华中科技大学

数据库系统原理实践报告

姓　　名：	许荣仁
学　　院：	计算机科学与技术
专　　业：	计算机科学与技术
班　　级：	CS1409
学　　号：	U201414803
指导教师：	左琼

分数	
教师签名	

2017 年 5 月 6 日

目　录

1 课程任务概述

实验内容由两大部分组成：

1）基本 SQL 操作＋DBMS 综合运用：8 学时（第 1-2 次上机），每次上机由指导老师及学生助教检查每个学生的任务完成情况和 DB 知识掌握情况；每次课后 1 天内提交电子版实验报告到指导教师邮箱。

2）小型数据库应用系统设计与开发：24 学时（第 3-8 次上机），其中，

第 3~4 次：选题、 ER 图设计、数据库功能模块设计、选择/学习使用开发工具；第 4 次课后 1 天内提交数据库初步设计报告（包括 ER 图、数据库模式设计、应用系统模块划分）到指导教师邮箱。

第 4~7 次：数据库应用系统开发。

第 7~8 次（18 周 8 学时）：当场检查。（检查必须在课内完成！请同学们抓紧时间，积极高效地完成实验任务！）

2 基本 SQL 操作

2.1 任务要求

1. 熟悉一种 DBMS 软件（Microsoft SQL Server 2008+ 或 MySQL 或 DM）的安装及使用（DM7 下载网址：http://www.dameng.com）。

2. 熟悉并掌握使用 SQL 语言进行：

- 数据库、模式、表的创建；
- 基本表上的增加、删除、修改、查询操作；
- 数据完整性约束的定义和检查。
- 视图操作。。

2.2 完成过程

1 创建数据库：DB*****(*为你的学号)

