华中科技大学计算机学院

数据库课程实验操作指导

徐丽萍 殷贤亮 卢炎生 编写

数据库系统课程教学组

二０一四年 三月

目录

1．DM 数据库的安装

此部分见安装文件自带的 DM_Install_zh.pdf 文件，十分详细。

2．DDL 使用方法

2.1.数据库创建

创建一个模式实际上定义了一个命名空间，在这个空间中可以进一步定义该模式包含的数据库对象，例如基本表、视图、索引等。

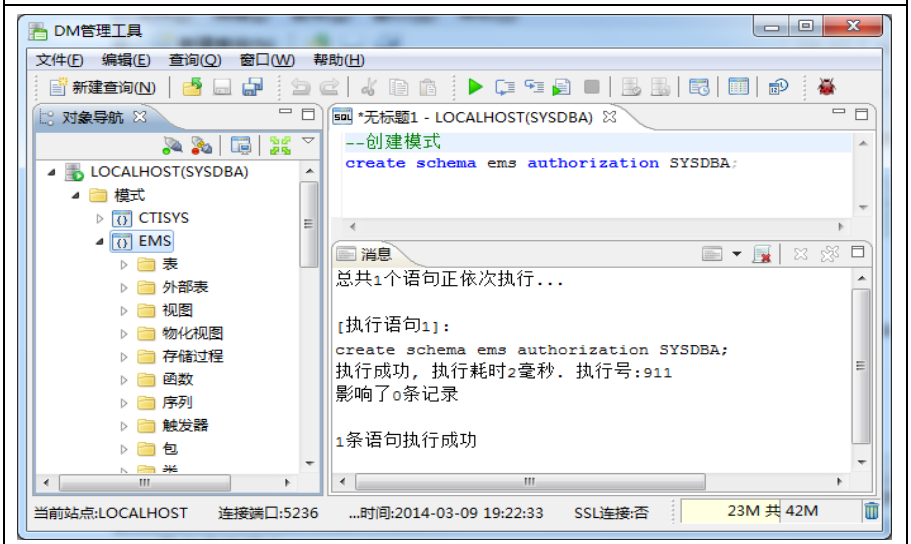
定义模式：CREATE SCHEMA <模式名> AUTHORIZATION <用户名>

例 1：创建名为 ems 的模式：create schema ems authorization SYSDBA;

这是一个简单的人事管理数据库。本例中的所用数据库对象均为 ems 建立。

图 2.1.1. 创建名为 ems 的模式

```
create schema ems authorization SYSDBA;
```



2.2.基本表的创建

创建基本表的命令为：CREATE TABLE table_name，在该命令中定义主码和外码时，可以使用列约束（Column Constraint）或表约束（Table Constraint）子句。

例 2：在 ems 中创建 employee（职员）表和 dept（部门）表：

employee（eno，ename，manager，salary，deptno）；

主码：eno ，外码：manager、deptno

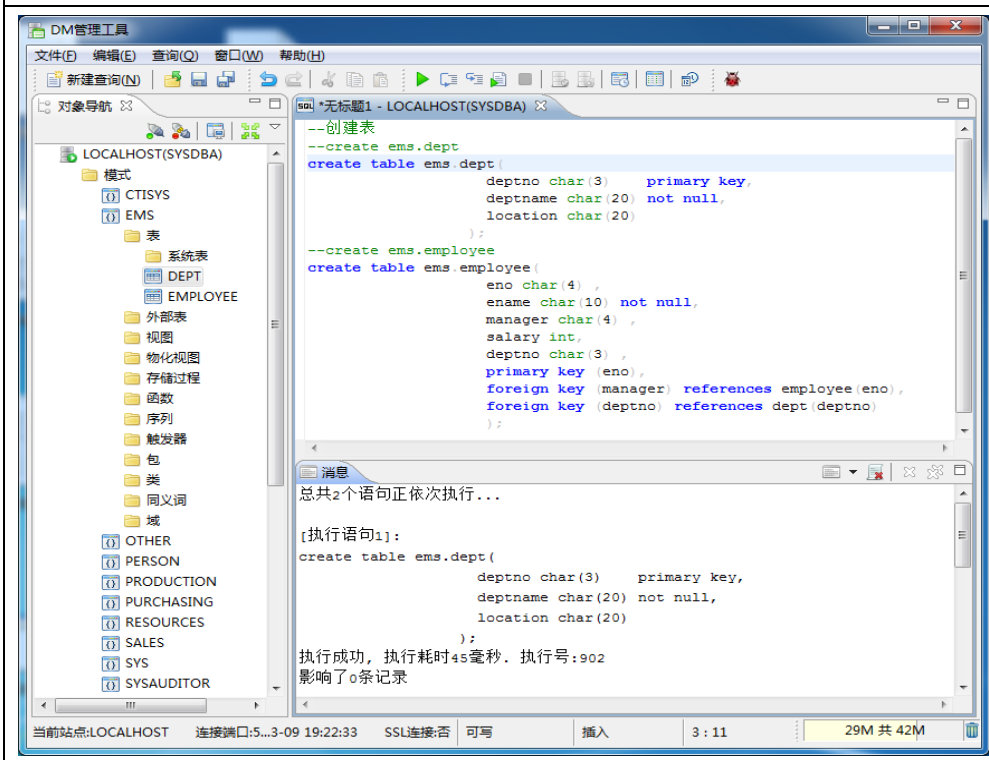
dept（deptno，dname，location）主码：deptno

注：employee 中的 manager 为相应雇员的直接领导或上司的 eno。

图 2.2.1 为使用 SQL 建表命令创建上述基本表的源代码。在查询分析器中输入该代码，点击执行按钮即可。

图2.2.1 SQL建表命令创建上述基本表的源代码

```
create table ems.dept(
                    deptno char(3)    primary key,
                    deptname char(20) not null,
                    location char(20)
                  );
create table ems.employee(
                    eno char(4) ,
                    ename char(10) not null,
                    manager char(4) ,
                    salary int,
                    deptno char(3) ,
                    primary key (eno),
                    foreign key (manager) references employee(eno),
                    foreign key (deptno) references dept(deptno)
                    );
```



2.3.视图的创建/删除

视图是组成数据库体系结构——三级模式两级映像结构中的外模式的基本单元，

SQL-Server 的视图定义命令为：

CREATE [OR REPALCE] VIEW [<模式名>.]<视图名>[(<列名> {,<列名>})] AS <查询说明> [WITH CHECK OPTION]|[WITH READ ONLY]; <查询说明>::=<表查询> | <表连接> <表查询>::=<子查询表达式>[ORDER BY 子句]	DROP VIEW [<模式名>.]<视图名> [RESTRICT | CASCADE];

