 train\_id\_t VARCHAR2(100);   
BEGIN  
 train\_id\_t := 1;  
 TrainPack.Get\_orders (train\_id\_t) ;   
 train\_id\_t := 2;  
 TrainPack.Get\_orders (train\_id\_t) ;   
END;

测试结果：





# 8.设计手动备份方案

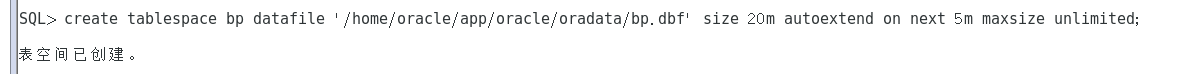
## 8.1设计简介

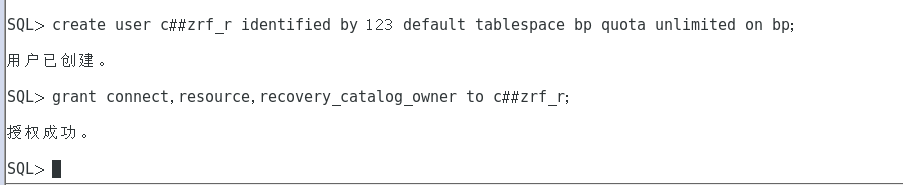
本项目设置手动备份方案——即采用Rman备份

## 8.2实验步骤

### 8.2.1创建恢复目录

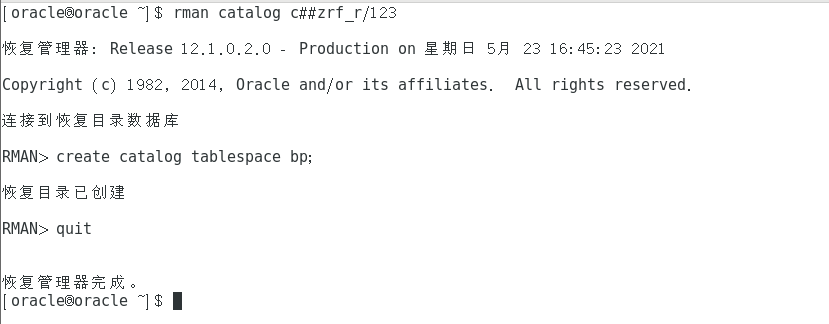
-- 创建恢复目录：用来存储RMAN资料库的   
create tablespace bp datafile '/home/oracle/app/oracle/oradata/bp.dbf' size 20m autoextend on next 5m maxsize unlimited;   
-- 在恢复目录数据库中创建RMAN用户并授权   
create user c##zrf\_r identified by 123 default tablespace bp quota unlimited on bp;   
grant connect,resource,recovery\_catalog\_owner to c##zrf\_r;