主要步骤：

（1） 新建查询

（2） 执行语句 create database DBU201414803

执行效果如图 2.1 所示

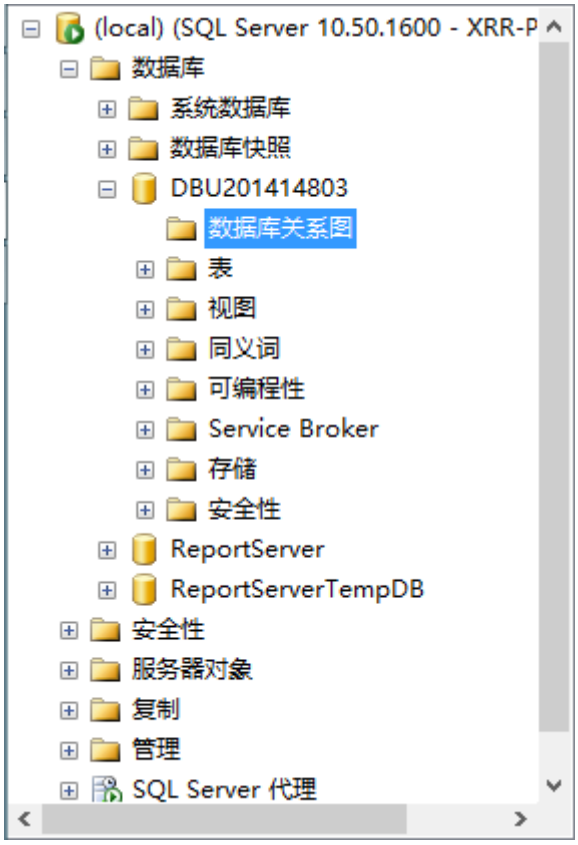


图 2.1 创建数据库结果

2 创建数据库模式

使用的 SQL 语句：create schema SPJ414803

执行结果如图 2.2 所示。

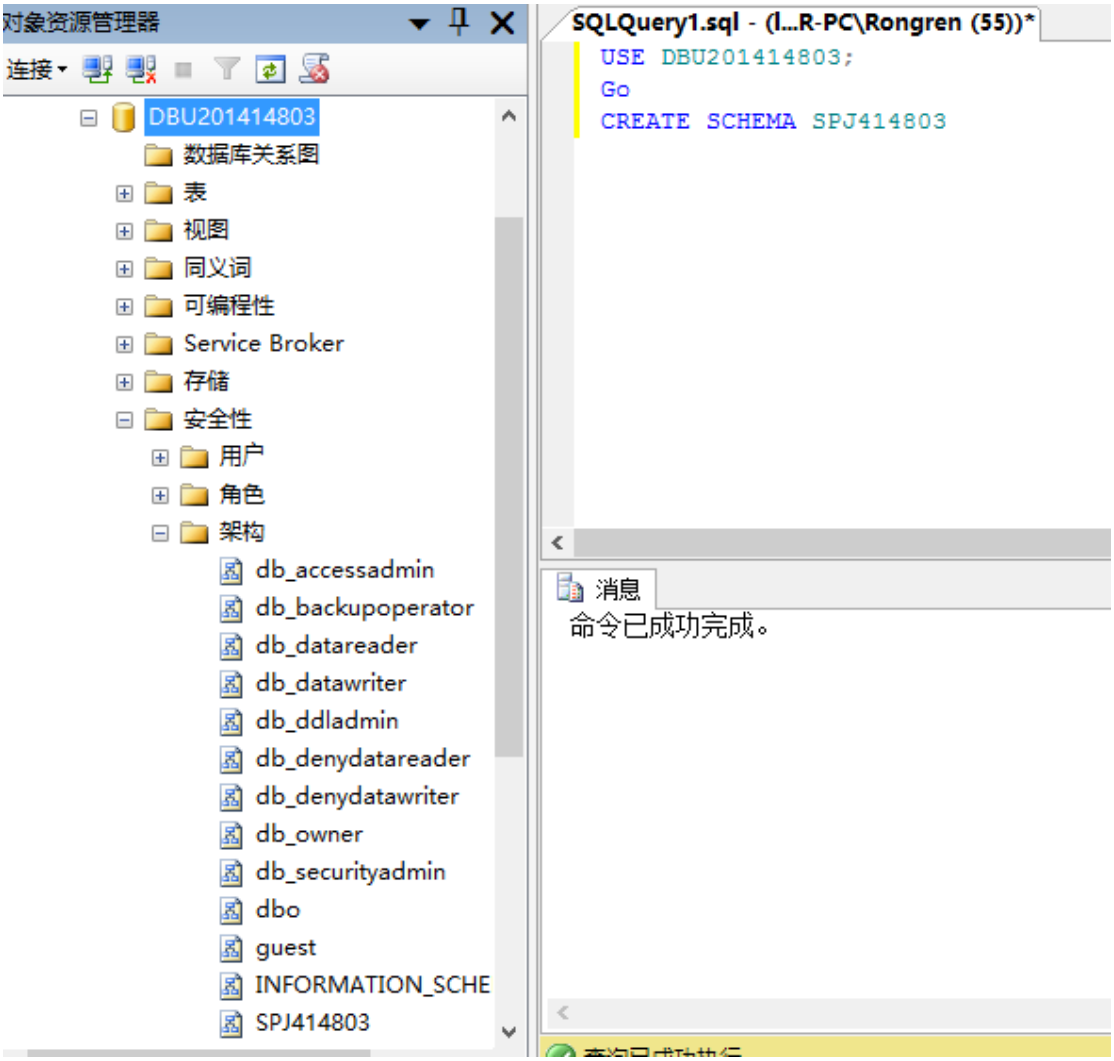


图 2.2 创建模式结果

3　创建 / 修改基本表

创建使用的 SQL 语句：

```
USE DBU201414803;
Go
CREATE TABLE SPJ414803.S
 (SNO CHAR(3) primary key,
 SNAME CHAR(7) not null,
 STATUS INT,
 CITY CHAR(5));
create table SPJ414803.P
 (PNO CHAR(3) primary key,
 PNAME char(7) not null,
 COLOR char(3),
 WEIGHT int default (10));
create table SPJ414803.J
```

```
 (JNO char(3) primary key,
 JNAME char(9) not null,
 CITY char(5));
create table SPJ414803.SPJ
 (SNO char(3),
 PNO char(3),
 JNO char(3),
 QTY int check (QTY>=1 and QTY <=10000),
 primary key (SNO,PNO,JNO),
 foreign key (SNO) references SPJ414803.S(SNO),
 foreign key (PNO) references SPJ414803.P(PNO),
 foreign key (JNO) references SPJ414803.J(JNO));
```

修改表结构使用的 SQL 语句：

```
USE DBU201414803;
go
alter table SPJ414803.P add Price float;
alter table SPJ414803.J add Pstart date not null;
alter table SPJ414803.J add Pend date;
alter table SPJ414803.J add constraint chk1 check (Pend>Pstart);
```

