

**期末项目设计报告**

|  |  |  |  |
| --- | --- | --- | --- |
| 题 目 | 基于Oracle的企业销售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 | 2018级 |
| 学生姓名 | 苟丞相 | 学号 | 201810414110 |
| 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** | | |  |

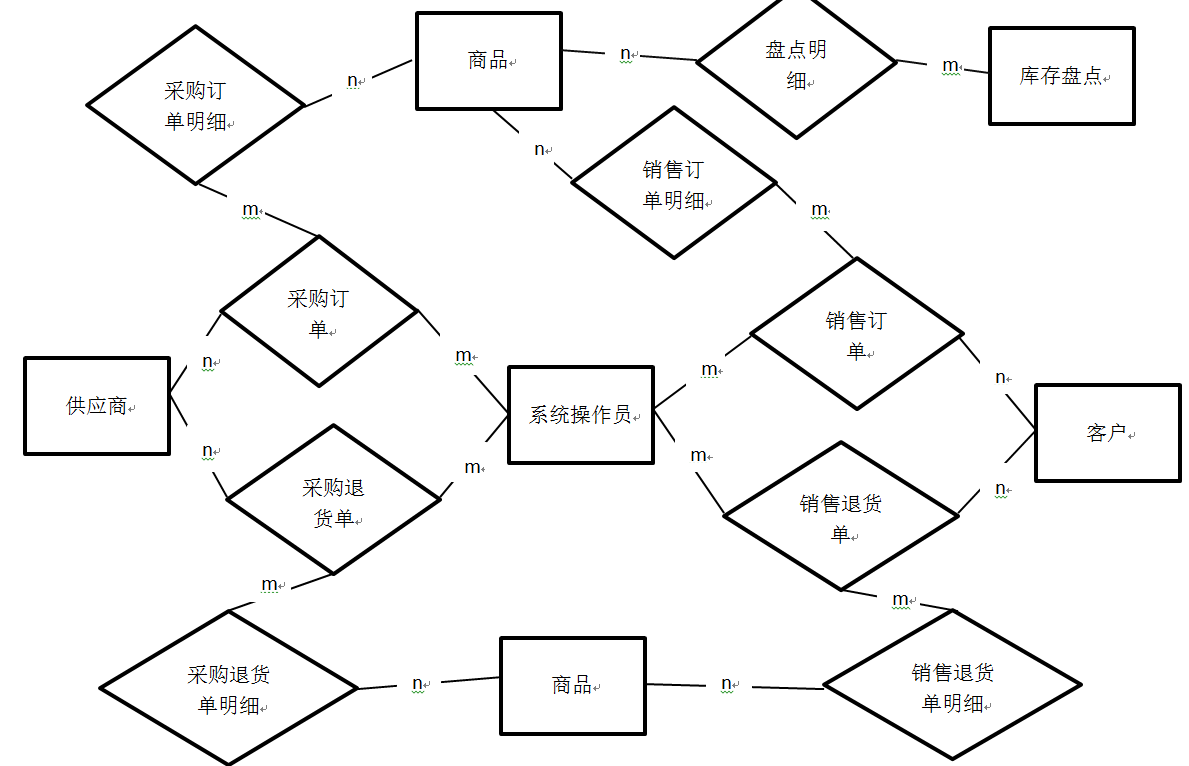
2019 年 11 月 18 日

1. 需求分析
   1. 研究背景和意义

经济大环境的变化使得规模经济的优势不再突出。固定的硬设备、人员数量、大量资金等资源投入占企业效益的比重变少，而软投入如管理、人力资源价值、服务、品牌附加值、渠道等要素资源的投入却能增加企业的效益。“速度冲击规模”的速度经济概念已经向企业提出。 那么对于日益发展壮大的企业集团，怎样才能找到一套功能强大可任意拓展、低运行成本、安全可靠的管理解决方案，来跟上企业的发展，跟上时代的发展呢? 最好的办法是利用Inter这个工具，架构自己的供需链管理平台，使无处不在的互联网为你所用，实现无处不在的管理目标。让Inter像电一样融入到企业的管理之中，提高企业在新经济时代的核心竞争力，通过对Inter的高效率使用，在激烈的市场竞争中，首先打赢第一仗“信息战”。正是基于对Inter的这种深刻认识，针对中国企业特点提出了基于Inter的网络商务管理解决方案——企业销售管理系统。

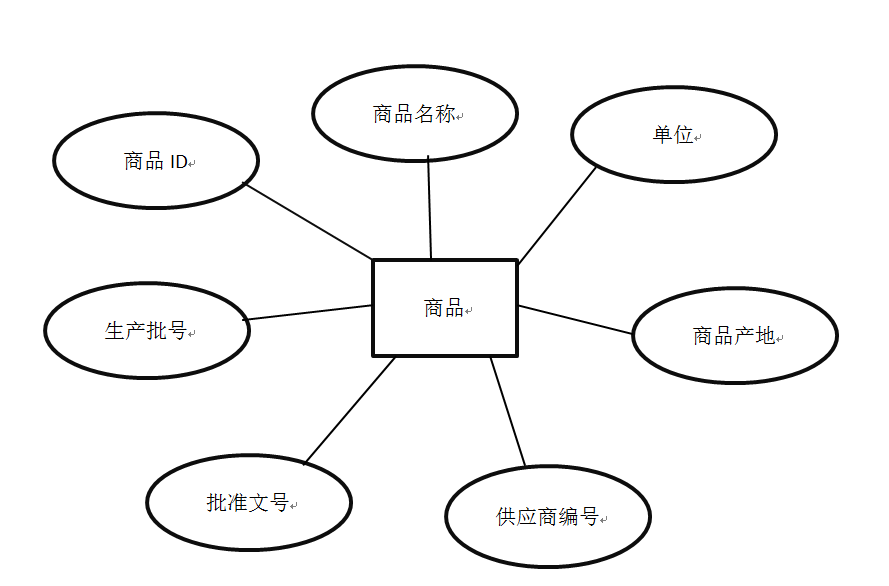
第二章 数据库设计

#### 1.总的E-R图

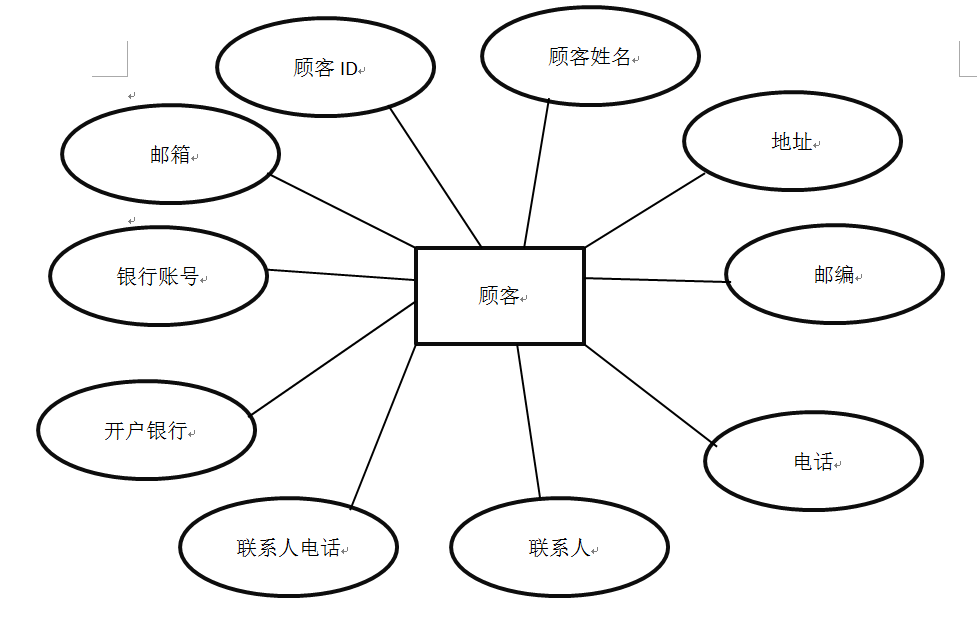


#### 2.分E-R图

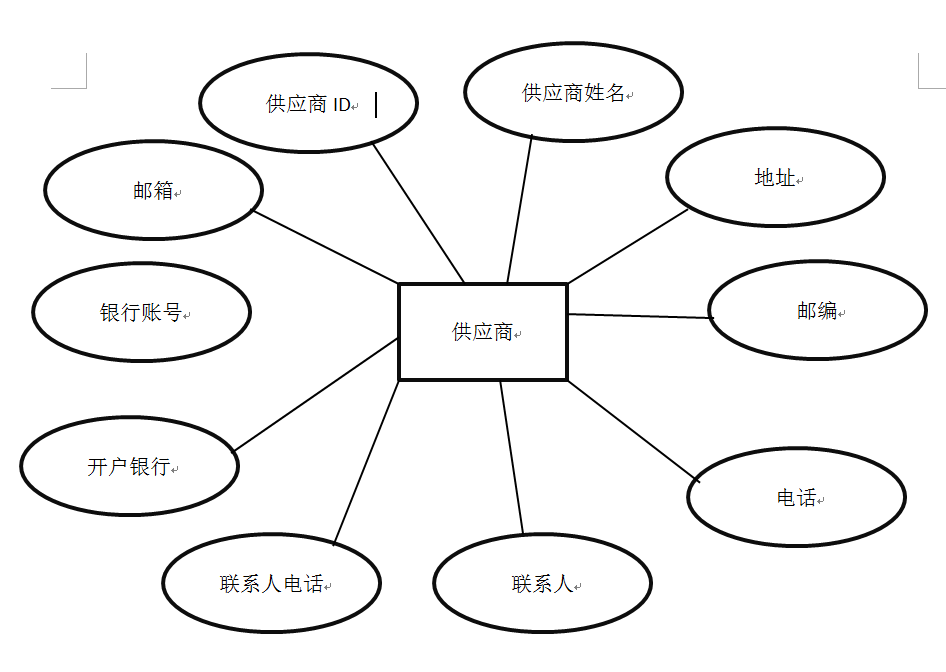
##### 2.1商品ER图



##### 2.2顾客ER图



##### 2.3供应商ER图



##### 2.4管理员ER图



#### 3.数据库表

##### 3.1系统数据库表

| **编号** | **表名** | **描述** |
| --- | --- | --- |
| 01 | customer | 顾客表 |
| 02 | dept | 部门表 |
| 03 | employee | 员工表 |
| 04 | goods | 商品表 |
| 05 | manager | 管理员表 |
| 06 | purchase\_item | 采购单明细表 |
| 07 | purchase\_order | 采购订单表 |
| 08 | return\_item | 采购退货明细表 |
| 09 | return\_order | 采购退货表 |
| 10 | sale\_item | 销售订单明细表 |
| 11 | sale\_order | 销售订单表 |
| 12 | sale\_return\_item | 退货销售明细表 |
| 13 | sale\_return\_order | 销售退货表 |
| 14 | stock | 库存表 |
| 15 | suppliers | 供应商表 |

##### 3.2顾客表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| customer\_id | 顾客编号 | NUMBER(20) | PK |  |
| customer\_name | 顾客名称 | Varchar(20) | NOT NULL |  |
| address | 顾客地址 | Varchar(255) |  |  |
| zip | 邮编 | Varchar(20) |  |  |
| telPhone | 联系电话 | Varchar(20) |  |  |
| linkMan | 联系人 | Varchar(20) |  |  |
| linkTel | 联系人号码 | Varchar(20) |  |  |
| bank | 开户银行 | Varchar(20) |  |  |
| bankAccoount | 银行账号 | NUMBER(20) |  |  |
| email | 邮箱 | Varchar(20) |  |  |

##### 3.2部门表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| dept\_id | 部门编号 | NUMBER(20) | PK |  |
| dept\_name | 顾客名称 | Varchar(20) | NOT NULL |  |

##### 3.3员工表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| employee\_id | 员工编号 | NUMBER(20) | PK |  |
| employee\_name | 员工姓名 | Varchar(20) | NOT NULL |  |
| employee\_password | 密码 | Varchar(20) | NOT NULL | ＭＤ５加密 |
| employee\_gender | 性别 | Varchar(5) |  |  |
| employee\_age | 年龄 | NUMBER(20) |  |  |
| employee\_phonenumber | 手机号码 | Varchar(20) |  |  |
| deptId | 部门ID | NUMBER(20) | FK |  |

##### 3.4商品表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| goods\_id | 商品编号 | NUMBER(20) | PK |  |
| goods\_name | 商品名称 | Varchar(20) | NOT NULL |  |
| unit | 单位 | Varchar(20) |  |  |
| space | 商品产地 | Varchar(255) |  |  |
| supplierId | 供应商编号 | Varchar(20) | FK |  |
| approveId | 批准文号 | Varchar(20) |  |  |
| batchId | 生产批号 | Varchar(20) |  |  |

##### 3.5管理员表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| manager\_id | 管理员编号 | NUMBER(20) | PK |  |
| manager\_name | 管理员姓名 | Varchar(20) | NOT NULL |  |
| manager\_password | 密码 | Varchar(20) | NOT NULL | ＭＤ５加密 |
| manager\_gender | 性别 | Varchar(5) |  |  |
| manager\_age | 年龄 | NUMBER(20) |  |  |
| manager\_phonenumber | 手机号码 | Varchar(20) |  |  |
| dept\_name | 部门名称 | Varchar(255) | FK |  |

