实验 2　SQL 数据定义和操作

课程名称：数据库系统

专业：计算机科学与技术

学号：3190105359

姓名：段皞一

一 实验目的

1. 掌握关系数据库语言 SQL 的使用
2. 使所有的 SQL 作业都能上机通过

二 实验平台

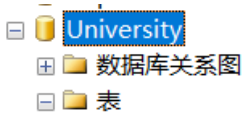
1. 数据库管理系统：SQL Server

三 实验内容和要求

1. 视图操作：通过视图的数据查询和数据修改

3.1 建立数据库

打开 Microsoft SQL Server Management Studio，连接自己的服务器，建立一个基于教材的详细的大学模式数据库。



在完成数据库University的建立后，在数据库的表选项中添加相应的一系列表。也可以使用SQL语言直接在新建查询中进行表的新建。

本次实验，采用SQL语句和图形界面结合的方式，进行表的创建。

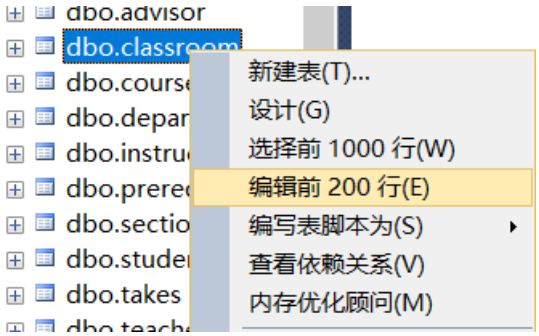
3.1.1创建表classroom

创建表“classroom”.相关代码如下所示：

```
create table classroom
(
building varchar(15),
room_number varchar(7),
capacity numeric(4,0),
primary key (building, room_number)
```

```
);
```

表建成以后，右键dbo.classroom单击选择“编辑前200行”选项，可以使用图形界面进行字段的添加。



完成添加字段。效果图如下：

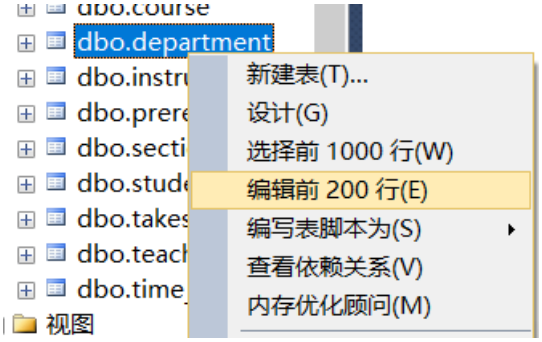
	building	room nu...	capacity
▶	Packard	101	500
	Painter	514	10
	Taylor	3128	70
	Watson	100	30
	Watson	120	50
*	*NULL*	*NULL*	*NULL*

3.1.2创建表department

创建表”department”,相关SQL语句如下所示：

```
create table department
(
dept_name varchar(20),
building varchar(20),
budget numeric(12,2) check (budget > 0),
primary key (dept_name)
);
```

创建表成功后，可以在表的栏目下查找到”dbo.department”记录。右键以后，在弹出的窗口栏中选择“编辑前200行”选项，即可进行该表相应字段的添加。



该表字段添加完成以后的示意图如下所示：

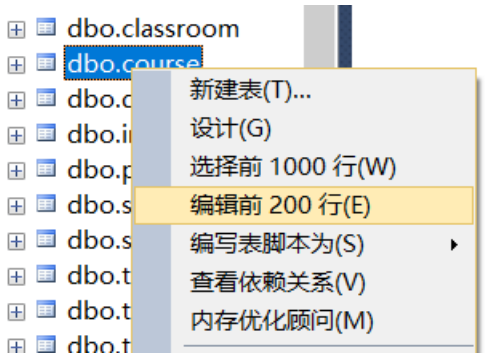
	dept_name	building	budget
	Biology	Watson	90000.00
	Comp.Sci.	Taylor	100000.00
	Elec.Eng.	Taylor	85000.00
	Finance	Painter	120000.00
	History	Painter	50000.00
	Music	Packard	80000.00
	Physics	Watson	70000.00
▸*	NULL	NULL	NULL

3.1.3 创建表 course

该表创建的相关代码如下所示：

```
create table course
(
course_id varchar(8),
title varchar(50),
dept_name varchar(20),
credits numeric(2, 0) check (credits > 0),
primary key (course_id),
foreign key (dept_name) references department
              on delete set null
);
```

创建成功以后，可以在表窗口栏下查到 dbo.course.右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



添加成功后的效果图如下：

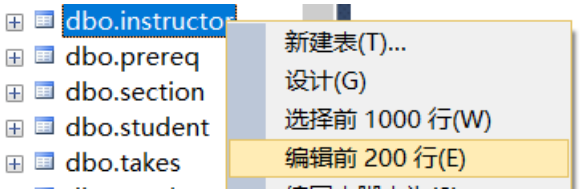
	course id	title	dept name	credits
▶	BIO-101	Intro. to Bi...	Biology	4
	BIO-301	Genetics	Biology	4
	BIO-399	Computati...	Biology	3
	CS-101	Intro. to C...	Comp.Sci.	4
	CS-190	Game Desi...	Comp.Sci.	4
	CS-315	Robotics	Comp.Sci.	3
	CS-319	Image Pro...	Comp.Sci.	3
	CS-347	Database ...	Comp.Sci.	3
	EE-181	Intro. to Di...	Elec.Eng.	3
	FIN-201	Investmen...	Finance	3
	HIS-351	World Hist...	History	3
	MU-199	Music Vid...	Music	3
	PHY-101	Physical Pr...	Physics	4
*	NULL	NULL	NULL	NULL

3.1.4 创建表 instructor

该表的创建代码如下所示：

```
create table instructor
(
ID varchar(5),
name varchar(20) not null,
dept_name varchar(20),
salary numeric (8,2) check (salary > 29000),
primary key(ID),
foreign key (dept_name) references department
on delete set null
);
```