创建表执行结果如图 2.3 所示

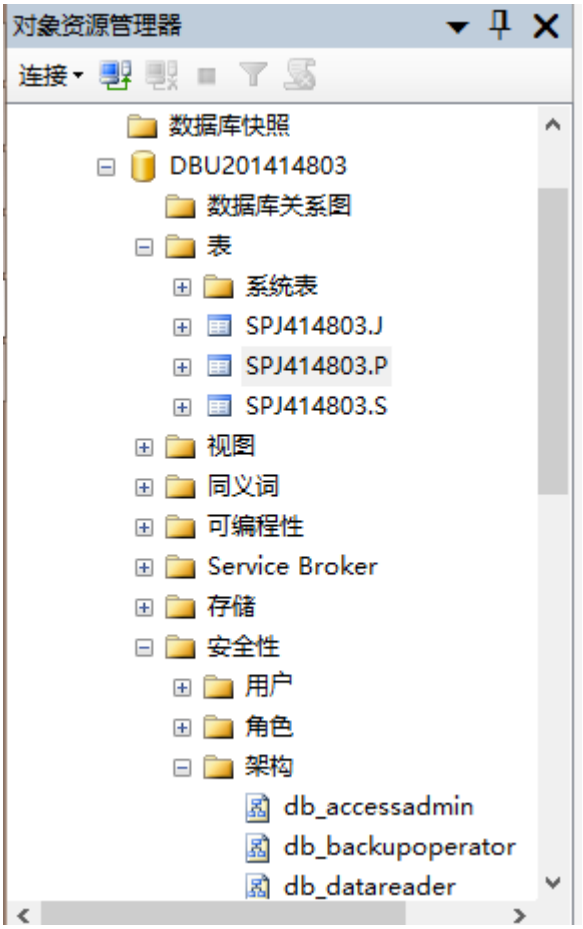


图 2.3 创建表结果

修改表结构执行结果如图 2.4 所示。

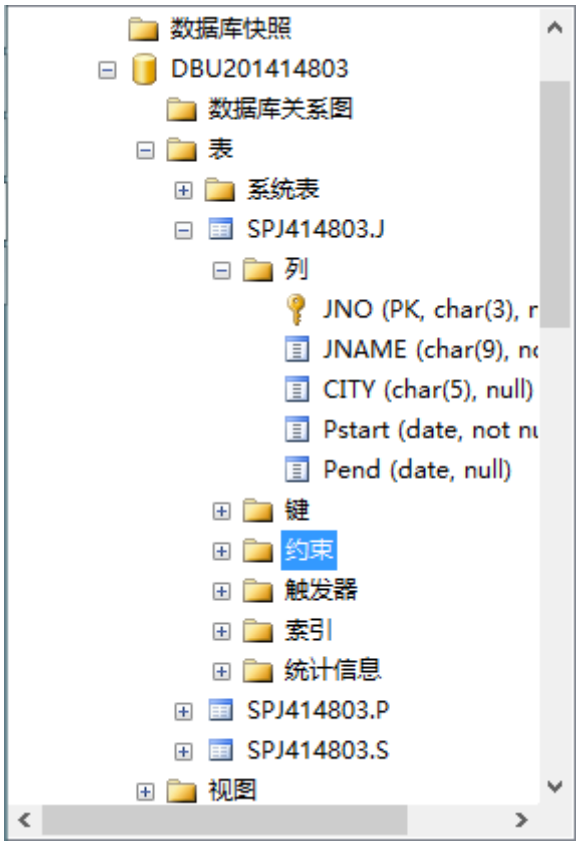


图 2.4 修改表结构结果

4 数据入库

数据入库使用的 SQL 语句：

```
use DBU201414803;
go
INSERT into SPJ414803.S
VALUES ('S1','精益',20,'天津'),
   ('S2','盛锡',10,'北京'),
   ('S3','东方红','30','北京'),
   ('S4','丰泰盛',20,'天津'),
   ('S5','为民',30,'上海');
insert into SPJ414803.P(PNO,PNAME,COLOR,WEIGHT)
   VALUES ('P1','螺母','红',12),
      ('P2','螺栓','绿',17),
      ('P3','螺丝刀','蓝',14),
      ('P4','螺丝刀','红',14),
      ('P5','凸轮','蓝',40),
      ('P6','齿轮','红',30);
insert into SPJ414803.J(JNO,JNAME,CITY,Pstart)
   VALUES ('J1','三建','北京','2017-05-03'),
      ('J2','一汽','长春','2017-05-03'),
```

```
        ('J3','弹簧厂','天津','2017-05-03'),
        ('J4','造船厂','天津','2017-05-03'),
        ('J5','机车厂','唐山','2017-05-03'),
        ('J6','无线电厂','常州','2017-05-03'),
        ('J7','半导体厂','南京','2017-05-03');
    insert into SPJ414803.SPJ
      VALUES ('S1','P1','J1',200),
        ('S1','P1','J3',100),
        ('S1','P1','J4',700),
        ('S1','P2','J2',100),
        ('S2','P3','J1',400),
        ('S2','P3','J2',200),
        ('S2','P3','J4',500),
        ('S2','P3','J5',400),
        ('S2','P5','J1',400),
        ('S2','P5','J2',100),
        ('S3','P1','J1',200),
        ('S3','P3','J1',200),
        ('S4','P5','J1',100),
        ('S4','P6','J3',300),
        ('S4','P6','J4',200),
        ('S5','P2','J4',100),
        ('S5','P3','J1',200),
        ('S5','P6','J2',200),
    ('S5','P6','J4',500);
```

批处理操作使用的 SQL 语句：

```
    USE DBU201414803;
    GO
    CREATE TABLE SPJ414803.BIG_SPJ
     (SNO char(3),
     PNO char(3),
     JNO char(3),
     QTY INT);
    INSERT INTO SPJ414803.BIG_SPJ
     SELECT *
     FROM SPJ414803.SPJ
     WHERE QTY>400;

    select *
    from SPJ414803.BIG_SPJ;
```

数据入库结果如图 2.5 和图 2.6 所示。

结果 消息

	SNO	SNAME	STATUS	CITY
1	S1	精益	20	天津
2	S2	盛锡	10	北京
3	S3	东方红	30	北京
4	S4	丰泰盛	20	天津
5	S5	为民	30	上海

	PNO	PNAME	COLOR	WEIGHT	Price
1	P1	螺母	红	12	NULL
2	P2	螺栓	绿	17	NULL
3	P3	螺丝刀	蓝	14	NULL
4	P4	螺丝刀	红	14	NULL
5	P5	凸轮	蓝	40	NULL
6	P6	齿轮	红	30	NULL

	JNO	JNAME	CITY	Pstart	Pend
1	J1	三建	北京	2017-05-03	NULL
2	J2	一汽	长春	2017-05-03	NULL
3	J3	弹簧...	天津	2017-05-03	NULL
4	J4	造船...	天津	2017-05-03	NULL
5	J5	机车...	唐山	2017-05-03	NULL
6	J6	无线...	常州	2017-05-03	NULL
7	J7	半导...	南京	2017-05-03	NULL

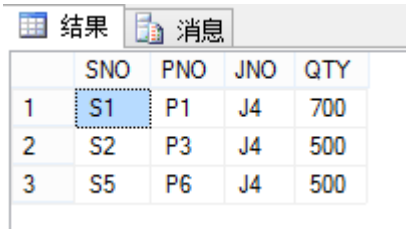
	SNO	PNO	JNO	QTY
1	S1	P1	J1	200
2	S1	P1	J3	100

图 2.5 数据入库结果

	SNO	PNO	JNO	QTY
2	S1	P1	J3	100
3	S1	P1	J4	700
4	S1	P2	J2	100
5	S2	P3	J1	400
6	S2	P3	J2	200
7	S2	P3	J4	500
8	S2	P3	J5	400
9	S2	P5	J1	400
10	S2	P5	J2	100
11	S3	P1	J1	200
12	S3	P3	J1	200
13	S4	P5	J1	100
14	S4	P6	J3	300
15	S4	P6	J4	200
16	S5	P2	J4	100
17	S5	P3	J1	200
18	S5	P6	J2	200
19	S5	P6	J4	500

