

《数据库系统》实验大纲

要求独立完成，严禁复制他人（含往届同学）SQL，平台能够发现并记录作弊行为。

每个小题完成后，保存好源代码，实验报告会用到。

低级错误害死程序员，例如拼写错误，就是因为往往忽略低级错误。

平台非常稳定，允许做任何操作，允许犯任何错误，都不用担心影响实验成绩。

总体介绍(请全面阅读本介绍，可以避免走弯路)

一、实验咨询 QQ 群，进群后修改昵称为学号-姓名, 节省我问学号时间

1、有问题咨询可以直接私信老师留言，等待老师回复你，不用申请加为好友。

2、**收到老师回复后，不用再发“感谢”**。原因是给你回复后，老师可能忙别的了，当听到 QQ 新消息，就会再次打开 QQ，你的“感谢”会浪费老师时间，**非常感谢同学们理解和支持**。

3、如果是研究生选修或者国际版插班的同学，学号不是 12 位数字的，正确账号是学号中数字后面补零后凑够 12 位。

4、选修、重修的同学：

如果发现你的账号无法登入，请私信群里李老师，给老师留言提供你的学号、姓名，以及你所跟班级不是原在班级，老师创建好账号后，也会留言通知你。

5、先将**全屏截图**发给老师，因为错误千差万别，老师看到系统提示的错误，才能尽快找到问题所在，不要仅仅截图一个 SQL 就让老师找错，找错误真很难的！！

6、QQ 截图：按[Ctrl]+[Alt]+[A]，然后鼠标选择截图区域完成截图拷贝，QQ 对话框粘贴截图给老师。

笑话一则：

某天考生物，其中有一题是看鸟的腿猜出鸟的名字。某生实在不懂，生气的把卷子一撕准备离开考场。监考老师很生气于是问他：“你是哪班的，叫什么名字？”某生把裤腿一掀，说：“你猜啊你猜啊。”

同学们，你千万不要就给老师一个只有一两行的截图，或者都不给老师截图，就问老师为什么你的实验不能通过，判断错误即使信息很全了，都很难马上找到错误点，所以，截全屏发给老师，否则就是笑话。

二、经验之谈

实验期间，如果出现错误找不到，不要轻易怀疑实验平台和答案，更不要怀疑 Oracle 数据库系统，工作以后也要记住一句经验之谈，“**一定是你错了，只是不知道自己错在哪里了**”。

三、排错方法和步骤

- 1、如果出错行数比较多，100 行结果，90%有错误，那可能是你的计算公示有问题，造成一系列数据大多数错误了。如果 100%，基本可以确定绝对是这个原因。随便哪一行数据，看看为什么错了。
- 2、如果大部分都正确，但是多几行或者少几行或者错几行，这可能的情况就是忽略的什么特殊情况没有考虑，例如同学刷成绩、两次以上不及格等等。
- 3、最后的解决办法，也是没有办法的办法，但是，实际工作往往都是靠这个办法解决问题，特别是对于统计数据更有效，那就是一个一个统计。

统计学生学分结果如下：

#	SID	NAME	CREDIT
1	200900130417	顾丽梅	4
2	200900130737	崔亚丽	4
3	200900130067	侯佳	6
4	200900130617	梁代红	6
5	200900130559	李国民	6
6	200900130539	李飞飞	6
7	200900130428	成宝华	6
8	200900130416	刘国政	6
9	200900130402	李志梅	6

可以用下面语句查询第一行学号和姓名是否一致，这样错误一般不会错。

```
Select * from pub.student where sid=200900130417
```

可以用下面语句查询第一行学生的所有选课和每门课的学分，口算加起来看看是否正确。

```
Select * from pub.student_course t1, pub.course t2 where t1.cid=t2.cid and sid=200900130147
```

也可以直接把这个学生学分求和，对比是否正确。

```
Select sum(credit) from pub.course t1
```

```
where cid in (select cid from pub.student_course where sid=200900130147 and score>=60)
```

如果第一行没有错，那就继续这样第二行，第三行，一直找下去。

也可能，在这样找的过程中，启发你的想法，想起可能忽略的问题。

四、实验目的

不仅仅熟练掌握 SQL 的语法，更加主要目的**熟练掌握各种常用的、逻辑复杂的 SQL**，这些 SQL 都是老师从实际工作中精心提炼的、非常有代表性的 SQL。

五、实验环境

如果在校园网内，不需要启动 vpn 就可以如下实验平台网站，否则，必须启动 vpn 并登入。

- 1、数据库系统: oracle 11
- 2、主平台网址: <http://211.87.227.203:7088>
- 3、备平台网址: <http://211.87.227.202:7088>
- 4、备平台网址: <http://211.87.227.201:7088>
- 5、浏览器: firefox (**特别强调一下，只能使用这个浏览器，其他浏览器严重不稳定**)
- 6、主账号/密码: userID/123 (ID 为本人学号, 以下相同, 例如 user201000300001)

选修的**研究生学号**不足 12 位的，后面加 4 个 0，例如学号 20131334 的账号为 user201313340000。

国际生账号为学号中数字部分再加上 5 个 0，例如学号 GJ2022039 的账号为 user202203900000。

- 7、修改密码命令: alter user 账户 identified by 新密码。例如:

```
alter user user202200390001 identified by sdu456
```

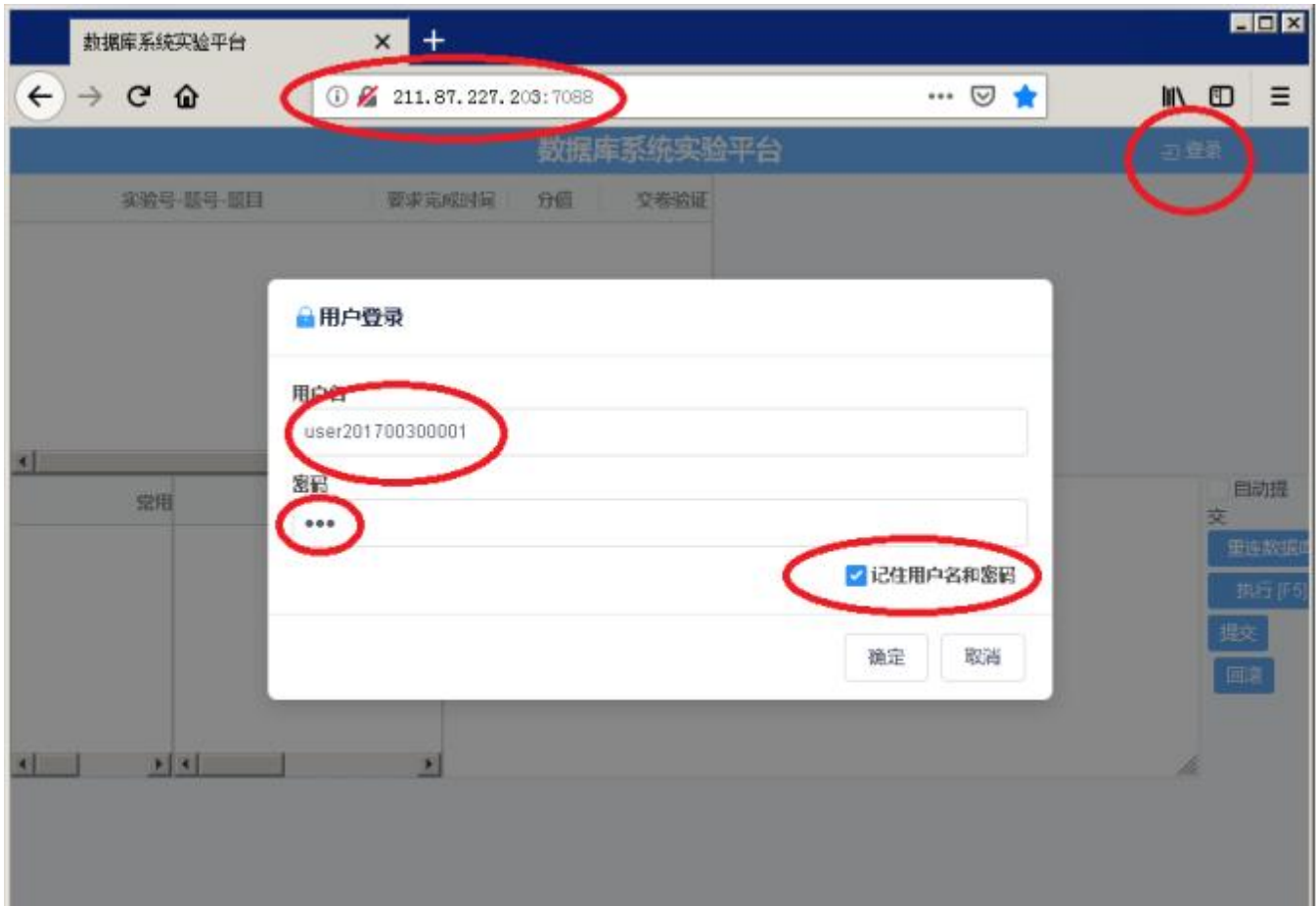
- 8、没有账号的选修或者插班的同学，直接私信 QQ 群中李老師，**发送学号、姓名、插班班级给老师**，老师创建好账号后会回复你。

六、临时账号

- 1、临时账号第一个作用题目得分后测试更多方法，第二个作用就是完成实验八需要两个账号
- 2、当某个题目已经完成并得分，自己的账号就不能再一次交卷验证了。
- 3、为了方便同学采用多方法测试同一个题目，系统为每个人设置了临时账号，临时账号为学号第一位变成 9，例如原来账号 user200020002000，临时账号 user900020002000，密码 123，姓名为[姓名]。
- 4、用自己的账号完成实验得分后，相应题目才会出现在自己的临时账号里。
- 4、临时账号成绩会每个整点清除，方便进行多次测试用。

七、实验操作说明

启动浏览器，输入网站地址，显示如下画面，点击登录，输入账号密码完成登录。



正确登录系统，显示主窗口。

提示：进入主窗口后，20 分钟不做任何操作或者登录总时长 60 分钟后，系统会自动断开与数据库连接，如果再进行操作，结果区会显示“您尚未登录”，这时需要执行【退出】，然后重新【登录】。