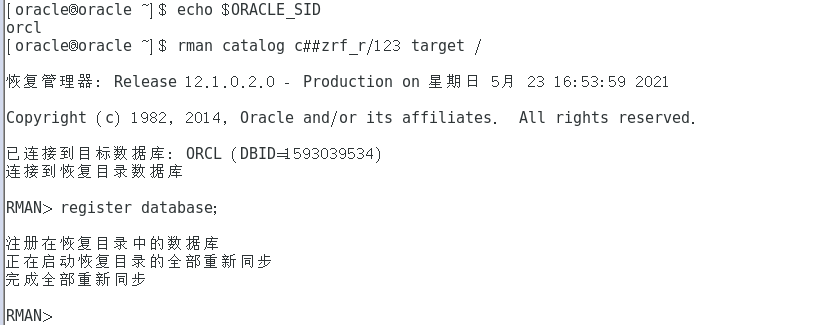




### 8.2.2连接RMAN恢复目录数据库

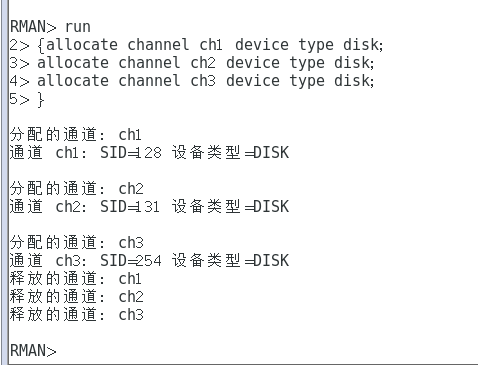
-- 连接RMAN恢复目录数据库   
rman catalog c##zrf\_r/123   
-- 创建恢复目录   
create catalog tablespace bp;   
-- 退出   
quit   
-- 确认环境信息   
echo $ORACLE\_SID   
-- 连接到目标数据库、连接到恢复目录数据库   
rman catalog c##zrf\_r/123 target /   
-- 向恢复目录注册数据库ORCL——此时就可以使用RMAN的恢复目录对目标数据库进行备份和恢复操作   
register database;

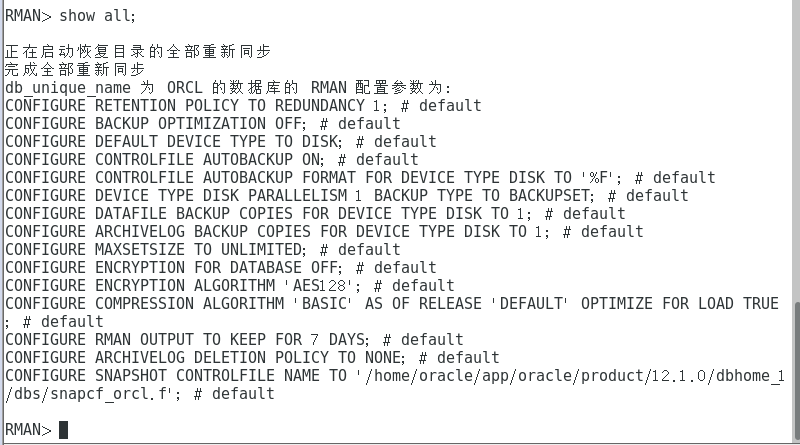




### 8.2.3通道分配

-- 手动通道配置   
run   
{   
allocate channel ch1 device type disk;   
allocate channel ch2 device type disk;   
allocate channel ch3 device type disk;   
}   
   
-- 显示已经配置过的有默认值的参数，其中包括通道参数   
show all;