##### 3.6采购单明细表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| purchase\_order\_id | 采购单编号 | NUMBER(20) | PK |  |
| suppliersId | 采购商编号 | Varchar(20) | FK |  |
| purchase\_order\_pay | 支付方式 | Varchar(20) |  |  |
| inDate | 采购日期 | Date |  |  |
| purchase\_order\_\_total | 总金额 | NUMBER(20,2) |  |  |

##### 3.7采购退货明细表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| return\_item\_id | 采购退货单明细编号 | NUMBER(20) | PK |  |
| returnOrderId | 采购退货单编号 | NUMBER(20) | FK |  |
| goodsName | 商品名称 | Varchar(20) | FK |  |
| return\_item\_price | 单价 | NUMBER(20,2) |  |  |
| return\_item\_count | 数量 | NUMBER(20) |  |  |

##### 3.8销售订单明细表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| sale\_item\_id | 销售订单明细编号 | NUMBER(20) | PK |  |
| saleOrderId | 销售订单编号 | NUMBER(20) | FK |  |
| sale\_item\_name | 商品名称 | Varchar(20) | FK |  |
| sale\_item\_price | 单价 | NUMBER(20,2) |  |  |
| sale\_item\_count | 数量 | NUMBER(20) |  |  |

##### 3.9销售订单表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| sale\_order\_id | 销售单编号 | NUMBER(20) | PK |  |
| customerId | 顾客编号 | Varchar(20) | FK |  |
| sale\_order\_pay | 支付方式 | Varchar(20) |  |  |
| sale\_order\_saleDate | 销售日期 | Date |  |  |
| sale\_order\_total | 总金额 | NUMBER(20,2) |  |  |

##### 3.10销售退货明细表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| sale\_return\_item\_id | 销售退货订单明细编号 | NUMBER(20) | PK |  |
| saleReturnOrderId | 销售退货订单编号 | NUMBER(20) | FK |  |
| goodsName | 商品名称 | Varchar(20) | FK |  |
| sale\_return\_item\_price | 单价 | NUMBER(20,2) |  |  |
| sale\_return\_item\_count | 数量 | NUMBER(20) |  |  |

##### 3.11销售退货表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| sale\_return\_order\_id | 销售退货单编号 | NUMBER(20) | PK |  |
| customerId | 顾客ＩＤ | Varchar(20) | FK |  |
| sale\_return\_order\_pay | 支付方式 | Varchar(20) |  |  |
| sale\_return\_order\_returnDate | 退货日期 | Date |  |  |
| sale\_return\_order\_total | 总金额 | NUMBER(20,2) |  |  |

##### 3.12库存表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| stock\_id | 库存编号 | NUMBER(20) | PK |  |
| goodsId | 商品编号 | NUMBER(20) | FK |  |
| stock\_count | 商品库存数量 | Varchar(20) | NOT NULL |  |
| stock\_area | 存放地方 | Varchar(255) |  |  |

##### 3.13供应商表

| **字段名** | **字段描述** | **类型/长度** | **约束** | **备注** |
| --- | --- | --- | --- | --- |
| suppliers\_id | 供应商编号 | NUMBER(20) | PK |  |
| supplier\_name | 供应商名称 | Varchar(20) | NOT NULL |  |
| address | 顾客地址 | Varchar(255) |  |  |
| zip | 邮编 | Varchar(20) |  |  |
| telPhone | 联系电话 | Varchar(20) |  |  |
| linkMan | 联系人 | Varchar(20) |  |  |
| linkTel | 联系人号码 | Varchar(20) |  |  |
| bank | 开户银行 | Varchar(20) |  |  |
| bankAccoount | 银行账号 | NUMBER(20) |  |  |
| email | 邮箱 | Varchar(20) |  |  |

第三章 实验过程

4.1 首先，在localhost\_pdborcl\_perfectism用户下创建四个表空间：

create tablespace users02 datafile '/home/oracle/app/oracle/oradata/orcl/users02\_1.dbf'

size 100m autoextend on next 50m maxsize unlimited,

'/home/oracle/app/oracle/oradata/orcl/users02\_2.dbf' size 100m autoextend on next 50m

maxsize unlimited extent management local segment space management auto;

create tablespace users03 datafile '/home/oracle/app/oracle/oradata/orcl/users03\_1.dbf'

size 100m autoextend on next 50m maxsize unlimited,

'/home/oracle/app/oracle/oradata/orcl/users03\_2.dbf' size 100m autoextend on next 50m

maxsize unlimited extent management local segment space management auto;

create tablespace users04 datafile '/home/oracle/app/oracle/oradata/orcl/users04\_1.dbf'

size 100m autoextend on next 50m maxsize unlimited,

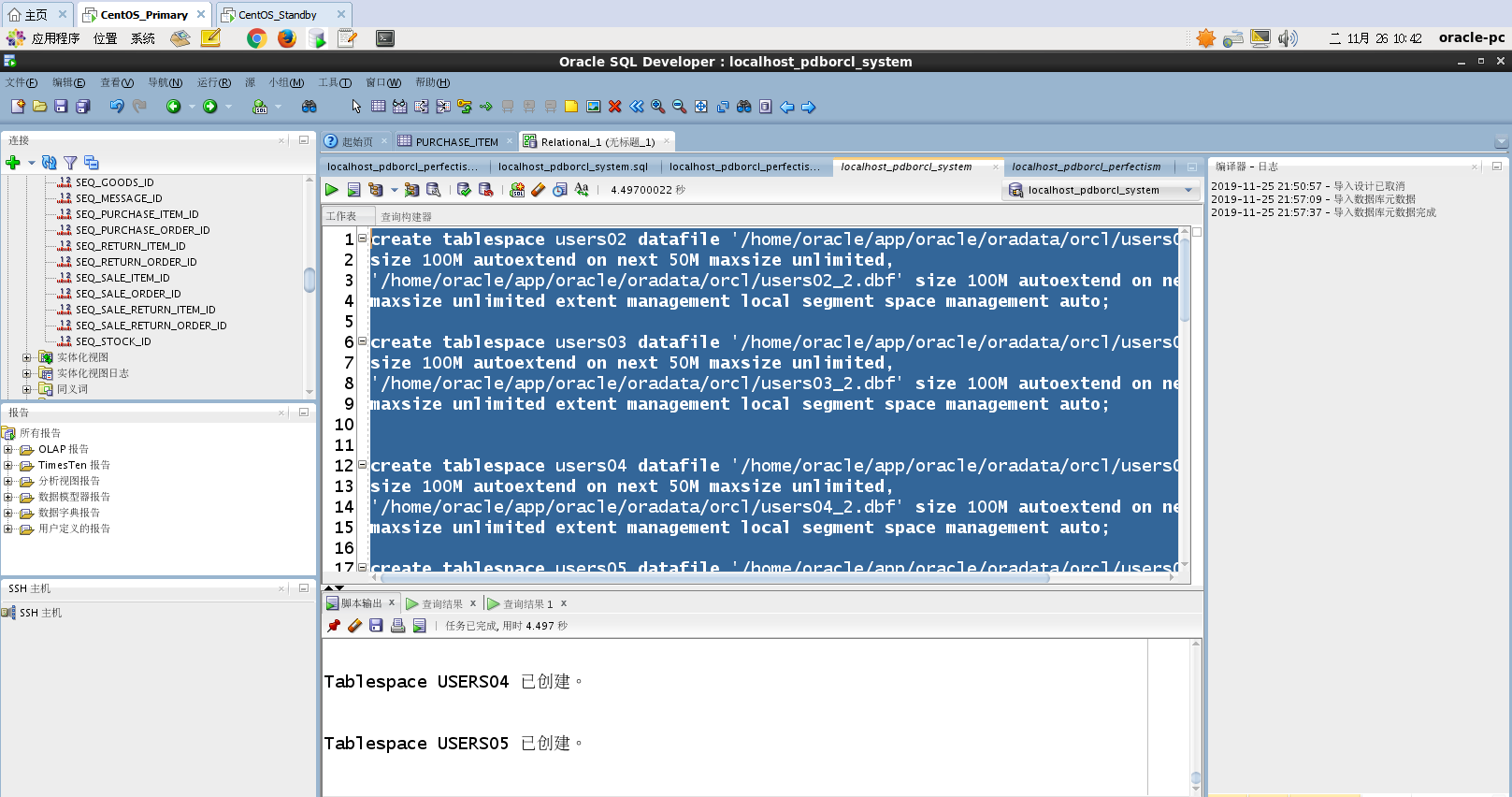
'/home/oracle/app/oracle/oradata/orcl/users04\_2.dbf' size 100m autoextend on next 50m

maxsize unlimited extent management local segment space management auto;

create tablespace users05 datafile '/home/oracle/app/oracle/oradata/orcl/users05\_1.dbf'

size 100m autoextend on next 50m maxsize unlimited,

'/home/oracle/app/oracle/oradata/orcl/users05\_2.dbf' size 100m autoextend on next 50m

maxsize unlimited extent management local segment space management auto;

分析：上述sql语句下用户localhost\_pdborcl\_perfectism下创建了四个表空间，分别是USERS0,USERS02,USERS03,USERS04,USERS05。

1. 在主库primary上在localhost\_pdborcl\_system下给创建用户perfectism 、outstand

--创建了两个角色、两个用户，并将角色的权限授予用户

select \* from new\_user;

select \* from dba\_roles ;

create role con\_res\_view\_perfectism;

create role sel\_del\_ins\_perfectism;

grant connect,resource,create view to con\_res\_view\_perfectism;

grant select ,delete,insert on userlog to con\_res\_view\_perfectism;

create user perfectism identified by 123 default tablespace users temporary tablespace temp;

create user outstand identified by 123 default tablespace users temporary tablespace temp;

alter user perfectism quota 50m on users;

alter user outstand quota 50m on users;

grant con\_res\_view\_perfectism to perfectism;

grant sel\_del\_ins\_perfectism to perfectism;

grant con\_res\_view\_perfectism to outstand;

grant sel\_del\_ins\_perfectism to outstand;

alter user perfectism quota unlimited on users;

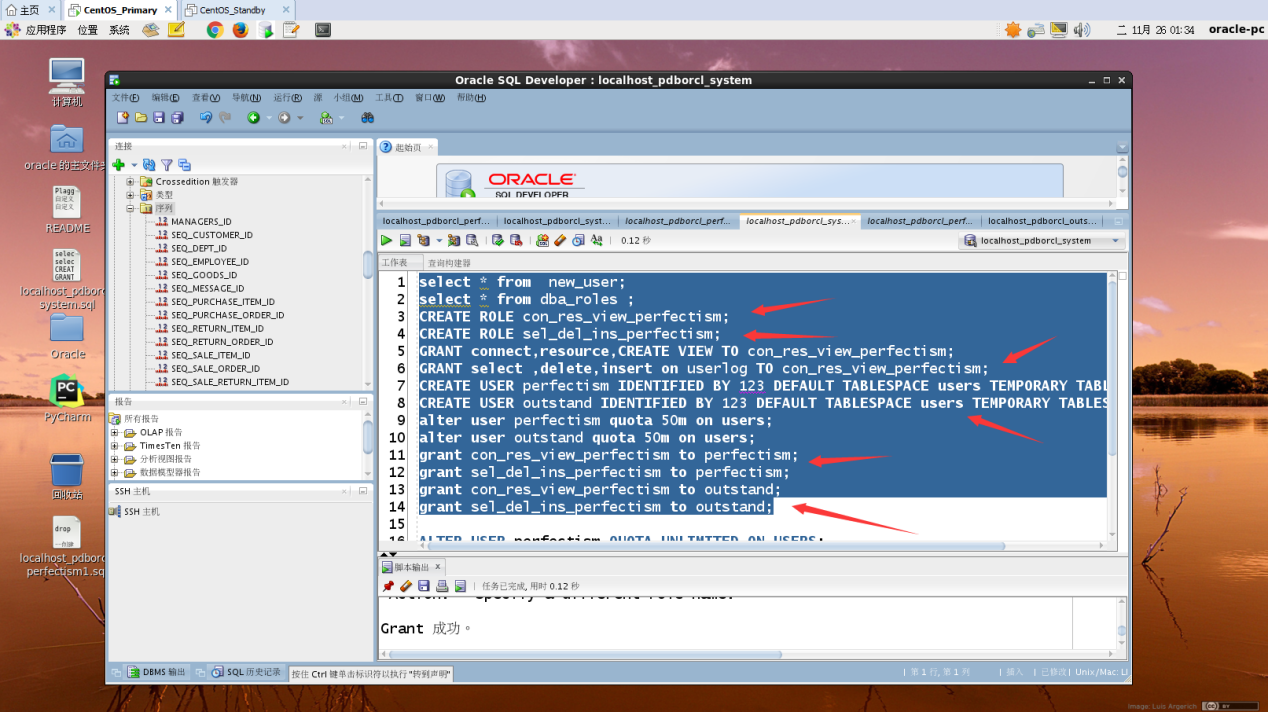
alter user perfectism quota unlimited on users02;