创建成功以后，可以在表窗口栏下查到 dbo. instructor. 右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



字段添加完成以后的效果图如下：

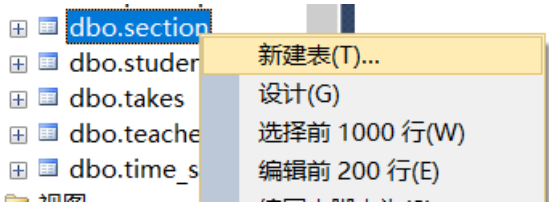
	ID	name	dept_name	salary
	10101	Srinivasan	Comp.Sci.	65000.00
	12121	Wu	Finance	90000.00
	15151	Mozart	Music	40000.00
	22222	Einstein	Physics	95000.00
	32343	El Said	History	60000.00
	33456	Gold	Physics	87000.00
	45565	Katz	Comp.Sci.	75000.00
	58583	Califieri	History	62000.00
	76543	Singh	Finance	80000.00
	76766	Crick	Biology	72000.00
	83821	Brandt	Comp.Sci.	92000.00
	98345	Kim	Elec.Eng.	80000.00
▸*	NULL	NULL	NULL	NULL

3.1.5 创建表 section

该表的创建的代码如下：

```
create table section
(
course_id varchar(8),
sec_id varchar(8),
semester varchar(6) check (semester in
('Fall', 'Winter', 'Spring', 'Summer')),
year numeric(4, 0) check (year > 1701 and year < 2100),
building varchar(15),
room_number varchar(7),
time_slot_id varchar(4),
primary key (course_id, sec_id, semester, year),
foreign key (course_id) references course
on delete cascade,
foreign key (building, room_number) references classroom
on delete set null
);
```

创建成功以后，可以在表窗口栏下查到 dbo.section.右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



添加字段以后的示意图如下：

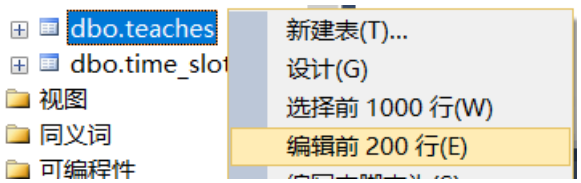
	course id	sec id	semester	year	building	room nu...	time slot...
	BIO-101	1	Summer	2009	Painter	514	B
	BIO-301	1	Summer	2010	Painter	514	A
	CS-101	1	Fall	2009	Packard	101	H
	CS-101	1	Spring	2010	Packard	101	F
	CS-190	1	Spring	2009	Taylor	3128	E
	CS-190	2	Spring	2009	Taylor	3128	A
	CS-315	1	Spring	2010	Watson	120	D
	CS-319	1	Spring	2010	Watson	100	B
	CS-319	2	Spring	2010	Taylor	3128	C
	CS-347	1	Fall	2009	Taylor	3128	A
	EE-181	1	Spring	2009	Taylor	3128	C
	FIN-201	1	Spring	2010	Packard	101	B
	HIS-351	1	Spring	2010	Painter	514	C
	MU-199	1	Spring	2010	Packard	101	D
	PHY-101	1	Fall	2009	Watson	100	A
▸*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

3.1.6创建表teaches

创建表的相关代码如下：

```
create table teaches
(
ID varchar(5),
course_id varchar(8),
sec_id varchar(8),
semester varchar(6),
year numeric(4, 0),
primary key (ID, course_id, sec_id, semester, year),
foreign key (course_id, sec_id, semester, year) references section
on delete cascade,
foreign key (ID) references instructor
on delete cascade
);
```

创建成功以后，可以在表窗口栏下查到 dbo. teaches. 右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



添加字段完成以后的示意图如下：

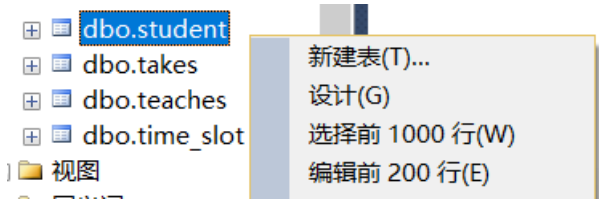
	ID	course id	sec id	semester	year
	10101	CS-101	1	Fall	2009
	10101	CS-315	1	Spring	2010
	10101	CS-347	1	Fall	2009
	12121	FIN-201	1	Spring	2010
	15151	MU-199	1	Spring	2010
	22222	PHY-101	1	Fall	2009
	32343	HIS-351	1	Spring	2010
	45565	CS-101	1	Spring	2010
	45565	CS-319	1	Spring	2010
	76766	BIO-101	1	Summer	2009
	76766	BIO-301	1	Summer	2010
	83821	CS-190	1	Spring	2009
	83821	CS-190	2	Spring	2009
	98345	EE-181	1	Spring	2009
▸*	NULL	NULL	NULL	NULL	NULL

3.1.7创建表student

创建相关表的相应代码如下所示：

```
create table student
(
ID varchar(5),
name varchar(20) not null,
dept_name varchar(20),
tot_cred numeric(3, 0) check(tot_cred >= 0),
primary key (ID),
foreign key (dept_name) references department
on delete set null
);
```

创建成功以后，可以在表窗口栏下查到 dbo. student. 右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