视图是用于定义终端用户数据来源的。在视图定义中可以使用复杂的 SELECT 命令。

例 3 :在 ems 中定义能够查询雇员年薪的视图 Annualsal 和统计雇员下属人数的视图 manager。

视图定义为：

annualsal（eno，ename，annualsal）；annualsal 为相应雇员的年薪。

manager（manager，name，clerknum）；clerknum 为相应雇员的直接下属个数。

在查询分析器中分别输入如图 2.3.1 所示的代码，即可创建要求的视图。

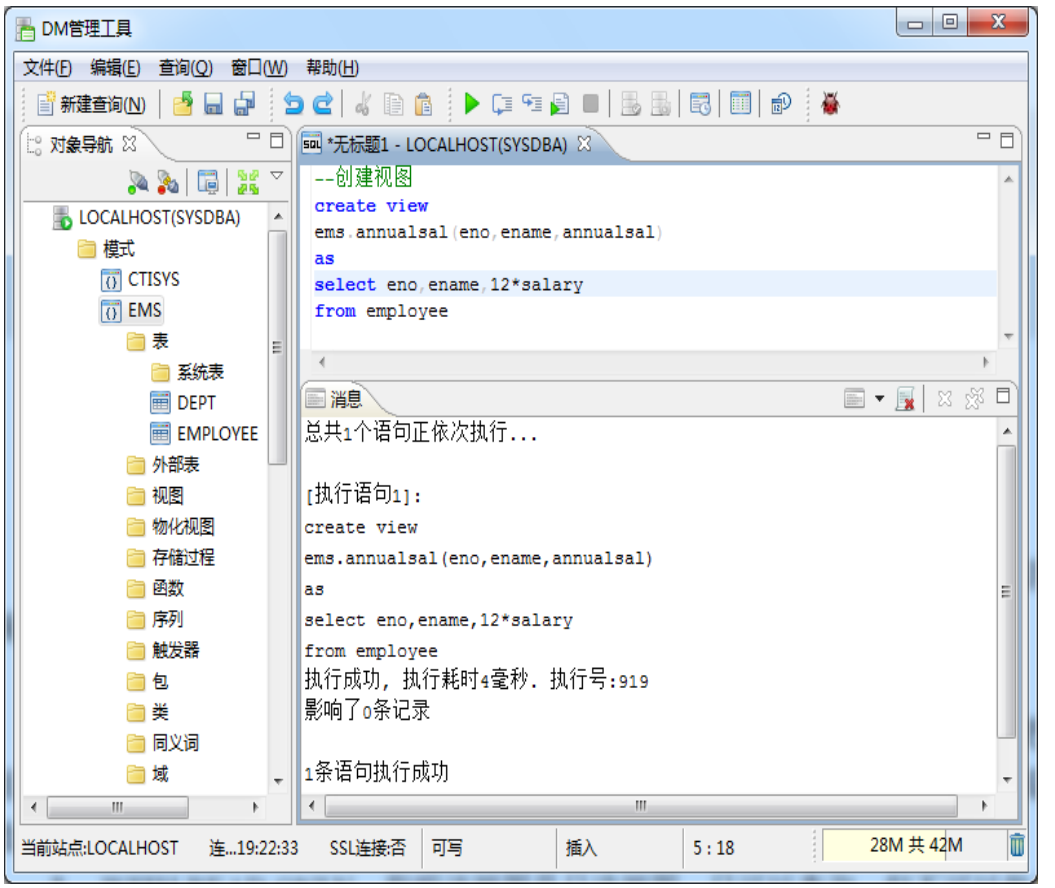
图2.3.1 SQL建表命令创建上述基本表的源代码

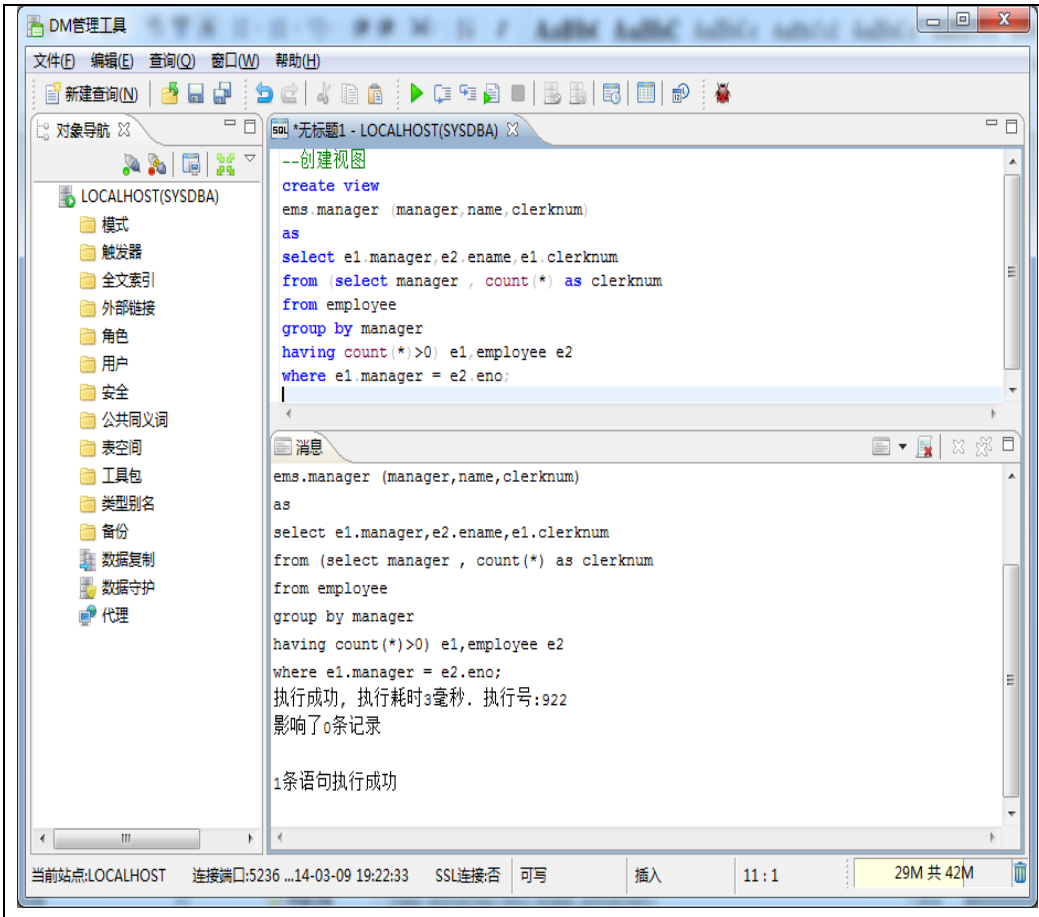
```
-- 创建annualsal视图
create view
ems.annualsal(eno,ename,annualsal)
as
select eno,ename,12*salary
from employee;
```

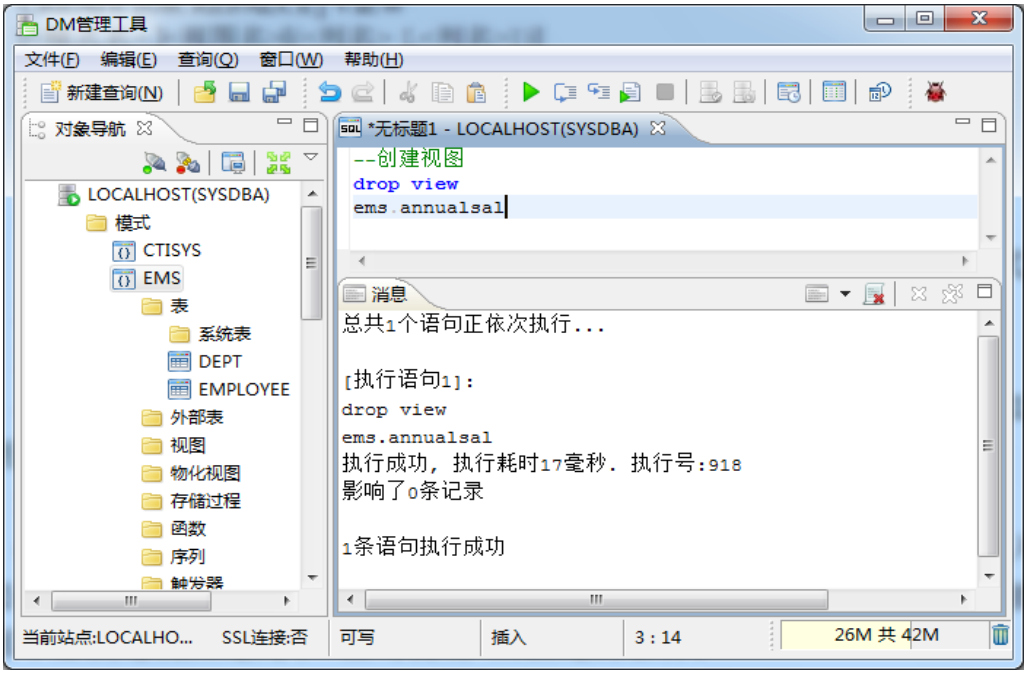
```
-- 删除annualsal视图
create view
ems.annualsal
```

```
--创建manager视图
create view
ems.manager (manager,name,clerknum)
as
select e1.manager,e2.ename,e1.clerknum
from (select manager , count(*)
as clerknum
from employee
group by manager
```

```
having count(*)>0) e1,employee e2
where e1.manager = e2.eno;
```




DM管理工具
文件(F) 编辑(E) 查询(Q) 窗口(W) 帮助(H)
新建查询(N)
对象导航
*无标题1 - LOCALHOST(SYSDBA)
--创建视图
create view
ems.manager (manager,name,clerknum)
as
select e1.manager,e2.ename,e1.clerknum
from (select manager , count(*) as clerknum
from employee
group by manager
having count(*)>0) e1,employee e2
where e1.manager = e2.eno;
LOCALHOST(SYSDBA)
模式
触发器
全文索引
外部链接
角色
用户
安全
公共同义词
表空间
工具包
类型别名
备份
数据复制
数据守护
代理
消息
ems.manager (manager,name,clerknum)
as
select e1.manager,e2.ename,e1.clerknum
from (select manager , count(*) as clerknum
from employee
group by manager
having count(*)>0) e1,employee e2
where e1.manager = e2.eno;
执行成功, 执行耗时3毫秒. 执行号:922
影响了0条记录
1条语句执行成功
当前站点:LOCALHOST
连接端口:5236 ...14-03-09 19:22:33
SSL连接:否
可写
插入
11 : 1
29M 共 42M


DM管理工具
文件(F) 编辑(E) 查询(Q) 窗口(W) 帮助(H)
新建查询(N)
对象导航
*无标题1 - LOCALHOST(SYSDBA)
--创建视图
drop view
ems.annualsal
LOCALHOST(SYSDBA)
模式
CTISYS
EMS
表
系统表
DEPT
EMPLOYEE
外部表
视图
物化视图
存储过程
函数
序列
触发器
消息
总共1个语句正依次执行...
[执行语句1]:
drop view
ems.annualsal
执行成功, 执行耗时17毫秒. 执行号:918
影响了0条记录
1条语句执行成功
当前站点:LOCALHO...
SSL连接:否
可写
插入
3 : 14
26M 共 42M

3．DML 使用方法

SQL 的 DML 包括插入（INSERT）、删除(DELETE)、修改(UPDATE)等命令。DML 命令的执行是可能造成数据库不一致的根源。因此，每一条语句在执行前，SQL-Server 都要验证语句是否符合完整性要求，包括实体完整性、参照完整性、用户定义完整性。

3.1.INSERT 命令

SQL 语言的插入命令：

```
INSERT [INTO] <表引用> [(<列名>{,<列名>})]
VALUES(<插入值>{,<插入值>});|(<插入值>{,<插入值>}){, (<插入值>{,<插入值>})};|
<查询说明>[<ORDER BY 表达式>];|
DEFAULT VALUES [RETURN <列名>{,<列名>} INTO
<结果对象>,{<结果对象>}]; <结果对象>::<数组>|<变量>
<表引用>::= [<模式名>.]<基表或视图名>
<基表或视图名>::= <基表名>|<视图名>
```

例 4：在 ems 中的 dept 表中表 3.1.1 中的数据；employee 表中表 3.1.2 中的数据。

表 3.1.1 基本表 dept 的数据

D01 Computer School	South1-405
D02 Communication Dept	South1-304
D03 Management School	kejilou-408