图 2.6 数据入库结果

批处理结果如图 2.7 所示。



	SNO	PNO	JNO	QTY
1	S1	P1	J4	700
2	S2	P3	J4	500
3	S5	P6	J4	500

图 2.7 批处理结果

5 查询操作及查询计划分析

使用的 SQL 语句：

```
use DBU201414803;
go
select distinct SNO
from SPJ414803.SPJ
where JNO='J1';

select distinct SNO
from SPJ414803.SPJ
where JNO='J1' and PNO='P1';

select distinct SPJ414803.SPJ.SNO
```

```
from SPJ414803.SPJ, SPJ414803.P
where SPJ.PNO=P.PNO and SPJ414803.SPJ.JNO='J1' and COLOR='红';

select distinct JNO
from SPJ414803.SPJ
where JNO not in(
 select JNO
 from SPJ414803.SPJ,SPJ414803.S,SPJ414803.P
 where SPJ.PNO=P.PNO and SPJ.SNO=S.SNO and COLOR='红' and CITY='天津');

SELECT DISTINCT JNO
FROM SPJ414803.SPJ SX
WHERE NOT EXISTS
(SELECT *
FROM SPJ414803.SPJ SY
WHERE SY.SNO = 'S1' AND
NOT EXISTS
(SELECT *
FROM SPJ414803.SPJ SZ
WHERE SZ.SNO=SY.SNO AND
SZ.PNO=SY.PNO));

SELECT JNO
FROM SPJ414803.SPJ
WHERE PNO='P1'    AND
 JNO IN (
 SELECT JNO
 FROM SPJ414803.SPJ
 WHERE PNO='P2');
```

查询执行结果如图 2.8 和图 2.9 所示。

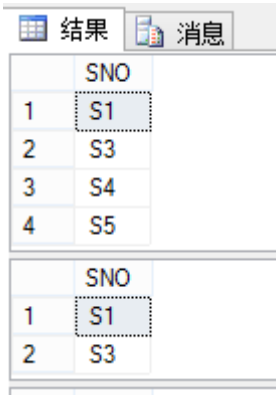
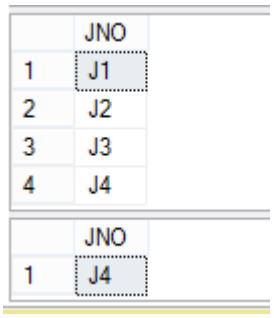


