****

**《大型数据库》**

**课内实验三**

指导老师： 盛 津 芳

专 业： 物联网工程

班 级： 物联网1802

学 号： 8208181125

姓 名： 王灏洋

2020 年 11 月

目录

[一、 实验内容 2](#_Toc57823079)

[1. 实验目的 2](#_Toc57823080)

[2. 实验内容 2](#_Toc57823081)

[二、 源程序清单 3](#_Toc57823082)

[1. 第一题 3](#_Toc57823083)

[2. 第二题 10](#_Toc57823084)

[3. 第三题，第四题 11](#_Toc57823085)

[4. 第五题 14](#_Toc57823086)

[三、 实验中遇到的问题及解决方案 15](#_Toc57823087)

# 实验内容

## 实验目的

本实验主要是熟悉ORACLE的基于大数据集（记录达到千万级）的相关数据库操作包括生成海量测试数据的更新操作、基于海量数据的索引维护、表空间、分区、视图、序列及相关的统计分析。

## 实验内容

1. 以常用“名字大全”与“百家姓”数据集为基础，生成不小于1千万条stud记录，要求，姓名的重复率不超过10%，学号以ABCD17EFGH为格式模板，即其中17是固定的，AB为从01到80，CD为从01到90，EF为01到50，GH为01到32；性别中，男、女占比为99%到99.5%。TEL与E-mail不作要求，但不能全空。Birthday要求从‘19940101’到‘19990731’分布。要求记录ORACLE数据文件的大小变化。（需要编制过程）
2. 分别测试stud有主键与没有主键情形下生成记录的时间。
3. 建立基于sname的索引index\_name，测试建立的时间与建立索引前后查询某一姓名及某一姓的时间长度。
4. 测试索引index\_name建立前后，分姓（简单地理解为姓名的第1，2位）的记录数统计时间。
5. 按学号首位建立10个分区分别为part\_0到part\_9，测试建立分区前后分首位统计人数与分专业（EF位）统计人数的时间差别。