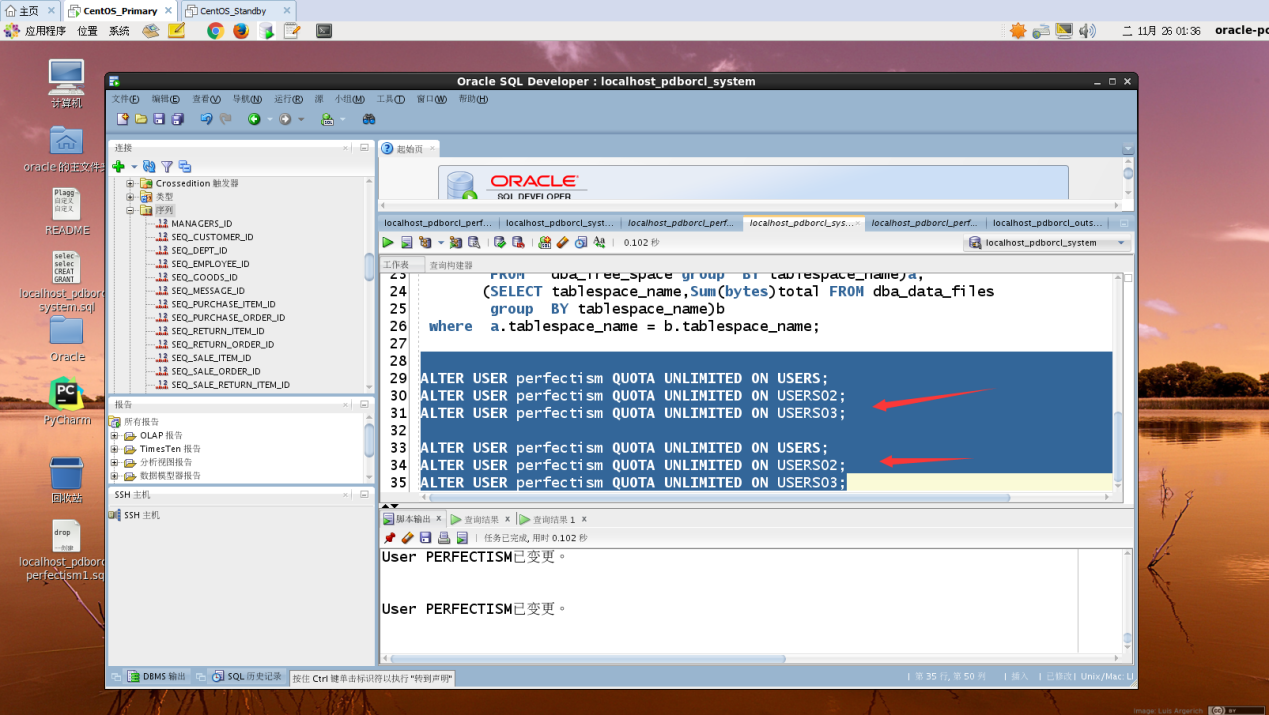
alter user perfectism quota unlimited on users03;

alter user perfectism quota unlimited on users;

alter user perfectism quota unlimited on users02;

alter user perfectism quota unlimited on users03;





分析：上面的语句就是新用户和所有的dba里面的角色，角色是权限的集合，接着创建了两个角色、两个用户，并将角色的权限授予用户,用户的默认表空间是users,用户perfectism和用户outstand

都有connect，resource角色的权限，创建视图的权限、对userlog表增、删、查的权限。

connect的默认权限:ALTER SESSION CREATE CLUSTER CREATE DATABASE LINK CREATE SEQUENCE CREATE SESSION CREATE SYNONYM CREATE TABLE CREATE VIEW

resource 的默认权限:CREATE CLUSTER CREATE INDEXTYPE CREATE OPERATOR CREATE PROCEDURE CREATE SEQUENCE CREATE TABLE CREATE TRIGGER CREATE TYPE。

查询表空间：

SELECT a.tablespace\_name "表空间名",Total/1024/1024 "大小MB",

free/1024/1024 "剩余MB",( total - free )/1024/1024 "使用MB",

Round(( total - free )/ total,4)\* 100 "使用率%"

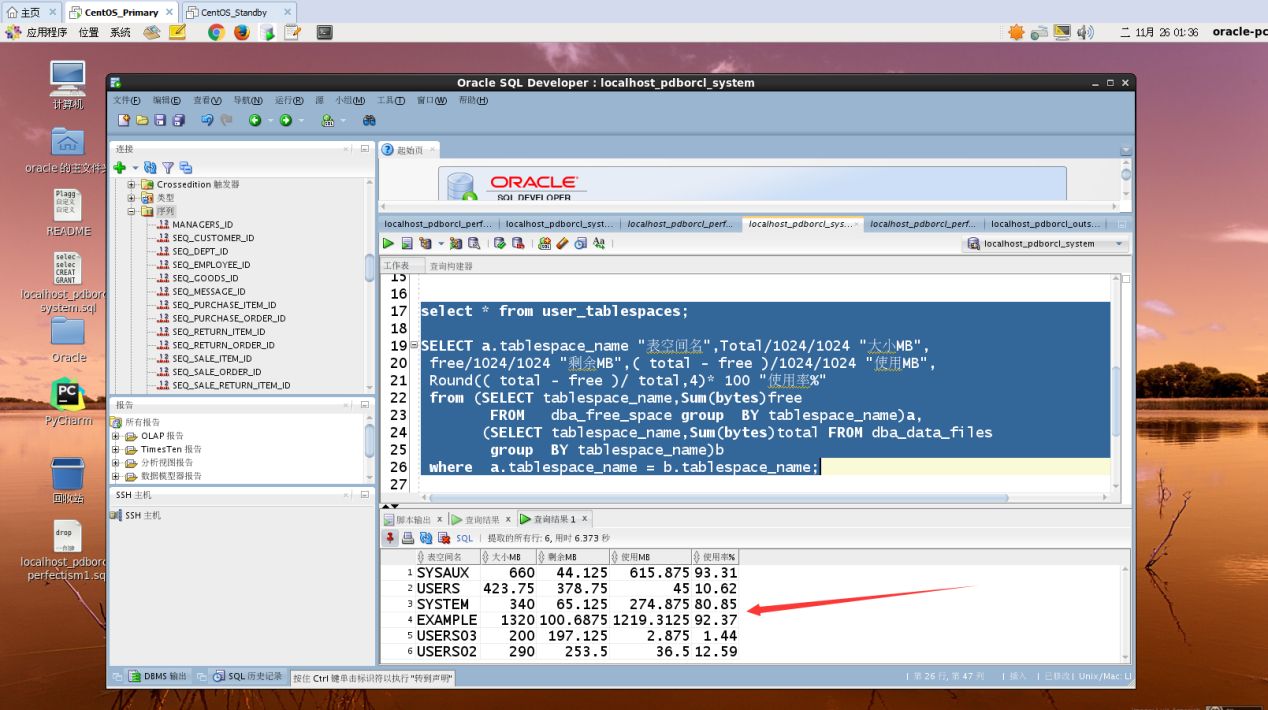
from (SELECT tablespace\_name,Sum(bytes)free

FROM dba\_free\_space group BY tablespace\_name)a,

(SELECT tablespace\_name,Sum(bytes)total FROM dba\_data\_files

group BY tablespace\_name)b

where a.tablespace\_name = b.tablespace\_name;

