

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的铁路售票系统的数据库设计 | | |
| 课程名称 | 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 | 蔡超 | 学号 | 201810414306 |
| 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 3](#_Toc74594366)

[2.类图设计 4](#_Toc74594367)

[3.数据库设计 5](#_Toc74594368)

[3.1admin(管理员表) 5](#_Toc74594369)

[3.2user(用户表) 6](#_Toc74594370)

[3.3station(车站表) 6](#_Toc74594371)

[3.4train\_order(订单表) 6](#_Toc74594372)

[3.5train\_num(车次表) 7](#_Toc74594373)

[3.6seat(座位表) 7](#_Toc74594374)

[3.7ticket(车票信息表) 7](#_Toc74594375)

[3.8设计方案 7](#_Toc74594376)

[3.8.1创建表空间 7](#_Toc74594377)

[3.8.2创建表 8](#_Toc74594378)

[3.8.3插入数据 12](#_Toc74594379)

[4.设计权限及用户分配方案 19](#_Toc74594380)

[4.1设计简介 19](#_Toc74594381)

[4.2代码设计 20](#_Toc74594382)

[4.3截图部分 20](#_Toc74594383)

[5.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 20](#_Toc74594384)

[6.设计手动备份方案 22](#_Toc74594385)

[6.1创建恢复目录 22](#_Toc74594386)

[6.2连接RMAN恢复目录数据库 22](#_Toc74594387)

[6.3通道分配 23](#_Toc74594388)

[6.4归档模式下备份与恢复 24](#_Toc74594389)

[6.5测试备份情况 25](#_Toc74594390)

[6.6测试恢复功能 25](#_Toc74594391)

[9.项目总结 27](#_Toc74594392)

# 1.项目简介

项目名称：铁路售票系统

本项目是基于Oracle的铁路售票系统数据库设计。

涉及角色/用户：管理员、用户

涉及表：管理员表、用户表、订单表、车次表、座位表、车站表、车票信息表

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。

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。

用户管理功能：主要指两个方面，一个是指用户对自己的信息进行管理，包括查看、修改等操作，另一个是指管理员对用户的用户管理，包括查询用户、删除用户。

订单管理功能：是由用户进行的，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。

票务查询功能：是由用户进行的，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。

登录注册功能：管理员和用户可以进行登录和注册的功能。

# 2.类图设计

图示

描述已自动生成

# 3.数据库设计

图示

描述已自动生成

## 3.1admin(管理员表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **字段** | **类型** | **主键，外键** | **可以为空** | **默认值** | **约束** | **说明** |
| admin\_id | Integer（20） | 主键 | 否 |  |  | 管理员ID |
| admin\_name | VARCHAR(100) |  | 否 |  |  | 管理员用户名 |
| admin\_password | VARCHAR(100) |  | 否 |  |  | 管理员密码 |
| admin\_phone | Integer（20） |  | 否 |  |  | 管理员电话号码 |
| admin\_real\_name | VARCHAR(100) |  | 否 |  |  | 管理员真实姓名 |
| admin\_address | VARCHAR(100) |  | 否 |  |  | 管理员居住地址 |

## 3.2user(用户表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **字段** | **类型** | **主键，外键** | **可以为空** | **默认值** | **约束** | **说明** |
| ID | Integer（20） | 主键 | 否 |  |  | 用户身份证ID |
| account | VARCHAR(100) |  | 否 |  |  | 用户账号 |
| password | VARCHAR(100) |  | 否 |  |  | 用户密码 |
| user\_name | VARCHAR(100) |  | 否 |  |  | 用户真实姓名 |
| phone\_num | Integer（20） |  | 否 |  |  | 用户手机号 |
| E-mail | VARCHAR(100) |  | 否 |  |  | 用户身份证号码 |
| vip\_level | Integer（10） |  | 是 | 空 |  | 用户等级 |
| address | VARCHAR(100) |  | 否 |  |  | 用户地址 |

## 3.3station(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 |

## 3.4train\_order(订单表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **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 |
| train\_id | VARCHAR(100) | 外键 | 否 |  |  | 车次号 |
| data | Date |  | 否 |  |  | 出发日期 |
| origin | VARCHAR(100) |  | 否 |  |  | 始发地 |
| destination | VARCHAR(100) |  | 否 |  |  | 目的地 |
| whole\_time | Time |  | 否 |  |  | 总耗时 |
| user\_name | VARCHAR(100) |  | 否 |  |  | 乘车人名字 |
| seat\_level | VARCHAR(100) |  | 否 |  |  | 座位等级 |
| seat\_num | Integer（20） |  | 否 |  |  | 座位号 |
| price | Float |  | 否 |  |  | 价格 |