数据库系统实验平台

211.87.227.213:7088

200020002000 李老師 [总分: 29 排名: 1/3] 下载大纲 退出

实验号-题号-题目	要求完成时间	分值	交卷验证	实际完成时间	得分	状态
7-1: 统计名字使用频率	2019-03-26	2.5	交卷验证	2019-03-29 17:38:05		按时完成
7-2: 统计名字的每个字使用频数	2019-03-26	2.5	交卷验证	2019-03-29 17:38:11		按时完成
7-3: 学院班级学号分达标情况统计表1	2019-05-26	2.5	交卷验证	2019-03-29 17:38:09		按时完成
7-4: 学院班级学号分达标情况统计表2	2019-05-26	2.5	交卷验证	2019-03-29 17:38:09		按时完成
8-1: 统计报表1	2019-06-02	2.5	交卷验证	2019-03-29 17:38:08		按时完成

2. 在学生表pub.student中统计名字 (姓名的第一位是姓氏, 不作统计, 名字指姓名的第二个之后的汉字) 的每个字使用的频数, 将统计结果放入test7_02中 (特别提示: 需要区别union和union all的不同), 表结构如下:

```
letter varchar(2) frequency number(10,2)
```

常用表及视图

历史SQL

执行时间

SQL编辑区

只能编辑、执行一条SQL

执行区

结果: 0

行数: 1279

耗时: 0.003s

#	SID	NAME	SEX	AGE	BIRTHDAY	DNAME	CLASS
1	200900131156	韩鹏程	女	22	1990-08-19 00:00:00	物理学院	2007
2	200900131158	焦正安	女	23	1989-02-11 00:00:00	计算机科学与技术学院	2009
3	200900131165	李文俊	女	24	1988-06-17 00:00:00		2010
4	200900131167	傅锐峰	女	19	1993-12-25 00:00:00	公共卫生学院	2006
5	200900131171	郎洪飞	女	21	1991-09-05 00:00:00	外国语学院	2010
6	200900131178	郭玲媛	女	23	1989-07-01 00:00:00	国际教育学院	2007
7	200900131200	江美芳	女	22	1990-03-13 00:00:00	生命科学学院	2006
8	200900131247	李俊德	女	22	1990-04-23 00:00:00	物理学院	2005
9	200900131267	曹永光	女	21	1991-10-05 00:00:00	物理学院	2008
10	200900131271	侯占龙	女	18	1994-11-20 00:00:00	公共卫生学院	2011
11	200900131298	冯志亮	女	23	1989-09-23 00:00:00	外国语学院	2004
12	200900131300	胡志龙	女	20	1992-08-04 00:00:00	马克思主义学院	2007
13	200900131307	范凤岗	女	18	1994-08-07 00:00:00	生命科学学院	2005
14	200900131309	曹明焕	女	21	1991-11-25 00:00:00	历史文化学院	2006
15	200900131316	唐三花	女	19	1993-11-24 00:00:00	公共卫生学院	2010

结果可以翻页

共 1279 条 < 1 2 3 ... 86 > 15 条/页

数据库系统实验平台

200020002000 李老師 [总分: 67 排名: 1/19] 下载大纲 退出

实验号-题号-题目	要求完成时间	分值	交卷验证	实际完成时间
4-3: 统计总学分	2022-04-25	1	交卷验证	
4-4: 设置院系编号	2022-04-25	1	交卷验证	
4-5: 几项内容综合	2022-04-25	1	交卷验证	
4-6: 剔除姓名中的空格	2022-04-25	1	交卷验证	
4-7: 规范性别	2022-04-25	1	交卷验证	

常用表及视图

执行时间

历史SQL

update test4_02 t
set (sum_score,avg_score)=
(select sum(score),round(avg(score),1)
from pub.student_course
where t.sid=aasid)

错误说明区

结果: ORA-00904

行数: 0

耗时: 0.0s

详述: ORA-00904: "AASID": 标识符无效

位置: 第5行, 第16列 第5行第16列 AASID标识符无效

主窗口主要分为 8 个区:

1、个人信息区, 显示学号、姓名、总得分、**班级排名** (点击排名可以显示全部详细排名列表)、下载大纲。

2、题目区，显示实验题目、**要求完成时间**、分值、交卷验证（点击每个题目的交卷验证，后台自动验证实验是否正确，如果正确给出得分，如果错误，给出错误提示）、得分、实际完成时间、状态（包含错误提示）。

状态列在实验完成后，状态列还会包含上传实验报告时间，点击这一行，详细说明区就会显示实验报告存在问题。

3、详细说明区，点击题目区一行题目时，显示本题目的详细说明。这个区域显示内容分两种情况，当实验开始没有取得分值前，这个区显示这个题目详细要求及说明；当实验完成取得分值后，这个区显示这个实验报告完成情况以及存在问题。

4、常用表区，显示当前自己所有表及视图、公共账号 pub 的所有表，单击一行表会在结果区显示表结构（如果这一行是视图则不显示结构），双击一行会在结果区显示表数据。

5、历史 SQL 区，显示历史上自己曾经执行过的所有 SQL，按执行时间倒序，双击一行则该 SQL 自动进入编辑区。

6、编辑区，**用来输入编辑一条 SQL 命令，提醒一次只能输入执行一条 SQL。**

7、执行区，自动提交如果选择，则没执行一条 SQL 后，系统自动执行 commit，否则需要手工执行“提交 commit”或者“回滚 rollback”。执行-运行编辑区的一条 SQL 指令，执行情况显示在执行区下方，执行结果显示在结果区。

“数据导出”功能，可以将查询后的结果数据导出到浏览器默认下载目录的“查询结果.xls”文件中。

8、结果区或者错误说明区，执行一条 select 执行后，显示返回的数据结果，如果数据较多，会自动分页显示。

八、 上机时间统计(仅对安排专门机房、固定时间的本科学生)

1、实验学时数：8*2

2、实验课可以在机房完成，也可以在宿舍提前完成。

3、上机无需签到，鼓励大家来机房上机。

九、 上传实验报告

1、实验完成后，完成实验报告并上传到指定服务器。

2、实验报告文件名：学号-姓名-实验题目（如果学号不是 12 位，使用登入账号后 12 位）。

例如：200020002000-李同学-实验 1-建表、删除表、插入数据.docx。

3、上传到服务器后，可以在相应实验的题目 1 状态后面看到上传日期（如下图），如果看不到，可能原因文件名中学号或者姓名或者题目有错误，修改后重新上传。

数据库系统实验平台						
实验号-题号-题目	要求完成时间	分值	交卷验证	实际完成时间	得分	状态
1-1: 创建test1_student表	2021-10-05	1	已交卷	2021-12-15 15:51:42	1	按时完成 / 报告已报:10-05 20:51
1-2: 创建test1_course表	2021-10-05	1	已交卷	2021-12-15 15:51:44	1	按时完成
1-3: 创建test1_student_course表	2021-10-05	1	已交卷	2021-12-15 15:51:45	1	按时完成
1-4: 表test1_student插入2行数据	2021-10-05	1	已交卷	2021-12-15 15:51:46	1	按时完成
1-5: 表test1_course插入2行数据	2021-10-05	1	已交卷	2021-12-15 15:51:48	1	按时完成

常用表及视图	执行时间	历史SQL
pub.teacher_course		
pub.teacher		
pub.student_testindex10		
pub.student_testindex		
pub.student_course_32		
pub.student_course		

十、常用系统视图

Oracle 系统视图:

- 1、当前用户所有表和视图: `select * from tab`
- 2、所有用户的所有表: `select * from all_tables`
- 3、所有用户的所有视图: `select * from all_views`
- 4、所有用户的所有表的列: `select * from all_tab_columns`

平台系统视图:

- 5、当前用户所有历史 SQL: `select * from sql`, 查询执行过的所有 sql。
- 6、视图 sql1 至 sql9, 对应实验一至实验九的所有执行过得 sql, 导出后写报告用。
- 7、全年级排名: `select * from rank`, 排名按成绩排名, 成绩相同, 按完成时间先完成排前面。

十一、命名要求

严格按照要求进行实验, 表名、列名、类型、长度、以及数据等严格按照要求, 不要做简称等变化, 否则系统会判为错误。

十二、如何排除 SQL 语法错误

排除语法错误按以下步骤, 很容易排除语法错。

撇开逻辑不管, 将你认为有把握的部分程序保留, 删除其他部分代码, 如果剩下部分能运行, 那就说明, 错误在删除的那部分里面, 然后一段一段增加被删除的内容, 直到出现运行错误, 那就可以确定就是最后这部分有语法错误, 当语法错误锁定在很小的范围, 基本上一眼就能找到错误地方了。

如果有子句, 你可以单独运行子句看看是否能正确运行; 如果子句引用了主句的列, 你可以将这

些列替换成常量进行测试，例如下面语句：

```
select * from pub.student_course t1
      where ..... and
      exists( select * from pub.student where t1.sid=sid and class='软件' )
```

将子句改成如下格式测试：

```
select * from pub.student where '20002000200'=sid and class='软件'
```

另外，这样的方法能排除运行错误，例如：“单行子查询返回多个行”，也可以通过这个办法来排除错误。

```
select sid,name,(select sum(score) from pub.student_course where sid=t1.sid) sum_score
      from pub.student t1;
```

将子句修改成如下格式测试：

```
select sum(score) from pub.student_course where sid='200020002000'
```

十三、 Oracle 相关知识简介

- 1、伪列：所谓伪列就是表中不存的列，可是像使用其他存在的列一样访问这些列。常用的伪列有 today、now、rownum、rowid、sysdate。
- 2、常用的几个函数：to_char()、to_date('2010-01-01 15:01:23' , ' yyyy-mm-dd hh24:mi:ss') 、取子串 substr()、子串查找 instr()。
- 3、Natural jion 和 jion 的语法有以下三种格式：

```
From table1 natural join table2
```

```
From table1 jion table2 using(sid)
```

```
From table1 jion table2 on table1.sid=table2.sid
```

没有：from table1 natural join table2 using(sid)

十四、 Oracle 与上课书上的差异之处

- 1、Oracle 字符串使用单引号分隔，不可以用双引号（双引号另有他用）。
- 2、Oracle 中没有 except 关键词，与其等价的是 minus 关键词。在使用 minus 时，select 语句不能够使用括号，正确格式如下：

```
select * from student where ..... minus select * from student .....,
```

错误写法是：

```
(select * from student where .....) minus (select * from student .....)
```


- 3、表别名的定义不能够有 As，例如正确的写法：

```
select * from student s,
```

错误的写法是：

```
select * from student as s。
```

- 4、重新命名的列名不可以在后面表达式中引用，除了可以出现在 order by 后面。

```
Select NC1, (select cname from ..... ) NCN,
```

```
    c2 * NC1, NCN || name    --这里引用 NC1、NCN 是非法的不允许
```

```
    c2 * (c1*12), (select cname from ..... ) || name --只能这样重复
```

```
from      .....
```

```
where NC1>100    --这里引用 NC1 是非法的不允许
```

```
order by NC1, NCN    --这里引用 NC1, NCN 是可以
```