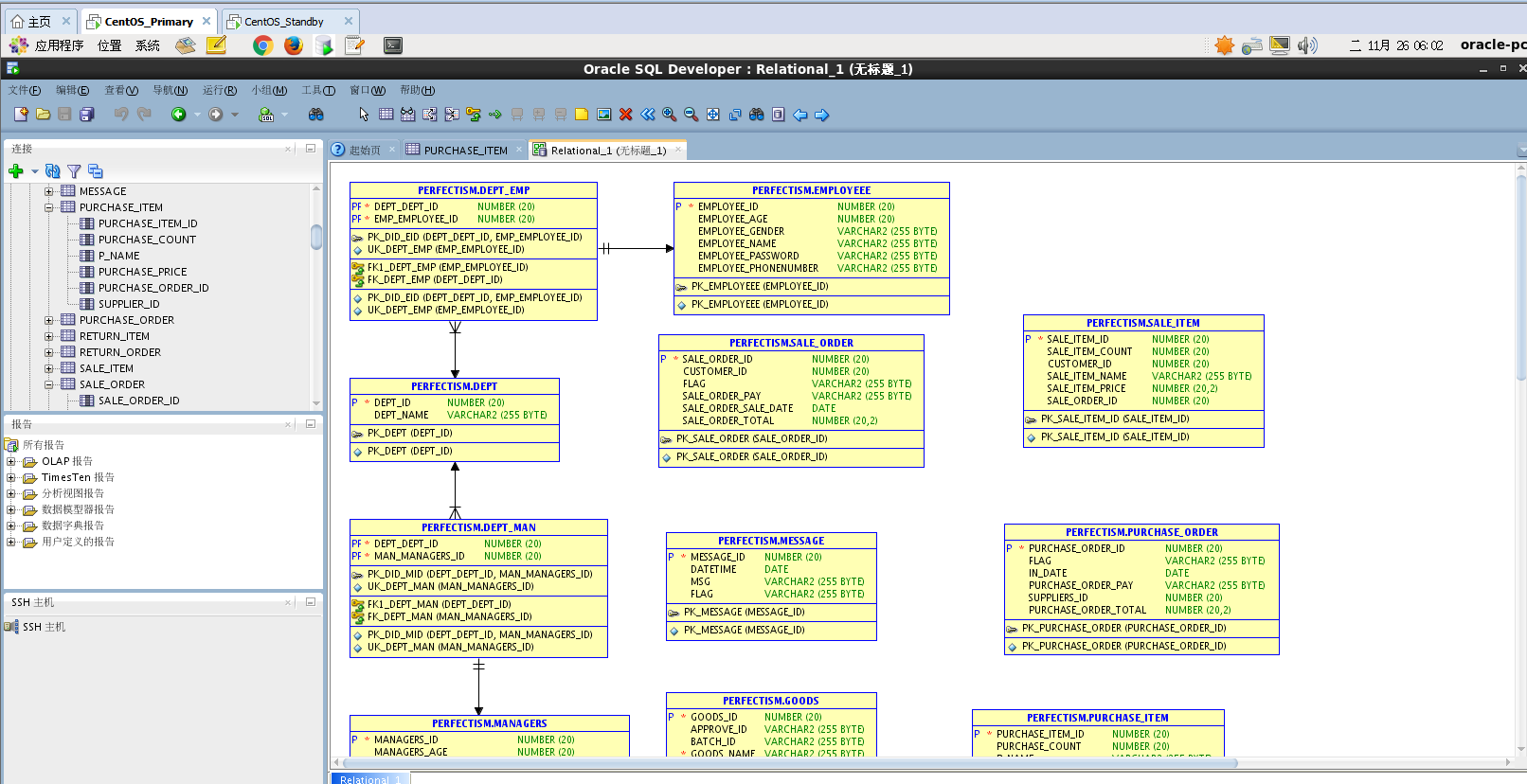


4.2 以用户perfectism 下做此次实验

在用户perfectism下创建18张表

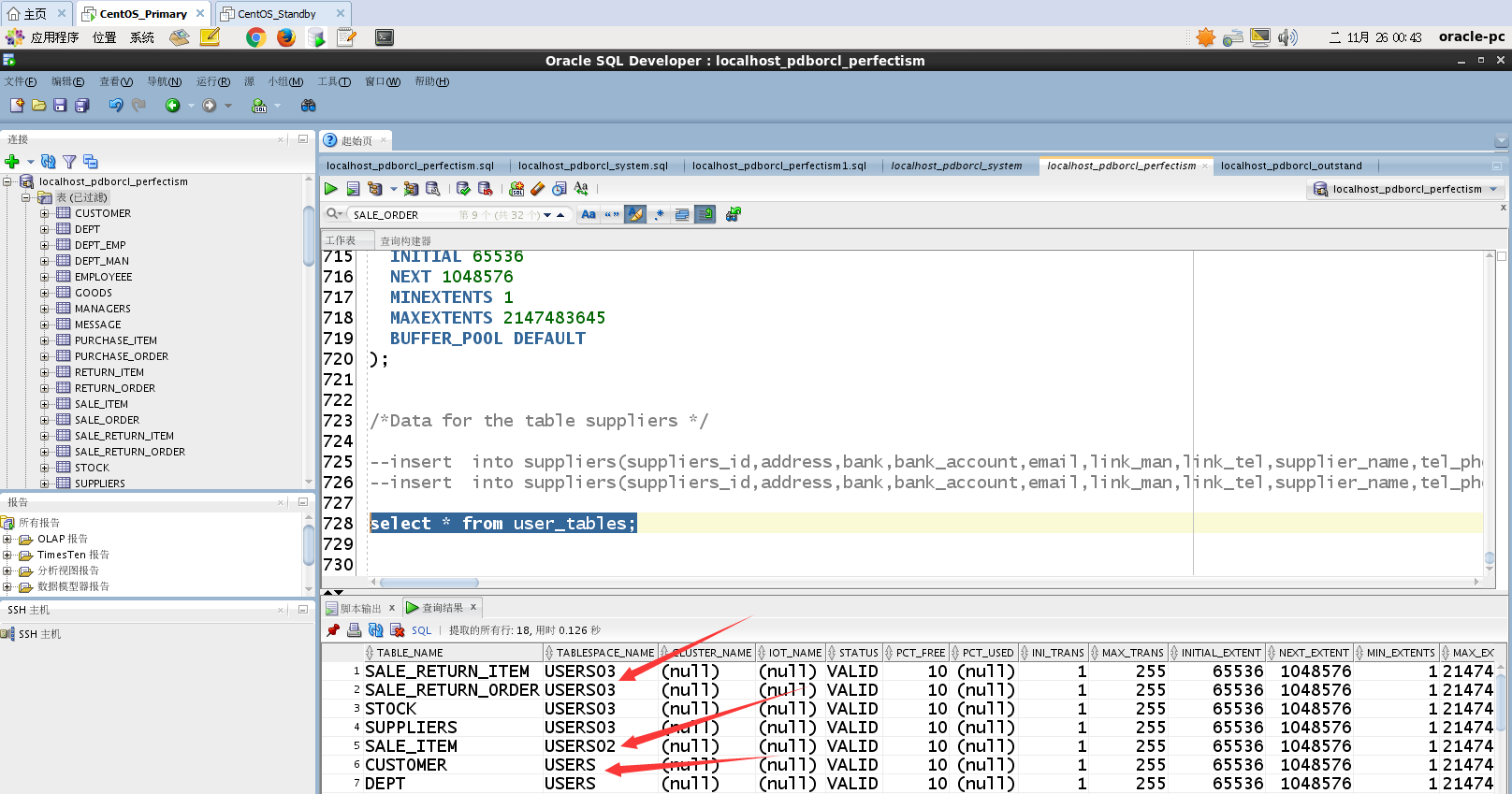
SALE\_RETURN\_ITEM、SALE\_RETURN\_ORDER 、STOCK 、 SUPPLIERS 、 SALE\_ITEM、 CUSTOMER、 DEPT、 DEPT\_EMP、 DEPT\_MAN 、 EMPLOYEEE、 GOODS、 MANAGERS 、MESSAGE 、 PURCHASE\_ITEM 、RETURN\_ITEM RETURN\_ORDER、PURCHASE\_ORDER、 SALE\_ORDER

关系图如下：



查询用户perfectism下的所有表信息

Select \* from user\_tables;



分析：从上面的截图可以看出，这些表处于不同的表空间，USERS,USERS02,USERS03都有表存在。

以其中插入了大量数据（共10万条数据，每张表2万条）的5张表来说明：

sale\_order表、stock表、purchase\_item表、customer表、goods表

--创建表sale\_order 创建主键为：PK\_SALE\_ORDER (sale\_order\_id)

create table sale\_order (

sale\_order\_id number(20) not null ,

customer\_id number(20) default null,

flag varchar(255) default null,

sale\_order\_pay varchar(255) default null,

sale\_order\_sale\_date date default null,

sale\_order\_total number(20,2) default null,

constraint pk\_sale\_order primary key (sale\_order\_id)

)

tablespace users

pctfree 10

initrans 1

storage

(

buffer\_pool default

)

nocompress

noparallel

partition by range (sale\_order\_sale\_date)

(

partition partition\_before\_2016 values less than (to\_date(' 2016-01-01 00:00:00', 'syyyy-mm-dd hh24:mi:ss', 'nls\_calendar=gregorian'))

nologging

tablespace users

pctfree 10

initrans 1

storage

(

initial 8388608

next 1048576

minextents 1

maxextents unlimited

buffer\_pool default

)

nocompress no inmemory

, partition partition\_before\_2017 values less than (to\_date(' 2017-01-01 00:00:00', 'syyyy-mm-dd hh24:mi:ss', 'nls\_calendar=gregorian'))

nologging

tablespace users02

pctfree 10

initrans 1

storage

(

initial 8388608

next 1048576

minextents 1

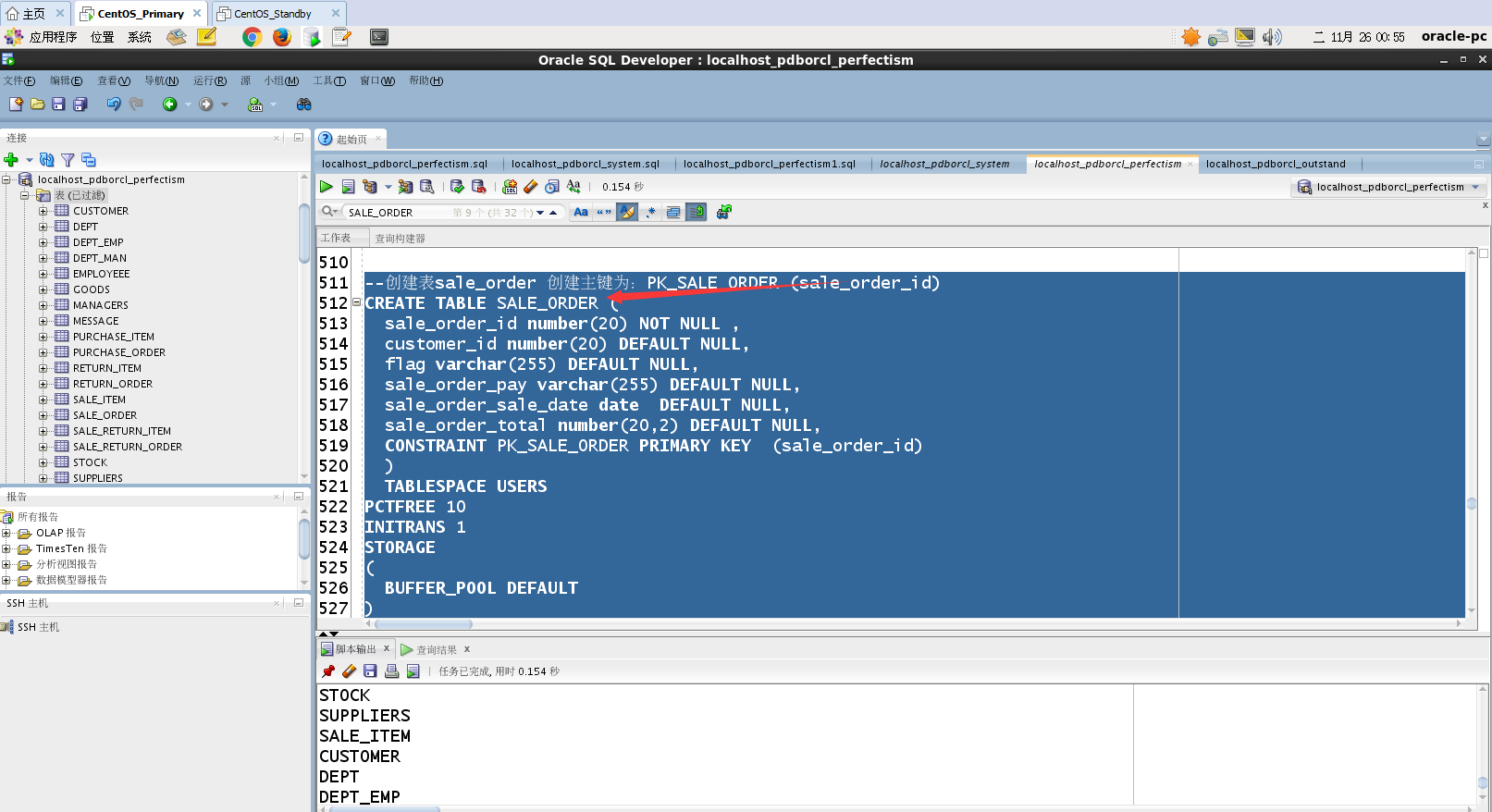
maxextents unlimited

buffer\_pool default

)

nocompress no inmemory

);



--创建表stock ，创建主键为：PK\_STOCK (stock\_id)

create table stock (

stock\_id number(20) not null ,

stock\_area varchar(255) default null,

stock\_count number(20) default null,

goods\_id number(20) default null,

constraint pk\_stock primary key (stock\_id)

)

logging

tablespace users03

pctfree 10

initrans 1

storage

(

initial 65536

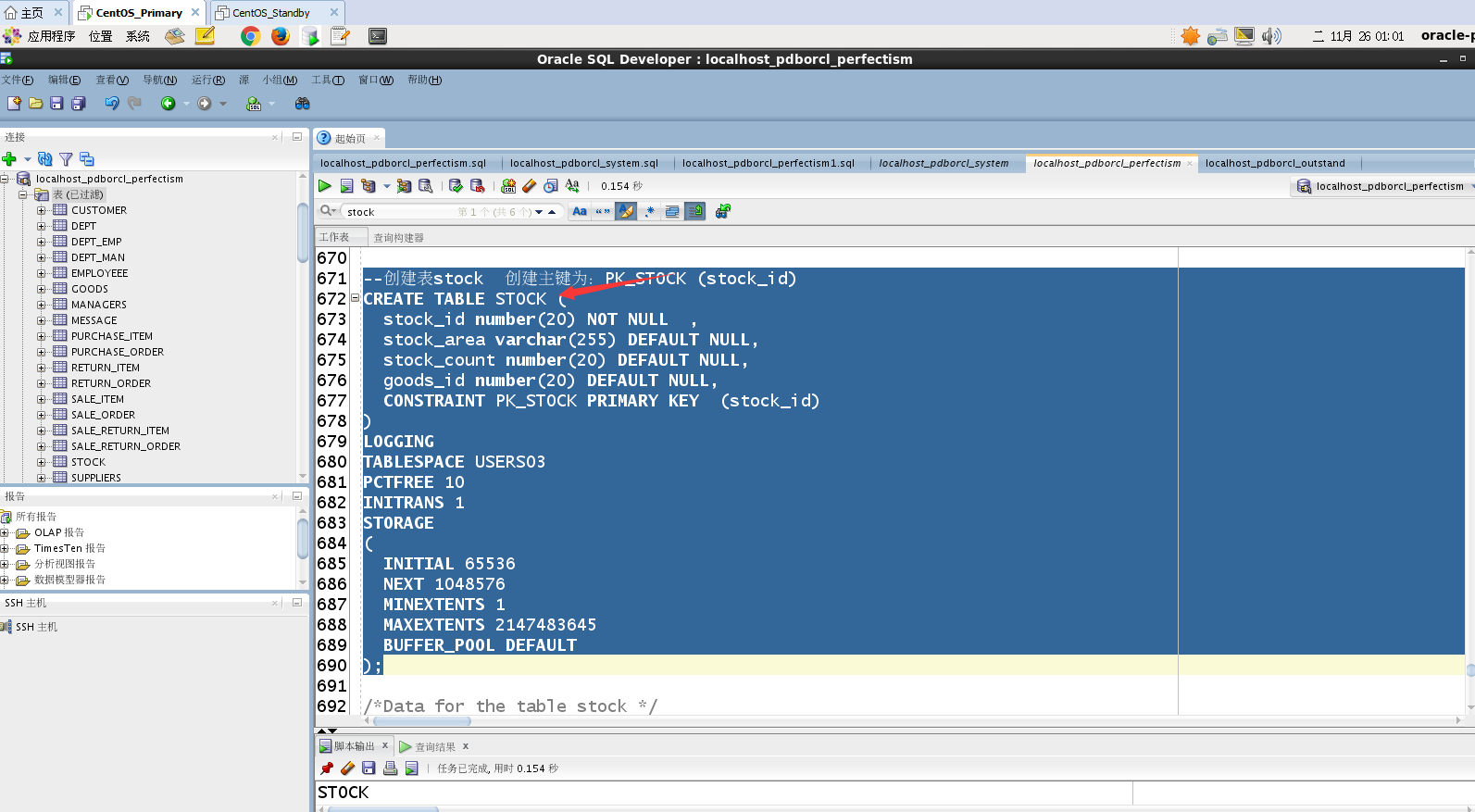
next 1048576

minextents 1

maxextents 2147483645

buffer\_pool default

);



--创建表purchase\_item 创建主键为：pk\_purchase\_item (purchase\_item\_id)

create table purchase\_item (

purchase\_item\_id number(20) not null,

purchase\_count number(20) default null,

p\_name varchar(255) default null,

purchase\_price number(20，2) default null,

purchase\_order\_id number(20) default null,

supplier\_id number(20) default null,

constraint pk\_purchase\_item primary key (purchase\_item\_id)