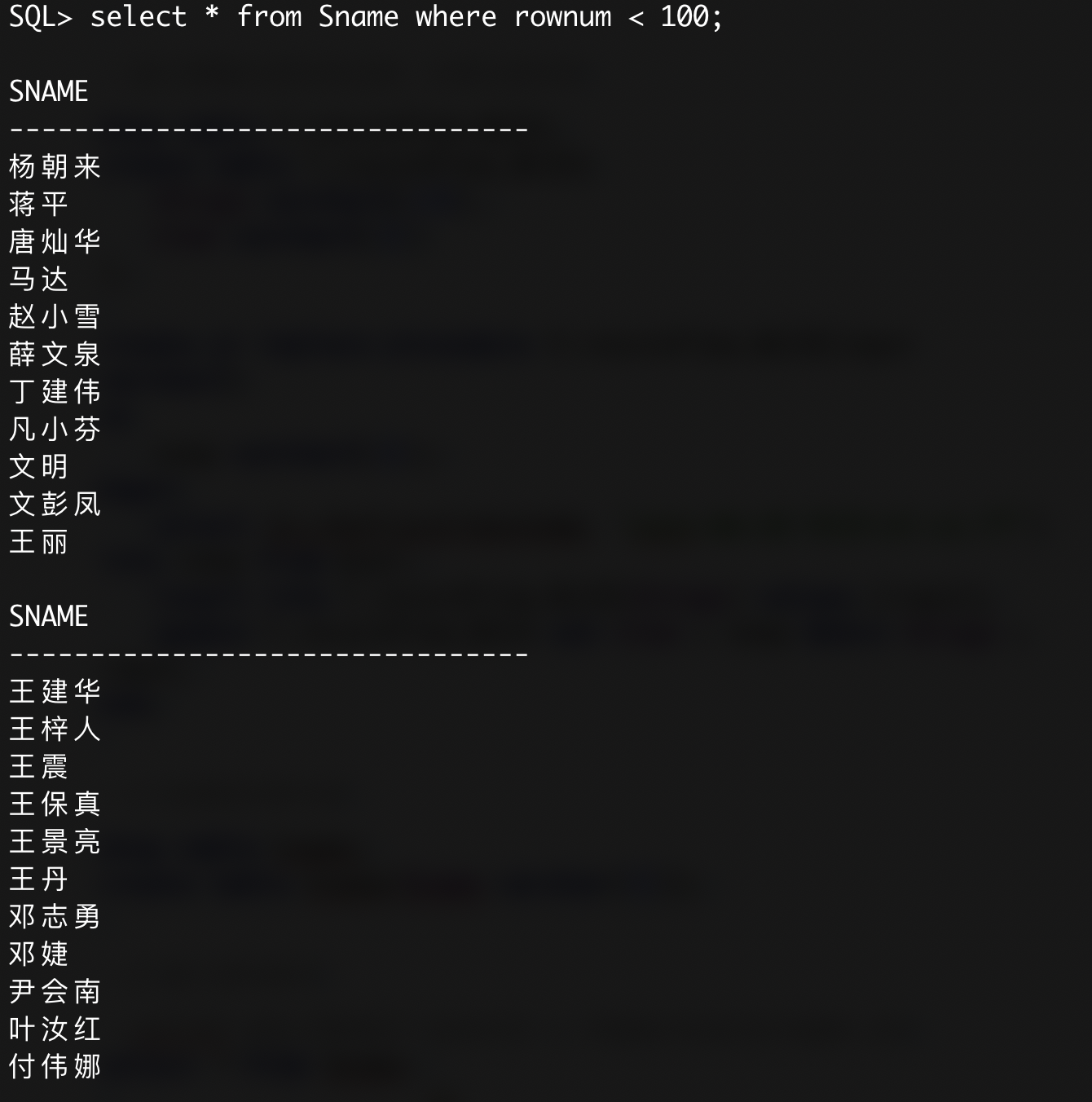
# 源程序清单

## 第一题

**题目：**以常用“名字大全”与“百家姓”数据集为基础，生成不小于1千万条stud记录，要求，姓名的重复率不超过10%，学号以ABCD17EFGH为格式模板，即其中17是固定的，AB为从01到80，CD为从01到90，EF为01到50，GH为01到32；性别中，男、女占比为99%到99.5%。TEL与E-mail不作要求，但不能全空。Birthday要求从‘19940101’到‘19990731’分布。要求记录ORACLE数据文件的大小变化。（需要编制过程）

*--1.以常用“名字大全”与“百家姓”数据集为基础，  
-- 生成不小于1千万条stud记录，  
-- 要求，姓名的重复率不超过10%，学号以ABCD17EFGH为格式模板，即其中17是固定的，  
-- AB为从01到80，CD为从01到90，EF为01到50，GH为01到32；  
-- 性别中，男、女占比为99%到99.5%。TEL与E-mail不作要求，但不能全空。  
-- Birthday要求从‘19940101’到‘19990731’分布。  
-- 要求记录ORACLE数据文件的大小变化。（需要编制过程）  
  
--建立精确记录时间过程，方便记录时间***drop table** T\_recordTime\_W125;  
**create table** T\_recordTime\_W125(  
 **things varchar2**(128),  
 **time varchar2**(32)  
);  
  
**create or replace procedure** *P\_recordTime\_W125*(input **varchar2**)  
**as** temp **varchar2**(32);  
**begin  
 select** *to\_char*(*systimestamp*, **'yyyy-mm-dd hh24:mi:ss.ff'**) **into** temp **from** dual;  
 **insert into** T\_recordTime\_W125(**things**) **values** (input);  
 **update** T\_recordTime\_W125 **set time** = temp **where things** = input;  
**end**;

*--1 创建基本姓名表***drop table** Sname;  
**create table** Sname(**Sname varchar**(32));  
  
*--2 插入基本姓名  
--sqlldr why/281612 control = /home/oracle/name.csv***select** *\** **from** Sname;