或者通过**再生表**办法引用：

```
Select nc1, nc1, c2*nc1, NCN || name
```

```
From (Select NC1, (select cname from ..... ) NCN,
```

```
      c2 , name
```

```
      from .....)
```

```
where NC1>100
```

```
order by NC1, NCN
```

- 5、创建视图语句支持覆盖功能选项。

```
Create or replace view viewname as .....
```

十五、 评分标准

- 1、每周实验都有**要求完成时间**，可以提前完成后面的实验。

数据库系统实						
实验号-题号-题目	要求完成时间	分值	交卷验证	实际完成时间	得分	
1 - 1 : 创建test1_student表	2019-09-22	1	已交卷	2019-09-02 17:30:15	1	按时完成
1 - 2 : 创建test1_course表	2019-09-22	1	已交卷	2019-09-02 17:30:17	1	按时完成
1 - 3 : 创建test1_student_course表	2019-09-22	1	已交卷	2019-09-02 17:30:18	1	按时完成
1 - 4 : 表test1_student插入2行数据	2019-09-22	1	已交卷	2019-09-02 17:30:19	1	按时完成
1 - 5 : 表test1_course插入2行数据	2019-09-22	1	已交卷	2019-09-02 17:30:30	1	按时完成
<						

- 2、每个小题按时完成计全分，次周完成计 80%得分,再之后完成计 70%得分。
- 3、总成绩求和为实验总分，按 10 分或者 15 分或者 20 分制折算后计入考试成绩。
- 4、实验全部完成，总成绩达到 100 分后，对机房上机不再作要求。
- 5、抄袭复制他人 SQL，一旦被查实将会被扣分。

十六、 测试数据

由于学生不可能在实验期间插入大量的数据，因此实验课之前老师已经在数据库中建了一个公用用户 pub，在这个用户下建立了实验用表，并且插入了大量的实验数据，本实验主要内容就是根据这些表进行操作。这些表已经授权给所有用户可以进行查询，但是不能够修改里面的数据。访问这些表的格式“用户名.表名”：

```
Select * from pub.student
```

表名	类型	数据行数	说明
COURSE	表	140	课程信息
DEPARTMENT	表	7	院系信息
DEPARTMENT_41	表	20	院系信息 1-实验三专用
STUDENT	表	4000	学生信息
STUDENT_41	表	4000	学生信息-实验三专用
STUDENT_42	表	3000	学生信息-实验三专用
STUDENT_31	表	4000	学生信息-实验四专用
STUDENT_COURSE	表	14000	学生选课
STUDENT_COURSE_32	表	14000	学生选课-实验四专用
TEACHER	表	200	教师信息
TEACHER_COURSE	表	40	教师授课信息

十七、 建议（可以不看）

- 1、要把实验当做实际开发工作一样对待，严格按设计要求执行。实际工作中对数据库的操作不仅仅是一门技术，更是一种技能，为什么是一种技能？你虽然会操作，别人一个小时完成的工作，你需要一天才能够完成，或者完成以后被发现和要求的有不一致，还需要重新返工，而且有的时候操作失误可能带来无法挽回的损失。因此，本实验要求大家严格按大纲要求正确输入表名、列名及数据。
- 2、学习计算机经常出现的现象就是，“一学就全会了，一做就全完成，一验收全错了（张冠李戴、缺斤

少两），一指点全改了”。所以，从开始就要严格要求自己，实际工作中，一点点也不能够有错误，从开始就养成一个严谨的好习惯。有时候前面错误会给后来工作带来很大的麻烦，例如：表名、列名错了，往往在发现错误的时候，数据已经输入很多，此时修改，需要先备份、再改表、再恢复数据。

实验一 建表、删除表、插入数据

一、 实验内容

创建 3 个表，为每个表输入 2 行数据，没有逻辑难度，只是熟悉环境，学会创建表。

表名、列名采用英文,oracle 不区分大小写，有 not null 的列代表不允许为空。

二、 相关知识

建表语句：

```
Create table test1_student (sid char(12),name varchar2(10));
```

insert 语句 1 次只能插入 1 行数据，插入多行有特殊语法，执行按钮一次只能执行一条 SQL，亦既一次只能执行一条 insert，因此编辑区一次也只能编辑一条 SQL 才能得到正确执行。

```
Insert into test1_student values('200020002000','王菲')。
```

下面两条 insert 一起执行会报错。

```
Insert into test1_dpstor values('200020002000','王菲');
```

```
Insert into test1_dpstor values('200020002001','张静');
```

一次插入多行数据的语法：

```
insert all  
  into test1_student values(200029002000,.....')  
  into test1_student values(200028002001,.....)  
select * from dual
```

删除已经创建的表：

```
Drop table test1_student
```

删除表中数据：

```
Delete from test1_student
```

删除表和删除表中数据一定不要混淆了。

三、 实验步骤

启动浏览器，输入地址：<http://211.87.227.203:7088>

点击【登录】，输入账号 userID（ID 为你的学号）、密码（初始密码为 123），进入如下主画面。

数据库系统实验平台

200020002000 李老師 [总分: 29 排名: 1/3] 下载大纲 退出

实验号-题号-题目	要求完成时间	分值	交卷验证	实际完成时间	得分	状态
7-1: 统计名字使用频率	2019-05-26	2.5	交卷验证	2019-03-29 17:38:05		按时完成
7-2: 统计名字的每个字使用频率	2019-05-26	2.5	交卷验证	2019-03-29 17:38:11		按时完成
7-3: 学院班级学分达标情况统计表1	2019-05-26	2.5	交卷验证	2019-03-29 17:38:09		按时完成
7-4: 学院班级学分达标情况统计表2	2019-05-26	2.5	交卷验证	2019-03-29 17:38:09		按时完成
8-1: 统计报表1	2019-06-02	2.5	交卷验证	2019-03-29 17:38:08		按时完成

2. 在学生库pub.student中统计名字(姓名的第一位是姓氏, 不作统计, 名字指姓名的第二个之后的汉字)的每个字使用的频率。将统计结果放入test7_02中(特别提示: 需要区别union和union all的不同), 表结构如下:

```
letter varchar(2) frequency number(10,2)
```

常用表及视图

历史SQL

执行时间

更新test10_01 on test10_02 set (sum_score, avg_score)= (select sum(scor...)

创建索引test10_02 on test10_02(birthday, name)

创建表test10_02 as select sid, name, birthday from pub.student

删除表test10_02

创建索引test10_01 on test10_02(birthday, name)

SQL编辑区

执行按钮一次只能执行一条SQL
因此这里也只能输入一条SQL才能正确执行

执行区

提交 回滚

行数: 3990
耗时: 0.016s

数据导出

#	SID	NAME	SEX	AGE	BIRTHDAY	DNAME	CLASS
1	200900131156	韩鹏程	女	22	1990-08-19 00:00:00	物理学院	2007
2	200900131158	焦正安	女	23	1989-02-11 00:00:00	计算机科学与技术学院	2009
3	200900131165	李文俊	女	24	1988-06-17 00:00:00		2010
4	200900131167	傅锐峰	女	19	1993-12-25 00:00:00	公共卫生学院	2006
5	200900131171	郎洪飞	女	21	1991-09-05 00:00:00	外国语学院	2010
6	200900131178	郭玲媛	女	23	1989-07-01 00:00:00	国际教育学院	2007
7	200900131200	江美芳	女	22	1990-03-13 00:00:00	生命科学学院	2006
8	200900131247	李俊德	女	22	1990-04-23 00:00:00	物理学院	2005
9	200900131267	曹永光	女	21	1991-10-05 00:00:00	物理学院	2008
10	200900131271	侯占龙	女	18	1994-11-20 00:00:00	公共卫生学院	2011
11	200900131298	冯志亮	女	23	1989-09-23 00:00:00	外国语学院	2004
12	200900131300	胡志龙	女	20	1992-08-04 00:00:00	马克思主义学院	2007
13	200900131307	范凤岗	女	18	1994-08-07 00:00:00	生命科学学院	2005
14	200900131309	曹明焕	女	21	1991-11-25 00:00:00	历史文化学院	2006
15	200900131316	唐三花	女	19	1993-11-24 00:00:00	公共卫生学院	2010

结果显示区

结果可以翻页

共 1279 条 < 1 2 3 ... 86 > 15 条/页

数据库系统实验平台

200020002000 李老師 [总分: 67 排名: 1/19] 下载大纲 退出

实验号-题号-题目	要求完成时间	分值	交卷验证	实际完成时间
4-3: 统计总学分	2022-04-25	1	交卷验证	
4-4: 设置院系编号	2022-04-25	1	交卷验证	
4-5: 几项内容综合	2022-04-25	1	交卷验证	
4-6: 剔除姓名中的空格	2022-04-25	1	交卷验证	
4-7: 规范性别	2022-04-25	1	交卷验证	

常用表及视图

执行时间

历史SQL

更新test4_02 t set (sum_score, avg_score)= (select sum(scor...)

更新test4_02 t set (sum_score, avg_score)= (select sum(scor...)

更新test4_01 t set (sum_score, avg_score)= (select sum(scor...)

创建表test4_02 as select t.*, 1000.00 sum_score, 100 avg_sco...