)

nologging

tablespace users

pctfree 10

initrans 1

storage

(

initial 65536

next 1048576

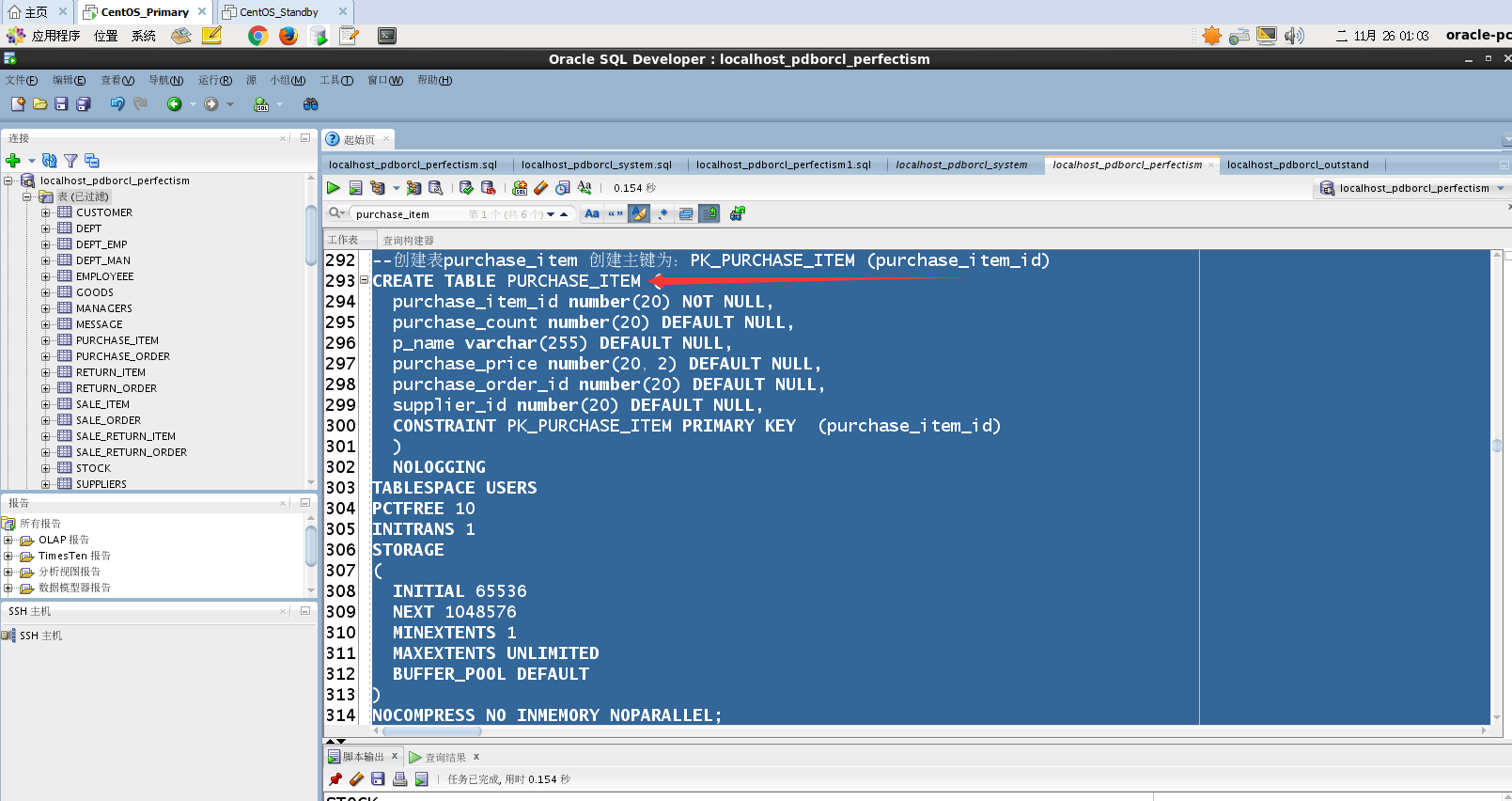
minextents 1

maxextents unlimited

buffer\_pool default

)

nocompress no inmemory noparallel;



--创建表customer：主键为：pk\_customer(customer\_id)

create table customer(

customer\_id number(20) not null ,

address varchar2(255) default null,

bank varchar2(255) default null,

bank\_account number(20) default null,

email varchar2(255) default null,

link\_man varchar2(255) default null,

link\_tel varchar2(255) default null,

customer\_name varchar2(255) default null,

tel\_phone varchar2(255) default null,

zip varchar(255) default null,

constraint pk\_customer primary key (customer\_id)

using index

(

create unique index pk\_customer on customer (customer\_id asc)

nologging

tablespace users

pctfree 10

initrans 2

storage

(

initial 65536

next 1048576

minextents 1

maxextents unlimited

buffer\_pool default

)

noparallel

)

enable

)

nologging

tablespace users

pctfree 10

initrans 1

storage

(

initial 65536

next 1048576

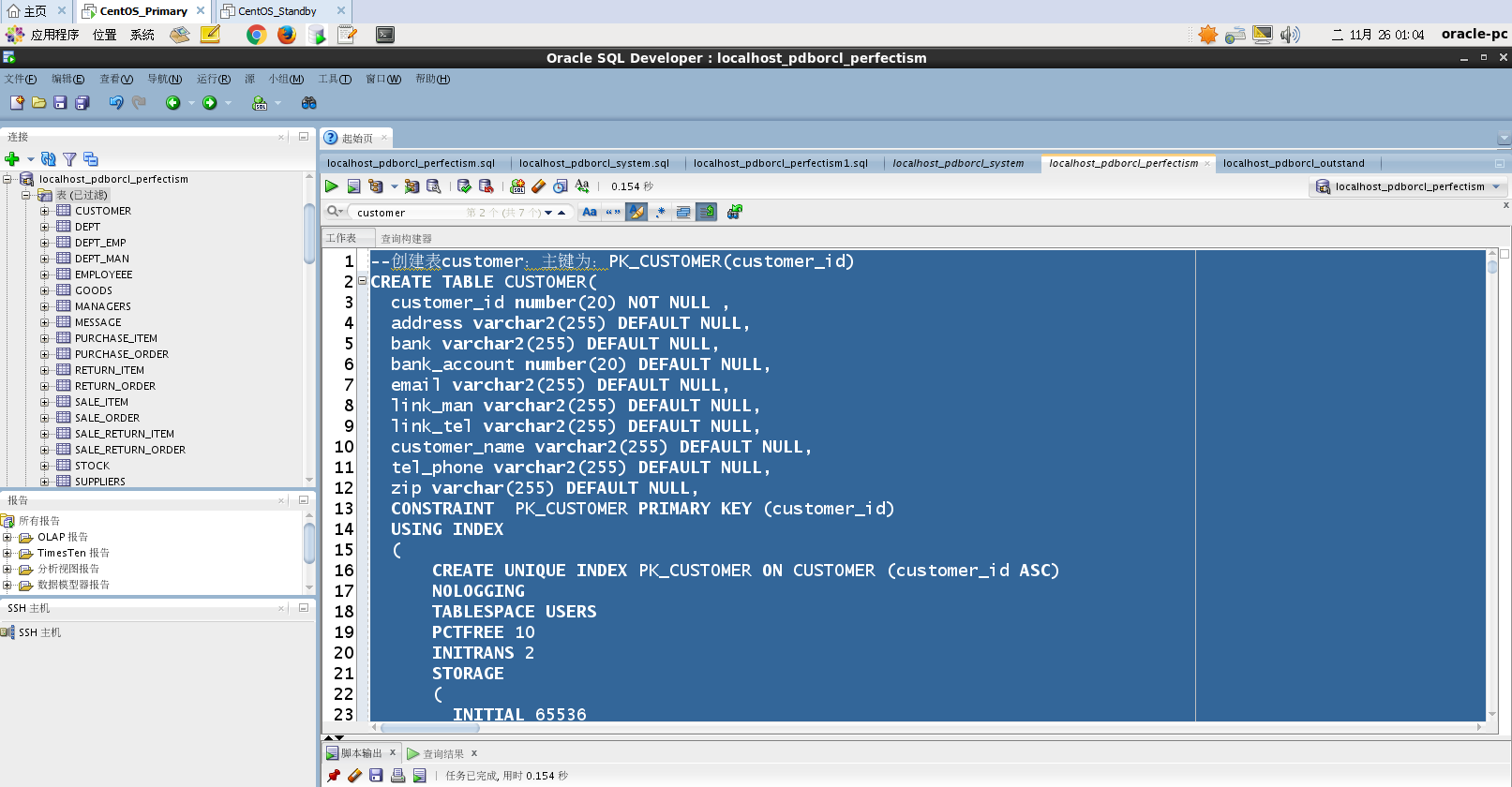
minextents 1

maxextents unlimited

buffer\_pool default

)

nocompress no inmemory noparallel;



--创建表goods 创建主键为：pk\_goods(goods\_id),唯一约束：uk\_goods(goods\_name)

create table goods (

goods\_id number(20) not null,

approve\_id varchar(255) default null,

batch\_id varchar(255) default null,

goods\_name varchar(255) not null,

price number(10,2) default null,

space varchar(255) default null,

supplier\_id number(20) default null,

unit varchar(255) default null,

constraint pk\_goods primary key (goods\_id)

)nologging

tablespace users

pctfree 10

initrans 1

storage

(

initial 65536

next 1048576

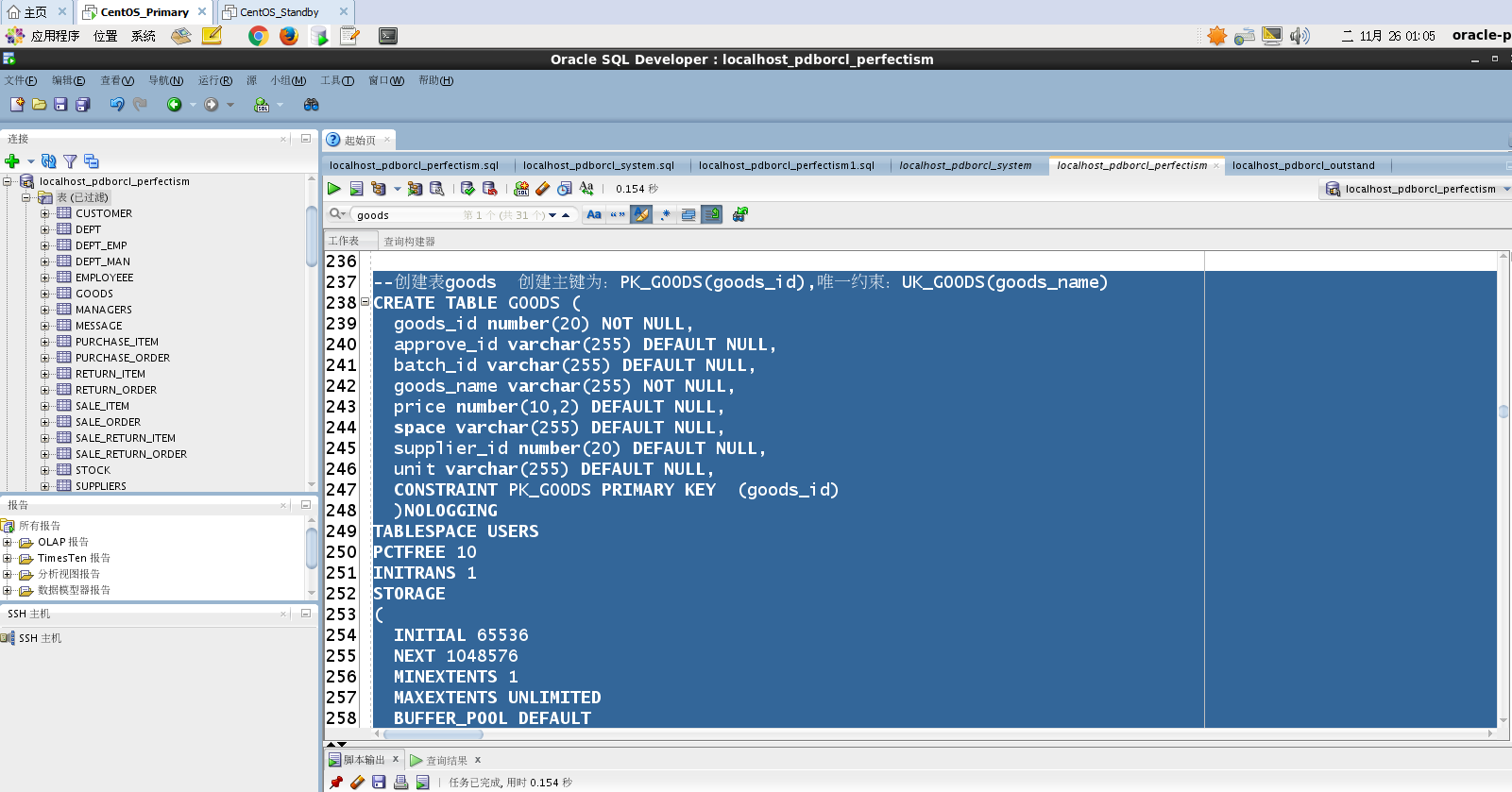
minextents 1

maxextents unlimited

buffer\_pool default

)

nocompress no inmemory noparallel;



由于其中有id自增的表，所以就对那些表设置了序列，具体有这些表：CUSTOMER、 DEPT EMPLOYEE 、GOODS、 MANAGERS、 MESSAGE、 PURCHASE\_ITEM、 PURCHASE\_ORDER 、RETURN\_ITEM、 RETURN\_ORDER、 SALE\_ITEM、 SALE\_ORDER、 SALE\_RETURN\_ITEM、 SALE\_RETURN\_ORDER、 STOCK

--实现自增

--创建序列

create sequence "seq\_customer\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_dept\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_employee\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_goods\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "managers\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_message\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_purchase\_item\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_purchase\_order\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_return\_item\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_return\_order\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_sale\_item\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_sale\_order\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