编辑字段成功后，可以看到相应的示意图如下：

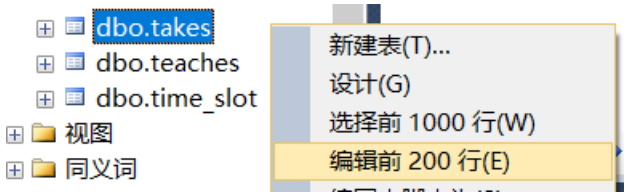
ID	name	dept name	tot cred
00128	Zhang	Comp.Sci.	102
12345	Shankar	Comp.Sci.	32
19991	Brandt	History	80
23121	Chavez	Finance	110
44553	Peltier	Physics	56
45678	Levy	Physics	46
54321	Williams	Comp.Sci.	54
55739	Sanchez	Music	38
70557	Snow	Physics	0
76543	Brown	Comp.Sci.	58
76653	Aoi	Elec.Eng.	60
98765	Bourikas	Elec.Eng.	98
98988	Tanaka	Biology	120
NULL	*NULL*	*NULL*	*NULL*

3.1.8创建表takes

该表创建的相关代码如下所示：

```
create table takes
(
ID varchar(5),
course_id varchar(8),
sec_id varchar(8),
semester varchar(6),
year numeric(4, 0),
grade varchar(2),
primary key (ID, course_id, sec_id, semester, year),
foreign key (course_id, sec_id, semester, year) references section
on delete cascade,
foreign key(ID) references student
on delete cascade
);
```

创建成功以后，可以在表窗口栏下查到 dbo.takes.右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



完成字段的添加以后，相关示意图如下：

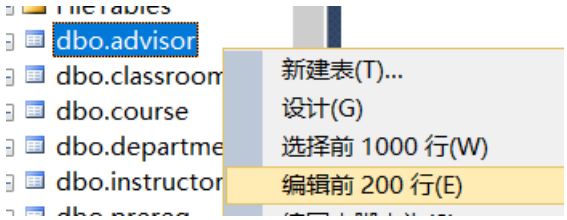
	ID	course id	sec id	semester	year	grade
	00128	CS-101	1	Fall	2009	A
	00128	CS-347	1	Fall	2009	A-
	12345	CS-101	1	Fall	2009	C
	12345	CS-190	2	Spring	2009	A
	12345	CS-315	1	Spring	2010	A
	12345	CS-347	1	Fall	2009	A
	19991	HIS-351	1	Spring	2010	B
	23121	FIN-201	1	Spring	2010	C+
	44553	PHY-101	1	Fall	2009	B-
	45678	CS-101	1	Fall	2009	A-
	45678	CS-319	1	Spring	2010	B
	54321	CS-101	1	Fall	2009	A-
	54321	CS-190	2	Spring	2009	B+
	55739	MU-199	1	Spring	2010	A-
	76543	CS-101	1	Fall	2009	A
	76543	CS-319	2	Spring	2010	A
	76653	EE-181	1	Spring	2009	C
	98765	CS-101	1	Fall	2009	C-
	98765	CS-315	1	Spring	2010	B
	98988	BIO-101	1	Summer	2009	A
	98988	BIO-301	1	Summer	2010	*NULL*
▶*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

3.1.9创建表advisor

该表创建的相关代码如下：

```
create table advisor
(
s_ID varchar(5),
i_ID varchar(5),
primary key (s_ID),
foreign key (i_ID) references instructor(ID)
on delete set null,
foreign key (s_ID) references student(ID)
on delete cascade
);
```

创建成功以后，可以在表窗口栏下查到 dbo.advisor.右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



完成字段的添加以后，相关示意图如下：

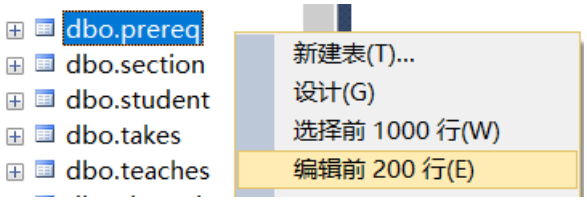
	s ID	i ID
	00128	45565
	12345	10101
	23121	76543
	44553	22222
	45678	22222
	76543	45565
	76653	98345
	98988	76766
▶*	NULL	NULL

3.1.10创建表prereq

创建表的相关代码如下：

```
create table prereq
(
course_id varchar(8),
prereq_id varchar(8),
primary key (course_id, prereq_id),
foreign key (course_id) references course
on delete cascade,
foreign key (prereq_id) references course
);
```

创建成功以后，可以在表窗口栏下查到 dbo.prereq.右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



完成字段的添加以后，可以看到示意图如下：

	course id	prereq id
	BIO-301	BIO-101
	BIO-399	BIO-101
	CS-190	CS-101
	CS-315	CS-101
	CS-319	CS-101
	CS-347	CS-101
	EE-181	PHY-101
▶*	NULL	NULL

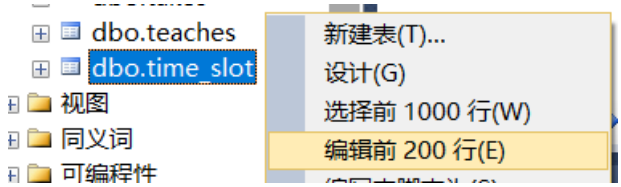
3.1.11创建表time_slot

该表创建的相关代码如下：

```
create table time_slot
```

```
(
time_slot_id varchar(4),
day varchar(1) check (day in ('M', 'T', 'W', 'R', 'F', 'S', 'U')),
start_time time,
end_time time,
primary key(time_slot_id, day, start_time)
);
```

创建成功以后，可以在表窗口栏下查到 dbo.time_slot.右键以后将弹出窗口栏，选择“编辑前 200 行”选项，即可进行相应字段的添加。



完成该表字段的添加以后，可以看到相关的示意图如下：

time_slot...	day	start_time	end_time
A	M	08:00:00	08:50:00
A	W	08:00:00	08:50:00
A	F	08:00:00	08:50:00
B	M	09:00:00	09:50:00
B	W	09:00:00	09:50:00
B	F	09:00:00	09:50:00
C	M	11:00:00	11:50:00
C	W	11:00:00	11:50:00
C	F	11:00:00	11:50:00
D	M	13:00:00	13:50:00
D	W	13:00:00	13:50:00
D	F	13:00:00	13:50:00
E	T	10:30:00	11:45:00
E	R	10:30:00	11:45:00
F	T	14:30:00	15:45:00
F	R	14:30:00	15:45:00
G	M	16:00:00	16:50:00
G	W	16:00:00	16:50:00
G	F	16:00:00	16:50:00
H	W	10:00:00	12:30:00
NULL	NULL	NULL	NULL