删除视图test4_02

SQL编辑区

执行按钮一次只能执行一条SQL
因此这里也只能输入一条SQL才能正确执行

执行区

提交 回滚

行数: 0
耗时: 0.0s

数据导出

结果: ORA-00904
行数: 0
耗时: 0.0s
详述: ORA-00904: "AASID": 标识符无效
位置: 第5行, 第16列 第5行第16列 AASID标识符无效

错误说明区

- 在 SQL 编辑区输入建表 SQL, 创建学生信息表(学生编号、姓名、性别、年龄、出生日期、院系名称、班级):

test1_student: sid char 12 not null、name varchar 10 not null、sex char 2、age int、

birthday date、dname varchar 30、class varchar 10。

建表语句常见错误如下：关键词拼写错误，少关键词、少逗号、少括号。

2. “交卷验证”——在题目区的实验 1-1 上点击“交卷验证”验证实验正确性，如果验证后有得分，继续下一个题目，否则删除重建。

交卷验证一般不会超过 5 秒，如果每题都超过 5 秒以上，请 QQ 私信李老师排查错误，谢谢。

3. 如果实验 1-1 题没有得分，可以通过“drop table test1_student”删除已经创建表，然后重新创建表，执行第 2 步再一次“交卷验证”。

常见错误：表名、列名、列类型和要求不一致，这里需要特别说明，一定要绝对一致。

4. 创建课程信息表(仅考虑一门课程最多一个先行课的情况)(课程编号、课程名称、先行课编号、学分)

test1_course: cid char 6 not null、name varchar 40 not null、fcid char 6、
credit numeric 4, 1 (其中 4 代表总长度，1 代表小数点后面长度)。

5. “交卷验证”——在题目区的实验 1-2 上点击“交卷验证”验证实验正确性。

6. 创建学生选课信息表(学号、课程号、成绩、教师编号、选课时间)

test1_student_course: sid char 12 not null、cid char 6 not null、
score numeric 5, 1 (其中 5 代表总长度，1 代表小数点后面长度)、tid char 6、sctime date。

7. “交卷验证”——在题目区的实验 1-3 上点击“交卷验证”验证实验正确性。

8. 给表 test1_student 插入如下 2 行数据。

学号	姓名	性别	年龄	出生日期	院系名称	班级
201800020101	王欣	女	21	1994-2-2	计算机学院	2010
201800020102	李华	女	20	1995-3-3	软件学院	2009

编辑框输入类似如下的插入 sql，然后按执行，既可以完成 1 行数据插入。

```
insert into test1_student  
values(200700030101,'某某某','男',19,to_date('20120202','yyyymmdd'),'某某学院','2010')
```

编辑框输入 select *from test1_student 后按执行，可以查询表 test1_student 中已经成功插入的数据，对比插入的数据是否和要求完全一致。

特别提醒：

insert 语句 1 次只能插入 1 行数据，插入多行有特殊语法，执行按钮一次只能执行一条 SQL，亦既一次

只能执行一条 insert，因此编辑区一次也只能编辑一条 SQL 才能得到正确执行。

引号一定也采用英文单引号，不能是双引号，也不能是中文的引号。下面的引号都是英文单引号，但是显示出来和中文单引号一样，如果复制到记事本，你就可以看到不是这个样子了。

9. “交卷验证”实验 1-4 正确性。

如果提交作业后，显示比答案少几行，错误原因是你插入的数据和下面数据不一样，系统要求插入的三行数据必须绝对一致。

解决办法：执行 `select * from test1_student` 查询插入数据的结果，和大纲逐条逐项对比。发现错误后，通过 `delete from test1_student where` 语句删除错误数据，然后重新插入正确的数据，再一次“交卷验证”。

常见错误：不一致或者多一位、少一位、多空格等。

10. 给表 test1_course 插入如下 2 行数据。

特别提醒：

插入空值使用 null，一次只能插入一行数据

课程号	课程名	先行课程号	学分
300001	数据结构		2
300002	数据库	300001	2.5

11. “交卷验证”实验 1-5 正确性。

12. 给表 test1_student_course 插入如下 2 行数据。

特别提醒：

一次只能插入一行数据

日期时间输入函数 `to_date(‘2020-01-03 15-26-26’,’ yyyy-mm-dd hh24-mi-ss’)`

学号	课程号	成绩	教师编号	选课时间
201800020101	300001	91.5	200101	2009-7-15 09:09:09
201800020101	300002	92.6	200102	2009-7-15 10:10:10

13. “交卷验证”实验 1-6 正确性。

14. 查询班级排行榜，执行：

```
select * from rank
```

实验二 检索查询

特别提醒:

1、实验二至实验九，都是使用 pub 用户的表 `pub.student`、`pub.student_course`、`pub.course` 等。

- 2、实验要求创建视图，最好不要采用先创建一个临时表，把表中数据通过几条 `update` 统计一下，然后依据这个临时表创建视图。原因是所以要求用视图，就是为了让掌握一种类型的用一条 `select` 的实现方法，毕业后实际工作中，本来一个视图显示到浏览器根本不需要程序，你上面办法就需要编写两行程序然后再调用视图，这样软件就太垃圾了。当然还有很多情况用临时表的办法就行不通，比如平台上常用视图 `sql`，每个人看到的都不一样，和自己学号关联，难道在数据库为每个人创建一个临时表吗？
- 3、定义视图语句不允许超过 600 字符，否则说明简单问题复杂化了，本大纲所有查询都不应该超过 600 字符，说明你的思路或者采用的方法有问题，你需要调整一下思路或者调整一下采用的方法。

一、实验步骤

1. 登录数据库系统实验平台，启动浏览器，输入如下地址：

<http://211.87.227.203:7088> 或者 <http://211.87.227.202:7088> 或者 `http://211.87.227.203:7088`

2. 假设实验二的题目 1 是 “找出所有有选课且成绩及格的学生的学号、总成绩。”

根据题目要求写出答案查询语句：

```
select sid,sum(score) from pub.student_course where score>=60 group by sid
```

3. 通过下面语句将查询结果创建到一个视图中 `test2_01`，其中 `test2` 代表实验二，`01` 代表题目 1。

```
Create or replace view test2_01 as select sid,sum(score) sum_score from pub.student_course  
where score>=60 group by sid
```

4. 点击题目实验 2-1 “交卷验证”，平台验证 `test2_01` 结果是否正确，如果正确给出得分，否则，给出错误原因。
5. 如果错误，则返回步骤 2，重复上面步骤，直到完成实验，获得本小题得分。

特别提醒：oracle 列别名定义不需要 `as`，

正确格式：`select sum(score) sum_score from`

错误格式：`select sum(score) as sum_score from`

更多 oracle 格式不同见大纲开始的 “《Oracle 与书上差异之处》”

二、常见错误

实验平台的测试数据量在现有服务器上运行，如果逻辑正确的情况下，运行任何 SQL 都不会超过 60 秒。如果出现下面的错误，就代表你的 `sql` 逻辑有问题，大多数情况表关联不对，或者条件不对。

1. 交卷验证, 出现“查询视图远远超时”, 创建的视图运行超过 60 秒。
2. 执行 sql 出现“ora-02393:超出 CPU 使用的调用限制”, 代表 sql 运行超过 1 分钟了。

三、实验题目

1. 找出没有选修任何课程的学生的学号、姓名(即没有选课记录的学生)。

自己认为查询语句正确后, 通过下面语句将查询语句创建成视图 test2_01

```
Create or replace view test2_01 as select .....
```

然后就可以点击实验二的题目 1 的【交卷验证】, 验证正确性, 正确后就有得分。

如果结果不对, 会提示错误几行或者多几行或者少几行, 这时你需要思考解题思路有没有遗漏的情况, 所有情况是否都有相应条件进行约束, 或者显示所得数据, 一条一条查看是否符合条件。

特别提醒:

1、逻辑错误尽量不要去问老师错在哪里了, 老师虽然知道答案但那不是唯一答案, 在不了解你的解题思路情况下, 根本不知道你的错误是思路错误, 还是实现的语句有错误。

2、如果出现错误一两行情况, 可以私信 QQ 群李老师, 提醒你可能你忽略的特殊情况。

2. 找出至少选修了学号为“200900130417”的学生所选修的一门课的学生的学号、姓名(不包含这名同学)。

自己认为查询语句正确后, 通过下面语句将查询语句创建成视图 test2_02

```
Create or replace view test2_02 as select .....
```

然后就可以点击实验二的题目 2 的【交卷验证】, 验证正确性, 正确后就有得分。

以下各题操作类似。

3. 找出至少选修了一门其先行课程号为“300002”号课程的学生的学号、姓名。
4. 找出选修了“操作系统”并且也选修了“数据结构”, 但是没有选修“程序设计语言”的学生的学号、姓名。
5. 找出姓名叫“李龙”的学生的学号及其选修全部课程的课程号、课程名和成绩。
6. 查询 2010 级、计算机科学与技术学院、操作系统的学生成绩表, 内容有学号、姓名、成绩。
7. 查询所有不姓张、不姓李、也不姓王的学生的学号 sid、姓名 name
8. 找出有间接先行课的所有课程的课程号、课程名称。
9. 查询选修了 300003 号课程的学生的 sid、name、score
10. 找出选修了所有课程的学生的学号、姓名。

实验三 复制表、删除数据

一、实验内容

将 pub 用户的表及数据复制到主账号下，对不符合要求的数据，使用 delete 语句进行删除。

二、预备知识

如何复制表

Create table 表名 as select.....from..... 语句将查询结果自动创建一个新表，例如 Create table test2 as select * from test1 就是实现表 test1 复制到 test2，当 select 查询语句存在表达式的，可以通过别名来指定列名。格式如（红色部分来指定列名）：

```
create table student_avg_score as select sno,sname,avg(score) avg_score from .....
```

更多格式和用法去网上查询。

三、实验题目

如何复制表

Create table 表名 as select.....from..... 语句将查询结果自动创建一个新表，如果 select *from 也就是实现表的复制，当 select 查询语句存在表达式的，可以通过属性更名来指定列名。格式如：

```
create table student_avg_score as select sno,sname,avg(score) avg_score from .....
```

红色部分来指定列名。

1. 将 pub 用户下的 Student_31 及数据复制到表 test3_01 中(注意：test3_xx 要建成表不是视图，否则删除数据时会显示无此权限)，用 delete 删除语句，删除表中的学号不全是数字的那些错误数据，学号应该是数字组成，不能够包含字母空格等非数字字符。

方法之一：用 substr 函数，例如 Substr(sid,1,1) 返回学号的第一位，判断是否是数字。

2. 将 pub 用户下的 Student_31 及数据复制到主账号的表 test3_02，用 delete 删除语句，删除表中的出生日期和年龄(截止到 2012 年的年龄，即年龄=2012-出生年份)不一致的那些错误数据。

函数 extract(year from birthday) 返回 birthday 的年份

3. 将 pub 用户下的 Student_31 及数据复制到主账号的表 test3_03，用 delete 删除语句，删除表中的性别有错误的那些错误数据（性别只能是“男”、“女”或者空值）。
4. 将 pub 用户下的 Student_31 及数据复制到主账号的表 test3_04，用 delete 删除语句，删除表中的院系名称有空格的、院系名称为空值的或者院系名称小于 3 个字的那些错误数据。
5. 将 pub 用户下的 Student_31 及数据复制到主账号的表 test3_05，用 delete 删除语句，删除表中的班级不规范的那些错误数据，不规范是指和大多数不一致。