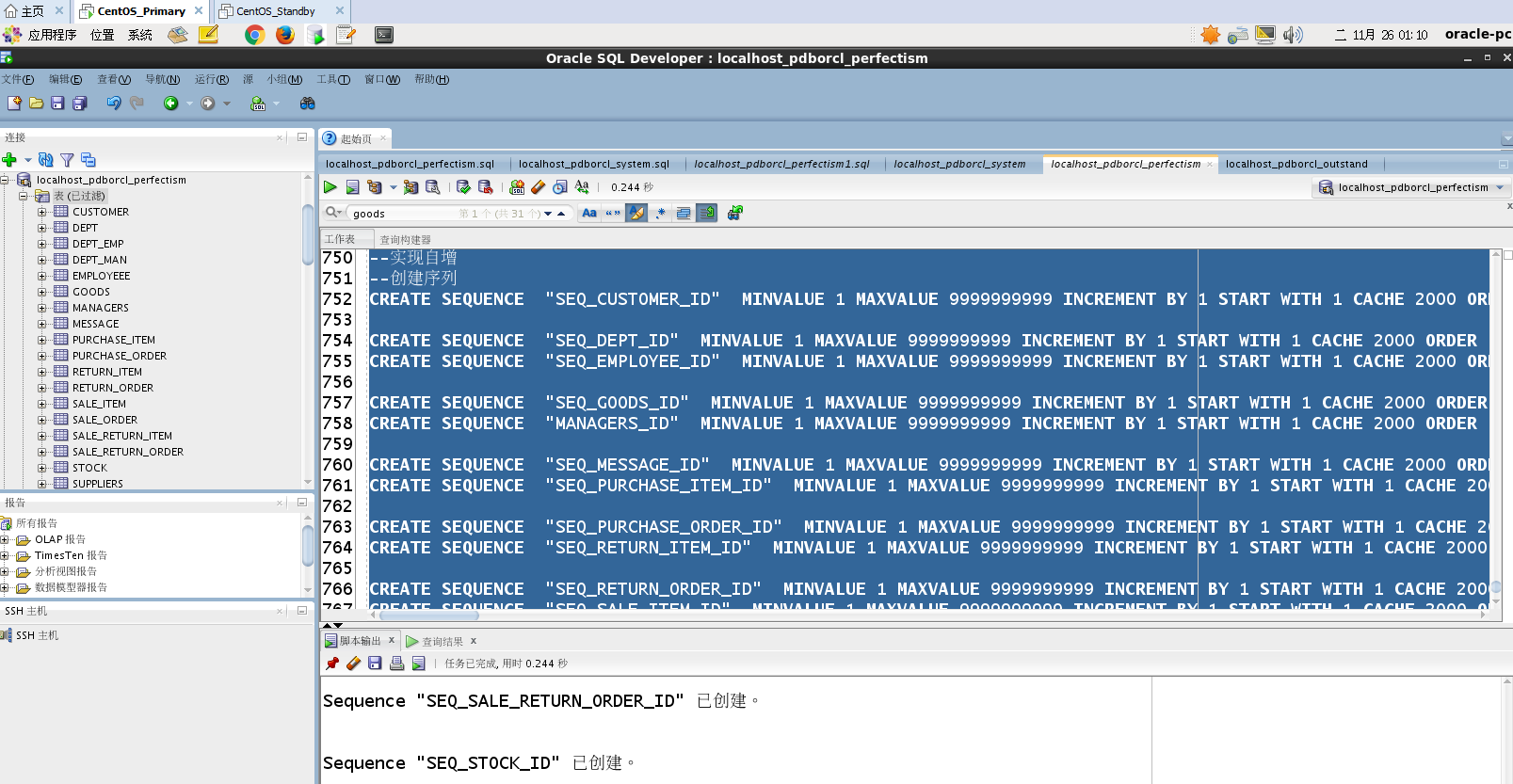
create sequence "seq\_sale\_return\_item\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

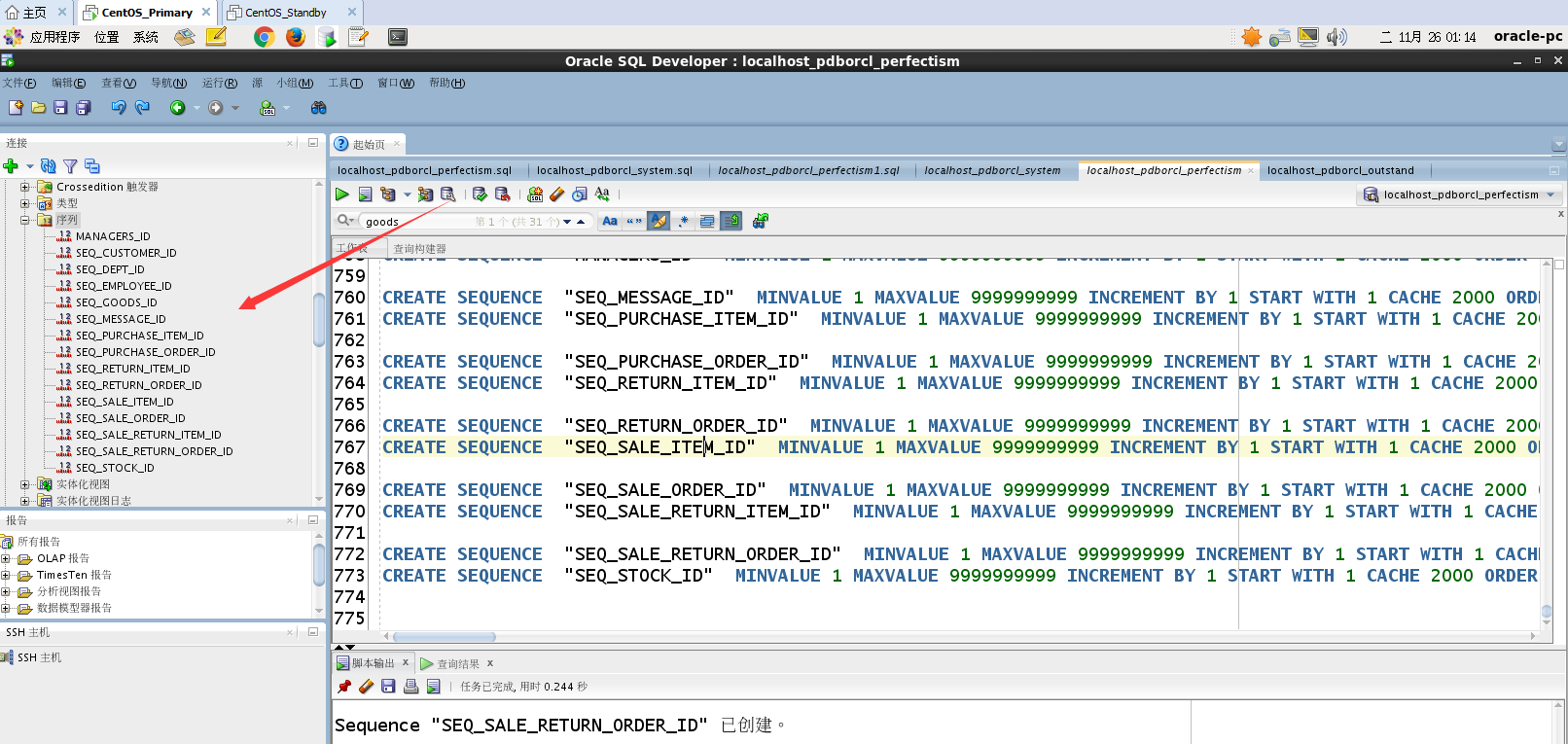
create sequence "seq\_sale\_return\_order\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

create sequence "seq\_stock\_id" minvalue 1 maxvalue 9999999999 increment by 1 start with 1 cache 2000 order nocycle nopartition ;

分析：

该语句创建了15个具有自增的序列：SEQ\_CUSTOMER\_ID,SEQ\_DEPT\_ID,SEQ\_EMPLOYEE\_ID, SEQ\_GOODS\_ID,MANAGERS\_ID,SEQ\_MESSAGE\_ID,SEQ\_PURCHASE\_ITEM\_ID ,SEQ\_PURCHASE\_ORDER\_ID,SEQ\_RETURN\_ITEM\_ID,SEQ\_RETURN\_ORDER\_ID,SEQ\_SALE\_ITEM\_ID ,SEQ\_SALE\_ORDER\_ID, SEQ\_SALE\_RETURN\_ITEM\_ID ,SEQ\_SALE\_RETURN\_ORDER\_ID ,SEQ\_STOCK\_ID，这些都是为15个表创建的，用以模拟自增，最小从1开始，最大为9999999999，步长为1，不循环，cache用于可见，防止拿到同一个序列数，一般cache元素的个数 <= 循环元素个数。





4.3 插入数据

--批量插入

declare

dt date;

v\_customer\_id number(20);

v\_id number(20);

v\_sale\_order\_pay varchar(255);

v\_sale\_order\_total number(20,2);

v\_flag varchar(255);

m number(20);

area varchar(255);

goods\_id number(20);

temp number(20);

p\_name varchar(255);

p\_price number(20,2);

supplier\_id number(20);

purchase\_order\_id number(20);

p\_count number(20);

v\_tel number(20);

email varchar(255);

tel\_phone varchar(255);

zip varchar(255);

p\_space varchar(255);

app\_id varchar(255);

p\_add varchar(255);

batch\_id varchar(255);

space varchar(255);

p\_bank varchar(255);

v\_name varchar(255);

begin

for i in 1..20000

loop

if i mod 2 =0 then

dt:=to\_date('2015-05-03','yyyy-mm-dd')+(i mod 60);

else

dt:=to\_date('2016-05-03','yyyy-mm-dd')+(i mod 60);

end if;

v\_customer\_id:=i;

--插入订单

v\_id:=seq\_sale\_order\_id.nextval; --应该将seq\_sale\_order\_id.nextval保存到变量中。

if i mod 2 =0 then

v\_sale\_order\_pay:='现金';

else

v\_sale\_order\_pay:='银行卡';

end if;

v\_name := 'wang' || i;

if i mod 2 =0 then

v\_flag:='已发货';

else

v\_flag:='未发货';

end if;

v\_sale\_order\_total:=dbms\_random.value(10000,5000);

insert/\*+append\*/into sale\_order(sale\_order\_id,customer\_id,flag,sale\_order\_pay,sale\_order\_sale\_date,sale\_order\_total) values ( v\_id,v\_customer\_id ,v\_flag,v\_sale\_order\_pay,dt,v\_sale\_order\_total);

m:=dbms\_random.value(100,10);

if i mod 2 =0 then

area:='a1';

else

area:='c2';

end if;

goods\_id :=i;

v\_id:=seq\_stock\_id.nextval;

insert /\*+append\*/ into stock(stock\_id,stock\_area,stock\_count,goods\_id) values (v\_id,area,m,goods\_id );

p\_count :=dbms\_random.value(100,10);

case

when i mod 5 =1 then

p\_name :='魅族';

p\_price:=3000.00;

supplier\_id:=1001;

when i mod 5=2 then

p\_name :='华为';

supplier\_id:=1002;

p\_price:=5400.00;

when i mod 5=3 then

p\_name :='三星';

p\_price:=3800.00;

supplier\_id:=1003;

when i mod 5=4 then

p\_name :='荣耀';

supplier\_id:=1004;

p\_price:=2100.00 ;

else

p\_name :='oppo';

supplier\_id:=1005;

p\_price:=4100.00;

end case;

purchase\_order\_id:=i;

v\_id:=seq\_purchase\_item\_id.nextval;

insert/\*+append\*/into purchase\_item(purchase\_item\_id,purchase\_count,p\_name,purchase\_price,purchase\_order\_id,supplier\_id) values (v\_id, p\_count,p\_name,p\_price,purchase\_order\_id,supplier\_id);

p\_count :=dbms\_random.value(9000,2000);

v\_id:=seq\_customer\_id.nextval;

v\_tel:='139888883'||i;

tel\_phone:='0790-7378785789'||i;

zip:='3274837'||i;

case

when i mod 5 =1 then

p\_add:='四川省';

p\_bank:='建设银行';

supplier\_id:=100100000;

email:='welcase@123.com';

when i mod 5=2 then

p\_add:='成都市';

supplier\_id:=100200000;

p\_bank:='中国银行';

email:='deffcase@123.com';

when i mod 5=3 then

p\_add :='浙江省';

p\_bank:='工商银行';

email:='johnsdifen@123.com';

supplier\_id:=100300000;

when i mod 5=4 then

p\_add :='湖南省';

supplier\_id:=100400000;

p\_bank:='邮政银行' ;

email:='perfectism@123.com';

else

p\_add :='山东省';

supplier\_id:=100500000;

p\_bank:='交通银行';

email:='sedeed@123.com';

end case;

insert/\*+append\*/into customer(customer\_id,address,bank,bank\_account,email,link\_man,link\_tel,customer\_name,tel\_phone,zip) values (v\_id,p\_add,p\_bank,supplier\_id,email, v\_name,v\_tel,'xxx电子商务有限公司',tel\_phone,zip);

v\_id:=seq\_goods\_id.nextval;

case

when i mod 5 =1 then

p\_name :='魅族';

p\_price:=3000.00;

supplier\_id:=1001;

when i mod 5=2 then

p\_name :='华为';

supplier\_id:=1002;

p\_price:=5400.00;

when i mod 5=3 then

p\_name :='三星';

p\_price:=3800.00;

supplier\_id:=1003;

when i mod 5=4 then

p\_name :='荣耀';

supplier\_id:=1004;

p\_price:=2100.00 ;

else

p\_name :='oppo';

supplier\_id:=1005;

p\_price:=4100.00;

end case;

app\_id:='13243434564'||i;

batch\_id:='65454213'||i;

p\_space:=dbms\_random.value(9000,2000);

insert into goods(goods\_id,approve\_id,batch\_id,goods\_name,price,space,supplier\_id,unit) values (v\_id,app\_id,'batch\_id',p\_name,p\_space,p\_add,supplier\_id,'台');