3.2 数据定义

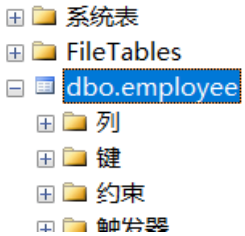
3.2.1 表的建立、删除和修改

建立一个表，SQL 语句如下所示：

```
create table employee
(
person_name char(10) not null,
```

```
street char(10),
city char(10)
);
```

可以看到索引下多了一个 dbo.employee.可见表创建成功。



修改一个表。相关的 SQL 语句的代码如下：

```
ALTER TABLE employee
add number_id char(10)
```

可以看到 employee 表下多了一个字段名 number_id.修改表成功。

	person n...	street	city	number id
*	NULL	NULL	NULL	NULL

实验中其他修改表的例子如下：

使用 SQL 语句进行类型长度的修改。比如将 varchar(10)改成 varchar(100)。

```
alter table userinfo
alter column name varchar(100);
```

使用 SQL 语句进行字段类型的修改。

```
alter table userinfo
alter column age float;
```

添加是否可以为空。

```
alter table userinfo
alter column age float not null;
```

添加主键。

```
alter table userinfo
add constraint KID primary key(ID);
```

新建字段。

```
alter table userinfo
add grade varchar(10);
```

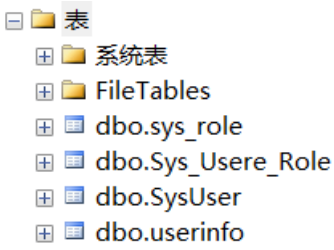
first：第一个位置；

after：在哪个字段之后；默认在最后一个字段的后面。

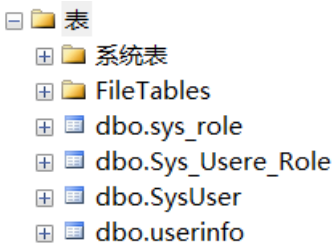
```
alter table userinfo
add newkey FIRST;
```

删除一个表，相关的 SQL 语句如下所示：

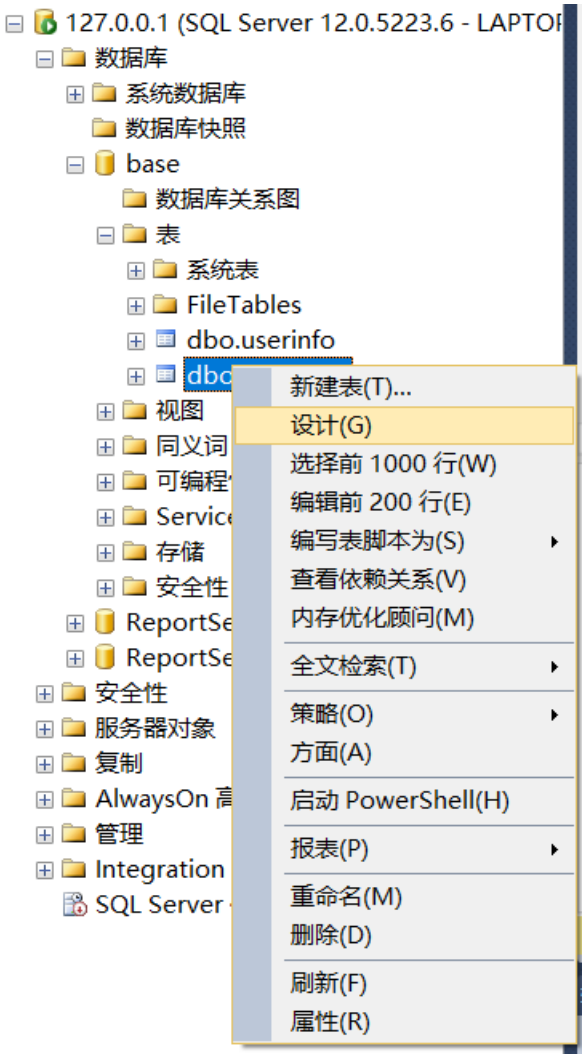
```
DROP TABLE employee
```



可以看到，索引下 dbo.employee 消失了。删除表成功。



通过视图方式修改表结构。右键相应的表，点击“设计”。



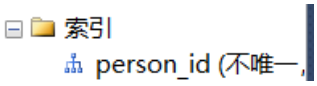
在弹出的对话框中，可以对表的各个字段的属性进行调整。

3.2.2 索引的建立、删除和修改

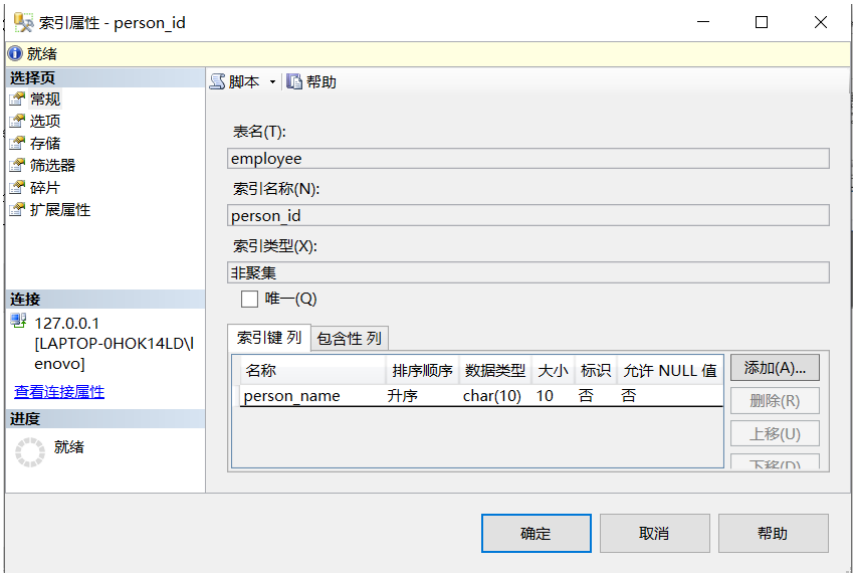
建立索引：

```
create index person_id
on employee(person_name)
```