图 2.8 查询结果



	JNO
1	J1
2	J2
3	J3
4	J4

	JNO
1	J4

图 2.9 查询结果

违反规则的插入操作结果如图 2.10 所示。

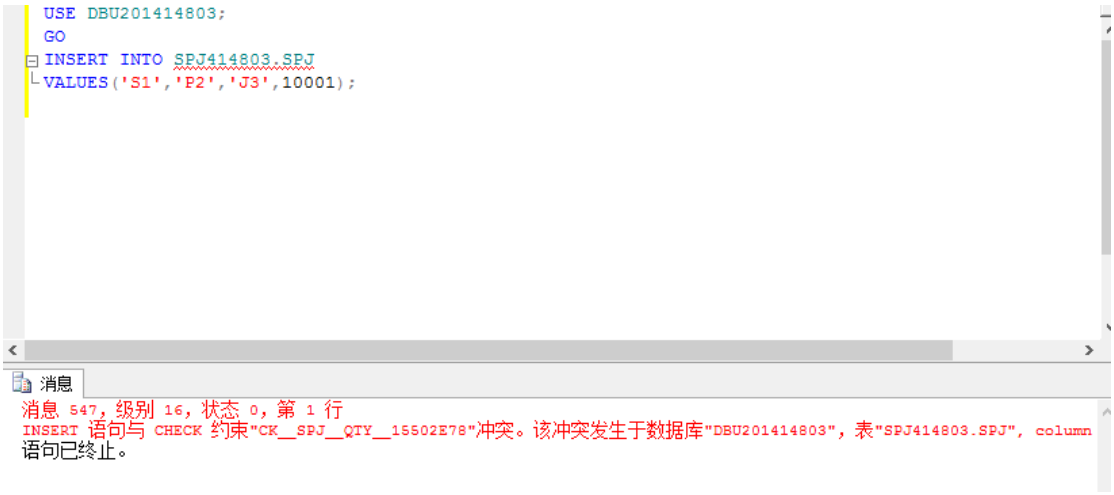


图 2.10 违反约束规则的插入操作

查询的物理路径如图 2.11 所示，方法是右键->显示预估的执行计划，然后就可以在执行计划窗口中看到结果了。

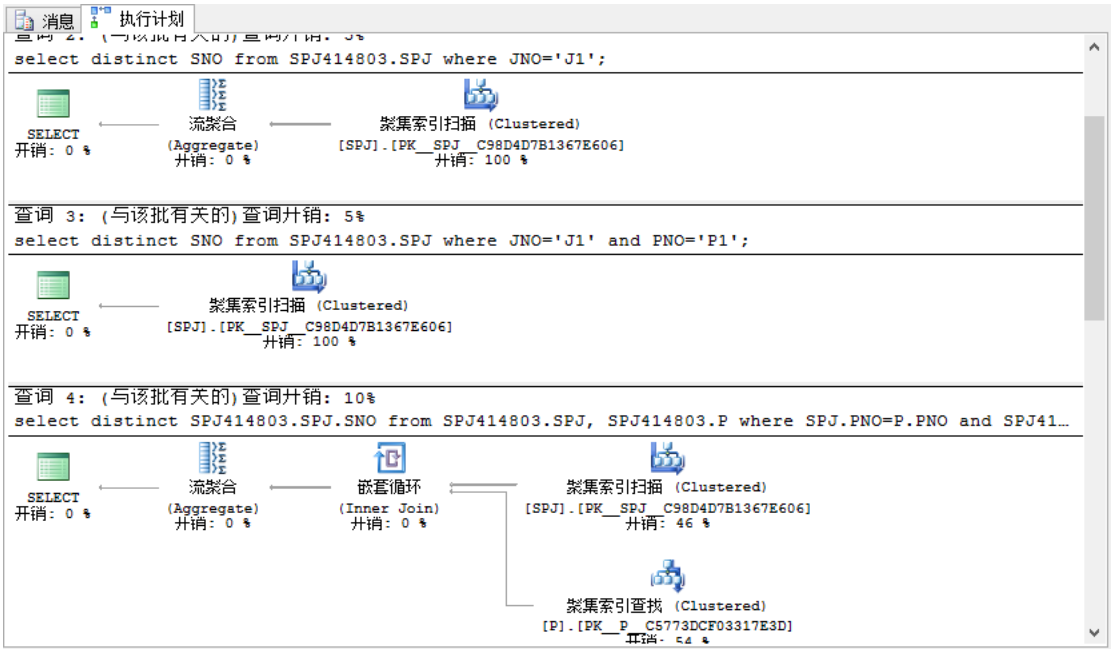


图 2.11 执行计划

6 查询 / 数据操纵 SQL 语句训练

使用的 SQL 语句：

```
use DBU201414803;
go

select SNAME,CITY
from SPJ414803.S;

select PNAME,COLOR,WEIGHT
from SPJ414803.P;

select JNO
from SPJ414803.SPJ
where SNO='S1';

select PNAME,QTY
from SPJ414803.SPJ,SPJ414803.P
where SPJ.PNO=p.PNO;

select PNO
from SPJ414803.SPJ,SPJ414803.S
where SPJ.SNO=s.SNO and CITY='上海';

select JNAME
from SPJ414803.SPJ,SPJ414803.S,SPJ414803.J
where SPJ.SNO=s.SNO and S.CITY='上海' and SPJ.JNO=j.JNO;
```

```
select JNO
from SPJ414803.J
where not exists
 ((SELECT * FROM SPJ414803.S WHERE CITY='天津' AND EXISTS
          (SELECT * FROM SPJ414803.SPJ WHERE SNO=S.SNO AND    JNO=J.JNO)));

update SPJ414803.P
set COLOR='蓝'
where COLOR='红';

UPDATE SPJ414803.SPJ
SET SNO='S3'
WHERE JNO='J4' AND SNO='S5' AND PNO='P6';

delete
from SPJ414803.SPJ
where SNO='S2';
delete
from SPJ414803.S
where SNO='S2';

insert into SPJ414803.SPJ
values('S2','J6','P4',200);
```

查询“厂”字结尾的 SQL 语句：

```
use DBU201414803;
go
select SPJ.PNO,p.PNAME,J.JNAME
from SPJ414803.SPJ,SPJ414803.J,SPJ414803.P
where SPJ.JNO=j.JNO and spj.PNO=p.PNO and JNAME like '%厂';
```

其他 SQL 语句：

```
use DBU201414803;
go
--一年内完工的项目
select SPJ.JNO,DATEDIFF(DAY,Pstart,Pend)
from SPJ414803.SPJ,SPJ414803.J
where SPJ.JNO=j.JNO and DATEDIFF(DAY,Pstart,Pend)<365;
--查出工程所用零件都是由异地供应商供应的工程号
select JNO
from SPJ414803.J jx
where CITY not in
 (select S.CITY
```

```
    from SPJ414803.SPJ,SPJ414803.S,SPJ414803.J jy
    where SPJ.JNO=jy.JNO and SPJ.SNO=S.SNO and SPJ.JNO=jx.JNO)
--求每个供应商 2016 年的零件销售额
select SUM(QTY*Price)
from SPJ414803.SPJ,SPJ414803.P
where SPJ.PNO=P.PNO
group by SNO
```

习题 5 执行结果如图 2.12 所示，因为设置的依赖项已被删除，所以最后一条插入语句不能执行。

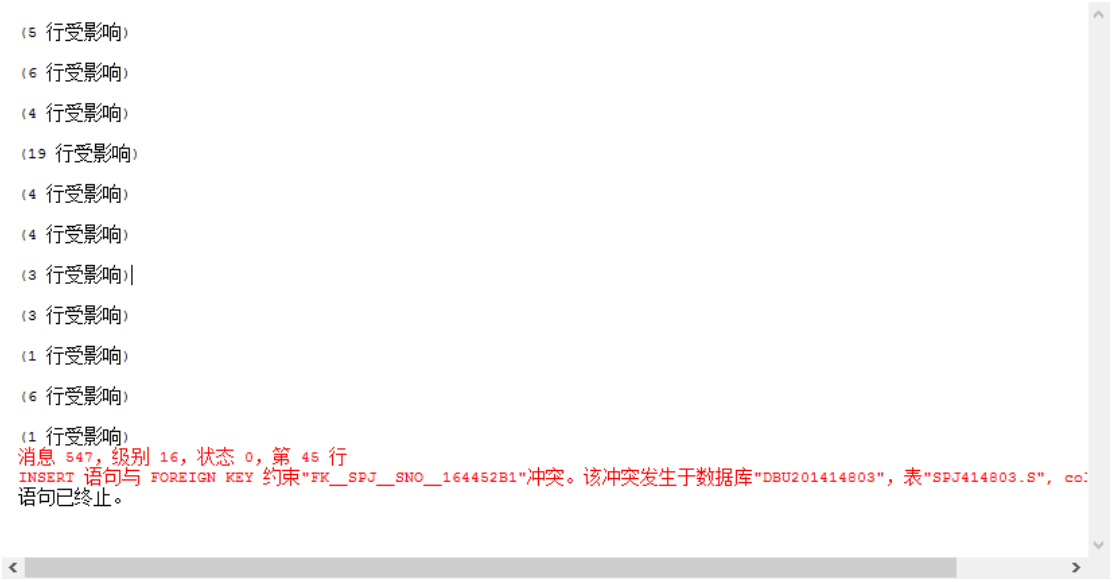


图 2.12 习题五执行结果

厂字查询结果如图 2.13 所示。

结果 消息

	PNO	PNAME	JNAME
1	P1	螺母	弹簧厂
2	P1	螺母	造船厂
3	P6	齿轮	造船厂
4	P6	齿轮	弹簧厂
5	P6	齿轮	造船厂
6	P2	螺栓	造船厂