--在触发器关闭的情况下，需要手工计算每个订单的应收金额：

select sum(purchase\_count\*purchase\_price) into m from purchase\_item,sale\_order where sale\_order.sale\_order\_id=purchase\_item.purchase\_order\_id;

if m is null then

m:=0;

end if;

temp:=i;

update sale\_order set sale\_order\_total = m where sale\_order\_id=temp;

if i mod 1000 =0 then

commit; --每次提交会加快插入数据的速度

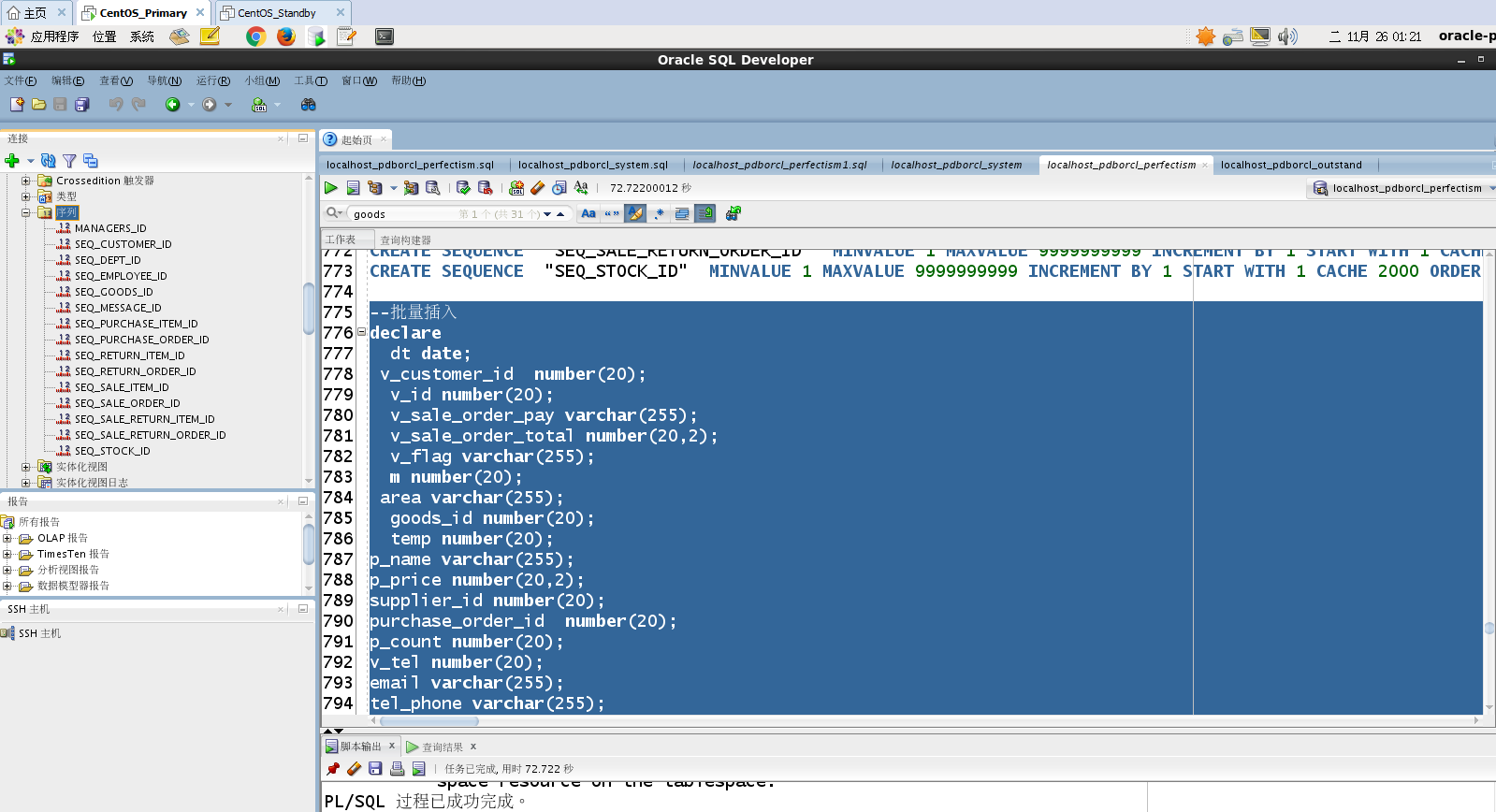
end if;

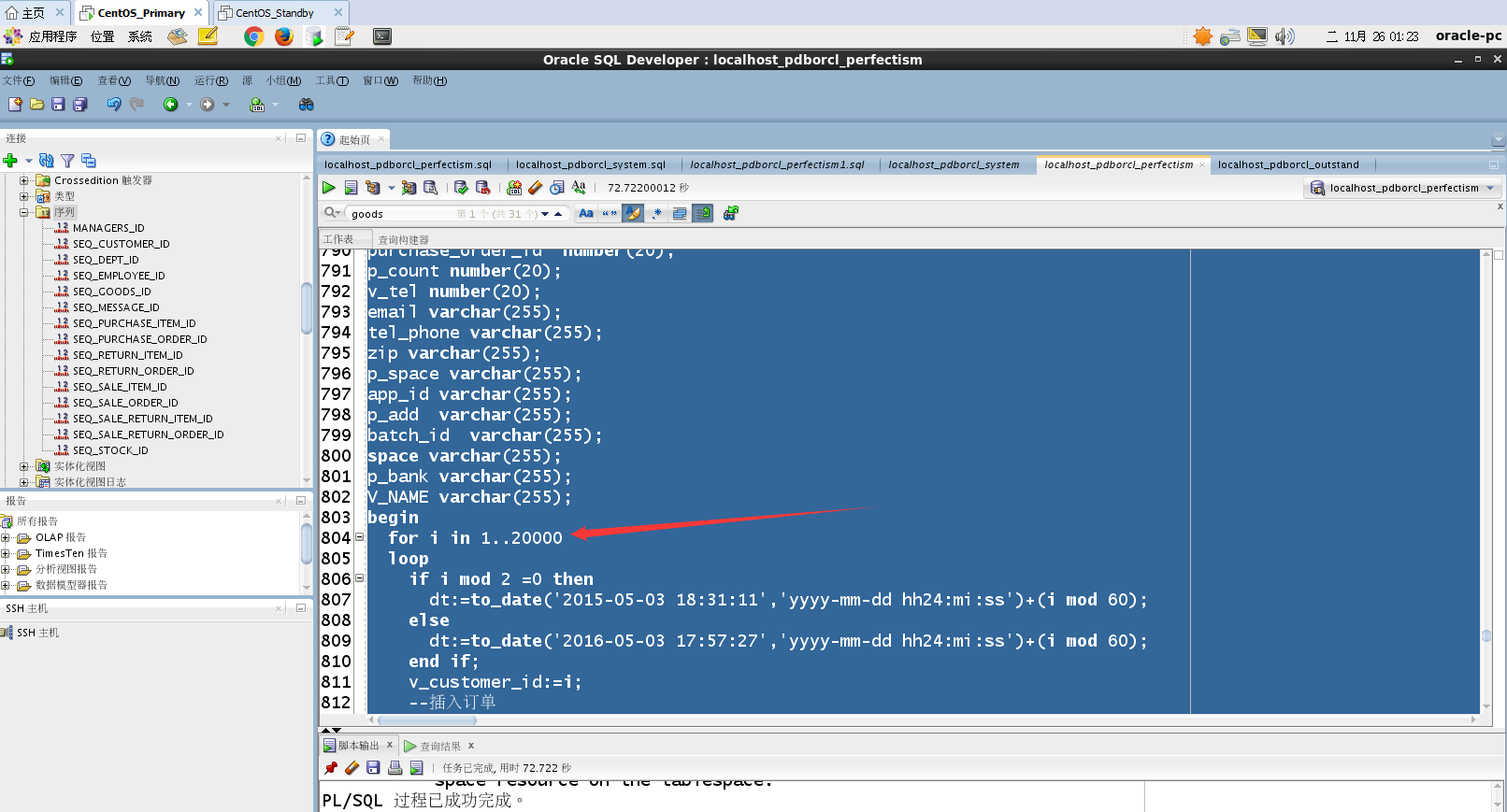
end loop;

end;

分析：

该sql语句在脚本里面先是声明了在插入表的用于存放数据的变量，有5张表的变量，其中包括时间类型，字符类型，数值类型，浮点类型，当然oracle与其它的数据库（MYSQL）不一样，它使用number(位数,保留位数)来设置的浮点数类型，在开始运行脚本之前，都会在变量声明之后跟上一个begin ,接着写循环的sql脚本，循环的次数是20000，就是说对sale\_order表,stock表, purchase\_item表,customer表,goods表这5张表分别插入了20000条数据。截图如下：