可以在搜索栏找到新建的索引。



之后还可以对索引的属性进行更改。



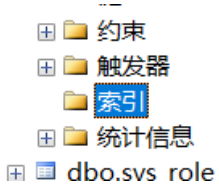
重建索引：

```
alter index person_id on employee
rebuild
```

删除索引：

```
drop index employee.person_id
```

可以看见，已经不存在索引 person_id. 删除索引成功。



3.2.3 视图的建立、删除和修改

视图的建立，相关代码如下所示：

```
CREATE VIEW member
AS
SELECT title, credits
FROM course
WHERE credits = 4
```

可以看见视图栏下出现了 dbo.member 的视图，新建试图成功。

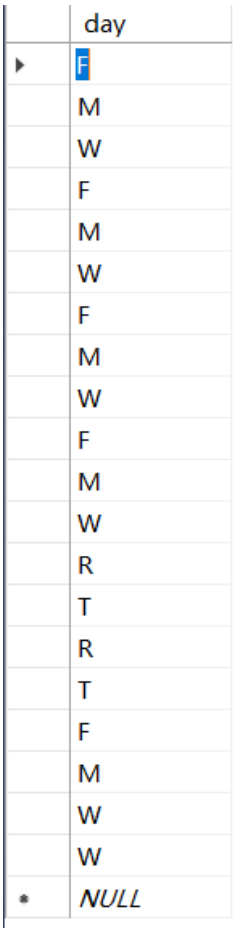
可以观察新建的视图。

	title	credits
	Intro. to Bi...	4
	Genetics	4
	Intro. to C...	4
	Game Desi...	4
	Physical Pr...	4
▶*	NULL	NULL

视图的修改，相关代码如下：

```
ALTER view dbo.member
as
select day
from time_slot
where start_time is not null
```

可以观察到 member 发生了改变，视图的修改成功。



	day
▶	F
	M
	W
	F
	M
	W
	F
	M
	W
	F
	M
	W
	R
	T
	R
	T
	F
	M
	W
	W
*	NULL

3.3 数据更新

3.3.1 用 insert 命令插入表命令

新增表记录的时候，我们可以插入单行数据，也可以插入多行数据。

在讲述如何新增表记录之前，先了解一下如何查询表，这是一个基础的 SQL 语句。我们输入以下代码查询表的内容：

```
select * from [dbo].[userinfo];
```