图 2.13 厂 结尾

异地供应查询结果如图 2.14 所示

	JNO
1	J2
2	J5
3	J6
4	J7

图 2.14 异地供应查询结果

7 视图定义和操作

创建“三建”视图的 SQL 语句：

```
use DBU201414803;
go

create view 三建 as
 select SPJ.PNO,SPJ.SNO,QTY
 from SPJ414803.SPJ,SPJ414803.J
 where JNAME='三建' and SPJ.JNO=J.JNO
```

视图查询的 SQL 语句：

```
SELECT [PNO]
      ,[QTY]
  FROM [DBU201414803].[dbo].[三建]
GO

SELECT [PNO]
      ,[SNO]
      ,[QTY]
  FROM [DBU201414803].[dbo].[三建]
  where SNO='S1'
GO
```

更新视图的 SQL 语句：

```
use DBU201414803;
go
update dbo.三建
set QTY=666
where PNO='P1' and SNO='S1'
```

创建视图的结果如图 2.15 所示。

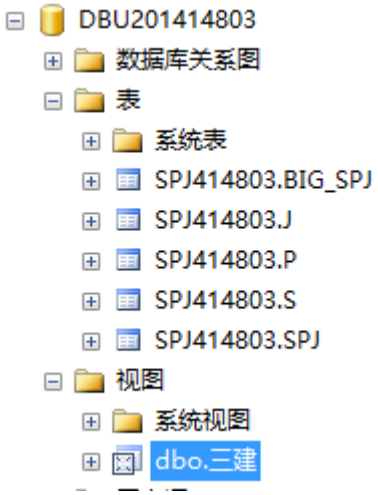


图 2.15 创建视图

视图查询结果如图 2.16 所示。

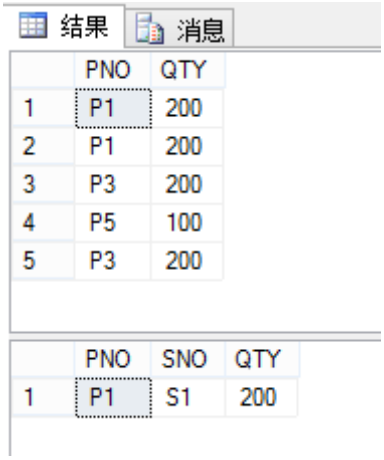


图 2.16 视图查询结果

查询更新后的视图的结果如图 2.17 所示，这是一个可更新的视图。

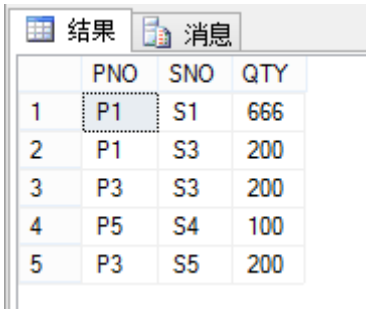


图 2.17 更新后的视图查询

2.3 任务总结

实验中遇到许多或大或小的问题，其中部分如下：

1. 无法使用数据库软件，通信被阻止，关闭防护软件的防火墙后仍然没有解决，最后卸载掉了防护软件，并且关闭 windows 自带的防火墙。
2. 创建模式后找不到模式在哪里。向同学求助后得知是在安全性-架构中。
3. 不知道如何在某一个模式下建表。向同学求助后得知只要在表名前加“模式名.”就可以了。
4. 修改表结构后，再插入课本的数据无法插入。因为 J 表要增加了一个不能为空的元组，解决办法是在 J 表的数据中都加入了一个日期数据。
5. 从视图查询数据时报错对象不存在。解决办法是在左侧对象资源管理器中右键视图名-编写视图脚本-select，在自动生成的脚本上进行修改。
6. 日期不能使用减法操作。查询资料后得知应该使用 datediff()函数。

3 DBMS 综合应用

3.1 任务要求

1）在第一次实验的基础上，通过移动公司手机信息数据库这个实例，使用 Microsoft SQL SERVER、DM 或 MySQL 进行 DBMS 综合运用；

2）强调用户管理、权限控制及完整性控制等操作。

3）练习 DBMS 上的完全备份方式：数据和日志文件的脱机备份、系统的备份功能。

3.2 完成过程

大致过程：1. 建表；2. 创建角色和用户；3. 给角色授权，给用户指派角色；4. 创建触发器；5.分别以不同的用户插入数据；6：根据题目要求，进行增删查改等多种操作

详细过程：

1. 以系统管理员的身份登录：考虑数据完整性要求，建表，建完整性约束；

   使用的 SQL 语句：

```
use MCPI;
go
create table BHall(
 Hno char(6) primary key,
 Haddr char(20),
 Hmanager char(6)
);
```

```
create table SaleRecord(
 Sno char(6) primary key,
 Ptype char(6),
 Sdate date,
 Salesman char(6),
 Hno char(6),
 foreign key (Hno) references BHall(Hno)
);
create table cellphone(
 Ptype char(6),
 Pdate date,
 primary key(Ptype, Pdate),
 Pprice float,
);
create table cellUser(
 pnumber char(12) primary key,
 username char(10) not null,
 uid char(20) not null,
 email char(24) check(email like '%@%.%')
);
create table phoneMFee(
 pnumber char(12),
 mdate date,
 primary key(pnumber, mdate),
 mfee1 float,
 mfee2 float,
 mfee3 float,
 mtotal as mfee1+mfee2+mfee3,
 foreign key (pnumber) references cellUser(pnumber)
);
create table phonePaid(
 pnumber char(12),
 pdate date,
 primary key(pnumber, pdate),
 foreign key (pnumber) references cellUser(pnumber),
 paid float
);
create table phoneFeeHistory(
 pnumber char(12),
 pmonth date,
 primary key(pnumber, pmonth),
 foreign key (pnumber) references cellUser(pnumber),
 fee float,
 CType int check(CType=1 or CType=0),
```