## 3.5train\_num(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) | 主键 | 否 |  |  | 车次号 |
| start\_time | Date |  | 否 |  |  | 出发时间 |
| arrive\_time | Date |  | 否 |  |  | 到达时间 |
| origin | VARCHAR(100) |  | 否 |  |  | 始发地 |
| destination | VARCHAR(100) |  | 否 |  |  | 目的地 |
| whole\_time | Time |  | 否 |  |  | 总耗时 |

## 3.6seat(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) | 外键 | 否 |  |  | 车次号 |
| spare\_num | Integer（20） |  | 否 |  |  | 余座 |
| total\_num | Integer（20） |  | 否 |  |  | 总的座位数 |

## 3.7ticket(车票信息表)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **字段** | **类型** | **主键，外键** | **可以为空** | **默认值** | **约束** | **说明** |
| train\_id<FK> | Integer（20） | 外键 | 否 |  |  | 车次号 |
| data | Date |  | 否 |  |  | 出发日期 |
| origin | VARCHAR(100) |  | 否 |  |  | 始发地 |
| destination | VARCHAR(100) |  | 否 |  |  | 目的地 |
| whole\_time | Time |  | 否 |  |  | 总耗时 |
| user\_name | VARCHAR(100) |  | 否 |  |  | 乘车人姓名 |
| ID | Integer（20） |  | 否 |  |  | 乘车人身份证ID |
| seat\_level | VARCHAR(100) |  | 否 |  |  | 座位等级 |
| price | Float |  | 否 |  |  | 车票价格 |

## 3.8设计方案

### 3.8.1创建表空间

创建表空间： train\_admin、train\_user、train\_public

代码设计：

-- 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;

截图：

表格

描述已自动生成

### 3.8.2创建表

表空间中存放的表:

train\_admin: admin

train\_user: user

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

代码设计：

-- admin 表的创建

create table admin

(

admin\_id number(20) primary key,

admin\_name VARCHAR(100) not null,

admin\_password VARCHAR(100) not null,

admin\_phone number(20) not null,

admin\_real\_name VARCHAR(100) not null,

admin\_address VARCHAR(100) not null

)

tablespace train\_admin

pctfree 10

initrans 1

maxtrans 255

storage

(

initial 64K

next 1M

minextents 1

maxextents unlimited

);

-- user表的创建

create table user

(

ID number(20) primary key,

account VARCHAR(100) not null,

password VARCHAR(100) not null,

user\_name VARCHAR(100) not null,

phone\_number number(20) not null,

E-mail VARCHAR(100) not null,

vip\_level number(20),

address VARCHAR(100) not null,

)

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 VARCHAR(100) not null,

station\_addr VARCHAR(100) 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,

train\_id VARCHAR(100) not null references train\_num(train\_id),

data Date not null,

origin VARCHAR(100) not null,

destination VARCHAR(100) not null,

whole\_time Time not null,

user\_name VARCHAR(100) not null,

seat\_level VARCHAR(100) not null,

seat\_num number(20) not null,

price float 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 VARCHAR(100) primary key,

start\_time Date not null,

arrive\_time Date not null,

origin VARCHAR(100) not null,

destination VARCHAR(100) not null,

whole\_time Time 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 VARCHAR(100) 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

(

train\_id number(20) not null references train\_num(train\_id),

data date not null,

origin VARCHAR(100) not null,

destination VARCHAR(100) not null,

whole\_time time not null,

user\_name VARCHAR(100) not null,

ID number(20) not null,

seat\_level VARCHAR(100) not null,

price float 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\_ADMIN';

select TABLE\_NAME from dba\_tables where TABLESPACE\_NAME='TRAIN\_USER';

select TABLE\_NAME from dba\_tables where TABLESPACE\_NAME='TRAIN\_PUBLIC';