*--3 制造足够多的姓名，为名字的每个字创造视图***drop view** V\_name1\_W125;  
**drop view** V\_name2\_W125;  
**drop view** V\_name3\_W125;  
**drop view** V\_name12\_W125;  
**drop view** V\_name123\_W125;  
**create view** V\_name1\_W125 **as select** *\** **from** (**select distinct** *substr*(**sname**,1,1) na1 **from** Sname **order by** dbms\_random.*VALUE*()) **where** *rownum*<=500;  
**create view** V\_name2\_W125 **as select** *\** **from** (**select distinct** *substr*(**sname**,2,1) na2 **from** Sname **order by** dbms\_random.*VALUE*()) **where** *rownum*<=1200;  
**create view** V\_name3\_W125 **as select** *\** **from** (**select distinct** *substr*(**sname**,3,1) na3 **from** Sname **order by** dbms\_random.*VALUE*()) **where** *rownum*<=20;  
**create view** V\_name12\_W125 **as select** *concat*(**na1**,**na2**) na12 **from** V\_name1\_W125,V\_name2\_W125;  
**create view** V\_name123\_W125 **as select** *concat*(**na12**,**na3**) na123 **from** V\_name12\_W125,V\_name3\_W125;

*--4 创建T\_studName\_W125表***drop table** T\_studName\_W125;  
**create table** T\_studName\_W125(**Sname varchar**(16));  
*--5 为姓名表T\_studName\_W125插入12600000条学生姓名***insert into** T\_studName\_W125(**Sname**) **select na12 from** V\_name12\_W125;  
**insert into** T\_studName\_W125(**Sname**) **select na123 from** V\_name123\_W125;   
**select** *count*(*\**) **from** T\_studName\_W125;



*--6.生成规范的学号表  
--每次生成两位，分四批生成***drop table** T\_studSno\_gh\_W125;  
**drop table** T\_studSno\_ef\_W125;  
**drop table** T\_studSno\_cd\_W125;  
**drop table** T\_studSno\_ab\_W125;  
**create table** T\_studSno\_gh\_W125 (**Sno number**(10));  
**create table** T\_studSno\_ef\_W125 (**Sno number**(10));  
**create table** T\_studSno\_cd\_W125 (**Sno number**(10));  
**create table** T\_studSno\_ab\_W125 (**Sno varchar2**(10));

*--建立过程：***declare**i **int**;  
**begin  
for** i **in** 1..32 **loop  
 insert into** T\_studSno\_gh\_W125 **values**(i);  
**end loop**;  
**for** i **in** 1..50 **loop  
 insert into** T\_studSno\_ef\_W125 **select** 170000 + i \* 100 + T\_studSno\_gh\_W125.**sno from** T\_studSno\_gh\_W125;  
 **commit**;  
**end loop**;  
**for** i **in** 1..90 **loop  
 insert into** T\_studSno\_cd\_W125 **select** i \* 1000000 + T\_studSno\_ef\_W125.**sno from** T\_studSno\_ef\_W125;  
 **commit**;  
**end loop**;  
**for** i **in** 1..80 **loop  
 insert into** T\_studSno\_ab\_W125 **select** *substr*(10000000000 + i \* 100000000 + T\_studSno\_cd\_W125.**sno**, 2, 10) **from** T\_studSno\_cd\_W125;  
 **commit**;  
**end loop**;  
**end**;

*--7.1批量处理sex, tel, email, birthday等数据  
-- 产生随机性别：男、女占比为99%到99.5%  
-- 在性别表T\_sex\_W125中，143人，男女分别71人，一人性别为其它。随机查询一条记录时，可使男、女占比为99.3%  
-- 函数F\_getSex\_W125返回值为随机性别***create or replace function** *F\_getSex\_W125* **return varchar2  
is**f\_numb **number**;  
f\_sex **varchar2**(8);  
**begin** *--dual是虚拟表，主要用来借用以调用系统函数* **select** *trunc*(DBMS\_RANDOM.*VALUE*(1, 143)) **into** f\_numb **from** dual;  
 **if** f\_numb <= 71 **then** f\_sex := **'男'**;  
 **elsif** f\_numb <= 142 **then** f\_sex := **'女'**;  
 **else** f\_sex := **'其他'**;  
 **end if**;  
 **return** f\_sex;  
**end**;