可以发现，当前是一个空表。

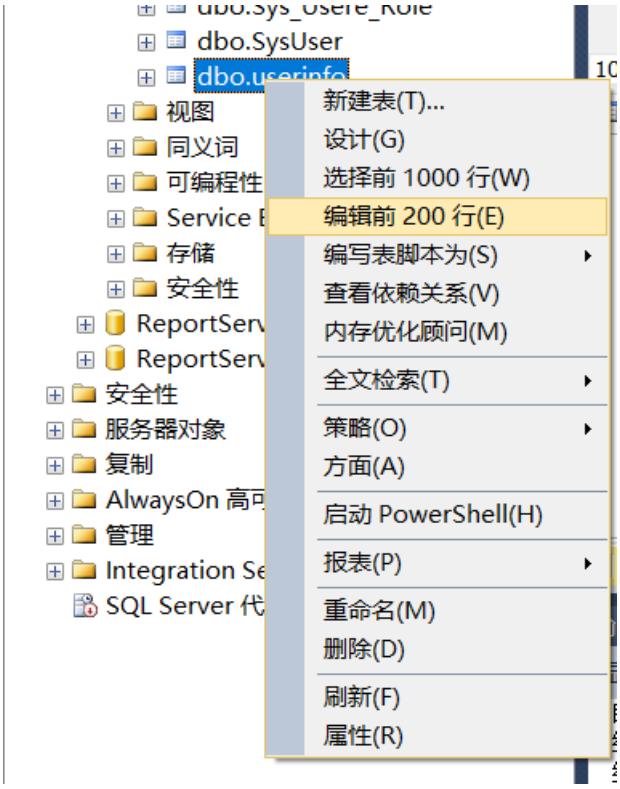
```
select * from [dbo].[userinfo];
```

0 %

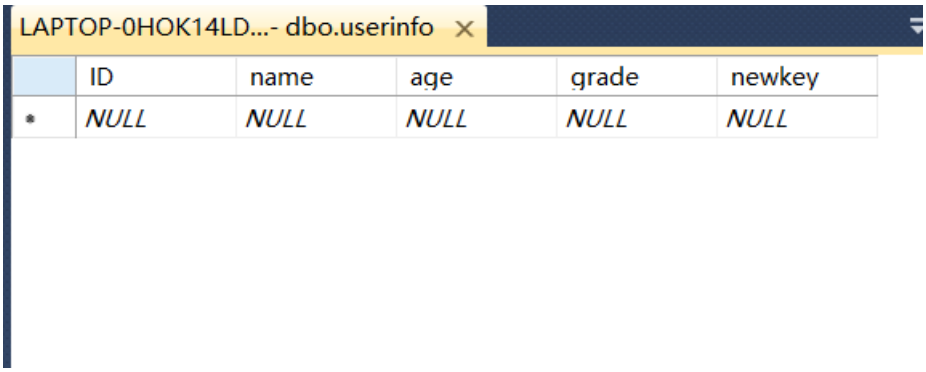
结果 消息

ID	name	age	grade	newkey

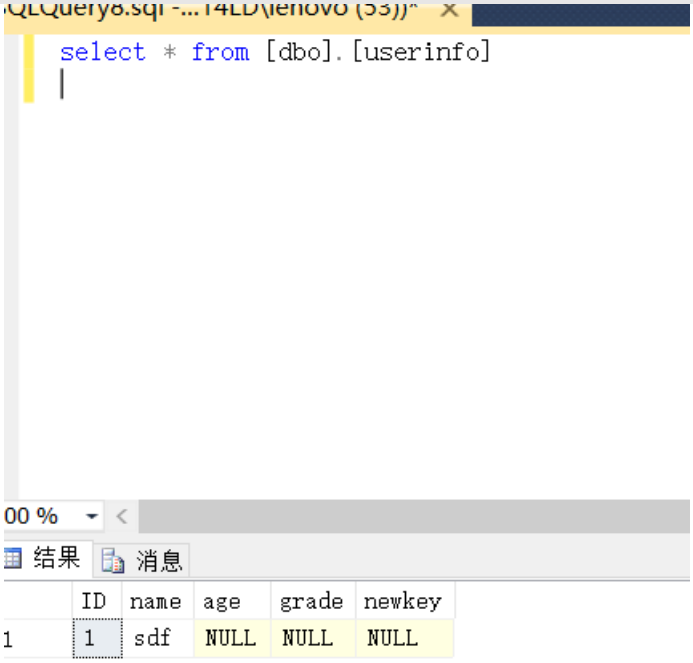
或者还可以从图形界面进行相关的查找。鼠标右键相应的表，选择编辑前 200 行。



可以观察到，当前的表也显示为空。




```
select * from [dbo].[userinfo]
```

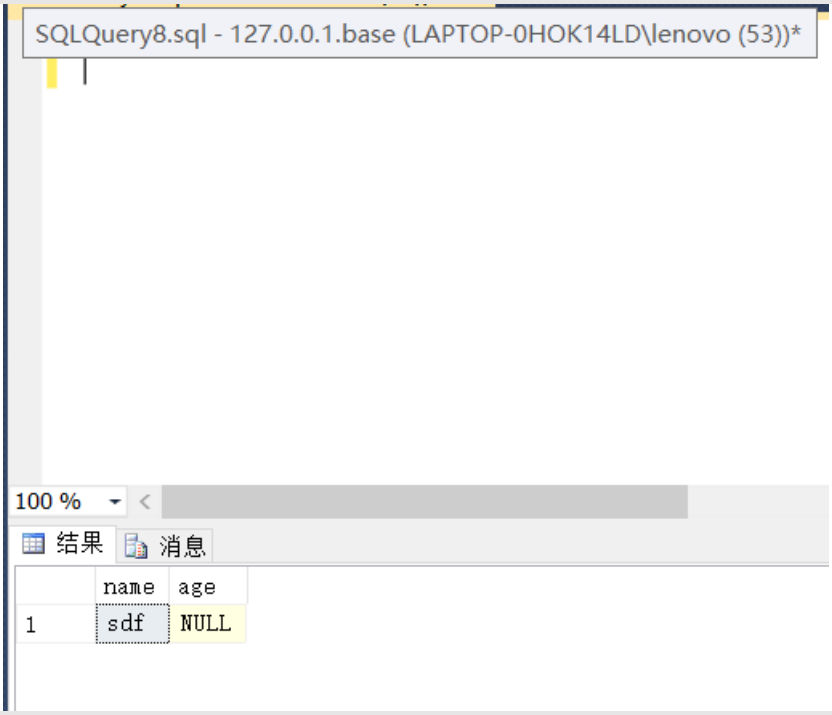



```
select name, age from [dbo].[userinfo]
```

```
select name, age from [dbo].[userinfo]
```

distinct/top 的用法：

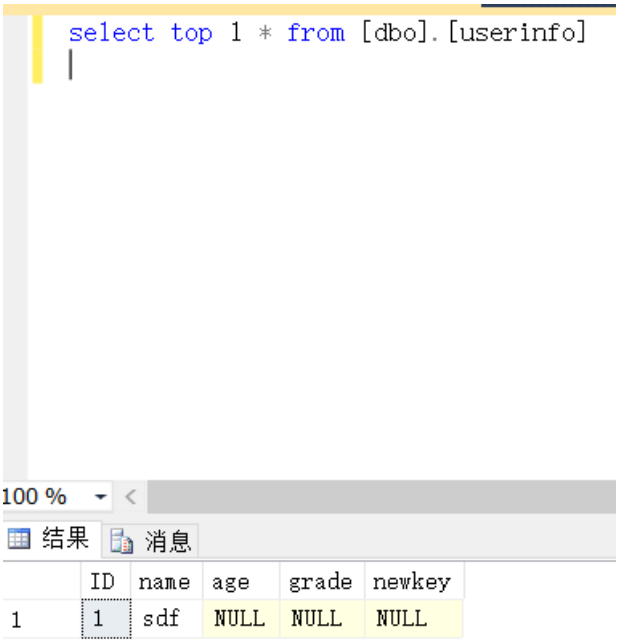
```
select distinct name, age from [dbo].[userinfo]
```




```
select top 0 name, age from [dbo].[userinfo]
```

```
select top 0 name, age from [dbo].[userinfo]
```


```
select top 1 * from [dbo].[userinfo]
```



可以用 SQL 语句进行新增表记录，相关代码如下：

```
insert into userinfo(age, grade, ID)
values(15, '一年级', 1031004);
```