截图：

文本

描述已自动生成

### 3.8.3插入数据

每个表插入数据：

总共插入数据：1000000条

admin：20000条

user：20000条

station：20000条

train\_order：10000条

train\_num：10000条

seat：10000条

ticket：10000条

代码设计：

-- user表的数据插入20000条

declare

i int;

admin\_id number(20);

admin\_name VARCHAR(100);

admin\_password VARCHAR(100);

admin\_phone number(20);

admin\_real\_name VARCHAR(100);

admin\_address VARCHAR(100);

begin

i:=1;

while i<=20000

loop

admin\_id:=i;

admin\_name:= 'admin'|| i;

admin\_password := '123'|| i;

admin\_phone := '123456'|| i;

admin\_real\_name := '蔡超'||i;

admin\_address := '成都市'||i;

insert into manage(admin\_id,admin\_name,admin\_password,admin\_phone,admin\_real\_name,admin\_address) values (admin\_id,admin\_name,admin\_password,admin\_phone,admin\_real\_name,admin\_address);

i:=i+1;

end loop;

commit;

end;

/

-- user表数据插入20000

declare

i int;

ID number(20);

account VARCHAR(100);

password VARCHAR(100);

user\_name VARCHAR(100);

phone\_number number(20);

E-mail VARCHAR(100);

vip\_level number(20);

address VARCHAR(100);

begin

i:=1;

while i<=20000

loop

ID:='5111232000030'|| i;

account:= '123'|| i;

password := '123456'|| i;

user\_name := 'cc'|| i;

phone\_number := '183986'||i;

E-mail := '25122545'||i;

vip\_level := '1'||i;

address := '成都市'||i;

insert into train\_user(ID,account,password,user\_name,phone\_number,E-mail,vip\_level,address) values (ID,account,password,user\_name,phone\_number,E-mail,vip\_level,address);

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 VARCHAR(100);

station\_addr VARCHAR(100);

begin

i:=1;

while i<=20000

loop

station\_id:=i;

station\_name:= '名称'|| 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);

train\_id VARCHAR(100);

data Date;

origin VARCHAR(100);

destination VARCHAR(100);

whole\_time Time;

user\_name VARCHAR(100);

seat\_level VARCHAR(100);

seat\_num number(20);

price float;

begin

i:=1;

while i<=10000

loop

order\_id:=i;

train\_id:='123214'||i;

if i mod 6 =0 then

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

elsif i mod 6 =1 then

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

elsif i mod 6 =2 then

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

elsif i mod 6 =3 then

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

elsif i mod 6 =4 then

data:=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;

origin:= '始发地'||i;

destination:= '目的地'||i;

whole\_time:=i;

user\_name := 'cc'||i;

seat\_level:=i;

seat\_num :='B'||i mod 60;

price:='54';

insert into train\_order(order\_id,train\_id,data,origin,destination,whole\_time,user\_name,seat\_level,seat\_num,price) values (order\_id,train\_id,data,origin,destination,whole\_time,user\_name,seat\_level,seat\_num,price);

i:=i+1;

end loop;

commit;

end;

/

-- train\_num表的数据插入10000条

declare

i int;

train\_id VARCHAR(100);

start\_time Date;

arrive\_time Date;

origin VARCHAR(100);

destination VARCHAR(100);

whole\_time Time;

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;

origin:= '始发地'||i;

destination := '目的地'|| i;

whole\_time := i;

insert into train\_num(train\_id,start\_time,arrive\_time,start\_station,arrive\_station,whole\_time) values (train\_id,start\_time,arrive\_time,start\_station,arrive\_station,whole\_time);

i:=i+1;

end loop;

commit;

end;

/

-- seat表的数据的插入10000条

declare

i int;

train\_id VARCHAR(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;

train\_id number(20);

data date;

origin VARCHAR(100);

destination VARCHAR(100);

whole\_time time;

user\_name VARCHAR(100);

ID number(20);

seat\_level VARCHAR(100);

price float;

begin

i:=1;

while i<=10000

loop

train\_id:='123214'||i;

if i mod 6 =0 then

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

elsif i mod 6 =1 then

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

elsif i mod 6 =2 then

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

elsif i mod 6 =3 then

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

elsif i mod 6 =4 then

data:=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;

origin:= '始发地'||i;

destination := '目的地'|| i;

whole\_time := i;

user\_name := 'cc'||i;

ID := '5111232000030'||i;

seat\_num :='B'||i mod 60;

price:='54';

insert into ticket(train\_id,data,origin,destination,whole\_time,user\_name,ID,seat\_level,price) values (train\_id,data,origin,destination,whole\_time,user\_name,ID,seat\_level,price);

i:=i+1;

end loop;

commit;

end;

/

-- 查询是否插入成功

select count(\*) from admin;

select count(\*) from user;

select count(\*) from station;

select count(\*) from train\_order;

select count(\*) from train\_num;

select count(\*) from seat;

select count(\*) from ticket;