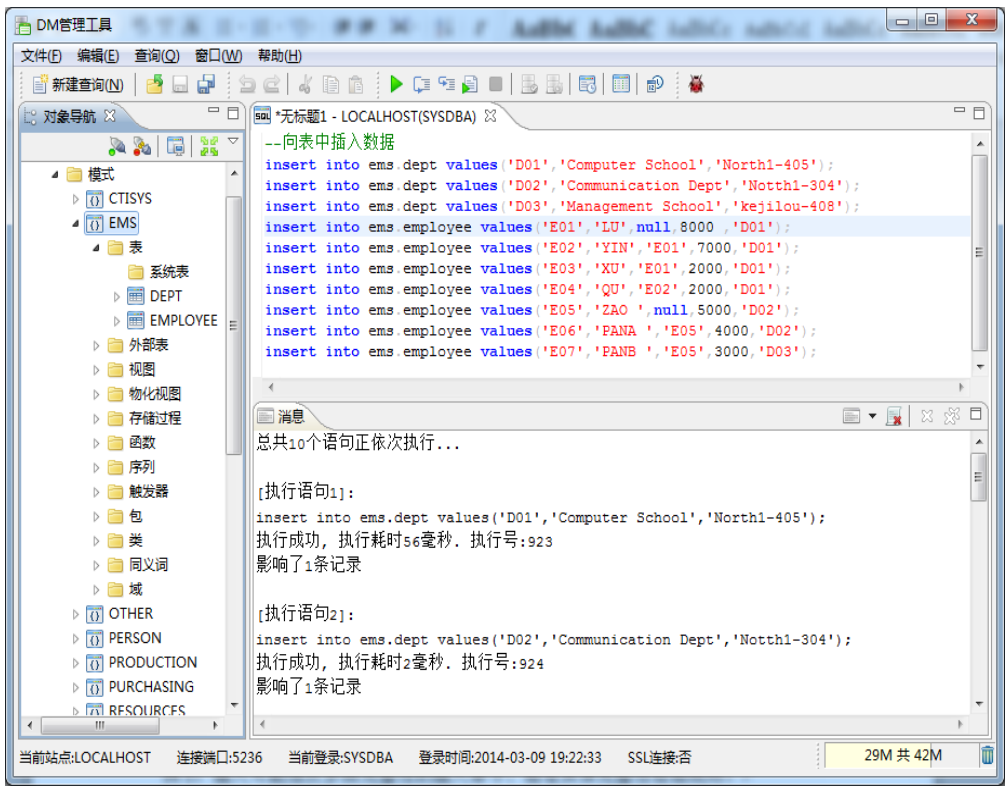
表 3.1.2 基本表 employee 的数据

E01	LU	null	8000	D01
E02	YIN	E01	7000	D01
E03	XU	E01	2000	D01
E04	QU	E02	2000	D01
E05	ZAO	null	5000	D02
E06	PANA	E05	4000	D02
E07	PANB	E05	3000	D03

图3.1.1 向基本表中插入数据的源代码

```
--向表中插入数据
insert into ems.dept values('D01','Computer School','North1-405');
insert into ems.dept values('D02','Communication Dept','Notth1-304');
insert into ems.dept values('D03','Management School','kejilou-408');
```

```
insert into ems.employee values('E01','LU',null,8000 ,'D01');
insert into ems.employee values('E02','YIN','E01',7000,'D01');
insert into ems.employee values('E03','XU','E01',2000,'D01');
insert into ems.employee values('E04','QU','E02',2000,'D01');
insert into ems.employee values('E05','ZAO ',null,5000,'D02');
insert into ems.employee values('E06','PANA ','E05',4000,'D02');
insert into ems.employee values('E07','PANB ','E05',3000,'D03');
```



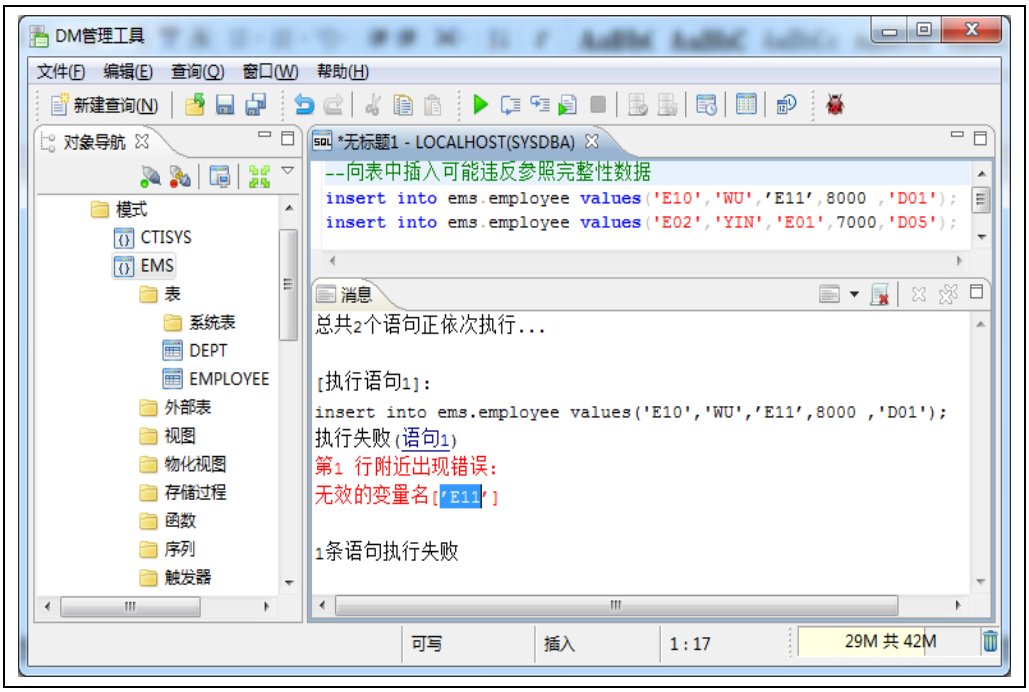
注意：在执行过程中，如果有一条语句出错误，再次执行时，在出错语句前面的语句就会被拒绝执行。因为，在查询分析器中的命令在执行时按顺序逐条执行，当执行到出错语句时停止执行，此时，在出错语句的前面各条语句的结果已经存入表中，故当重复执行时，它们违反了实体完整性。

例 5：插入可能违反参照完整性的插入命令，验证实体完整性检验规则。

在查询分析器中执行图 3.1.2 所示的命令，系统将拒绝执行。

图 3.1.2 插入可能违反参照完整性的插入命令

```
--向表中插入可能违反参照完整性数据
insert into ems.employee values('E10','WU',' E11' ,8000 ,'D01');
insert into ems.employee values('E02','YIN','E01',7000,'D05');
```



3.2.DELETE 命令

SQL 的删除命令为：

```
DELETE
[FROM] <表引用>
[WHERE <条件表达式>][RETURN <列名>{,<列名>} INTO <结果对象>,{<结果对象>}];
<表引用>::= [<模式名>.]<基表或视图名> <基表或视图名>::=
<基表名>|<视图名>
<结果对象>::=<数组>|<变量>
```

执行 DELETE 命令后，系统会删除满足命令中条件表达式的所有元组。这种删除只是逻辑的。因此，当再次插入一个与被删除的元组具有相同关键字的元组时，被认为违反了实体完整性。

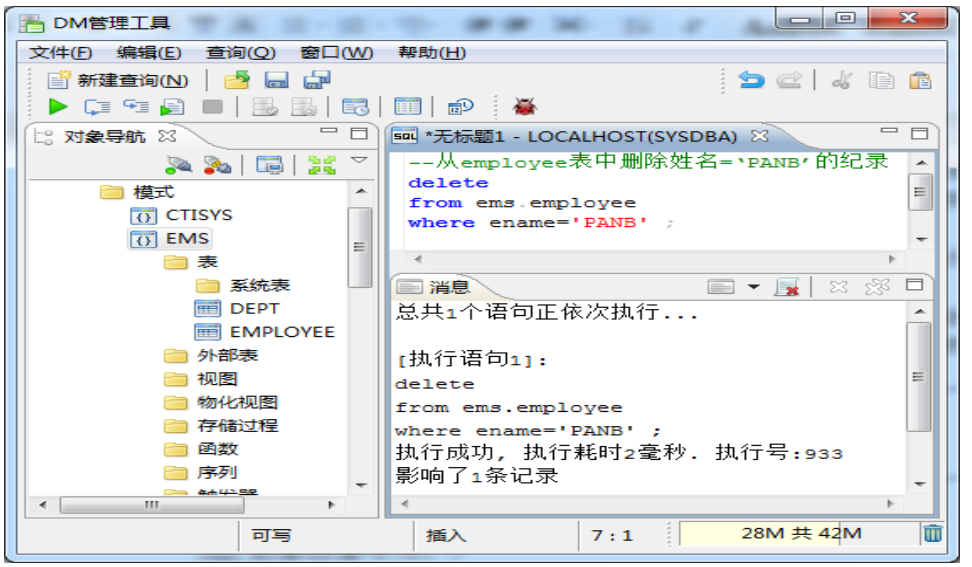
例 6：删除姓名= ‘PANB’ 的纪录。在查询分析器中输入并执行如图 3.2.1 所示的代码即可完成要求的删除操作。

删除操作可能会引起参照完整性的破坏，对于这些操作系统会根据完整性定义执行或拒绝执行。例如：在没有删除 PANB 纪录前，试图删除他所在的部门的操作就会引起系统的警告，并拒绝执行该操作。

图3.2.1 删除姓名= ‘PANB’ 的纪录

```
--从employee表中删除姓名= ‘PANB’ 的纪录
delete
```

```
from ems.employee
where ename='PANB' ;
```



3.3.UPDATE 命令

SQL 的修改命令为：

```
UPDATE <表引用>|<子查询> {<单列修改子句>|<多列修改子句>}
<子查询>::=[<common_table_2>]
<common_table_2>::= (<不带 INTO 查询表达式>) [[AS] <表别名> [<新生列>]]
<单列修改子句>::= SET<列名>=<<值表达式>|DEFAULT>{,<列名>=<<值表达式>|DEFAULT>}[FROM
    <表引用>{,<表引用>}][WHERE <条件表达式>][RETURN <列名>{,<列名>} INTO <结果对象>];
<多列修改子句>::= SET <列名>{,<列名>}= <subquery>;
<表引用>::= [ <模式名>.]<基表或视图名>
<基表或视图名>::= <基表名>|<视图名>
<结果对象>::= <数组>|<变量>
```

