 **(北京)**

# CHINA UNIVERSITY OF PETROLEUM

数据库原理上机报告

实验三 数据库的查询和数据库的安全性及完整性

院系名称： **信息学院**

专业名称： **计算机\_**

学生学号： 2019011777

学生姓名： 刘康来

完成日期 2021 年 11 月 3 日

**《数据库原理》上机报告**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **报告名称** | ***数据库的查询和数据库的安全性及完整性*** | | | **粘贴**  **照片** |
| **姓 名** | 刘康来 | **班 级** | 计算机19-3 |
| **成 绩** |  | | |
| **1．上机实践目的与准备知识（简介，300字以内）**  1.实验目的  （1）掌握复杂查询的表示；  （2）掌握GROUP BY、HAVING子句的作用和使用方法；  （3）掌握创建用户和对用户授权操作。  （4）掌握实体完整性、参照完整性、自定义完整性的设置  （5）掌握触发器的定义，初步了解T-SQL的变量定义、数据类型等  2. 实验准备  （1）了解子查询的表示方法；  （2）了解连接查询的表示方法；  （3）了解SELECT语句GROUP BY、HAVING子句的作用和使用方法；  （4）了解创建用户和对用户授权操作。  （5）了解完整性的定义。  **（6）了解触发器的定义。** | | | | |
| **2．主要实践内容与具体操作步骤（实践内容完成情况要有描述，如执行的SQL命令等，有运行结果截图，图大小以保证文字清晰为准）**  （1）查询男性雇员和女性雇员的人数；  select case  when sex = 1 then concat(cast(count(Name) as varchar(10)), '(男性)')  when sex = 0 then concat(cast(count(Name) as varchar(10)), '(女性)') end as '人数' from Employees group by sex;  -- scalar select distinct case  when Sex = 1 then '男性'  when Sex = 0 then '女性' end as sex, (select count(\*)  from Employees as y -- from Employees natural join Departments; wrong!  where x.Sex = y.sex) as number from Employees as x;      （2）查询员工人数超过2人的部门名称和员工的人数；  select DepartmentName, count(Name) as '员工人数'  from Employees natural join Departments group by DepartmentName having count(Name) > 2;    （3）查询研发部员工的姓名和出生日期（只要年月日），要求各列标题用中文表示；  select Name as '姓名', Birthday as '出生日期' from Employees natural join Departments where DepartmentName = '研发部';    （4）根据员工号定义用户，完成以下任务：  请用SQL的 GRANT和 REVOKE语句(加上视图机制)完成以下授权定义或存取控制功能:  (4.1)用户王明对三个表有 SELECT权限、INSERT和 DELETE权限。  (4.2) 用户刘星对Employees表有 SELECT权限,对Address字段具有更新权限。  (4.3) 用户张新具有修改这三个表的结构的权限。  (4.4) 用户杨兰具有从每个部门员工中 SELECT最高工资、最低工资、平均工资的权限,她不能查看每个人的工资。  create user '王明'; grant select, insert, delete on YGGL.\* to '王明'; create user '刘星'; grant select on Employees to '刘星'; grant update on Employees.Address to '刘星'; grant update(Address) on Employees to '刘星'; create user '张新'; grant select, insert, delete, update on YGGL.\* to '张新'; -- grant alter table on YGGL.\* to '张新'; create user '杨兰'; create view Em\_come as  select DepartmentName, max(Income), min(Income), avg(Income)  from Employees natural join Departments natural join Salary  group by DepartmentName; grant select on Em\_come to '杨兰';  (5)在数据库YGGL中创建表ES\_U（Eno,OldIncome,NewIncome）：   * 主码命名为pk1，主码为（eno）； * 定义约束条件ck1，NewIncome的值不能为空； * 定义约束条件ck2，要求OldIncome和NewIncome值范围都是[2000,20000]。   create table ES\_U(  Eno varchar(10) primary key,  OldIncome int check (OldIncome between 2000 and 20000),  NewIncome int not null check (NewIncome between 2000 and 20000) );  (6)修改employees表，性别的数据类型改为char（2），取值只能是‘男’、‘女’、‘F’、‘M’这四个值  alter table Employees modify Sex char(2);  -- change the values at first. update Employees set Sex = case  when Sex = '1' then 'M'  else 'F'  end;   alter table Employees  add check (Sex = '男' or Sex = '女' or Sex = 'F' or Sex = 'M');  (7)设计多个违反上述完整性的数据，分别用insert向ES\_U、employees表插入。  insert into ES\_U values('a', 0, 0); insert into Employees values('100', 'a', '小学', 20000101, 0, 1, '0', 0, 0);    （8）在Salary表上定义触发器，当更新数据后，如果Income值变大，就在ES\_U表中插入一条记录。设计插入一条合法数据，触发触发器。  create trigger raise\_salary before update on ES\_U  for each row  begin  if NEW.NewIncome > OLD.NewIncome then  set NEW.OldIncome = OLD.NewIncome;  end if;  end;  insert into ES\_U values ('a', null, 5000);  update ES\_U set NewIncome = '6000' where Eno = 'a';    （9）创建表employees\_log，只有员工人数一个属性，向表中插入值为employees表的总人数。在employees表上定义触发器：当插入或删除一个员工时，就在employees\_log表中修改员工人数。插入一个员工记录进行测试。  insert into Employees\_log value((  select count(\*)  from Employees  ));  -- Unsafe query: 'Update' statement without 'where' updates all table rows at once create trigger add\_num\_employees after delete on Employees  for each row  begin  update Employees\_log  set Num\_employees = (select count(\*) from Employees)  where 1;  end;  create trigger minus\_num\_employees after insert on Employees  for each row  begin  update Employees\_log  set Num\_employees = (select count(\*) from Employees)  where 1;  end;  delete from Employees where EmployeeID = '1'; insert into Employees values('1', 'b', '高中', 20000201, 'M', 2, '1', 0, 0);  select \* from Employees\_log;      （10）自行设计一个针对YGGL某个表的触发器。  -- function must have return...create procedure *abort*() begin rollback; end;create trigger check\_department before delete on Departments for each row begin if OLD.DepartmentID in (select DepartmentID from Employees) then -- how to terminate the operation? -- signal sqlstate '45000'; -- 45000 or 45001 ? call *abort*(); end if; end;delete from Departmentswhere DepartmentID = 0; | | | | |
| **3．总结与问题分析（100字以上）**  遇到的问题：  如何删除约束，查看约束？  alter table Employees add check (Sex = '男' or Sex = '女' or Sex = 'F' or Sex = 'M');  alter table Employees add constraint name\_sex check (Sex = '男' or Sex = '女' or Sex = 'F' or Sex = 'M');  一般是命名约束，再删除，可还是报错，用了以下方法解决：  alter table Employees drop constraint CONSTRAINT\_1;  如何在触发器中取消操作？  在sql 标准中，可用 rollback 解决问题，但mariadb 数据库报了安全问题。  1. signal sqlstate '45000'; -- 45000 or 45001 ? 这句命令解决。。。  2. 写了一个procedure，里面可以用roolback。尝试去写 function，发现必须要有返回值，其实忽略即可。  总结：易用性固然重要，但更重要的是安全，完整性；遇到问题，还是要 Google 或查使用手册，谁知道设计者怎么想的。 | | | | |