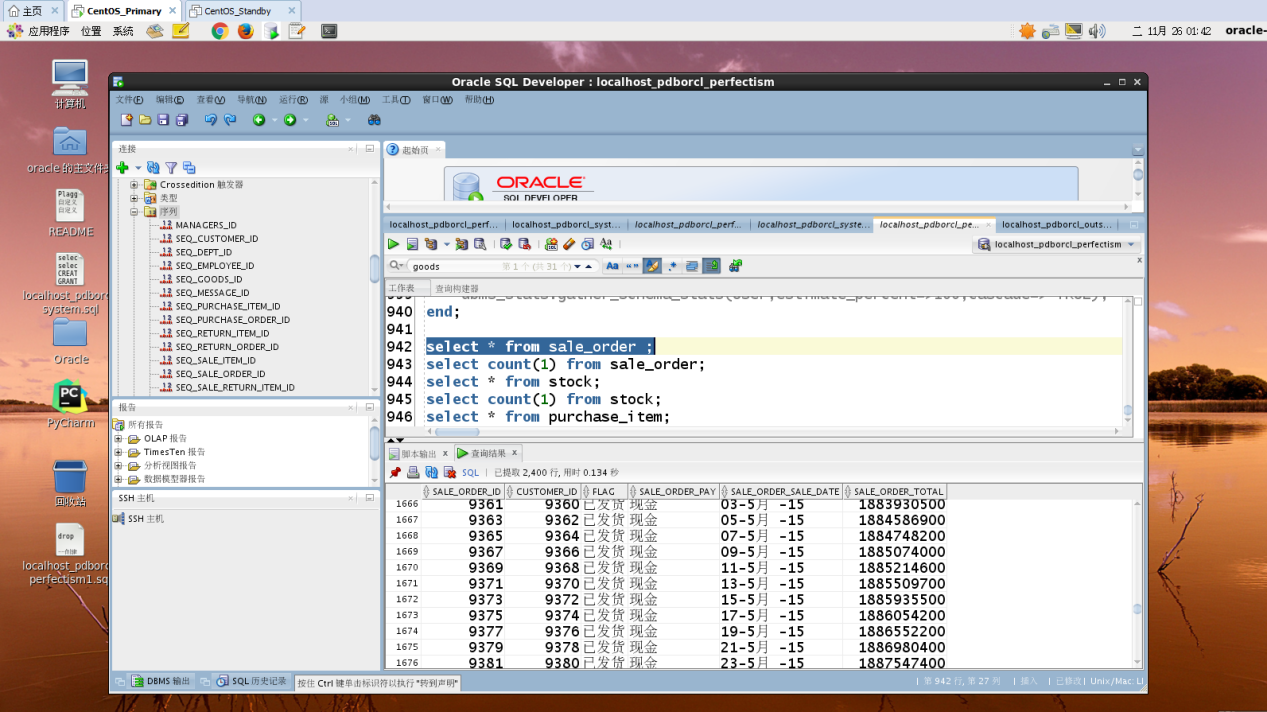
下面对其进行验证：分别查询每张表的数据。

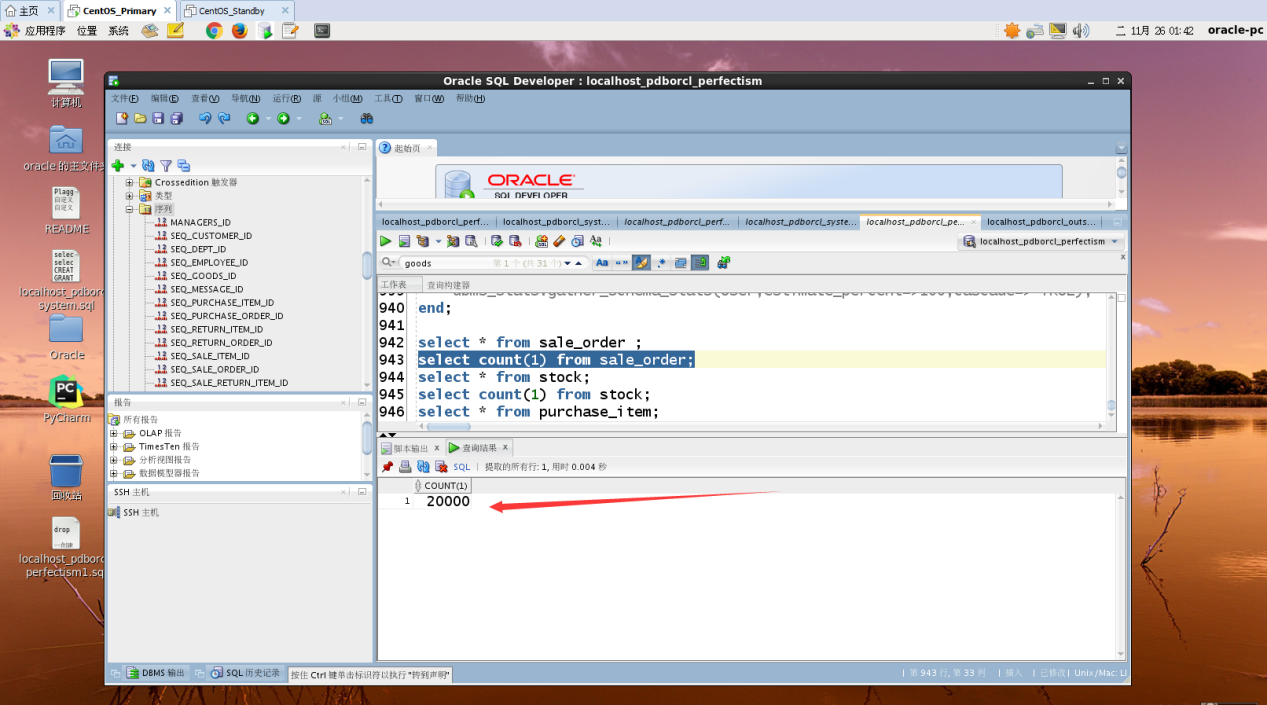
查询sale\_order表：

select \* from sale\_order ;

select count(1) from sale\_order;

截图如下：



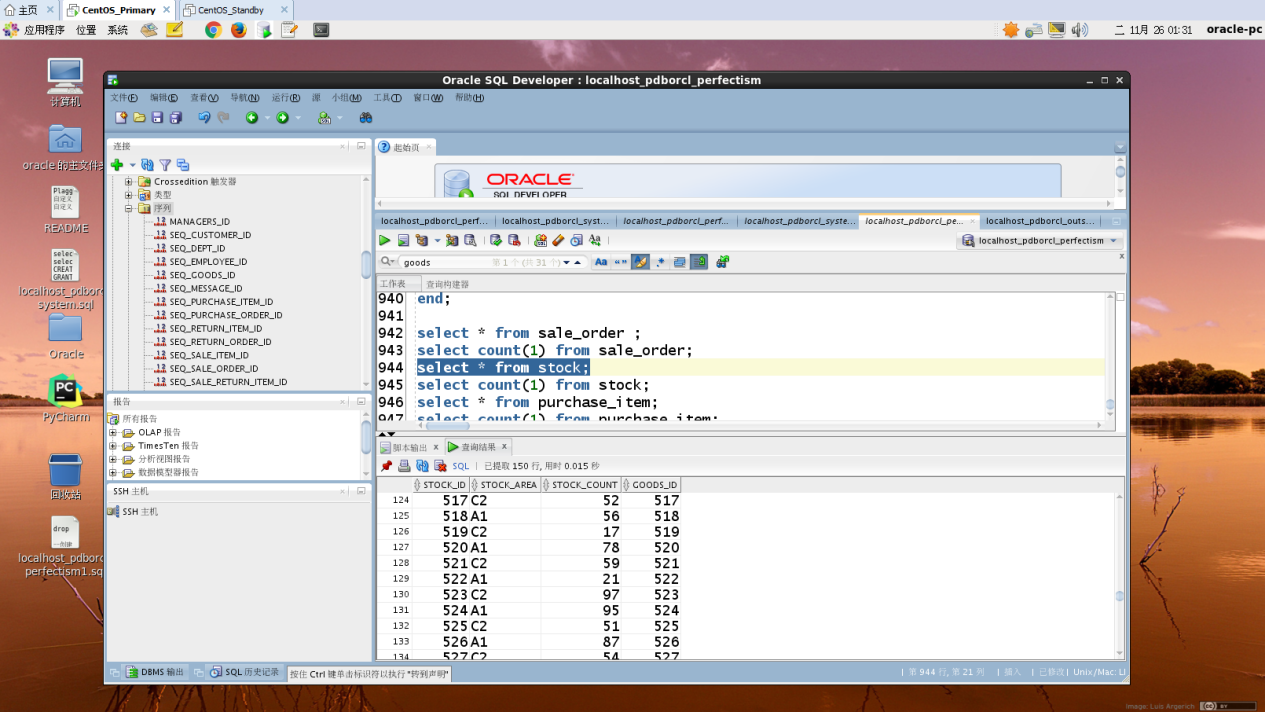


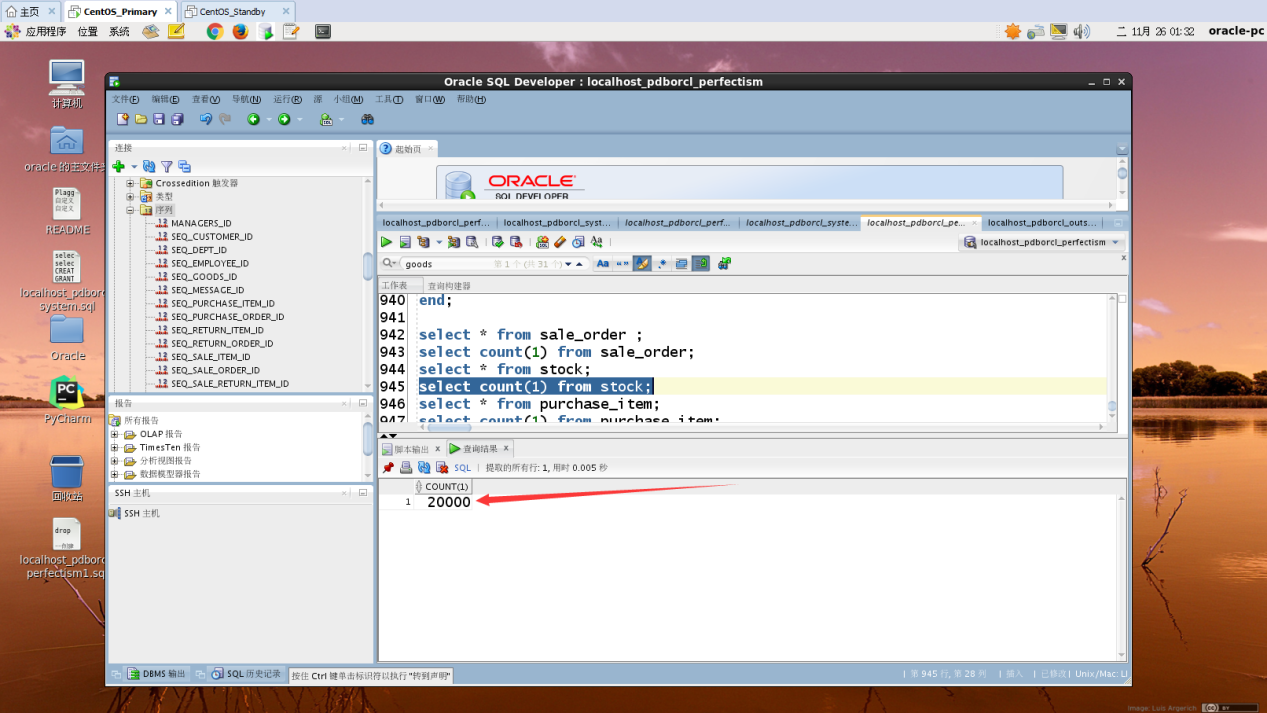
查询stock表：

select \* from stock;

select count(1) from stock;

截图如下：



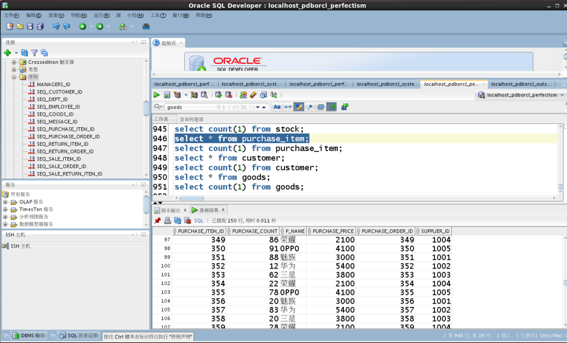


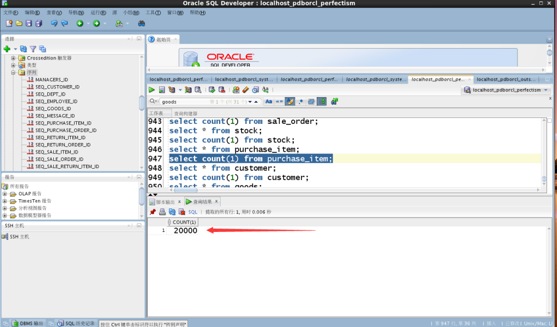
查询purchase\_item表：

select \* from purchase\_item;

select count(1) from purchase\_item;

截图如下：



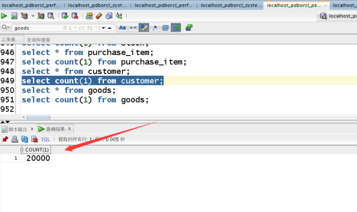


查询customer表：

select \* from customer;

select count(1) from customer;

截图如下：

CREATE OR REPLACE

PACKAGE BODY MYPACKAGE AS

function getTotalSal(pid in number)

return number

as

price purchase\_item.purchase\_price%type ;

pcount purchase\_item.purchase\_count%type ;

begin

select purchase\_price ,purchase\_count into price,pcount from purchase\_item where purchase\_item\_id = pid ;

dbms\_output.put\_line( price\*nvl( pcount,0));

return price\*nvl( pcount,0) ;

end ;

procedure queryEmpList(dno in number , costomerList out coscursor) AS

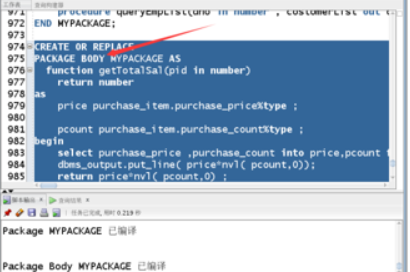
BEGIN

open costomerList for

select \* from customer where customer\_id = dno;

END queryEmpList;

END MYPACKAGE;



第四章 总结报告

现在市面上流行的也就那几大数据库（mysql,oracle,sql server），但是说不好,以后可能又会有新的好用的数据库，比如现在又有阿里巴巴的ocean数据库，远超oracle，就我而言，我可能接触的最多的是MySQL，因为但是刚刚接触编程语言设计数据库的时候就用的MySQL，当然，它适用于小中型企业，数据量不是特别大，操作起来也比较方便一点，最重要的一点是开源。

 ​人总是要跟上时代的步伐才行，多掌握一种数据库也是好的，而且oracle在很多大公司使用的相当频繁，一直处于数据库层面的中流砥柱，所以学好Oracle相当有必要，我是先跟着老师入门，对其有一定的了解，比如oracle的结构的结构，文件类型等等，有一个大概的了解，之后才学习它具体的对数据的操作方法，当然学习了这么久的Oracle，就让我来谈谈它与MySQL的一些区别吧。 安装就不说了，MySQL是要简单一点，配置也很简单，这个安装配置的复杂性主要体现在服务器上。当然我也有安装他们两个数据库的经历，就mysql而言是要简单一点，oralce就对服务器硬盘等等的要求也极高，硬盘至少要高于30个G（我的是80G的虚拟机）,这个经历是惨痛的，但是就是因为日志备份文件找不到适合空间存放而重装。我对Oracle对象管理及使用思想有了进一步的了解，通过动手实现Oracle对象管理及使用，更加深刻的理解了Oracle对象管理及使用的特点。在实验中实现了表结构的建立、修改、查看、删除操作。实现了索引的建立、修改、查看、删除操作。实现了视图的建立、查询、修改、删除操作。实现了 同义词的建立、查询、修改、删除操作，比较对同义词的操作与对原数据库对象的操作是否一致。了解并实现了用alter table old\_table\_name rename to new\_table\_name表名进行修改。通过create index index\_name on table\_name(column\_name1asc|desc,column\_name2 asc|desc....)创建基于多列的索引。掌握序列的建立、查询、修改、删除操作，利用序列向数据库表中插入数据。并且在实验过程中，回顾书本上的理论知识，巩固了我的知识。