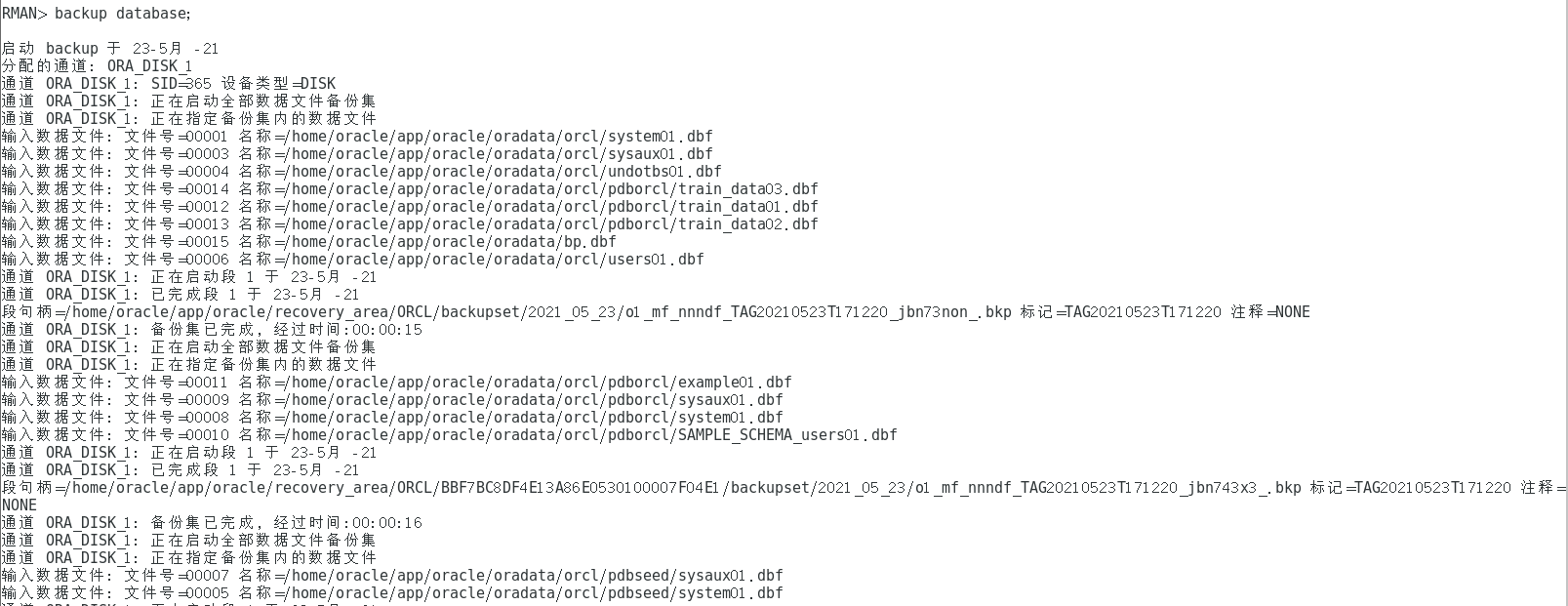


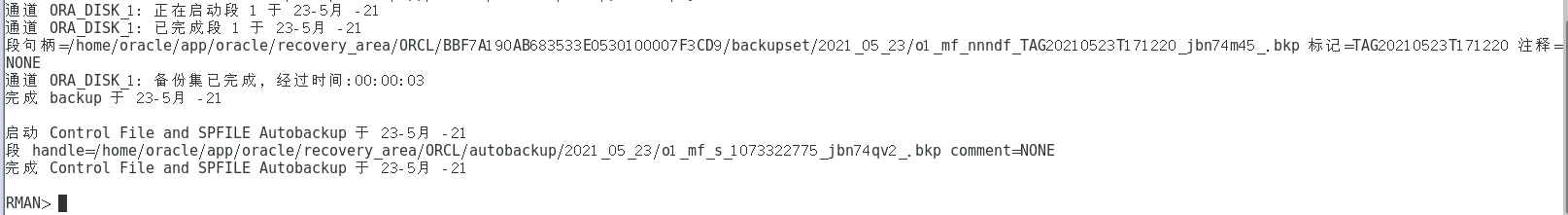
### 8.2.4归档模式下备份与恢复

-- 查看数据库是否处于归档模式下   
archive log list;   
-- 关闭数据库   
shutdown immediate   
-- 重启并设置成归档模式   
startup mount;   
alter database archivelog;   
archive log list;   
alter database open;   
-- 连接到目标数据库、连接到恢复目录数据库   
rman catalog c##zrf\_r/123 target /   
-- 备份和恢复整个数据库   
backup database;



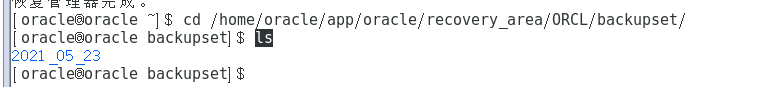






### 8.2.5测试备份情况

-- 切换到保存路径   
cd /home/oracle/app/oracle/recovery\_area/ORCL/backupset/   
-- 查看文件   
ls