注意，要一一对应。插入多行记录的方法类似，同样需要一一对齐。

```
insert into userinfo(age, grade, ID)
values(15, '一年级', 1031004),
(18, '四年级', 1031005),
(10, '一年级', 1031006),
(13, '六年级', 1031007),
```

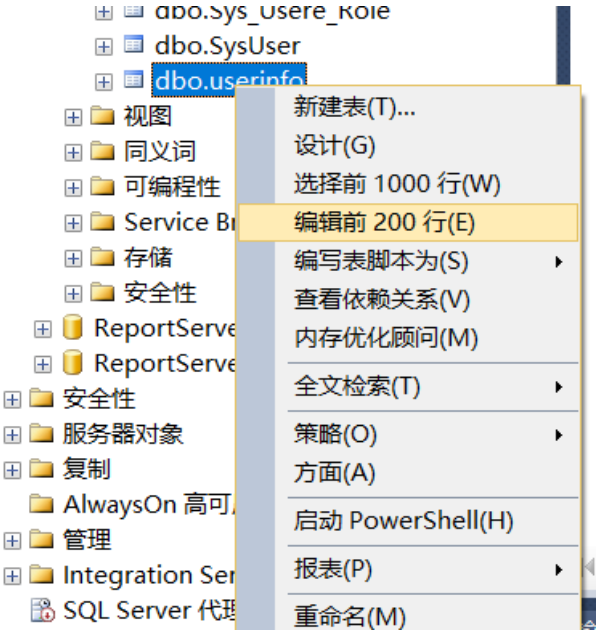
```
(90, '二年级', 1031048);
```

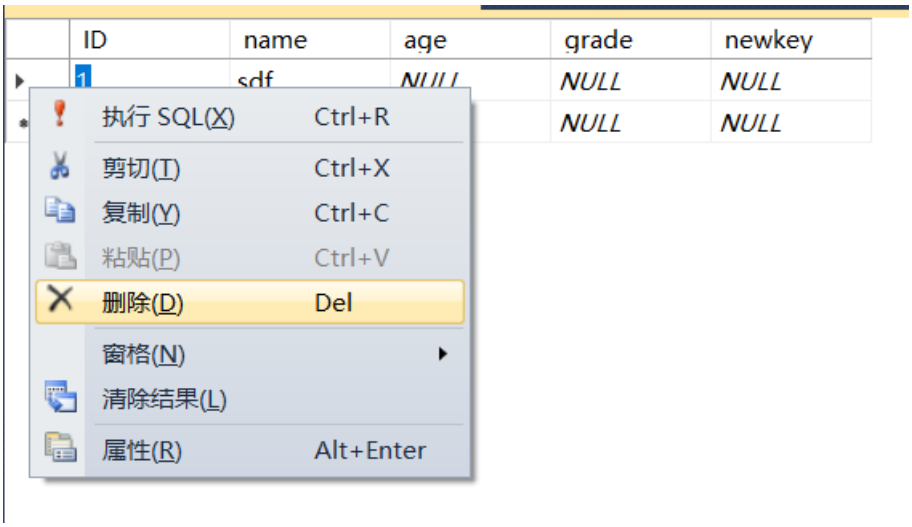
插入其他表的大数据集：

```
insert into userinfo(age, grade, ID)
select age, grade, email from [dbo][bigdata];
```

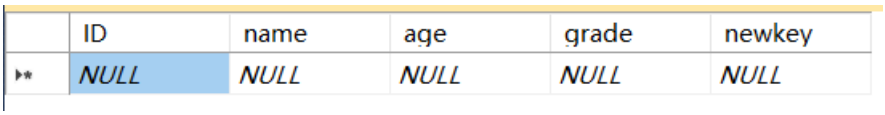
3.3.2 用 delete 命令删除表命令

选择相应的表格，右键选择“编辑前 200 行”，即可在弹出的窗口中进行表记录的删除操作。





删除完毕：



同样地，也可以使用 SQL 语句进行表记录的删除操作。

```
delete from [dbo].[userinfo]
where name = 'zhangsan';
```

3.3.3 用 updata 命令修改表数据

```
update department
set building = 'Watson'
where dept_name = 'Physics'
```

观察到 physics 的系大楼改到了’Watsond’.

	Physics	Watson	70000.00

3.4 数据查询

3.4.1 单表查询

单表查询的代码案例：

```
//选出ID=-1的1字段
select * from [dbo].[userinfo]
where ID = -1;
//选出RoleName以"管理员"作为结尾的字段
select * from [dbo].[userinfo]
where RoleName like '%管理员'
//选出RoleName以"仓库"作为开头的字段
select * from [dbo].[userinfo]
where RoleName like '仓库%'
//选出RoleName包含"中心"的字段
select * from [dbo].[userinfo]
where RoleName like '%中心%'
//选出name是三个字节的字段
select * from [dbo].[userinfo]
where name like '___'
```

Between 用法。限制条件表达式，指定表达式范围值。

```
select * from [dbo].[userinfo]
where name between 100 and 200;

select * from [dbo].[userinfo]
where name between '9DDBL006' and '9GDBL006';

select * from [dbo].[userinfo]
where time between '2015-07-29 12:00:00.001' and '2019-5-18
23:59:59.999';
```

In 语句的用法实例如下：

```
select * from [dbo].[userinfo]
where StudantNo in (1503, 1501, 1505);
```

取反的用法如下：

```
select * from [dbo].[userinfo]
where StudantNo not in (1503, 1501, 1505);

select * from [dbo].[userinfo]
```

```
where StudantName not in ('张三', '李四', '王五');

select * from [dbo].[Students]
where StudantName in (select studentname from
[dbo].[Student_Lesson]);
```

exists 语法，exits 语句返回布尔值 True 或者 False。

```
select a.StudentNo, a.StudentName, a.Age from [dbo].[Students] a
where exists (select ID from [dbo].[Student_Lesson] b
where a.StudentNo = b.StudentNo);
```