提示：这个题知识点是学会用 SQL 找出不规范的数据，而不是靠猜测或者人眼找到不规范，如果没有学会好解决办法，请参考大纲最后的提示。

6. 将 pub 用户下的 Student_31 及数据复制到主账号的表 test3_06，用 delete 删除语句，删除表中的错误数据，不规范的数据也被认为是那些错误数据。
- 学号不全是数字；
 - 出生日期和年龄不一致的（年龄=2012-出生年份）；
 - 姓名有空格的或者长度小于 2 个字的；函数 length() 返回字符串长度。
 - 性别有错误的（只能是“男”、“女”、空值）；
 - 院系名称有空格的、院系名称为空值的；
 - 院系名称小于 3 个字的；
 - 班级数据有错误的（需要先找到班级里面的错误）。

保留最后全部正确的数据。

7. 将 pub 用户下的 Student_course_32 及数据复制到主账号的表 test3_07，用 delete 删除语句，删除其中的错误数据，错误指如下情况：
学号在学生信息 pub.student 中不存在的；
8. 将 pub 用户下的 Student_course_32 及数据复制到主账号的表 test3_08，用 delete 删除语句，删除其中的错误数据，错误指如下情况：
课程号和教师编号在教师授课表 pub.teacher_course 中不同时存在的，即没有该教师教该课程；
9. 将 pub 用户下的 Student_course_32 及数据复制到主账号的表 test3_09，用 delete 删除语句，删除其中的错误数据，错误指如下情况：
成绩数据有错误（需要先找到成绩里面的错误）。

提示：这个题知识点是学会用 SQL 找出不规范的数据，而不是靠猜测或者人眼找到不规范，如果没有学会好解决办法，请参考大纲最后的提示。

10. 将 pub 用户下的 Student_course_32 及数据复制到主账号的表 test3_10，用 delete 删除语句，删除其中的错误数据，错误指如下情况：
- (1) 学号在学生信息 pub.student 中不存在的；
 - (2) 课程号在课程信息 pub.course 中不存在的；
 - (3) 教师编号在教师信息 pub.teacher 中不存在的；
 - (4) 课程号和教师编号在教师授课表 pub.teacher_course 中不存在的；
 - (5) 成绩数据有错误（需要先找到成绩里面的错误）。

保留最后正确的数据。

检查你有没有中枪？

一、简单写法非要复杂写：

复杂写法：Delete from test3_01 where sid in (select sid from test3_01 where);

简单写法：delete from test3_01 where;

有的时候，简单写法复杂写，因为空值原因还造成逻辑错误：

简单写法：delete from test3 where sex='男' or sex='女' or sex is null

复杂写法：delete from test3 where sex not in (select sex from test3 where sex='男' or sex='女' or sex is null)

上面这两句话删除的结果是不一样的，因为是空值原因。

二、正确删除步骤：

- 1、通过 select * from student where找出要删除的语句。
- 2、人工检查是不是找出来的数据就是符合要删除的条件的数据，如果错误修改 where。
- 3、将查询改成删除语句，只需将“select *”换成“delete”就可以了。
- 4、如果不查询直接删除，那你可能需要多次建表，如果真的业务数据，每删错一次你就需要半天时间去从备份数据中恢复数据，业务系统中从备份数据恢复是很辛苦。

实验四 复制表、修改表结构、修改数据

一、实验内容

利用 oracle 管理平台完成对表的结构、数据进行修改，每一个问题可以通过多个 SQL 语句完成。

二、实验题目

1. 将 pub 用户下表 student_41 及数据复制到主账号的表 test4_01 中, 使用 alter table 语句为表增加列:
“总成绩:sum_score”。

使用 update 语句, 利用 pub.student_course, 统计 “总成绩”;

2. 将 pub 用户下表 student_41 及数据复制到主账号的表 test4_02 中, 使用 alter table 语句为表增加列
“平均成绩:avg_score” (小数点后保留 1 位)。

利用 pub.student_course, 统计 “平均成绩”, 四舍五入到小数点后 1 位

3. 将 pub 用户下表 student_41 及数据复制到主账号的表 test4_03 中, 使用 alter table 语句为表增加列:
“总学分:sum_credit”。

使用 update 语句, 利用 pub.student_course、pub.course, 统计 “总学分”;

这是需要注意:

1、成绩及格才能够计算所得学分。

2、一门课多个成绩都及格(刷分情况)只计一次学分。

4. 将 pub 用户下表 student_41 及数据复制到主账号的表 test4_04 中。

根据列院系名称 dname 到 pub.department 找到对应院系编号 did, 将对应的院系编号回填到院系名称列 dname 中, 如果表中没有对应的院系名称, 则列 dname 中内容不变仍然是原来的内容。

5. 将 pub 用户下表 student_41 及数据复制到主账号的表 test4_05 中, 使用 alter table 语句为表增加 4 个列: “总成绩:sum_score”、“平均成绩:avg_score”、“总学分:sum_credit”、“院系编号:did varchar(2)”。

(1) 利用 pub.student_course、pub.course, 统计 “总成绩”;

(2) 利用 pub.student_course、pub.course, 统计 “平均成绩”, 四舍五入到小数点后 1 位;

(3) 利用 pub.student_course、pub.course, 统计 “总学分”;

(4) 根据院系名称到 pub.department 和 pub.department_41 中, 找到对应编号, 填写到院系编号中, 如果都没有对应的院系, 则填写为 00。

说明: 执行 update 后, 在查询表中数据, 可能出现顺序变化, 这是正常, 因为数据在表中是无序。需要顺序的时候可以通过 orderby 实现。

6. 将 pub 用户下的 Student_42 及数据复制到主账号的表 test4_06 中, 对表中的数据进行整理, 修复那些

不规范的数据：

剔除姓名列中的所有空格；

7. 将 pub 用户下的 Student_42 及数据复制到主账号的表 test4_07 中, 对表中的数据进行整理, 修复那些不规范的数据：

对性别列进行规范（需要先确定哪些性别数据不规范，也就是那些和大多数不一样的就是不规范的）；

8. 将 pub 用户下的 Student_42 及数据复制到主账号的表 test4_08 中, 对表中的数据进行整理, 修复那些不规范的数据：

对班级列进行规范（需要先确定哪些班级不规范）。

9. 将 pub 用户下的 Student_42 及数据复制到主账号的表 test4_09 中, 对表中的数据进行整理, 修复那些不规范的数据：

年龄为空值的根据出生日期设置学生年龄（截止到 2012 年的年龄，即年龄=2012-出生年份），年龄不为空值的不要改变。

10. 将 pub 用户下的 Student_42 及数据复制到主账号的表 test4_10 中, 对表中的数据进行整理, 修复那些不规范的数据：

(1) 剔除姓名列中的所有空格；

(2) 剔除院系名称列中的所有空格；

(3) 对性别列进行规范（需要先确定哪些性别数据不规范，也就是那些和大多数不一样的就是不规范的）；

(4) 对班级列进行规范（需要先确定哪些班级不规范）。

(5) 年龄为空值的根据出生日期设置学生年龄（截止到 2012 年的年龄，即年龄=2012-出生年份），年龄不为空值的不要改变。

实验五 报表统计

一、实验内容

- 1. 学习掌握 sum, count 函数和 group by 的用法。
- 2. 学习掌握一个表和自己进行建立连接的查询。
- 3. 部分题目可以使用 union all 来解决。

二、实验题目

- 1. 在学生表 pub.student 中统计名字（姓名的第一位是姓氏，其余为名字，不考虑复姓）的使用的频率，将统计结果放入表 test5_01 中，表结构如下。

First_name varchar(4)	frequency numeric(4)
国强	1034
红	1232
卫东	2323
.....	

- 2. 在学生表 pub.student 中统计名字（姓名的第一位是姓氏，不作统计，名字指姓名的第二个之后的汉字）的每个字使用的频率，将统计结果放入表 test5_02 中，表结构如下。

letter varchar(2)	frequency numeric(4)
锋	1034
红	1232
鹏	2323
.....	

- 3. 先创建“学院班级学分达标情况统计表 1” test5_03，依据 pub.student， pub.course， pub.student_course 统计形成表中各项数据，成绩>=60 为及格（成绩>100 也算及格，也就是不用加条件成绩<=100，本实验以下各小题相同）计入学分，总学分>=10 算作达标，院系为空值的数据不统计在下表中，表结构：院系名称 dname、班级 class、学分达标人数 p_count1、学分未达标人数 p_count2、总人数 p_count。

Dname	class	P_count1	P_count2	P_count
varchar(30)	varchar(10)	Int	int	int
计算机学院	2006	4	5	9

计算机学院	2007			
软件学院	2006			
.....				

排错提示：

平台提示有多行数据错误，特别是行数比较多，可以从结果的第一行开始，一个列一个列进行单独统计，找到错误的是那个列数，然后对照你形成这个列的部分寻找原因。假设输出结果如上图，可以这样统计：

`select count(*) from pub.STUDENT where dname='计算机学院' and class=2006`，检查统计计算机学院 2006 级是不是 9。

`Select count(*)`检查统计计算机学院 2006 级达标人数是不是 4。

`Select count(*)`检查统计计算机学院 2006 级不达标人数是不是 5。

通过这样办法找到错在那个列，然后再去对应的 SQL 部分找错误就容易了。

4. 创建“学院班级学分达标情况统计表 2”test5_04, 依据 pub. student, pub. course, pub. student_course 统计形成表中各项数据，成绩 ≥ 60 为及格计入学分，2008 级及之前的班级总学分 ≥ 8 算作达标，2008 级之后的班级学分 ≥ 10 算作达标，院系为空值的数据不统计在下表中，表结构：院系名称 dname、班级 class、学分达标人数 p_count1、学分未达标人数 p_count2、总人数 p_count。

Dname varchar(30)	class varchar(10)	P_count1 int	P_count2 int	P_count int
计算机学院	2006			
计算机学院	2007			
软件学院	2006			
.....				