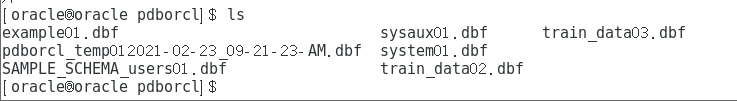
### 8.2.6测试恢复功能

-- 关闭数据库   
shutdown immediate;   
-- 退出数据库   
exit   
-- 切换到数据文件存储路径   
cd /home/oracle/app/oracle/oradata/orcl/pdborcl   
-- 查看数据文件   
ls   
-- 删除train\_data01.dbf   
rm -rf train\_data01.dbf   
-- 再次确认   
ls   
-- 启动数据库   
sqlplus /nolog   
conn /as sysdba   
startup   
-- 此时因为缺少数据文件无法启动   
-- 检查此时数据库状态   
select status from v$instance;   
-- 连接RMAN   
rman target sys/123   
-- 恢复数据库   
restore database;   
-- 同步恢复   
recover database;   
-- 打开数据库   
alter database open resetlogs;   
-- 再次启动数据库，启动成功，检查此时数据库状态，此时状态已经打开   
startup   
select status from v$instance;   
-- 再次切换到数据文件存储路径   
cd /home/oracle/app/oracle/oradata/orcl/pdborcl   
-- 查看数据文件，删除的已经还原   
ls