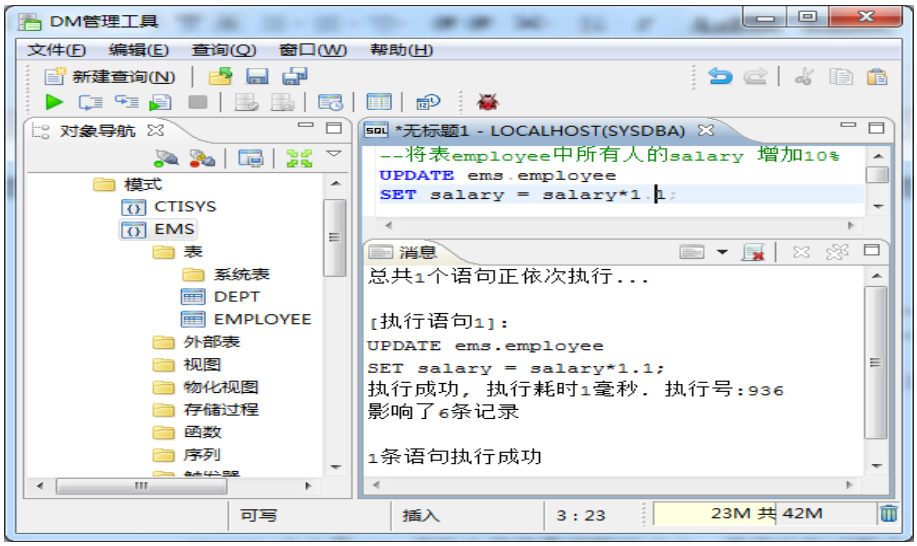
执行 UPDATE 命令后，系统会按照 SET 子句修改满足命令中条件表达式的所有元组。如果使用修改命令更新关键字的值，而该关键字在其他表中作为外码存在时，操作可能违反参照完整性。系统将拒绝执行。

例 7：将表 employee 中所有人的 salary 增加 10%。该操作如图图 3.3.1 所示：

图3.3.1 将表employee中所有人的salary 增加10%。

```
--将表employee中所有人的salary 增加10%
UPDATE ems.employee
```

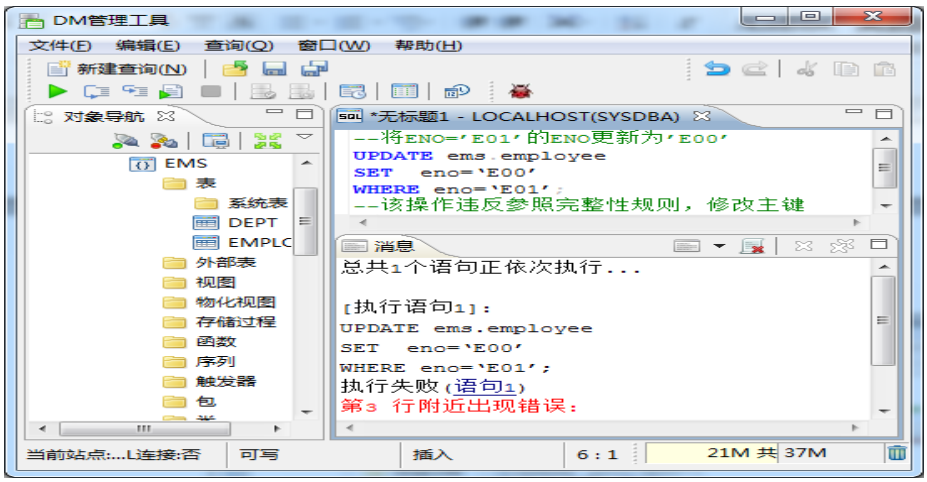
```
SET salary = salary*1.1;
```



例 8：将 ENO=’ E01’的 ENO 更新为’ E00’ 。该操作如图图 3.3.2 所示：

图3.3.2 将ENO=’ E01’的ENO更新为’ E00’

```
--将ENO=’ E01’的ENO更新为’ E00’
UPDATE ems.employee
SET eno= ‘E00’
WHERE eno= ‘E01’ ;
--该操作违反参照完整性规则，修改主键
```



4．SELECT 命令

SELECT 命令是 SQL 语言中使用最频繁的命令 ,其变化形式繁多 ,灵活运用 SELECT 命令可以完成任意复杂的查询要求。SELECT 命令的基本语法为:

```
SELECT <选择列表>
FROM [<模式名>.]<基表名>｜<视图名> [<相关名>]
[<WHERE 子句>]
[<CONNECT BY 子句>]
[<GROUP BY 子句>]
[<HAVING 子句>]
[ORDER BY 子句];
```

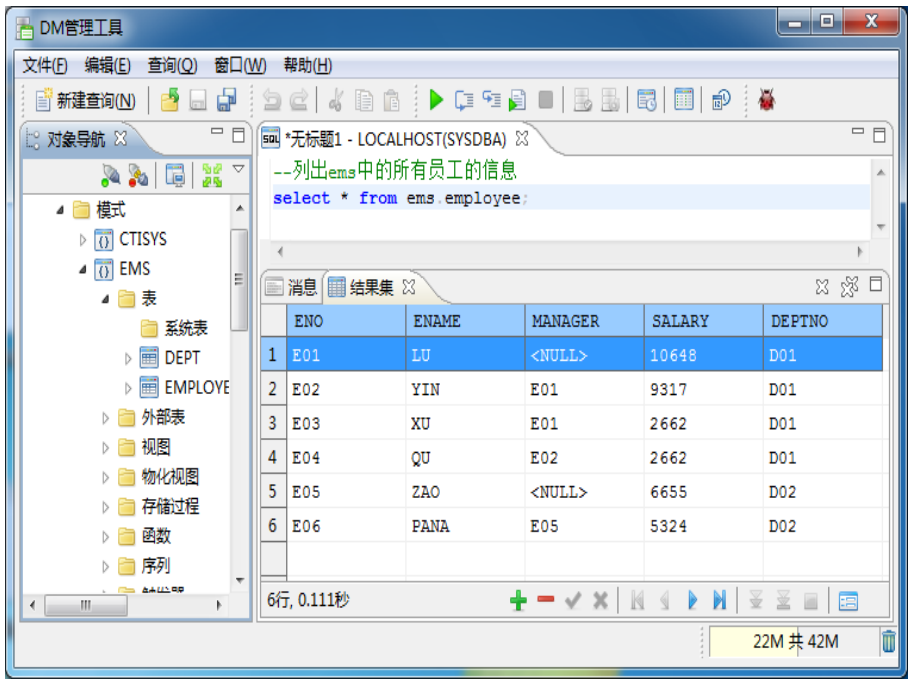
可选项 WHERE 子句用于设置对于行的检索条件。不在规定范围内的任何行都从结果表中去除。GROUP BY 子句逻辑地将由 WHERE 子句返回的临时结果重新编组。结果是行的集合，一组内一个分组列的所有值都是相同的。HAVING 子句用于为组设置检索条件。由于 SELECT 命令非常复杂,上面的语法描述还无法完全表达，请参考用户手册中的 DM_SQL.pdf 。下面只举几个典型例子。

4.1.简单查询

例 9：列出 ems 中的所有员工的信息。

图4.1.1 列出ems中的所有员工的信息。

```
--列出ems中的所有员工的信息。
select * from ems.employee;
```



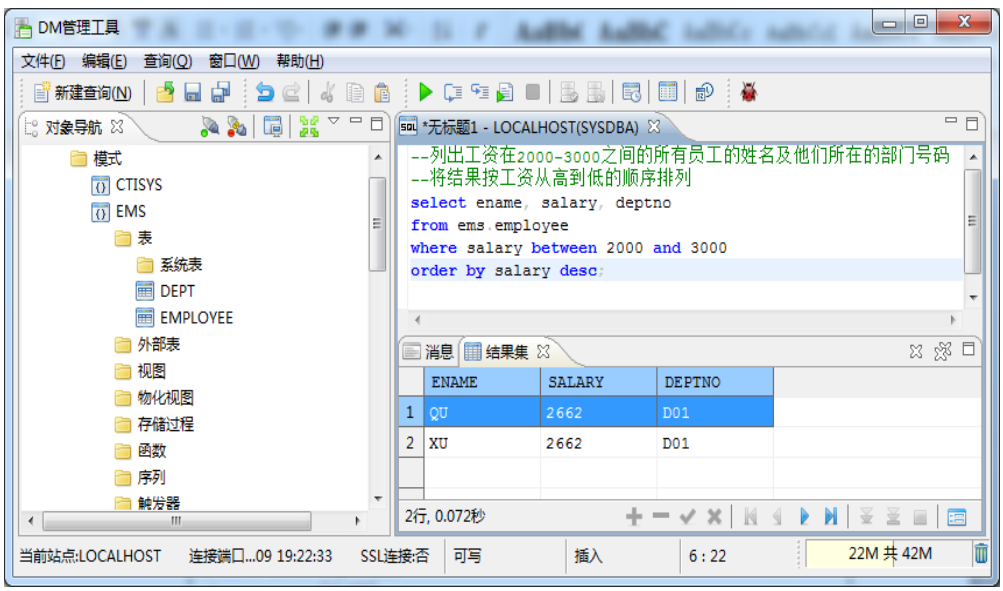
4.2.使用谓词的查询

谓词有 IN、 LIKE、 NULL、 EXISTS、 BETWEEN 等，在查询时 WHERE 子句中可以使用。

例 10：列出工资在 2000-3000 之间的所有员工的姓名及他们所在的部门号码,将结果按工资从高到低的顺序排列。在查询分析器中输入并执行图 4.2.1 中的代码即可得到所要的结果。

图4.2.1 列出工资在2000-3000之间的所有员工的姓名及他们所在的部门号码,将结果按工资从高到低的顺序排列。

```
--列出工资在2000-3000之间的所有员工的姓名及他们所在的部门号码
--将结果按工资从高到低的顺序排列
select ename, salary, deptno
from ems.employee
where salary between 2000 and 3000
order by salary desc;
```