注意事项：

5.5 题至 5.10 题，全部建成视图, 即 test5_05、test5_06、test5_07、test5_08、test5_09、test5_10 建成视图。

5.5 题、5.6 题、5.7 题、5.8 题，学生一门课程有多次成绩（刷成绩），仅仅考虑最高成绩，也就是只用他的最好成绩参加统计。

5.9 题、5.10 题不用考虑刷成绩情况，亦即忽略刷成绩的情况。

实验要求创建视图，最好不要采用先创建一个临时表，把表中数据通过几条 update 统计一下，然后依据这个临时表创建视图。原因是所以要求用视图，就是让你掌握一种类型的用一条 select 的实现方法，毕业后实际工作中，本来一个视图显示到浏览器根本不需要程序，你上面办法就需要编写两行程序然后再调用视图，这样软件就太垃圾了。当然还有很多情况用临时表的办法就行不通，比如平台上常用视图 sql，每个人看到的

都不一样，和自己学号关联，难道在数据库为每个人创建一个临时表吗？

5. 查询各院系(不包括院系名称为空的)的数据结构平均成绩 avg_ds_score、操作系统平均成绩 avg_os_score，平均成绩四舍五入到个位，创建视图 test5_05，视图内容示例如下：

Dname	Avg_ds_score	Avg_os_score
马克思主义学院	72	70
软件学院	77	74
艺术学院	77	76
医学院	74	73

6. 查询” 计算机科学与技术学院” 的同时选修了数据结构、操作系统两门课的学生的学号 sid、姓名 name、院系名称 dname、数据结构成绩 ds_score、操作系统成绩 os_score，创建视图 test5_06，视图内容示例如下：

Sid	Name	Dname	Ds_score	Os_score
200020002000	张三	计算机科学与技术学院	87	78
200020002001	李四	计算机科学与技术学院	87	78
200020002002	王五	计算机科学与技术学院	78	77

7. 查询计算机科学与技术学院的选修了数据结构或者操作系统的学生的学号 sid、姓名 name、院系名称 dname、数据结构成绩 ds_score、操作系统成绩 os_score，创建视图 test5_07，视图内容示例如下：

Sid	Name	Dname	Ds_score	Os_score
200020002000	张三	计算机科学与技术学院		78
200020002001	李四	计算机科学与技术学院	87	78
200020002002	王五	计算机科学与技术学院	77	

8. 查询计算机科学与技术学院所有学生的学号 sid、姓名 name、院系名称 dname、数据结构成绩 ds_score、操作系统成绩 os_score，创建表 test5_08，视图内容示例如下：

Sid	Name	Dname	Ds_score	Os_score
200020002000	张三	计算机科学与技术学院		78
200020002001	李四	计算机科学与技术学院	87	78

200020002002	王五	计算机科学与技术学院	77	
200020002003	照照	计算机科学与技术学院		

9. 创建“成绩及格的学生每个成绩人数百分比统计表”视图 `test5_09`，依据 `pub.student_course` 统计形成视图中各项数据，视图结构：成绩 `score`、对应成绩人数 `count1`、及格总人数 `count0`、站总人数的百分比 `percentage`（即前面两列相除后乘以 100）。其中百分比四舍五入到小数点后面 2 位。输出按成绩排序。上面说的“人数”理解成“人次”更加准确，亦既一个同学同一门考试考两次，算两人次。

视图内容示例如下：

Score	Count1	Count0	percentage
60	66	1590	10. 01
61	67	1590	10. 02
.....		1590	
100	6	1590	1. 11

10. 创建“每个课程成绩在 60-149 分之间每段成绩的人数百分比统计表”视图 `test5_10`，依据 `pub.course`，`pub.student_course` 统计形成视图中各项数据，仅仅统计成绩在 60-149（包含 60 和 149）之间成绩，视图结构：课程编号 `cid`、课程名称 `cname`、成绩 `score`（这个列内容不要包含空格）、对应成绩人数 `count1`、及格总人数 `count0`（仅统计 60-149 分）、站总人数的百分比 `percentage`（即前面两列相除后乘以 100，用 `round` 函数四舍五入到小数点后面 2 位）。输出按课程编号、成绩排序。上面说的“人数”理解成“人次”更加准确，亦既一个同学同一门考试考两次，算两人次。这个题要求用 `trunc` 函数生成 `score` 列。

知识点 1：分组不仅可以用一个列分组，还可以用表达式分组，还可以用两个列或表达式分组。

知识点 2：确保只返回一个数据 `select` 可以理解成另外一个 `sql` 的表达式。

知识点 3：`trunc` 函数，可以截断小数点后面，也可以小数点前，还可以是日期。

提示：理解下面这句话，会启发怎么形成 `score` 列。

```
select distinct cid,score,to_char(trunc(score,-1),'fm000') || '-' || to_char(trunc(score-1)+9,'fm000')
from pub.student_course t1 where score >=60 and score <=149。
```

如果看了上面提示能解决最后，否则看进一步提示：

给出下面填空，相应“.....”里面填空就能实现这个答案。

```
Select cid,
       (select ..... from pub.course t2 where ..... ) cname,
```

```

..... score,
Count(*) count1,
(select count (*) from pub.student_course t3 where ..... ) count0,
Round(.....) percentage
From pub.student_course t1
Group by .....

```

或者喜欢用派生表的格式：

```

select t1.cid,name cname,score,count1,count0,
round(count1/count0*100,2) percentage
from pub.course t1,
(select cid,
..... score,
count(*) count1
from pub.student_course
where .....
group ..... ) t2,
(select cid ,count(*) count0
from pub.student_course
where .....
group by ..... ) t3
where t1.cid=t2.cid and t2.cid=t3.cid

```

视图内容示例如下：（060-069 之间是减号，不是下划线）

cid	Cname	Score	Count1	Count0	percentage
200001	数据结构	060-069	200	2000	10.07
200001	数据结构	070-079	400	2000	20.03
200001	数据结构	080-089	600	2000	30.44
200001	数据结构	130-139	800	2000	40.34
200002	数据库	060-069	500	1000	50.12
200002	数据库	090-099	500	1000	50.43
200002	数据库	120-129	500	1000	50.43

实验六 创建视图、删除视图

一、实验内容

oracle 管理平台, 针对公共用户 pub 下的表, 完成创建视图、查询验证视图、删除视图。视图名为 test6_ (题号, 题号长度两位, 前面补零), 例如 test6_01。

二、实验题目

例如: 找出年龄小于 20 岁的所有学生的学号、姓名、年龄

正确执行: `create view test6_00 as select sid, name, age from pub.student where age>20`

Oracle 扩展后方便写法:

`create or replace view test6_00 as select sid, name, age from pub.student where age>20`

1. 找出年龄小于 20 岁且是“物理学院”的学生的学号、姓名、院系名称, 按学号排序。
2. 查询统计 2009 级、软件学院所有学生的学号、姓名、总成绩 (列名 sum_score) (如果有学生没有选一门课, 则总成绩为空值)。
3. 查询所有课的最高成绩、最高成绩人数, test6_06 有四个列: 课程号 cid、课程名称 name、最高成绩 max_score、最高成绩人数 max_score_count (一个学生同一门课成绩都是第一, 只计一次, 需要考虑刷成绩情况, 一个同学选了一个课程多次, 两次都是最高分。如果结果错一行, 可能就是没有考虑这种情况, 这里是为了考核“去重复计数”知识点的)。如果没有学生选课, 则最高成绩为空值, 最高成绩人数为零。

提示: 参考讲义关于标量子查询 (只返回包含单个属性的单个元组)。

4. 找出选修了“操作系统”并取得学分或者选修“数据结构”并且取得学分, 但是没有选修“程序设计语言”或者没有取得这门课的学分的男学生的学号、姓名。
5. 查询 20 岁的所有有选课的学生的学号、姓名、平均成绩 (avg_score, 此为列名, 下同) (平均成绩四舍五入到个位)、总成绩 (sum_score)

Test6_05 有四个列, 并且列名必须是: sid、name、avg_score、sum_score。通过下面方式实现列名定义:

```
create or replace view test6_05 as select sid, name, (表达式) avg_score, (表达式) sum_score
from .....
```

6. 找出同一个同学同一门课程有两次或以上不及格的所有学生的学号、姓名 (即一门课程需要补考两次或以上的学生的学号、姓名)。
7. 找出选修了所有课程并且每门课程每次考试成绩均及格的学生的学号、姓名。

8. 找出选修了所有课程并且得到所有课程的学分（即每门课程最少一次考试及格）的学生的学号、姓名。
9. 查询统计 2010 级、化学与化工学院的学生总学分表，内容有学号、姓名、总学分 sum_credit。（不统计没有选课的学生）。
10. 查询学生表中每一个姓氏及其人数（不考虑复姓，用到取子串函数 substr(string, position, length)），test6_10 有两个列：second_name、p_count

实验七 索引重要性……提高速度

一、实验内容

对比有无索引情况下数据检索速度,学会如何能够使用索引,掌握如何查询是否使用索引了。

二、知识介绍,通过执行计划,了解索引使用情况

1. Pub 用户下 100 万行的表 student_10 (sid、name、sid1),按 sid、name 建立了两个索引。

```
create UNIQUE index student_10_sid on student_10(sid)
```

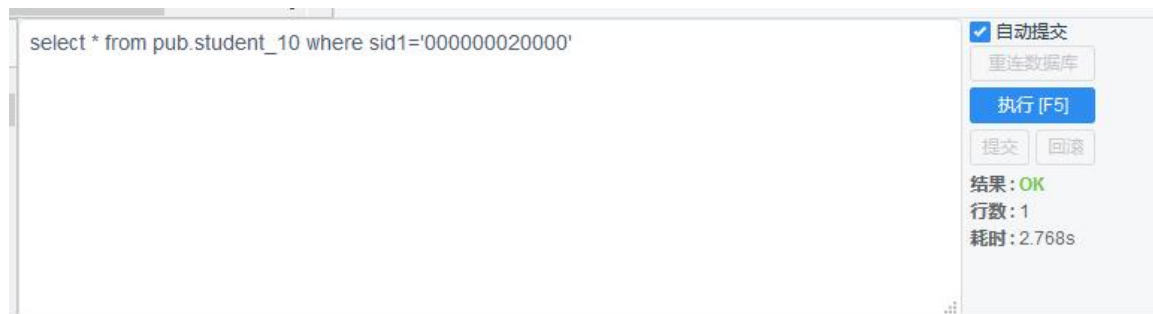
```
create index student_10_name on student_10(name)
```

2. 对比索引列 sid 和非索引列 sid1 进行查询,观察耗时

```
select * from pub.student_10 where sid='000000020000',耗时在毫秒级
```



```
select * from pub.student_10 where sid1='000000020000', 耗时在 1 秒以上
```



从上面这两句话对比,可以看到有索引的列比没有索引的列进行查询速度相差百倍,这不能简单理解成相差 1 秒。因为现在是 100 万行数据进行对比查询,如果是 1000 万*1 万行数据,那用时就是相差 1 万秒了。实际工作中,往往不会是一个表,可能是两个表以上进行连接,当两个 1000 万行的表进行笛卡尔关联时,复杂性就是 1000 万*1000 万,那用时可能就是相差 1000 万秒了。

下面 7000 行数据的实验内容能进一步来说明索引的重要性。

3. 什么是执行计划

执行计划就是数据库系统执行一句命令是按照什么样的顺序和方法完成的,其中包括是否使用了索引,

先使用的是那个索引，后使用的是那个索引， 按什么条件使用的索引。

更多知识参考

<https://www.cnblogs.com/Dreamer-1/p/6076440.html>

4. 怎么获取执行计划

执行下面命令，提交系统解释如下 SQL 的执行计划

```
explain plan for select * from pub.student_10 where sid='000000020000'
```