*--7.2产生随机电话  
-- 函数F\_getTel\_W125会返回一个随机手机号***drop table** T\_tel\_W125;  
**create table** T\_tel\_W125(**tel varchar2**(4));  
**insert into** T\_tel\_W125 **values**(**'132'**);  
**insert into** T\_tel\_W125 **values**(**'135'**);  
**insert into** T\_tel\_W125 **values**(**'156'**);  
**insert into** T\_tel\_W125 **values**(**'151'**);  
**insert into** T\_tel\_W125 **values**(**'138'**);  
**insert into** T\_tel\_W125 **values**(**'139'**);  
**insert into** T\_tel\_W125 **values**(**'183'**);  
**insert into** T\_tel\_W125 **values**(**'187'**);  
**insert into** T\_tel\_W125 **values**(**'153'**);  
**insert into** T\_tel\_W125 **values**(**'150'**);  
**insert into** T\_tel\_W125 **values**(**'186'**);  
**insert into** T\_tel\_W125 **values**(**'188'**);  
**create or replace function** *F\_getTel\_W125* **return varchar2  
is** getTel **varchar2**(12);  
 numb\_head **varchar2**(4);  
 numb\_other **varchar2**(16);  
**begin  
 select tel into** numb\_head **from** (**select tel from** T\_tel\_W125 **order by** DBMS\_RANDOM.*VALUE*()) **where** *rownum* = 1 ;  
 **select** *substr*(*cast*(DBMS\_RANDOM.*VALUE*() **as varchar2**(32)), 3, 8) **into** numb\_other **from** dual;  
 getTel := numb\_head || numb\_other;  
 **return** getTel;  
**end**;

*--7.3随机产生邮箱  
-- 函数F\_getEmail\_W125会返回一个随机邮箱号***drop table** T\_email\_W125;  
**create table** T\_email\_W125 (**email varchar2**(16));  
**insert into** T\_email\_W125 **values**(**'126'**);  
**insert into** T\_email\_W125 **values**(**'139'**);  
**insert into** T\_email\_W125 **values**(**'sohu'**);  
**insert into** T\_email\_W125 **values**(**'sina'**);  
**insert into** T\_email\_W125 **values**(**'163'**);  
**insert into** T\_email\_W125 **values**(**'foxmail'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
**insert into** T\_email\_W125 **values**(**'qq'**);  
  
**create or replace function** *F\_getEmail\_W125* **return varchar2  
is** getEmail **varchar2**(32);  
 emName **varchar2**(16);  
 emOwn **varchar2**(16);  
**begin  
 select** *substr*(*cast*(DBMS\_RANDOM.*VALUE*() **as varchar2**(32)), 3, 11) **into** emName **from** dual;  
 **select email into** emOwn **from** (**select email from** T\_email\_W125 **order by** DBMS\_RANDOM.*VALUE*()) **where** *rownum* = 1;  
 getEmail := emName || **'@'** || emOwn || **'.com'**;  
 **return** getEmail;  
**end**;

*--7.4产生随机生日  
-- 函数f\_getBirthday\_j432会返回一个在[19940101,19990701]时间内的日期***create or replace function** *F\_getBirthday\_W125* **return date  
is**birthday **date**;  
**begin  
 select** *to\_date*(*trunc*(DBMS\_RANDOM.*VALUE*(2449354, 2451186)), **'J'**) **into** birthday **from** dual;  
 **return** birthday;  
**end**;

*--7.5生成完整学生表除学号和姓名之外的所有信息  
--建立基本信息模板表***create table** T\_sequenceId(  
 **id number**(10),  
 **sex varchar2**(32),  
 **tel varchar2**(32),  
 **email varchar2**(32),  
 **birthday date**);

*--存储大量的学生其他信息，千万级以上***create table** T\_studOtherInformation\_w125 (  
 **id number**(16),  
 **sex varchar2**(32),  
 **tel varchar2**(32),  
 **email varchar2**(32),  
 **birthday date**);