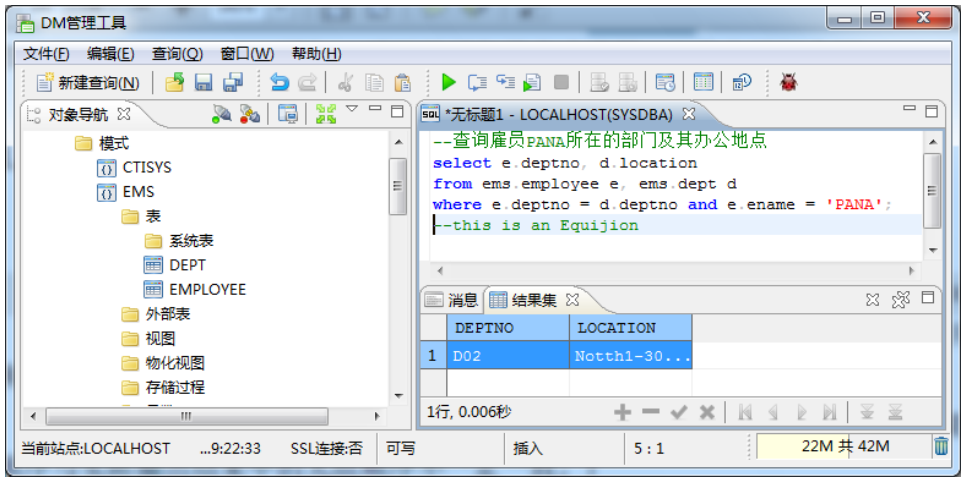
4.3.连接查询

连接查询通过表间的关联字段实现相关查询，连接查询分为等值联接查询、不等值连接查询、外联接查询、自我连接查询等。在书写较为复杂的查询命令时，建议使用别名来提高效率。

例 11：查询雇员 PANA 所在的部门及其办公地点；

图 4.3.1 查询雇员 PANA 所在的部门及其办公地点；

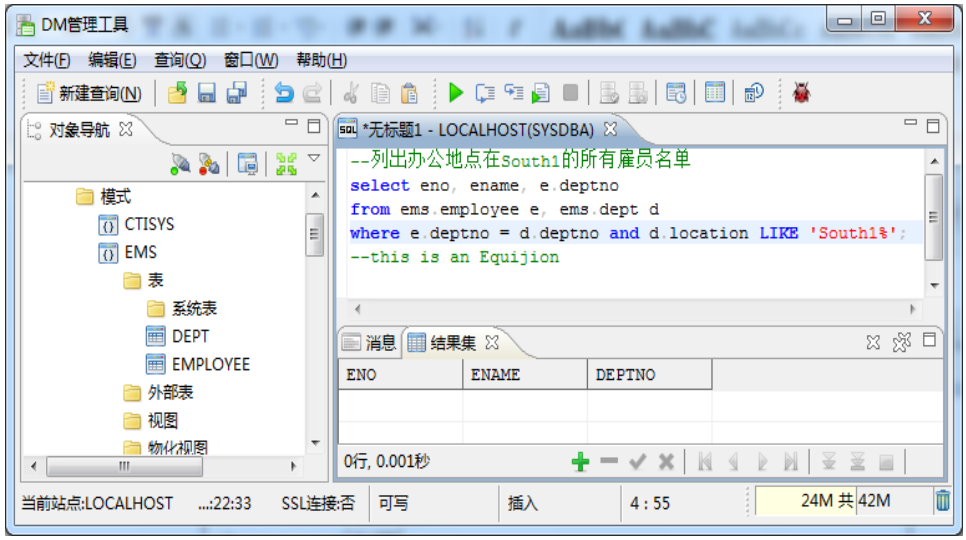
```
--查询雇员PANA所在的部门及其办公地点
select e.deptno, d.location
from ems.employee e, ems.dept d
where e.deptno = d.deptno and e.ename = 'PANA';
--this is an Equijion
```



例 12：列出办公地点在 South1 的所有雇员名单；

图 4.3.2 列出办公地点在 South1 的所有雇员名单

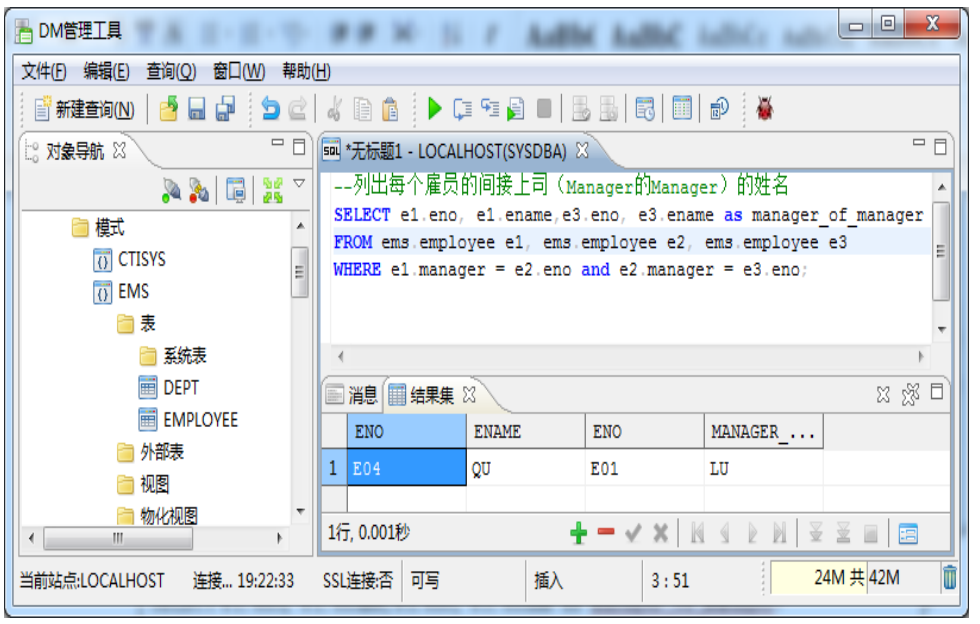
```
--列出办公地点在South1的所有雇员名单
select eno, ename, e.deptno
from ems.employee e, ems.dept d
where e.deptno = d.deptno and d.location LIKE 'South1%';
--this is an Equijion
```



例 13：列出每个雇员的间接上司（Manager 的 Manager）的姓名。

图4.3.3 列出每个雇员的间接上司（Manager的Manager）的姓名。

```
--列出每个雇员的间接上司（Manager的Manager）的姓名
SELECT e1.eno, e1.ename,e3.eno, e3.ename
as manager_of_manager
FROM ems.employee e1, ems.employee e2, ems.employee e3
WHERE e1.manager = e2.eno and e2.manager = e3.eno;
```



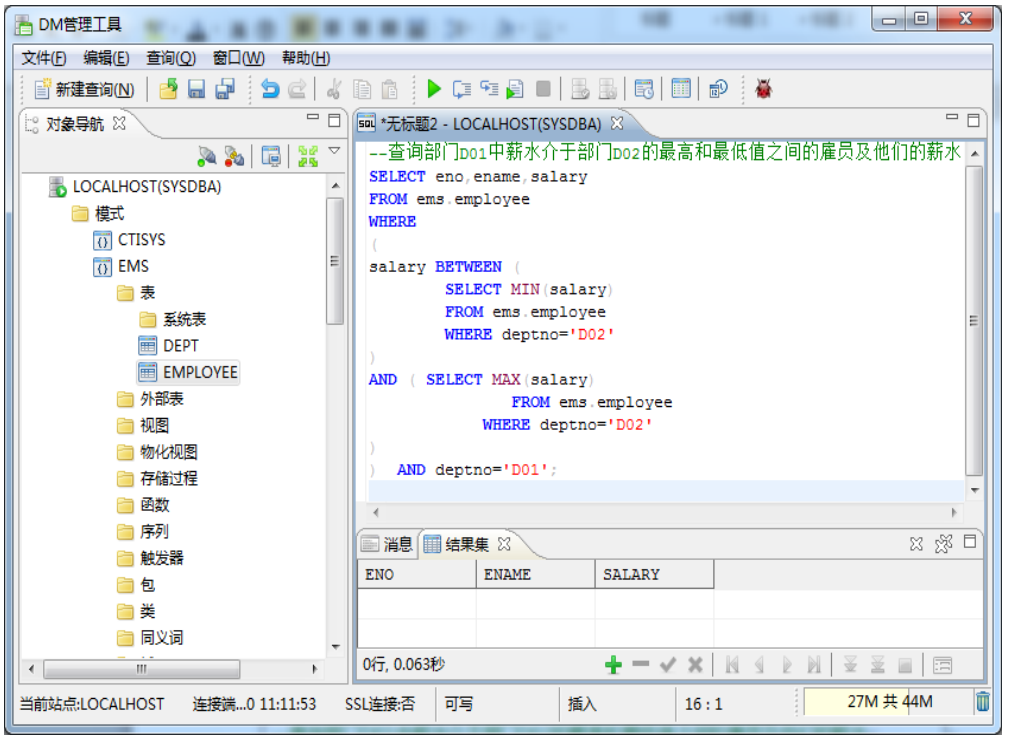
4.4.复杂查询

例 14 :查询部门 D01 中薪水介于部门 D02 的最高和最低值之间的雇员及他们的薪水。

图4.3.4 查询部门D01中薪水介于部门D02的最高和最低值之间的雇员及他们的薪水。

```
--查询部门D01中薪水介于部门D02的最高和最低值之间的雇员及他们的薪水
SELECT eno,ename,salary
FROM ems.employee
WHERE
(
   salary BETWEEN (
          SELECT MIN(salary)
          FROM ems.employee
          WHERE deptno='D02'
          )
```

```
        AND ( SELECT MAX(salary)
              FROM ems.employee
              WHERE deptno='D02'
            )
) AND deptno='D01';
```



5．DCL 的使用方法

5.1.SQL Server 登录管理

5.2.用户管理

5.2.1.DM 中有哪些管理用户

DM 首次提出了“三权分立”的概念，在 DM 数据库中有三类管理用户，系统管理员（SYSDBA），系统安全员（SYSSSO）和系统审计员（SYSAUDITOR）。SYSDBA 负责配置达梦数据库参数、管理数据库对象、自主访问权限的分配、数据导入导出以及数据库的备份和恢复等职责；SYSSSO 负责对系统进行强制访问控制，定义新的数据库安全员；SYSAUDITOR 负责设置审计策略。“三权分立”的思想是将系统中所有的权限按照类型进行划分，为每个管理员分配相应的权限，管理员之间的权限互不相交，从而达到相互制约、相互协助的目的，具有较高的安全性和较强的灵活性。