执行下面命令，查询刚刚提交系统解释的 SQL 的执行计划。

```
SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY);
```

PLAN_TABLE_OUTPUT							
Plan hash value: 2380613049							

Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time	

0	SELECT STATEMENT		1	74	3 (0)	00:00:01	
1	TABLE ACCESS BY INDEX ROWID	STUDENT11	1	74	3 (0)	00:00:01	
* 2	INDEX UNIQUE SCAN	STUDENT11_SID	1		2 (0)	00:00:01	

Predicate Information (identified by operation id):							

2 - access("SID"='000000020000')							

简单说明上面执行计划，第一步系统用条件 sid= '00000020000' 在索引 student_10_sid 中快速找到对应的那行数据的物理地址 rowid（因为 student_10_sid 是唯一索引，所以系统找到一行后就不再继续找了），第二步利用物理地址 rowid 在表 student_10 中直接取出对应那一行数据。

可以看到这条语句能使用索引 student_10_sid，估算用时为 1 秒（实际用时为 0.032 秒）。

其中名字解释：

名称（name）：表名称或者索引名称

基数（Rows）：Oracle 估计的当前操作的返回结果集行数

字节（Bytes）：执行该步骤后返回的字节数

耗费（COST）、CPU 耗费：Oracle 估计的该步骤的执行成本，用于说明 SQL 执行的代价，理论上越小越好（该值可能与实际有出入）

时间（Time）：Oracle 估计的当前操作所需的时间

*代表后面会有说明，说明使用的条件是什么。

5. 执行下面命令，解释用时多的 select 的执行计划

```
explain plan for select * from pub.student_10 where sid1='000000020000'
```

执行下面命令，查询用时多的 select 的执行计划


```
SELECT * FROM TABLE(DBMS_XPLAN.DISPLAY);
```

PLAN_TABLE_OUTPUT							
Plan hash value: 2824533770							

	Id	Operation	Name	Rows	Bytes	Cost (%CPU)	Time

	0	SELECT STATEMENT		1	74	21290 (2)	00:04:16
	* 1	TABLE ACCESS FULL	STUDENT11	1	74	21290 (2)	00:04:16

Predicate Information (identified by operation id):							

1 - filter(TO_NUMBER("SID")=000000020000)							
Note							

- dynamic sampling used for this statement (level=2)							

简单说明上面执行计划，系统用条件 `sid1= '00000020000'` 在表 `student_10` 中全表扫描一遍所有数据，取出符合条件的数据。系统即便是找到一行符合条件的数据，也要继续对全表进行扫描，因为它不知道是否还有符合条件的数据。思考为什么？

可以看到这条语句没用任何索引，估算用时为 4 分（实际用时 1.585 秒）。

对比两个 SQL 命令的执行计划，更容易理解索引为什么能提高速度。

6. 同样的方法对比下面两组查询语句。

```
Select count(*) from pub.student_10 where name like '阙%'
```

和

```
Select count(*) from pub.student_10 where substr(name,1,1)='阙'
```

以及

```
select * from pub.student_10 where sid=' 000000020000'
```

和

```
select * from pub.student_10 where sid=000000020000
```

通过对比，会发现每组第二句不能使用索引，为什么？

7. 相关知识

对索引列进行 `=`, `>=`, `<=`, `>`, `<`, **between** 等运算可以使用索引。

对索引列进行 **Like** 运算时，表达式前面没有 `%` 等通配符时则可以使用索引，为什么？理解索引原理就明白了。

对索引列进行运费后不能使用按列建立的索引，例如 `where sid=000000020000`（系统实际转换成

to_number(sid)=000000020000)，所以不能使用 sid 建立的索引。

如果将索引 student_sid 用 create index student_sid on student_10(to_number(sid))来创建，则两个语句正好相反，即条件 where sid=000000020000 能用索引，where sid=' 000000020000' 反而不能用索引。

假设日期类型列 birthday 有索引，当需要查询一个月份时，to_char(datecol,';yyyymm')='201801'不能使用索引，但是可以通过 datecol>=to_date('20180101','yyyymmdd') and datecol< to_date('20180201','yyyymmdd')解决，这个技巧非常实用。

三、 题目 1

1. 将 pub 用户下表 student 的 3 个列 sid,name,birthday 复制到表 test7_01 中。
2. 执行如下查询，观察运行速度（5 秒以上）。

查询 Samefirstname 相同姓氏的人数。

```
select * from
(select sid,name,birthday,
(select count(*) from test7_01 where substr(name,1,1)=substr(t1.name,1,1)) samefirstname
from pub.student_testindex t1)
where samefirstname=7
```

3. 为 test7_01 创建一个仅仅一个索引 test7_01_index，保证上面 SQL 耗时在 1 秒内。
4. 交卷验证。
5. 如果验证失败，提示耗时小于 2 秒，可以多次尝试提交阅卷。

四、 题目 2

1. 将 pub 用户下表 student 的 3 个列 sid,name,birthday 复制到表 test7_02 中。
2. 将出生日期部分修改成一天：

```
Update test7_02 set birthday=to_date('19881018','yyyymmdd') where substr(sid,12,1)= '0' ;
```

3. 为 test7_02 创建一个仅仅一个索引 test7_02_index，保证下面 SQL 耗时在 1 秒内。

Samenamebirthday 同名同生日的人数，Samebirthday 相同出生日期的人数

```
select * from
(select sid,name,birthday,
(select count(*) from test7_02 where name=t1.name and birthday=t1.birthday)
samenamebirthday,
(select count(*) from test7_02 where birthday=t1.birthday) samebirthday
from pub.student_testindex t1)
where samebirthday=403
```

4. 交卷验证。
5. 如果验证失败，提示耗时小于 2 秒，可以多次尝试提交阅卷。

五、题目 3

1. pub 用户下表 student 已经用下面两句 SQL 创建了两索引。
Create index student_birthday on student(birthday);
Create index student_name on student(name);
2. 下面 SQL 在访问 pub.student 时不能用索引 student_name 因此耗时超过 2 秒，在逻辑不变情况下，修改 SQL 中标为记红色的子查询的 where 条件部分，不要修改其它地方，使其能使用索引。
说明：因为 pub.student_testindex 数据行数太少，不能通过修改主句 where 绕过问题。
查询 samefirstname 同姓氏的人数。

```
select * from
    (select sid,name,birthday,
        (select count(*) from pub.student
            where substr(name,1,1)=substr(t1.name,1,1)
        ) samefirstname
    from pub.student_testindex t1) where samefirstname=7
```
3. 修改以后验证耗时在 2 秒之内，将修改以后语句创建成视图 create view test7_03 as select ……。
4. 交卷验证

六、题目 4

1. pub 用户下表 student 已经用下面两句 SQL 创建了两索引。
Create index student_birthday on student(birthday);
Create index student_name on student(name);
2. 下面 SQL 在访问 pub.student 时不能用索引 student_birthday 因此耗时超过 1 秒，在逻辑不变情况下，修改 SQL 中标为记红色的子查询的 where 条件部分，不要修改其它地方，使其能使用索引。
说明：因为 pub.student_testindex 数据行数太少，不能通过修改主句 where 绕过问题。

```
select * from
    (select sid,name,birthday,
        (select count(*) from pub.student
            where to_char(birthday,'yyyymm')=to_char(t1.birthday,'yyyymm')
        ) sameyearmonth,
        (select count(*) from pub.student
```

```
where extract (year from birthday) =extract (year from t1.birthday)
```

```
) sameyear
```

```
from pub.student_testindex t1) where sameyearmonth=35
```

3. 修改以后验证耗时在 1 秒之内，将修改以后语句创建成视图 create view test7_04 as select ……。
4. 交卷验证。
5. 如果验证失败，提示耗时小于 2 秒，可以多次尝试提交阅卷。

七、题目 5

1. pub 用户下表 student 已经用下面两句 SQL 创建了两索引。

```
Create index student_birthday on student(birthday);
```

```
Create index student_name on student(name);
```

2. 下面 SQL 在访问 pub.student 时不能用索引 student_birthday 因此耗时超过 1 秒，在逻辑不变情况下，修改 SQL 中标为记红色的子查询的 where 条件部分，不要修改其它地方，使其能使用索引。

说明：因为 pub.student_testindex 数据行数太少，不能通过修改主句 where 绕过问题。

查询 nextbirthday 晚一天出生的人数

```
select * from
```

```
(select sid,name,birthday,
```

```
(select count(*) from pub.student
```

```
where birthday-1=t1.birthday
```

```
) nextbirthday
```

```
from pub.student_testindex t1) where nextbirthday=7
```

3. 修改以后验证耗时在 1 秒之内，将修改以后语句创建成视图 create view test7_05 as select ……。
4. 交卷验证
5. 如果验证失败，提示耗时小于 2 秒，可以多次尝试提交阅卷。

实验八 提交 commit 和回滚 rollback、实体授权

一、实验内容

主账号登入主平台网址: <http://211.87.227.203:7088>,

临时账号登入备用平台网址: [http:// 211.87.227.202:7088](http://211.87.227.202:7088),

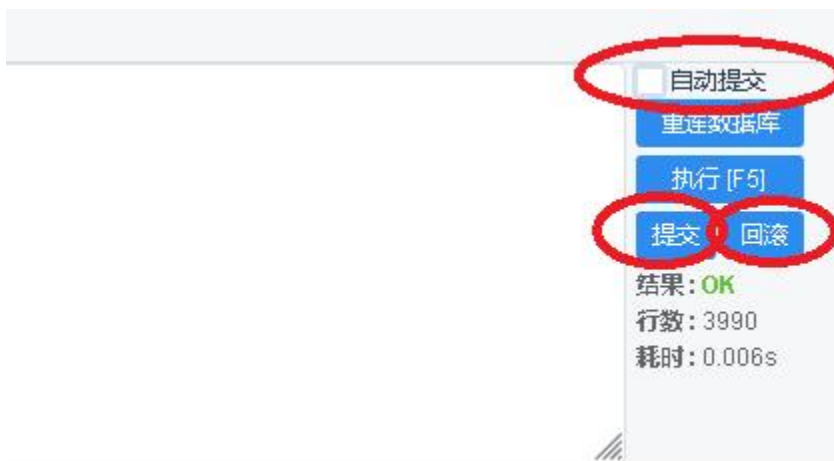
临时账号是主账号第五位换成 9, 例如 User200020002000 的临时账号为 user900020002000, 临时账号下面简写 user9Id。