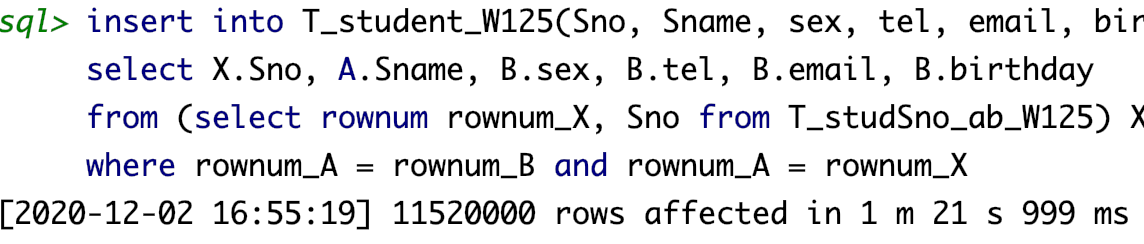
*--生成1万条数据模版***begin  
 delete from** T\_sequenceId;  
 **for** i **in** 0..9999 **loop  
 insert into** T\_sequenceId(**id**, **sex**, **tel**, **email**, **birthday**) **values** (i, *F\_getSex\_W125*(), *F\_getTel\_W125*(), *F\_getEmail\_W125*(), *F\_getBirthday\_W125*());  
 **end loop**;  
**end**;  
**select** *\** **from** T\_sequenceId **where** *ROWNUM* < 100;

*--每1w条数据插入一次，得到12000000数据***begin  
 delete from** T\_studOtherInformation\_w125;  
 **for** i **in** 1..1200 **loop  
 insert into** T\_studOtherInformation\_w125(**id**, **sex**, **tel**, **email**, **birthday**) **select** i \* 10000 + T\_sequenceId.**id as** MSISDN, T\_sequenceId.**sex**, T\_sequenceId.**tel**, T\_sequenceId.**email**, T\_sequenceId.**birthday from** T\_sequenceId;  
 **commit**;  
 **end loop**;  
**end**;

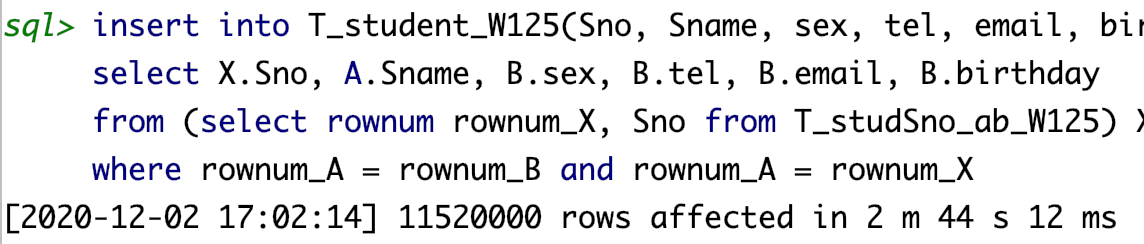
## 第二题

**题目：**分别测试stud有主键与没有主键情形下生成记录的时间。

*--'学生信息表没有主键，生成11520000条记录'***create table** T\_student\_W125 (  
 **Sno varchar2**(10), *--学生学号* **Sname varchar2**(32),  
 **sex varchar2**(32),  
 **tel varchar2**(32),  
 **email varchar2**(32),  
 **birthday date**);  
  
*--生成完整的学生信息表：将学号，姓名和其他信息，放在一起***insert into** T\_student\_W125(**Sno**, **Sname**, **sex**, **tel**, **email**, **birthday**)  
**select** X.**Sno**, **A**.**Sname**, B.**sex**, B.**tel**, B.**email**, B.**birthday  
from** (**select** *rownum* rownum\_X, **Sno from** T\_studSno\_ab\_W125) X, (**select** *rownum* rownum\_A, **Sname from** T\_studName\_W125) **A**, (**select** *rownum* rownum\_B, **sex**, **tel**, **email**, **birthday from** T\_studOtherInformation\_w125) B  
**where** rownum\_A = rownum\_B **and** rownum\_A = rownum\_X;  
*--学生信息表没有主键，生成11520000条记录*