pbalance float
);
建表的结果如图 3.1 建表结果图 3.1 所示，7 张表全部创建完成：

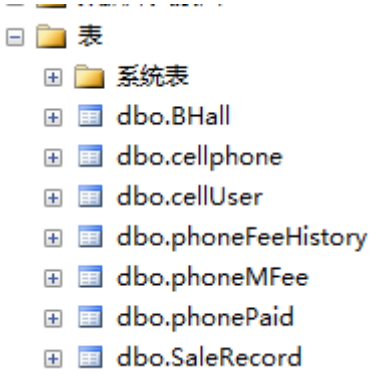


图 3.1 建表结果

2. 按照应用场景 2 中的表访问要求，对营业厅操作员和用户进行权限的授予与回收。

2.1 授予角色 rolem（营业厅管理员角色）：表 1-7 的所有权限，并允许权限的转授，将该角色授予 1 个管理员用户 manager；

2.2 授予角色 role1（营业厅销售员角色）：表 1-3 的增删查改权限，将该角色授予 3 个销售员用户 sales1,sales2,sales3；

2.3 授予角色 role2（营业厅业务员角色）：表 4-6 的增查权限 及 各表除主码以外的属性列上的删改权限，表 7 的查询权限，将该角色授予 2 个业务操作员用户 oper1,oper2；

2.4 授予角色 roleu（手机用户角色）：表 3-7 的查询权限，将该角色授予手机用户 user1。

实现过程：先创建 rolem,role1,role2,rolu4 个角色，再创建 7 个用户，然后分别对角色进行授权，最后把角色指派给对应的用户。

具体的 SQL 语句：

创建角色：

```
create role rolem;
create role role1;
create role role2;
create role roleu;
```

创建用户：

```
create user manager without login;
create user sales1 without login;
create user sales2 without login;
create user sales3 without login;
create user oper1 without login;
create user oper2 without login;
```

```
create user user1 without login;
```
角色授权：
```
grant all privileges
on object::BHall
to rolem
with grant option
grant all privileges
on object::cellphone
to rolem
with grant option
grant all privileges
on object::phoneFeeHistory
to rolem
with grant option
grant all privileges
on object::phoneMFee
to rolem
with grant option
grant all privileges
on object::phonePaid
to rolem
with grant option
grant all privileges
on object::SaleRecord
to rolem
with grant option

grant insert,delete,select,update
on object::BHall
to role1
grant insert,delete,select,update
on object::cellphone
to role1
grant insert,delete,select,update
on object::SaleRecord
to role1

grant insert,select
on object::cellUser
to role2
grant insert,select
on object::phoneMFee
to role2
grant insert,select
```

```
on object::phonePaid
to role2
grant select
on object::phoneFeeHistory
to role2
grant update(Haddr,Hmanager)
on object::BHall
to role2
grant update(Ptype,Sdate,Salesman,Hno)
on object::SaleRecord
to role2
grant update(Pprice)
on object::cellphone
to role2

grant select
on object::cellphone
to roleu
grant select
on object::cellUser
to roleu
grant select
on object::phoneMFee
to roleu
grant select
on object::phonePaid
to roleu
grant select
on object::phoneFeeHistory
to roleu
```

用户指派角色：

```
exec sp_addrolemember 'rolem','manager'
exec sp_addrolemember 'role1','sales1'
exec sp_addrolemember 'role1','sales2'
exec sp_addrolemember 'role1','sales3'
exec sp_addrolemember 'role2','oper1'
exec sp_addrolemember 'role2','oper2'
exec sp_addrolemember 'roleu','user1'
```

创建后的角色如图 3.2 和图 3.3 用户如和所示：

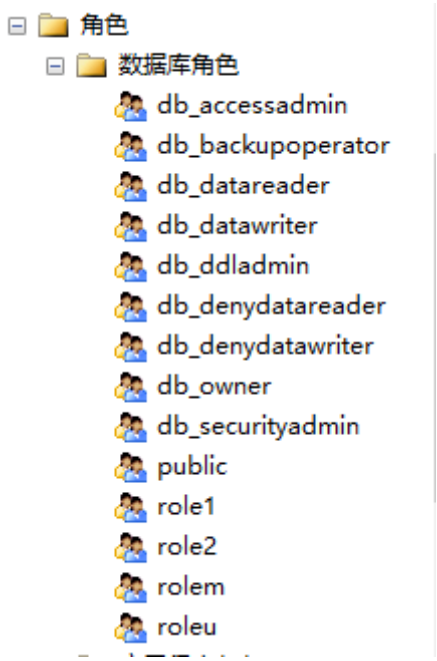


图 3.2 角色

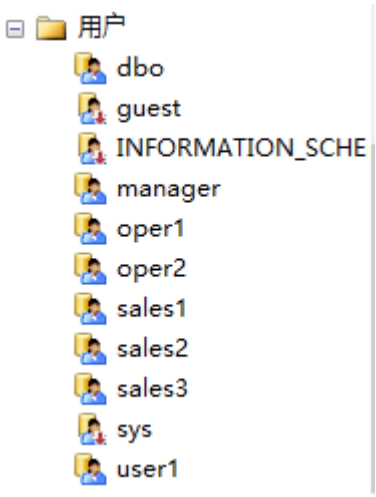


图 3.3 用户

3. 以营业厅销售员和业务员的身份各自登录：分别实现对表 1-6 的数据插入。数据插入使用 insert into 语句即可，以不同的身份需要在最前方加上 execute as user='user'这一语句。插入表 1-3 的 SQL 语句如下：

```
execute as user='sales1'
insert into BHall
values('001','珞瑜路 1037','张三'),
('002','保卫处','李四'),
('003','东四食堂','王五')

insert into SaleRecord
values('001','s101','2011-1-1','小王','001'),
 ('002','n302','2011-3-3','小李','001'),
 ('003','n302','2011-3-5','小王','002')
```

```
insert into cellphone
 values('s101','2011-1-1',1599),
 ('n302','2011-3-3',1999),
 ('n302','2011-3-5',1899)
```