测试提交 commit 和回滚 rollback 的作用, 了解锁等待、授权等知识。

二、实验步骤

1. 使用主账号 userID 登录数据库, 简称主窗口。

关闭自动提交复选框。按【提交】执行 commit, 按【回滚】执行 rollback。亦可以通过输入 commit 或者 ollback 按【执行】实现提交和回滚。



2. 使用临时账号 user9ID 登录数据库, 简称备用窗口。关闭自动提交复选框。

在临时账号下将 pub.teacher 复制创建表 test8_00, 然后将其所有权限给主账号。

3. 主账号访问临时账号的表之前, 需要在临时账号中将相应的表的相应的权限授权给主账号, 这样主账号才可以查询操作临时账号的相应的表。

在主账号下可以执行 `select * from user9Id.test8_00` 查询临时账号的表 test8_00 的数据, 如果没有授权, 则会提示表没有表找到。

如果临时账号执行 `grant select on test8_00 to userID`, 即授权表 test8_00 的 select 权限给用户 userID, 主账号的查询语句就可以正确执行, 并查询到相应的结果。

4. 常用的授权、命令:

grant select on test8_00 to userID 授权表 test8_00 的 select 权限给用户 userID。

grant update on test8_00 to userID 授权表 test8_00 的 update 权限给用户 userID。

grant insert on test8_00 to userID 授权表 test8_00 的 insert 权限给用户 userID。

grant delete on test8_00 to userID 授权表 test8_00 的 delete 权限给用户 userID。

grant all on test8_00 to userID 授权表 test8_00 的 all 权限给用户 userID。

revoke select on test8_00 from userID 收回表 test8_00 的 insert 权限从用户 userID。

5. 按表中序号在相应窗口执行对应的命令（主账号访问临时账号表需要授权）。

序号	窗口	题号	执行语句	结果
1	备用窗口		Update test8_00 set age=88	
2	备用窗口	结果 1	select * from test8_00	88
3	备用窗口		Commit	
4	备用窗口		Rollback	
5	备用窗口		Update test8_00 set age=age+1	
6	备用窗口		Rollback	
7	备用窗口		Commit	
9	备用窗口		Update test8_00 set age=age+2	
10	备用窗口		commit	
11	备用窗口	结果 2	select * from test8_00	
12	备用窗口		rollback	
13	主窗口	结果 3	select * from user9ID.test8_00	
14	备用窗口		Update test8_00 set age=age-2	
15	备用窗口		Update test8_00 set age=age-2	
16	备用窗口	结果 4	select * from test8_00	
17	主窗口	结果 5	select * from user9ID.test8_00	
18	主窗口		Commit	
19	主窗口	结果 6	select * from user9ID.test8_00	
20	主窗口		Rollback	
21	主窗口		Update user9ID.test8_00 set age=age-10 注意 锁等待现象，不是死机, 可以继续后面步骤	
22	备用窗口	结果 7	select * from test8_00	

23	备用窗口		Create table test8_01 as select * from test8_00	
24	备用窗口		Rollback 这里会出现回退无效的提示：自动提交功能设置为启用时无法回退，继续就可以。	
25	备用窗口	结果 8	select * from user9ID.test8_00	
26	主窗口	结果 9	select * from user9ID.test8_00	
27	主窗口		rollback	
28	主窗口	结果 10	select * from user9ID.test8_00	

6. 通过上面的操作以后，在每次查询的时候的年龄是多少？根据你的判断得出结果，然后按步骤进行实验验证，在 **主账号** 下创建一个表 test8_10(test varchar(20),age numeric (3)),插入 10 行数据，分表存放 10 个结果。

注意事项：插入数据以后，一定不要忘记执行 commit，因为开始已经关闭了自动提交选项。或者选中自动提交选项。

test	age
结果 1	88
结果 2	
结果 3	
结果 4	
结果 5	
结果 6	
结果 7	
结果 8	
结果 9	
结果 10	

7. “交卷验证”实验八。

实验九 条件数据插入

一、 实验内容

学会复制表结构、学会插入数据，特别是学会如何避免重复插入，也就是如何避免插入已经存在的数据。

二、 实验题目 1

1. 创建表 test9_01，表的结构同 pub.student_11_1 一样。
2. 为 test9_01 的 sid 创建唯一不重复索引。
3. 将 pub 用户下的 Student 中性别是“女”的数据添加到 test9_01 中。
4. 将 pub 用户下的 Student_11_1 中性别是“女”的数据添加到 test9_01 中, 如果某个学号已经包含在 test9_01 中, 这个记录就不要再插入了（即不要插入重复学号的数据）。
5. 将 pub 用户下的 Student_11_2 中性别是“女”的数据添加到 test9_01 中, 如果某个学号已经包含在 test9_01 中, 这个记录就不要再插入了（即不要插入重复学号的数据）。
6. 要求完成上述功能，请采用 1 条 create table、1 条 create index、3 条 insert 共 5 条 SQL 方式完成。

三、 实验题目 2

7. 创建表 test9_02，表的结构同 pub.student_11_1 一样。
8. 为 test9_02 的 sid 创建唯一不重复索引。
9. 将 pub 用户下的 Student 中性别是“女”的且 pub.student_course 中存在不及格成绩的同学添加到 test9_02 中。
10. 将 pub 用户下的 Student_11_1 中性别是“女”的且 pub.student_course 中存在不及格成绩的同学数据添加到 test9_02 中, 如果某个学号已经包含在 test9_02 中, 这个记录就不要再插入了（即不要插入重复学号的数据）。
11. 将 pub 用户下的 Student_11_2 中性别是“女”的且 pub.student_course 中存在不及格成绩的同学数据添加到 test9_02 中, 如果某个学号已经包含在 test9_02 中, 这个记录就不要再插入了（即不要插入重复学号的数据）。
12. 要求完成上述功能，请采用 1 条 create table、1 条 create index、3 条 insert 共 5 条 SQL 方式完成。

提示

1、实验二，题目 6：

思路 1：

```
Select cid, name, (select max ( ) from pub.student_course t1 where t1.xx=t0.xx ..... ) max_score,  
(select max() from pub.student_course t2.where t2.xx=t0.xx ..... and score| no in (select max ( ) from  
pub.student_course t1 where t1.xx=t0.xx ..... ) ) max_score2,  
(select count(.....) from from pub.student_course t1 where t1.xx=t0.xx ..... ) max_count  
From pub.course t0
```

思路 2：

With

```
tmp1 as (select cid max(course) max_score from pub.student_course group by..... ) ,--每门课最大成绩  
tmp2 as (select cid max(course) max_score2 from pub.student_course where 排除最大成绩可以使用 tmp1  
group by.....), --每门课第二成绩  
tmp3 as (select cid count(.....) max_count from pub.student_course where 可以使用 tmp1 group by.....) --每  
门课最大成绩人数
```

```
Select ..... from pub.couse t1,tmp1,tmp2,tmp3
```

Where

思路 3：

```
Select ..... from pub.couse t1,  
(select cid max(course) max_score from pub.student_course group by..... ) t2,--每门课最大成绩  
(select cid max(course) max_score2 from pub.student_course where 排除最大成绩 group by.....) t3, --每门  
课第二成绩  
(select cid count(.....) max_count from pub.student_course group by.....) t3, --每门课最大成绩人数  
Where
```

说明：思路 1 和思路 3 对比，思路 1 更加简单，一个子查询就是解决一个值统计。效率会低一些。

2、实验三，题目 5：

找出不规范的班级的办法如下，执行一下就明白了：

```
Select class, count(*) from pub.student_31 group by class
```

3、实验三，题目 9：

思路：

```
Select distinct score from pub.student_course_32 order by score
```

将来需要补充题

附加题：设计

- 1、创建视图 test9_03，从表 pub.course_10 用递归查找“英语 355”这门课的所有先行课、先行课的先行课、先行课的先行课的先行课……

```
SELECT *  
FROM pub.COURSE_10  
START WITH NAME =' 英语 355'  
CONNECT BY PRIOR fCID=cid
```

- 2、创建视图 test9_04，从表 pub.course_10 用递归查找“语文 1”这门课的所有下行课、下行课的先行课、下行课的先行课的下行课……

```
SELECT *  
FROM pub.COURSE_10  
START WITH NAME =' 语文 1'  
CONNECT BY PRIOR CID=fcid
```

- 3、创建视图 test9_04，从表 pub.course_10、pub.student_course_10 递归查找选修了下行课，却没有选修先行课的那些学生的选课信息。

```
select * from pub.STUDENT_COURSE_10 t0  
  where exists(select cid from pub.COURSE_10  
  where cid not in (select cid from pub.STUDENT_COURSE_10 where t0.sid=sid)  
  and cid in (select cid from pub.COURSE_10  
  START WITH cid=t0.cid  
  CONNECT BY PRIOR fCID=cid))  
  order by sid,cid
```

- 4、将 pub.student_course_10 复制创建表 test9_05，利用表 pub.course_10、pub.student_course_10，通过递归实现选修了下行课，却没有选修先行课，插入的表 test9_05 中。

```
select t1.sid,t0.cid from pub.COURSE_10 t0, pub.student_course_10 t1  
start with t0.cid=t1.cid  
connect by prior t0.fcid=t0.cid  
minus select sid,cid from pub.student_course_10;
```

- 5、语文 12 的所有先行课，后续课

```
select * from pub.COURSE_10  
START WITH NAME =' 语文 12'  
CONNECT BY PRIOR fcid=cid  
union  
select * from pub.COURSE_10  
START WITH NAME =' 语文 12'  
CONNECT BY fcid= PRIOR cid
```

- 6、语文 12 的所有亲戚课程

```
select * from pub.COURSE_10  
START WITH cid in (  
select cid from pub.COURSE_10  
START WITH NAME =' 语文 12'
```

```
CONNECT BY prior fcid= cid
)
CONNECT BY PRIOR cid=fcid
order by cid
```

1. 找出选修了“操作系统”并且获得学分并且也选修了“数据结构”且没有获得学分，但是没有选修“程序设计语言”的所有女学生的学号、姓名。

```
select sid,name from pub.student t1 where sex='女' and sid in(
select sid from pub.student_course where score>59 and cid in (select cid from pub.course where
name='操作系统'))
and sid in(
select sid from pub.student_course where cid in (select cid from pub.course where name='
数据结构'))
and sid not in(
select sid from pub.student_course where score>59 and cid in (select cid from pub.course where
name='数据结构'))
and sid not in(
select sid from pub.student_course where cid in (select cid from pub.course where name='
程序设计语言'))
```