*--学生信息表有主键，生成11520000条记录，开始时间***drop table** T\_student\_W125;  
**create table** T\_student\_W125 (  
 **Sno varchar2**(10) **primary key** , *--学生学号* **Sname varchar2**(32),  
 **sex varchar2**(32),  
 **tel varchar2**(32),  
 **email varchar2**(32),  
 **birthday date**);  
**insert into** T\_student\_W125(**Sno**, **Sname**, **sex**, **tel**, **email**, **birthday**)  
**select** X.**Sno**, **A**.**Sname**, B.**sex**, B.**tel**, B.**email**, B.**birthday  
from** (**select** *rownum* rownum\_X, **Sno from** T\_studSno\_ab\_W125) X, (**select** *rownum* rownum\_A, **Sname from** T\_studName\_W125) **A**, (**select** *rownum* rownum\_B, **sex**, **tel**, **email**, **birthday from** T\_studOtherInformation\_w125) B  
**where** rownum\_A = rownum\_B **and** rownum\_A = rownum\_X;  
*--学生信息表有主键，生成11520000条记录，完成时间*

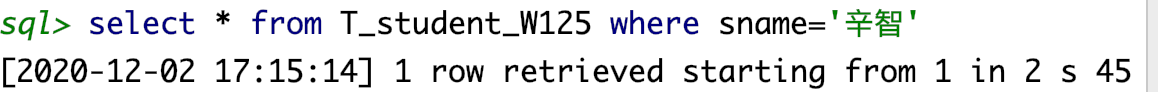


**alter session set** nls\_date\_format = **'yyyy-mm-dd'**;  
**select** *\** **from** (**select** *\** **from** T\_student\_W125 **order by** dbms\_random.*value*()) **where** *rownum* <= 100;  
**select** *count*(*\**) **from** T\_student\_W125;

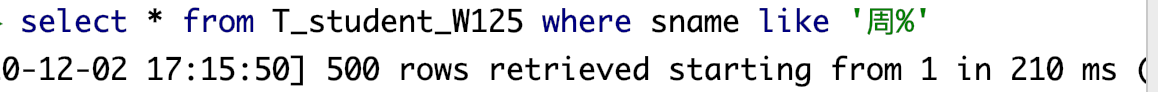
## 第三题，第四题

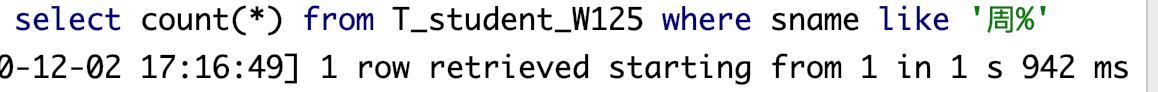
**题目：**建立基于sname的索引index\_name，测试建立的时间与建立索引前后查询某一姓名及某一姓的时间长度。

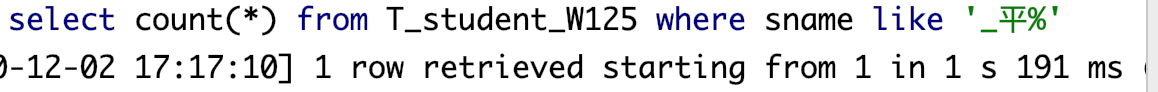
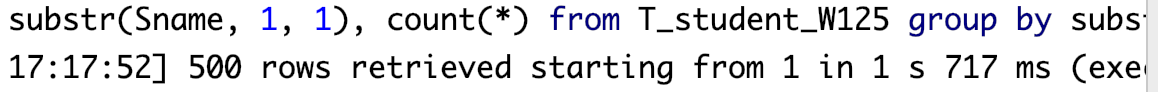
**题目：**测试索引index\_name建立前后，分姓（简单地理解为姓名的第1，2位）的记录数统计时间。

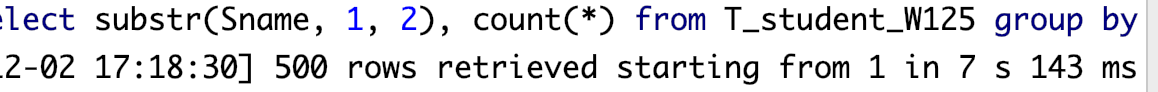
*--没有索引  
--没有姓名索引，查询一条姓名***select** *\** **from** T\_student\_W125 **where sname**=**'辛智'**;  
**

*--没有姓名索引，查询某一姓氏人数***select** *\** **from** T\_student\_W125 **where sname like '周%'**;