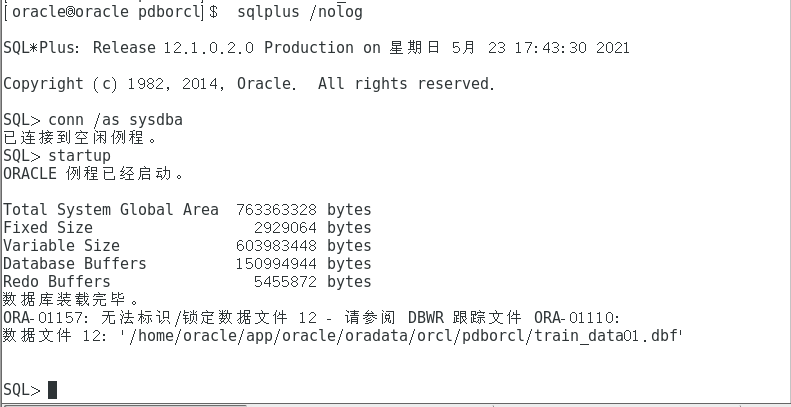
测试截图：

1、删除文件并查看删除成功与否：

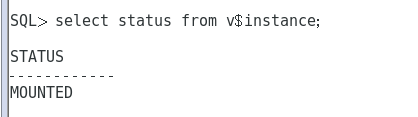




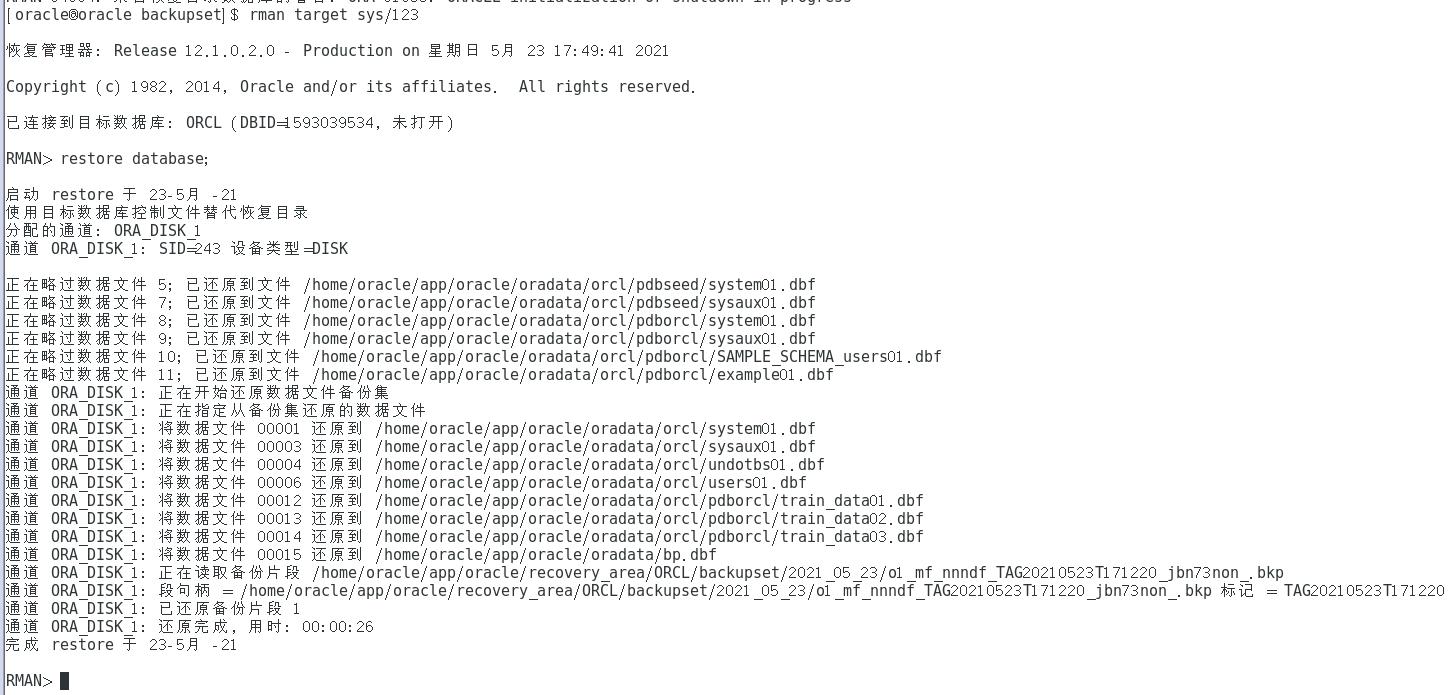
2、尝试打开数据库，此时由于删除了数据文件已经打不开了：

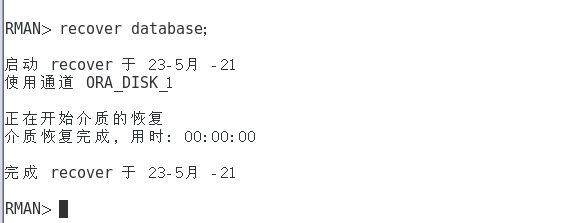


3、查看数据库状态：



4、执行恢复操作：

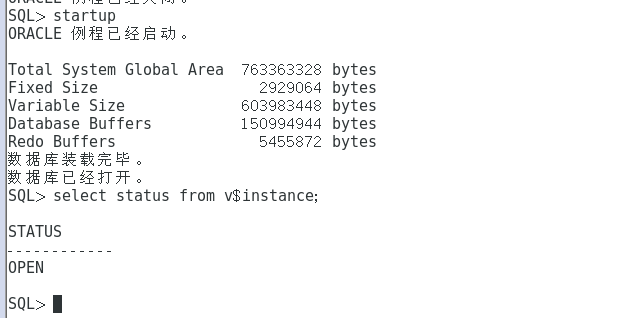






5、验证文件是否已经恢复成功





# 9.项目总结

本次项目，是对我们本学期所有Oracle学习的总结，在本次期末项目中，我自行设计了一个信息系统的数据库项目，自拟题目为逸趣行-列车售票系统名称。并且设计了项目涉及的表及表空间使用方案，总共设计了三个表空间、七张表，共插入了十万条数据。在设计权限及用户分配方案阶段，我设计了两类角色，两个用户。并且在数据库中建立一个程序包，在包中用PL/SQL语言设计了存储过程和函数，实现比较复杂的业务逻辑，用模拟数据进行执行计划分析，并且设计了手动备份方案，是基于RMAN设计的备份。

本次项目每个阶段，老师都在课程讲过相关的知识，因此在本次项目制作过程中，我的进展较为顺利。其中最有挑战性的部分，我认为是备份部分，在该部分，我尝试过自动备份，通过expdp编写脚本，并设置自动运行时间来进行备份，但是备份完毕之后的恢复过程并不理想，因此我又尝试了老师在课堂上最开始讲过的脱机备份，但是总体效率不高，需要人主动将每一个数据文件复制以作恢复。因此最后，我选择了RMAN设计的备份方案，RMAN支持多种备份手段，并且备份、恢复简单。

此次项目，我对Oracle数据库操作有了更进一层的理解。