返回记录排序，示例代码如下：

```
select * from [dbo].[Students]
order by parentid;

select * from [dbo].[Students]
order by parentid desc;

select * from [dbo].[Students]
order by parentid, ordernum;
```

表关联查询，交叉关联示例代码：

```
select * from [dbo].[Students] a
inner join [dbo].[Classes] b
on a.ClassesNo = b.ClassNo;
```

左关联：左表的全部数据和右表满足关联的数据。

```
select * from [dbo].[Students] a
left join [dbo].[Classes] b
on a.ClassesNo = b.ClassNo;
```

聚合函数的用法：

```
select avg(Score) from [dbo].[Score]
select sum(Score) from [dbo].[Score]
select min(Score) as SumScore from [dbo].[Score]
select max(Score) as SumScore from [dbo].[Score]
```

取别名：

```
select sum(Score) as SumScore from [dbo].[Score]
```

LEN()函数，返回指定字符串表达式的字符数：

```
select *, LEN() as len_grade from [dbo].[Classes]
```

DATALENGTH()返回表达式的字节数：

```
select *, DATALENGTH() as len_grade from [dbo].[Classes]
```

随机数的产生：

```
select rand();
select  floor(rand()*10);
//
select rand();
select  ceiling(rand()*1000);
```

具体的案例，查询秋季开课的字段。

```
select *
from takes
where semester = 'Fall'
```

	ID	course_id	sec_id	semester	year	grade
1	00128	CS-101	1	Fall	2009	A
2	单击可选择整个行		1	Fall	2009	A-
3	12345	CS-101	1	Fall	2009	C
4	12345	CS-347	1	Fall	2009	A
5	44553	PHY-101	1	Fall	2009	B-
6	45678	CS-101	1	Fall	2009	A-
7	54321	CS-101	1	Fall	2009	A-
8	76543	CS-101	1	Fall	2009	A
9	98765	CS-101	1	Fall	2009	C-

3.4.2 多表查询

多表查询的示例如下：

```
select *
from teaches, course
where teaches.course_id = course.course_id
```

示意图如下：

	ID	course_id	sec_id	semester	year	course_id	title	dept_name	credits
2	10101	CS-315	1	Spring	2010	CS-315	Robotics	Comp. Sci.	3
3	10101	CS-347	1	Fall	2009	CS-347	Database System Concepts	Comp. Sci.	3
4	12121	FIN-201	1	Spring	2010	FIN-201	Investment Banking	Finance	3
5	15151	MU-199	1	Spring	2010	MU-199	Music Video Production	Music	3
6	22222	PHY-101	1	Fall	2009	PHY-101	Physical Principles	Physics	4
7	32343	HIS-351	1	Spring	2010	HIS-351	World History	History	3
8	45565	CS-101	1	Spring	2010	CS-101	Intro. to Computer Science	Comp. Sci.	4
9	45565	CS-319	1	Spring	2010	CS-319	Image Processing	Comp. Sci.	3
10	76766	BIO-101	1	Summer	2009	BIO-101	Intro. to Biology	Biology	4
11	76766	BIO-301	1	Summer	2010	BIO-301	Genetics	Biology	4
12	83821	CS-190	1	Spring	2009	CS-190	Game Design	Comp. Sci.	4
13	83821	CS-190	2	Spring	2009	CS-190	Game Design	Comp. Sci.	4
14	98345	EE-181	1	Spring	2009	EE-181	Intro. to Digital Systems	Elec. Eng.	3

3.4.3 嵌套子表查询

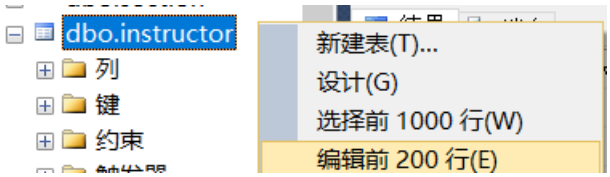
```
select dept_name, avg_salary
from(  select dept_name, avg(salary) as avg_salary
       from instructor
       group by dept_name)
where avg_salary > 30000
```

执行该嵌套查询语句后的示意图如下所示：

	dept_name	avg_salary
1	Biology	72000.000000
2	Comp.Sci.	77333.333333
3	Elec.Eng.	80000.000000
4	Finance	85000.000000
5	History	61000.000000
6	Music	40000.000000
7	Physics	91000.000000

3.5 视图操作

通过视图的数据查询和数据修改方法是，右键相应的表，选择“编辑前 200 行”选项，即可进行查询和修改操作。



	ID	name	dept name	salary
▶	10101	Srinivasan	Comp.Sci.	65000.00
	12121	Wu	Finance	90000.00
	15151	Mozart	Music	40000.00
	22222	Einstein	Physics	95000.00
	32343	El Said	History	60000.00
	33456	Gold	Physics	87000.00
	45565	Katz	Comp.Sci.	75000.00
	58583	Califieri	History	62000.00
	76543	Singh	Finance	80000.00
	76766	Crick	Biology	72000.00
	83821	Brandt	Comp.Sci.	92000.00
	98345	Kim	Elec.Eng.	80000.00
*	NULL	NULL	NULL	NULL

3.6 其他操作

时间的获取：

```
//格式1
select getdate()
//格式2
select getutcdate()
```

效果如下：

1	2021-03-18 12:52:34.000

1	2021-03-18 04:53:02.040

Convert 函数：把日期转换成新的数据类型。

```
//格式1
select convert(varchar(10), getdate(), 110);
//格式2
select convert(varchar(10), getdate(), 111);
//格式3
select convert(varchar(20), getdate(), 120);
```

	(无列名)
1	03-18-2021

	(无列名)
1	2021/03/18

	(无列名)
1	2021-03-18 13:02:17