5.2.2. 如何创建用户

创建用户的命令是 CREATE USER，创建用户所涉及的内容包括为用户指定用户名、认证模式、口令、口令策略、空间限制、只读属性以及资源限制。其中用户名是代表用户帐号的标识符，它的命名规则是：

1. 必须以字符开始；
2. 长度为 1～128 个字符；
3. 从第二个字母开始，可以包括大小写字母、数字、_、$和#等字符。

表 5.2.2.1　用来创建用户的 CREATE USER 命令的完整语法格式

```
CREATE USER 用户名 IDENTIFIED BY "口令"
PASSWORD POLICY 口令策略
ENCRYPT BY <口令>
DISKSPACE LIMIT 空间限制
[NOT] READ ONLY LIMIT 资源限制
[NOT] ALLOW IP 地址
[NOT] ALLOW DATETIME 时间
DEFAULT TABLESPACE <表空间名>
```

表 5.2.2.2　创建对失败登录次数进行控制的用户，如果用户失败的登录次数达到 3 次，这个用户帐号将被锁定。

```
CREATE USER USER1 IDENTIFIED BY PASSWORD
LIMIT FAILED_LOGIN_ATTEMPS 3, PASSWORD_LOCK_TIME 5;
```

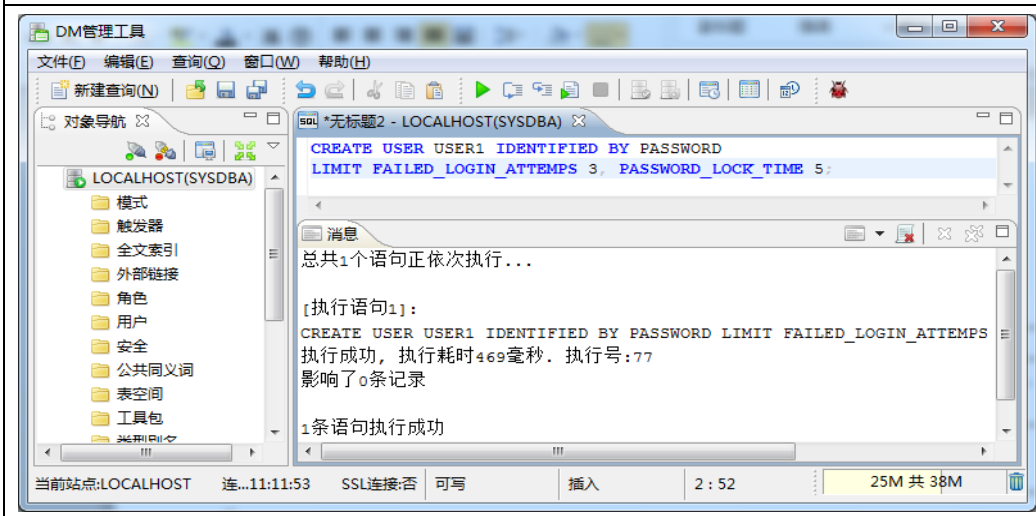


表 5.2.2.3　创建对修改口令进行限制的用户，要求用户在 30 天内必须把口令修改过 5 次后，才能使用过去用过的口令。

```
CREATE USER USER3 IDENTIFIED BY PASSWORD
LIMIT PASSWORD_REUSE_TIME 30, PASSWORD_REUSE_MAX 5;
```

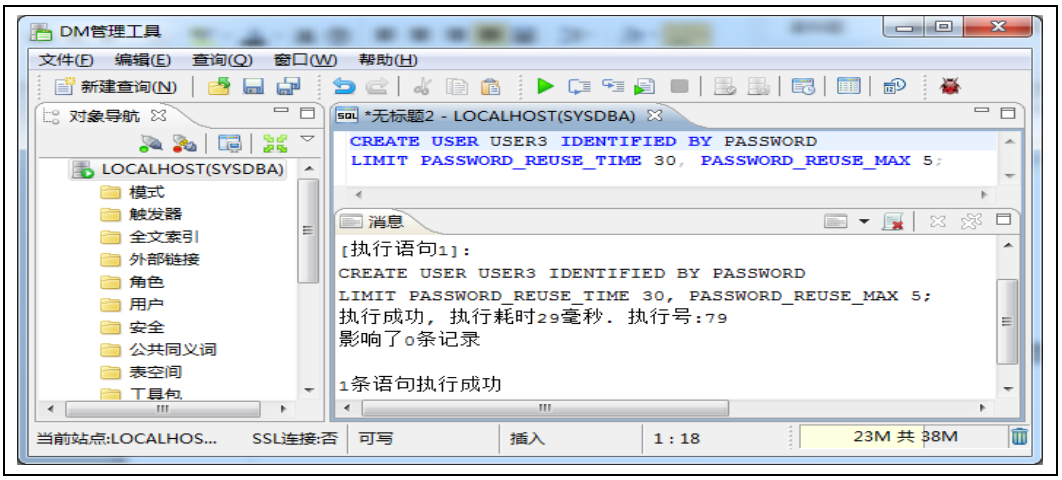
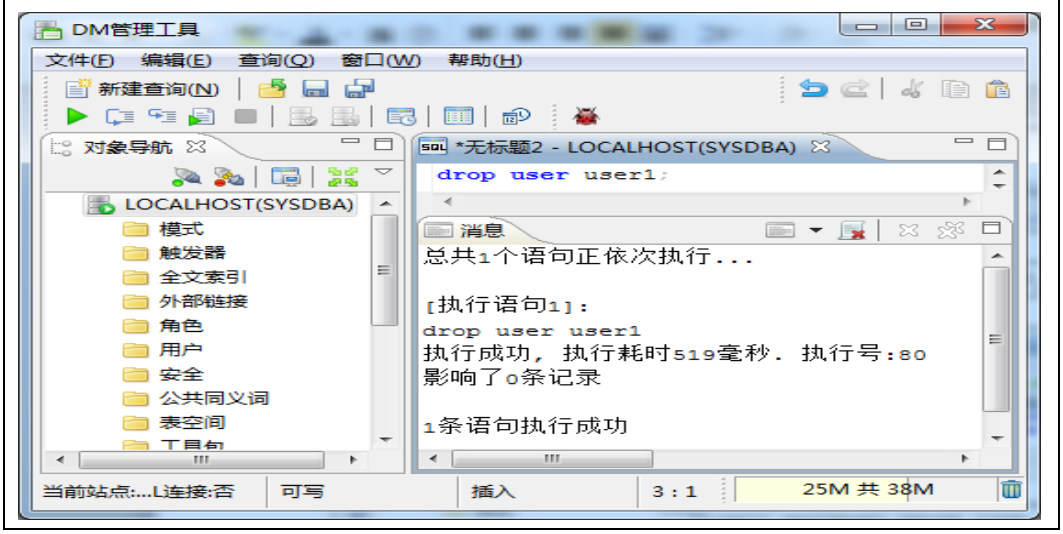


表 5.2.2.4 删除用户。
drop user user1;



5.3.授权用户（GRANT、REVOKE）

数据库权限的授予者一般是数据库管理员。普通用户被授予了某种数据库权限及其转授权时，系统允许它把所拥有的数据库权限再授予其他用户。

表 5.3.1 GRANT/REVOKE 命令的语法格式
GRANT 权限 1，权限 2,..... TO 用户 1，用户 2,..... WITH ADMIN OPTION;
REVOKE 数据库权限 1，数据库权限 2.... FROM 用户 1，用户 2....

GRANT 命令执行后，所有指定用户都将获得指定的权限。如果希望把一个权限授予所有用户，可以用 PUBLIC 代替所有的用户名。选项"WITH ADMIN OPTION"的功能是使得权限的获得者可以再将权限授予其他用户。

需要注意的是，数据库权限可以转授，但是回收时不能间接回收。假设有三个用户，第一个用户将某个数据库权限以“WITH ADMIN OPTION”的方式授予第二个用户，第二个用户将这个权限授予第三用户，那么当地一个用户从第二个用户回收这个权限时，并不能同时从第三个用户回收这个权限，第三个用户这时仍然具有这个权限。

5.3.1. 对象权限

对象权限总是针对表、视图、存储过程而言，它决定了能对这些对象执行哪些操作（如 UPDATE DELETE INSERT SELECT EXECUTE），不同类型的对象支持不同类型的操作。