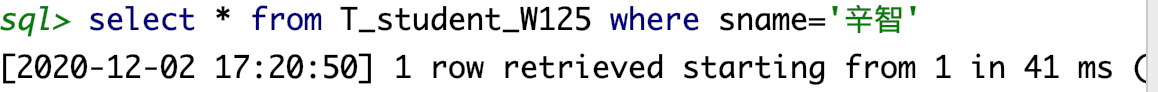
****  
*--没有姓名索引，统计某一姓氏人数***select** *count*(*\**) **from** T\_student\_W125 **where sname like '周%'**;

  
*--没有姓名索引，统计某一姓名第二个字相同人数***select** *count*(*\**) **from** T\_student\_W125 **where sname like '\_平%'**;

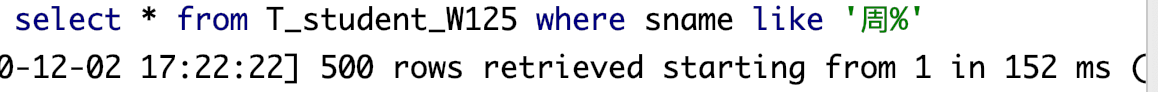
  
*--没有索引，分姓1位的记录数统计时间。***select** *substr*(**Sname**, 1, 1), *count*(*\**) **from** T\_student\_W125 **group by** *substr*(**Sname**, 1, 1);  
*--没有索引，分姓2位的记录数统计时间。***select** *substr*(**Sname**, 1, 2), *count*(*\**) **from** T\_student\_W125 **group by** *substr*(**Sname**, 1, 2);



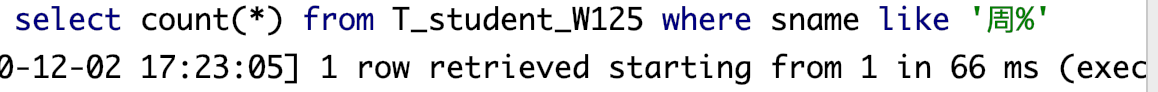
*--有索引***drop index** indexName;  
*--开始创建姓名索引***create index** indexName **on** T\_student\_W125(**Sname**);  
*--有姓名索引，查询一条姓名，开始时间***select** *\** **from** T\_student\_W125 **where sname**=**'辛智'**;



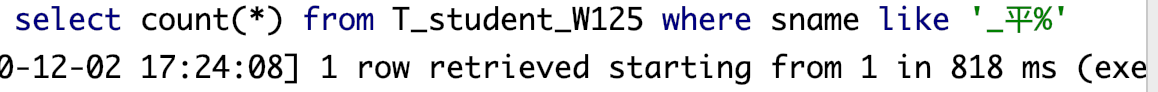
*--有姓名索引，查询某一姓氏人数的开始时间***select** *\** **from** T\_student\_W125 **where sname like '周%'**;



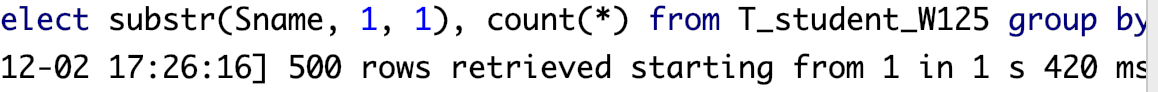
*--有姓名索引，统计某一姓氏人数***select** *count*(*\**) **from** T\_student\_W125 **where sname like '周%'**;



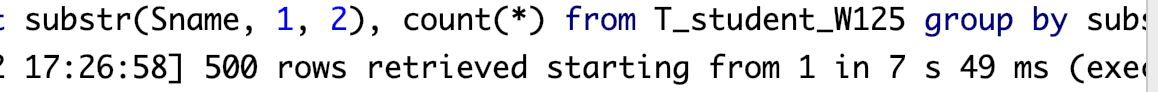
*--有姓名索引，统计某一姓名第二个字相同人数***select** *count*(*\**) **from** T\_student\_W125 **where sname like '\_平%'**;



*--有索引，分姓1位的记录数统计时间。***select** *substr*(**Sname**, 1, 1), *count*(*\**) **from** T\_student\_W125 **group by** *substr*(**Sname**, 1, 1);