截图：

图形用户界面, 应用程序

中度可信度描述已自动生成

# 4.设计权限及用户分配方案

## 4.1设计简介

用户：

c##train\_manage

拥有角色：c##t\_manage

分配表空间：train\_admin、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

## 4.2代码设计

-- 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\_admin;

-- 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;

## 4.3截图部分

文本

描述已自动生成

# 5.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

包名： Train

函数名：Get\_count(order\_id\_t NUMBER)

功能：输入订单ID，查询订单表,返回与该订单对应的起始车站信息

过程名：Get\_orders(train\_id\_t VARCHAR)

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

代码设计：

-- 创建包

create or replace PACKAGE Train IS

FUNCTION Get\_count(order\_id\_t NUMBER) RETURN VARCHAR;

PROCEDURE Get\_orders(train\_id\_t VARCHAR);

END Train;

-- 创建函数和过程

create or replace PACKAGE BODY Train IS

FUNCTION Get\_count(order\_id\_t NUMBER) RETURN VARCHAR

AS

M VARCHAR2(100);

N VARCHAR2(100);

BEGIN

select train\_id into N from train\_order where order\_id=order\_id\_t;

select origin into M from train\_num where train\_id=N;

RETURN M;

END;

PROCEDURE Get\_orders(train\_id\_t VARCHAR)

AS

N NUMBER(20);

L date;

S VARCHAR(100);

R VARCHAR(100);

begin

--使用游标

DBMS\_OUTPUT.PUT\_LINE('输出起始站：');

select origin 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 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;

/

-- 测试函数

select Train.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 VARCHAR(100);

BEGIN

train\_id\_t := 1;

TrainPack.Get\_orders (train\_id\_t) ;

train\_id\_t := 2;

TrainPack.Get\_orders (train\_id\_t) ;

END;

# 6.设计手动备份方案

本项目设置手动备份方案

### 6.1创建恢复目录

代码设计：

-- 创建恢复目录：用来存储RMAN资料库的

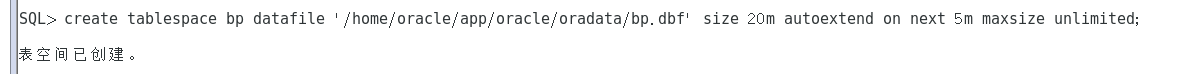
create tablespace bp datafile '/home/oracle/app/oracle/oradata/bp.dbf' size 20m autoextend on next 5m maxsize unlimited;

-- 在恢复目录数据库中创建RMAN用户并授权

create user c##cc\_r identified by 123 default tablespace bp quota unlimited on bp;

grant connect,resource,recovery\_catalog\_owner to c##cc\_r;

截图：



### 6.2连接RMAN恢复目录数据库

代码设计：

-- 连接RMAN恢复目录数据库

rman catalog c##cc\_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##cc\_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;

截图：

图片包含 图形用户界面

描述已自动生成

### 6.3通道分配

代码设计：

-- 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;

截图：

文本

描述已自动生成

### 6.4归档模式下备份与恢复

代码设计：

-- 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##cc\_r/123 target /

-- 备份和恢复整个数据库

backup database;

截图：

文本, 信件

描述已自动生成

### 6.5测试备份情况

代码设计：

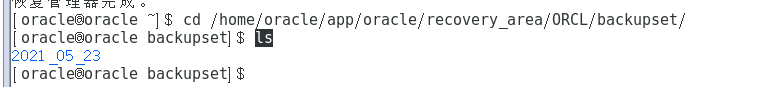
-- 切换到保存路径

cd /home/oracle/app/oracle/recovery\_area/ORCL/backupset/

-- 查看文件

ls

截图：



### 6.6测试恢复功能

代码设计：

-- 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

截图：

文本, 信件

描述已自动生成 表格

低可信度描述已自动生成 图片包含 应用程序

描述已自动生成 图片包含 表格

描述已自动生成 表格

描述已自动生成

# 9.项目总结

在本次期末项目中，我自行设计了一个铁路售票系统的数据库项目。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。本次项目所运用到的知识是对我们本学期所有Oracle学习的总结，其中所用到的每个知识点都是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。我认为最难的部分是备份部分，在该部分，通过尝试了几种方法后，我最终选择了RMAN设计的备份方案，因为RMAN支持多种备份手段，并且备份、恢复简单。通过此次项目的独立实践，我对Oracle数据库有了进一步的理解，相信在以后的实践中能学到更多。