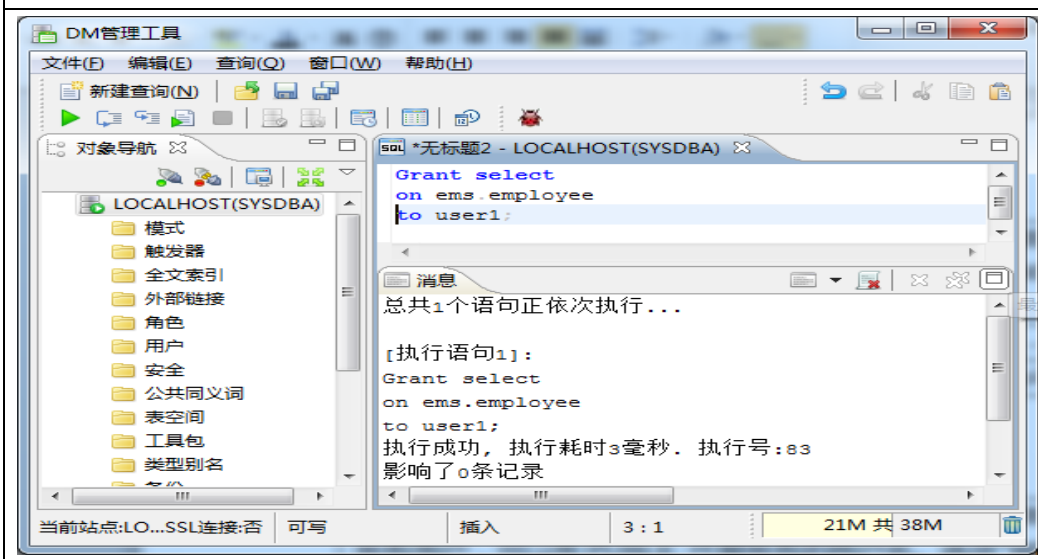
表 5.3.1.1 各种对象的可能操作列举。

对 象	操 作
表	SELECT、UPDATE、DELETE、INSERT、REFERENCE
视图	SELECT、UPDATE、DELETE 、INSERT
存储过程	EXECUTE
列	UPDATE、SELECT

例 15 为 ems 的用户 ‘user1’ 授予表 employee 查询权限。

图 5.3.1.1 为 ems 的用户 ‘user1’ 授予表 employee 查询权限。

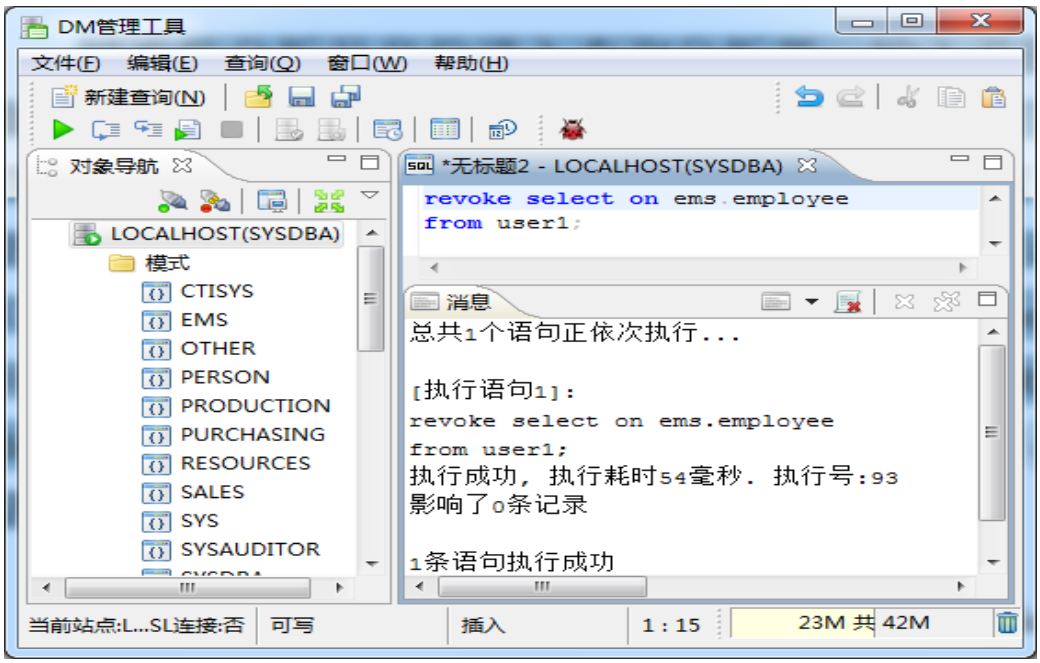
```
Grant select
on ems.employee
to user1;
```



例 16 回收为 ems 的用户‘user1’ 授予表 employee 查询权限。

图 5.3.1.2 回收为 ems 的用户‘user1’ 授予表 employee 查询权限。

```
revoke select on ems.employee
from user1;
```



5.3.2. 语句权限

语句权限指数据库用户执行某种语句的操作权，如创建数据库、表、存储过程等。这些语句虽然（如 CREATE 命令）包含有操作对象，但这些对象在操作前并不存在于数据库中。

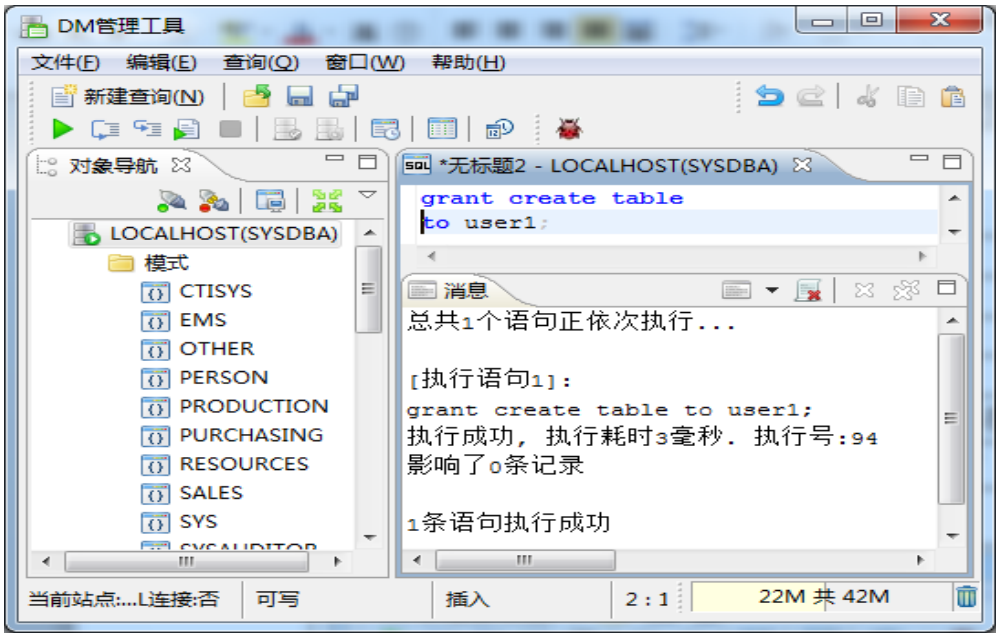
表 5.3.2.1 语句权限总结表

语句	含义
CREATE DTATBASE	创建数据库
CREATE TABLE	创建表
CREATE VIEW	创建视图
CREATE RULE	创建规则
CREATE DEFAULT	创建缺省
CREATE PROCEDURE	创建存储过程
BACKUP DATABASE	备份数据库
BACKUP LOG	创建事务日志

例 17 使用语句权限设置命令，授予用户‘user1’在数据库 ems 上建立新表的权限。

图 5.3.2.1 授予用户‘user1’在数据库 ems 上建立新表的权限。

```
grant create table
to user1;
```



DM7.0 中有角色的概念，角色的设置可以降低权限管理的繁杂性，在大的系统中，虽然有很多数据库用户，但可以将这些用户进行分类，每类定义一种角色，只要对角色进行权限设置，就可以简单管理用户了。角色的设置在此不赘述了。

6．游标的使用

由于 DM 支持的 SQL 语言是面向集合的语言，一条 SQL 语句可以产生或处理多条记录，而 PL/SQL 语言是面向记录的，一组主变量一次只能存放一条记录，所以仅使用主变量并不能满足 SQL 语句向应用程序输出数据的要求。为解决这一问题，SQL 语言引用了游标来协调这两种不同的工作方式，从而为应用程序逐个处理查询结果提供了强有力的手段。

6.1.游标的定义

表 6.1.1 定义游标的命令
CURSOR 游标名[FAST | NO FAST] IS|FOR SELECT 语句; 游标名 CURSOR; CURSOR 游标名;

6.2.游标的操作

- 打开游标：为了处理游标中的数据，首先要打开游标

表 6.2.1 打开游标的命令
OPEN 游标名

OPEN 游标名 FOR SELECT 语句;
由于游标的作用域仅在其所在的批处理中,当建立游标的批处理文件执行结束后,游标会被自动释放。因此在此打开操作前，要重新定义一次，因前面定义的游标已经被释放。如果要保留游标的作用域，可以在定义时指定其作用域为 GLOBAL。否则为 LOCAL。

- 读取游标中的数据：当游标成功打开后，就可以从游标中逐行读取数据

表 6.2.2 读取游标中数据的命令
FETCH [NEXT |PRIOR|FIRST|LAST|ABSOLUTE N|RELATIVE N] <游标名> [INTO <主变量名>{,<主变量名>}];
FETCH 命令可以将一行数据中各列的值依次赋给指定的变量，需要注意的是，变量的类型、数目要与游标中的一行的各列相对应

- 游标的关闭与释放：游标在使用完后应及时关闭，以释放它所占用的内存空间。

  关闭游标的语法格式为：CLOSE 游标名

  当游标关闭后，不能再从游标中获取数据。如果需要，可以再次打开游标。

表 6.2.3 游标属性	
属性	描述
CURSOR%ISOPEN	判断当前游标是否已经打开 ,如果打开 ,该属性值为 TRUE，否则为 FALSE
CURSOR%ROWCOUNT	表示到目前为止，用 FETCH 语句已经取到的行数
CURSOR%FOUND	表示最后一次 FETCH 操作是否从游标中取到一行数据，如果已经取到，其值为 TRUE，如果没有取到，其值为 FALSE
CURSOR%NOTFOUND	和 CURSOR%FOUND 正好相反

在大多数情况下，游标的应用步骤是：打开游标；开始循环；从游标中取出一行数据并进行处理；关闭循环；关闭游标，这一类应用称为游标用于循环。但还有一种循环与这种类型不同，这就是游标 FOR 循环。用于 FOR 循环的游标与一般游标的区别在于不需要显式的打开、关闭、取数据，测试数据的存在、定义存放数据的变量等。