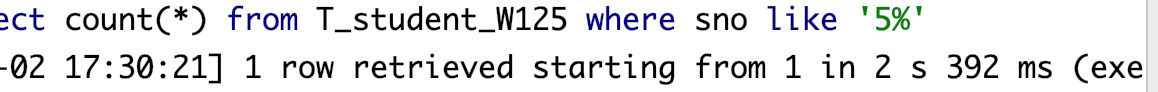
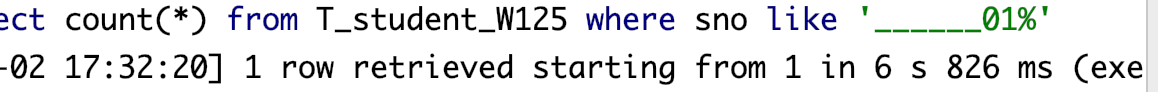


*--有索引，分姓2位的记录数统计时间。***select** *substr*(**Sname**, 1, 2), *count*(*\**) **from** T\_student\_W125 **group by** *substr*(**Sname**, 1, 2);



## 第五题

**题目：**按学号首位建立10个分区分别为part\_0到part\_9，测试建立分区前后分首位统计人数与分专业（EF位）统计人数的时间差别。

*--没有分区，按学号首位ID统计人数***select** *count*(*\**) **from** T\_student\_W125 **where sno like '5%'**;  
  
*--没有分区，按专业统计人数***select** *count*(*\**) **from** T\_student\_W125 **where sno like '\_\_\_\_\_\_01%'**;  
  
*--重新创建学生表，并分区***drop table** T\_student\_W125;  
**create table** T\_student\_W125(  
 **Sno varchar2**(10) **primary key**,*--学生学号* **Sname varchar2**(32),  
 **sex varchar2**(32),  
 **tel varchar2**(32),  
 **email varchar2**(32),  
 **birthday date**)**partition by range**(**sno**)(  
**partition** part\_0 **values less than** (**'1000170000'**),  
**partition** part\_1 **values less than** (**'2000170000'**),  
**partition** part\_2 **values less than** (**'3000170000'**),  
**partition** part\_3 **values less than** (**'4000170000'**),  
**partition** part\_4 **values less than** (**'5000170000'**),  
**partition** part\_5 **values less than** (**'6000170000'**),  
**partition** part\_6 **values less than** (**'7000170000'**),  
**partition** part\_7 **values less than** (**'8000170000'**),  
**partition** part\_8 **values less than** (**'9000170000'**),  
**partition** part\_9 **values less than** (**maxvalue**));  
**insert into** T\_student\_W125(**sno**,**sname**, **sex**, **tel**, **email**, **birthday**) **select** X.sno,**A**.sname, B.sex, B.tel, B.email, B.birthday **from** (**select** *rownum* rownum\_X,sno **from** t\_student\_sno\_ab\_j432) X, (**select** *rownum* rownum\_A,sname **from** t\_stu\_name\_j432) **A**, (**select** *rownum* rownum\_B,sex,tel,email,birthday **from** t\_stud\_other\_information\_j432) B **where** rownum\_A = rownum\_B **and** rownum\_A = rownum\_X;  
  
*--为学生表添加其他约束***alter table** T\_student\_W125 **add constraint** ck\_student\_sex **check**(**sex in**(**'男'**,**'女'**,**'其它'**));  
**alter table** T\_student\_W125 **add constraint** ck\_student\_email **check**(**email like '%@%.%'**);  
**alter table** T\_student\_W125 **add constraint** ck\_student\_birthday **check**(**birthday**>=*to\_date*(**'19940101'**,**'yyyymmdd'**) **and birthday**<=*to\_date*(**'19990731'**,**'yyyymmdd'**));  
  
*--有分区，按学号首位ID统计人数***select** *count*(*\**) **from** T\_student\_W125 **where sno like '5%'**;  
  
*--有分区，按专业统计人数***select** *count*(*\**) **from** T\_student\_W125 **where sno like '\_\_\_\_\_\_01%'**;

**因为使用的是标准版，所以无法分区，但是代码可以使用**

# 实验中遇到的问题及解决方案

* 使用的docker oracle不支持中文

解决方法：设置linux支持中文

* 导入数据出现问题

解决方法：使用ctl文件进行数据导入