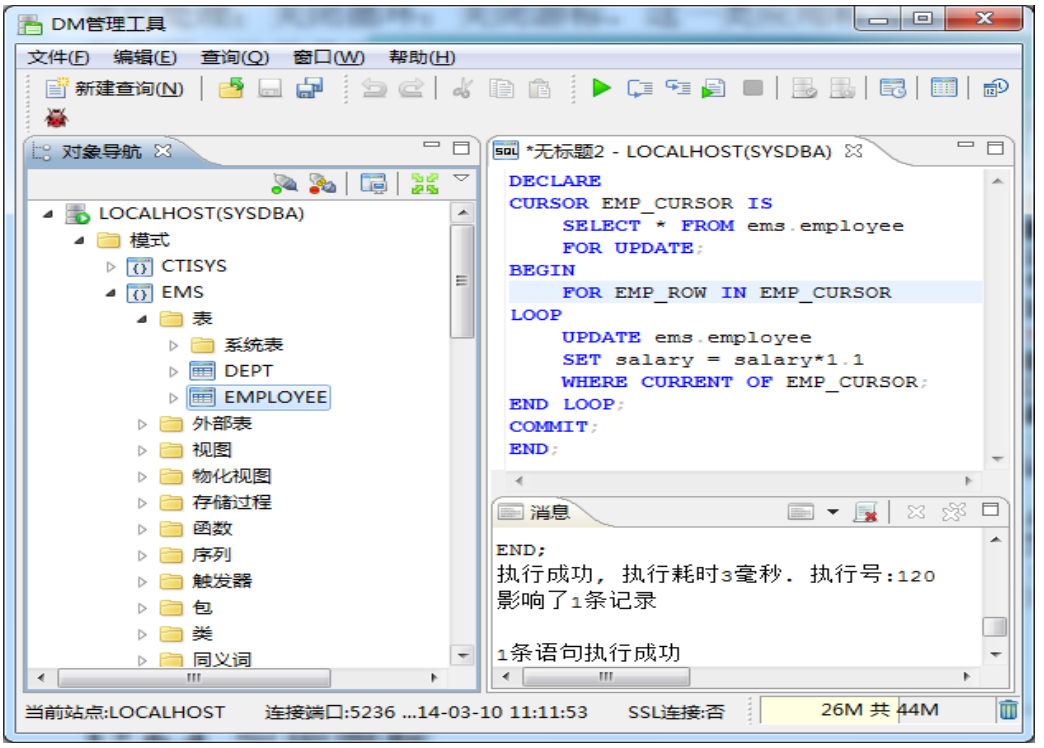
例 18 进行游标 FOR 循环，将表 employee 中所有人的 salary 增 10%。

图6.2.1 进行游标FOR循环，将表employee中所有人的salary增10%。

```
--游标for循环
DECLARE
CURSOR EMP_CURSOR IS
    SELECT * FROM ems.employee
    FOR UPDATE;
BEGIN
    FOR EMP_ROW IN EMP_CURSOR
LOOP
```

```
    UPDATE ems.employee
    SET salary = salary*1.1
    WHERE CURRENT OF EMP_CURSOR;
END LOOP;
COMMIT;
END
```

从上面的例子中我们可以看到游标FOR循环确实很好的简化了游标的开发，我们不再需要OPEN、FETCH和CLOSE语句，也不需要用%FOUND属性检测是否到最后一条记录，这一切DM服务器隐式的帮我们完成了。



7．数据库的备份和恢复

达梦数据库管理系统支持数据库和表空间的完全备份和增量备份。完全备份是指一个备份包含指定数据库或表空间的所有数据。当用户进行完全备份时，系统需要做的事情就是将一个数据库或表空间所包含的文件内容都拷贝到用户指定的备份文件中。完全备份包含了一个数据库或表空间的所有数据，这样的一个备份通常很大，需要占用很多的磁盘空间，尤其是系统在运行过程中，定期产生的完全备份所占用磁盘上的空间将非常大，同时，其备份过程也会很长。为了解决这个问题，DM 提供了增

量备份的功能，即不用备份当前数据库或表空间的所有数据，而只是以该数据库或表空间最近的一个备份为基础，仅仅拷贝自该备份以来所有被修改过的数据，形成一个增量备份。相对于完全备份而言，增量备份小得多，同时其备份的时间也会更短。

完全备份与增量备份各有缺点和优点。完全备份的优点是，只需要一次备份的数据，就能完全将数据库还原到备份时的状态，缺点是由于对整个数据库的数据备份，因此备份文件需要比较大的磁盘空间。由于增量备份是基于数据库的某一次备份(增量备份或是完全备份)做的操作，因此在还原时若增量备份所依赖的基础备份丢失，则该备份文件就变得不可用，相对于完全备份而言，不够可靠；另一方面，由于只是对某一段时间内修改数据的备份，因此备份文件比较小，备份速度比较快。需要说明的是，增量备份所依赖的基础备份有且仅有一个完全备份。

8．实验练习

实验 1： 基本表的创建、数据插入

（1）建立教学管理中的三个基本表：

Students（S＃，SNAME，AGE，SEX） 学生（学号，姓名，年龄，性别）

Courses（C＃，CNAME，SCORE，PC＃） 课程（课程号，课程名，学分，先行课号）

SC（S＃，C#，GRADE） 选修（学号，课程号，成绩）

（2）用 INSERT 命令输入数据。

表 6 基本表 Students 的数据：

S1	LU	20	M
S2	YIN	19	M
S3	XU	18	F
S4	QU	18	F
S6	PAN	14	M
S8	DONG	24	M

表 7 基本表 Courses 的数据

C1	数学	4	M
C2	英语	8	M
C3	数据结构	4	F
C4	数据库	3.5	F
C5	网络	4	M

表 8 基本表 SC 的数据（空格为未选修）：

S# / C#	S1	S2	S3	S4	S6	S8
C1	85	90	89	84	88	87
C2	73	NULL	86	82	75	85
C3	88	80			90	NULL
C4	89	85		NULL	92	88
C5	73	NULL				87

实验 2：数据查询

（1）列出选修课程号为C2的学生学号与姓名。

（2）检索选修课程名为“数学”的学生学号与姓名。

（3）检索没有选修C2课程的学生姓名与年龄。

（4）检索选修全部课程的学生姓名。

实验 3：数据修改、删除

（1）把C2课程的非空成绩提高10%。

（2）在SC表中删除课程名为“物理”的成绩所对应的元组。

（3）在S和SC表中删除学号为S8的所有数据。

实验 4：视图的操作

（1）建立男生学生的视图，属性包括学号，姓名，选修课程名和成绩。

（2）在男生视图中查询平均成绩大于80分的学生学号和姓名。

实验 5：库函数，授权的控制

（1）计算每个学生选修课程的门数、平均成绩。

（2）建立一个合法的用户，将SC表的查询权限授予该用户。

（3）使用GRANT语句，把对基本表students、Courses、SC的使用权限授予其他用户。

实验 6：数据库的备份、恢复

（1）使用完全备份将你的实验数据库备份到软盘。

（2）删除你所建立的数据库。

（3）恢复你的数据库。

（4） 在恢复后的数据库上撤销你建立的基本表和视图。

9．数据库课程设计基本要求

9.1.设计目标

1）运用数据库设计理论设计一个较完善的有实际意义的数据库；

2）掌握目前流行数据库管理系统SQL Server 2000的使用与应用开发技术；

3）为数据库开发相应应用程序，构成完整的数据库应用系统；

4）将设计在SQL Server 2000上实现。

9.2.基本要求

本课程设计，按照数据库原理课程中有关数据库应用系统设计章节的内容，主要从以下几个方面要求设计者必须提交相应的设计文档。

1）问题定义

在设计的第一阶段按软件工程要求给出系统定义，进行需求分析，设计出系统的信息模型即E-R图。

2）库文件结构（关系模式）

选定关系模型作为系统的数据模型，在信息模型的基础上设计合理的数据库文件结构，主要考虑规范化和实际应用需要，一般要求达到三范式，如果需要降低范式时应对冗余数据及适当的反规范化设计进行说明。

3）完整性考虑

关系模型的三类完整性约束条件在设计的过程中是必须考虑的，数据之间的关联应详细说明，要求使用DBMS对联系进行适当定义和编辑。对有些统计数据可使用触发器（请参考有关资料）。

4）库文件分类

应用系统中的库文件通常分为：

- 主文件：系统的核心数据，包含有需要永久保存的数据，是数据库系统的共享资源。
- 事务文件：记录事务处理的轨迹，保存数据更新的必要信息，供复核和数据恢复用，属于面对应用的局部数据，为了让设计者体会数据库应用程序的设计技巧，在此要求设计者将此类文件的结构设计出来。
- 工作文件：应用程序在工作时对应的库文件。
- 临时文件：存放应用程序执行过程的某些轨迹，供系统缓冲或恢复之用。

为了主文件的安全和提高并发度，通常情况下，将应用程序接收的数据写入临时文件和工作文件（请思考如果将这些数据用变量表示，会用什么不同）。因此，在你的系统中建议区分各类文件。

5）并发控制

数据库系统中的数据是全局共享的，因此在应用程序开发的过程中，应考虑多用户并发执行的情况，建议在开发前对工作区进行分配，每个文件尽量在同一工作区打开。建议遵循两段锁协议和共同开发中数据库文件的打开顺序。

6）安全性考虑

数据库的安全性是至关重要的，建议为系统开发密钥功能，对关键数据应采用隐码存放。

7）系统体系结构

系统可以使用自含式的DBMS开发，也可以使用嵌入式DBMS，可以设计为单机版或网络版。

8）用户接口设计

用户接口是系统最终提交给用户的操作界面，可使用菜单式也可使用按钮式。但应使应用程序和数据库相互隔离，禁止将数据库直接暴露给用户。

9）应用程序功能设计

应用系统的基本功能应根据实际目标来设定，通常有增、删、改、查、备份、恢复、密钥等功能。

9.3.实验系统参考题目

选题说明：2或3人一组，自由组合，从下列题目中任选一个，在课程设计期间按要求完成设计任务，并提交一份完整的设计报告和已调通的应用系统模型。

1） 学籍管理系统

2） 教学管理系统

3） 学生管理系统

4） 财务管理系统

5） 银行储蓄管理系统

6） 网上销售系统

7） 仓储管理系统

8） 图书管理系统

9） 超市收银系统

10）自选题目

9.4.文档内容

在课程设计提交的设计报告中，应包括以下内容：

1）系统分析、设计

2）数据字典、数据库结构

3）源程序代码

文档请按照软件工程的要求与格式书写。