插入表 4-6 的 SQL 语句如下：

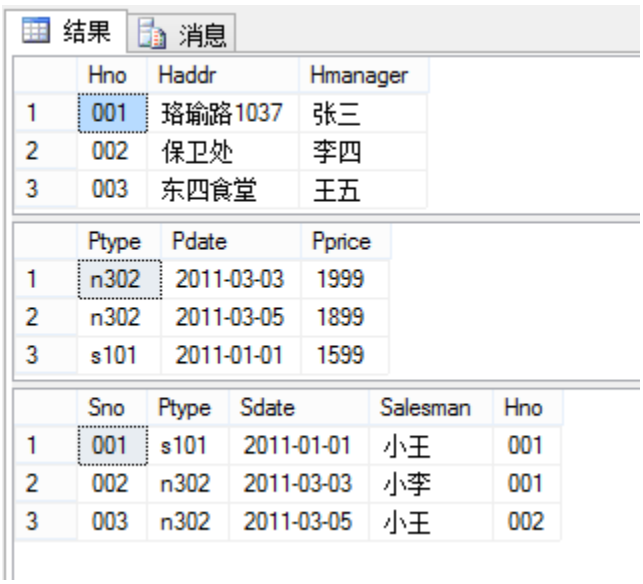
```
execute as user='oper1'
insert into cellUser
 values('15122223333','陈小兵','382325677895346782','ifxnbk@gmail.com'),
 ('15144334433','李欣博','382326735895346782','lixbbo@live.com'),
 ('15144222233','王世明','382326711111346782','whuimk@live.com'),
 ('15144111333','陈秋鹤','382387657895346782','ifqqhe@gmail.com'),
 ('15267367933','能世木','309785835895346782','nguimu@live.com')

insert into phonePaid
 values('15122223333','2017-01-01',100)
insert into phonePaid
 values('15144222233','2017-02-01',100)

insert into phoneMFee(pnumber,mdate,mfee1,mfee2,mfee3)
 values('15122223333','2017-02-02',10,10,10);
insert into phoneMFee(pnumber,mdate,mfee1,mfee2,mfee3)
 values('15122223333','2017-03-02',11,12,13)

insert into phoneMFee(pnumber,mdate,mfee1,mfee2,mfee3)
 values('15144222233','2017-03-02',40,40,40);
```

插入数据后的表 1-3 如所示，4-6 如所示：



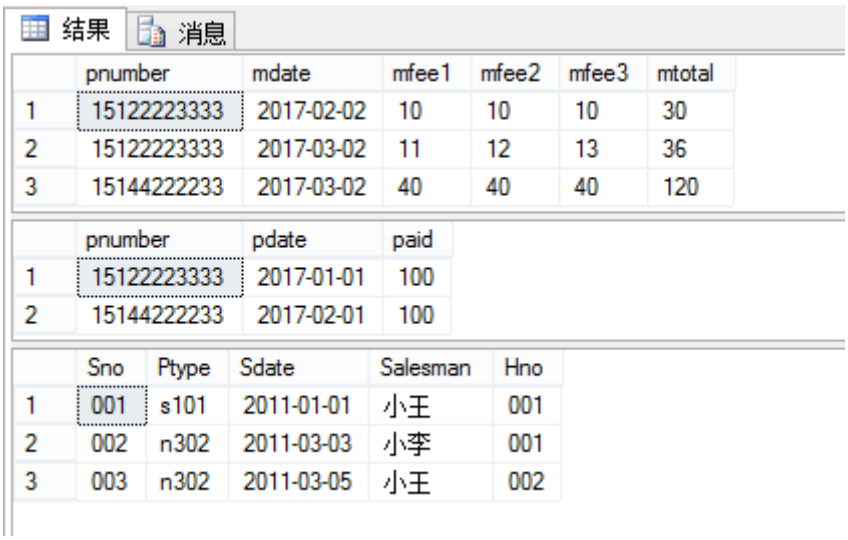
结果 消息

	Hno	Haddr	Hmanager
1	001	珞瑜路1037	张三
2	002	保卫处	李四
3	003	东四食堂	王五

	Ptype	Pdate	Pprice
1	n302	2011-03-03	1999
2	n302	2011-03-05	1899
3	s101	2011-01-01	1599

	Sno	Ptype	Sdate	Salesman	Hno
1	001	s101	2011-01-01	小王	001
2	002	n302	2011-03-03	小李	001
3	003	n302	2011-03-05	小王	002

图 3.4 表 1-3 插入数据后的查询结果



结果 消息

	pnumber	mdate	mfee1	mfee2	mfee3	mtotal
1	15122223333	2017-02-02	10	10	10	30
2	15122223333	2017-03-02	11	12	13	36
3	15144222233	2017-03-02	40	40	40	120

	pnumber	pdate	paid
1	15122223333	2017-01-01	100
2	15144222233	2017-02-01	100

	Sno	Ptype	Sdate	Salesman	Hno
1	001	s101	2011-01-01	小王	001
2	002	n302	2011-03-03	小李	001
3	003	n302	2011-03-05	小王	002

图 3.5 表 4-6 插入数据后的查询结果

4. 由管理员创建触发器，并验证其执行效果：

4.1 触发器 1：每月底计算当月总话费，插入表 7 扣费记录；

4.2 触发器 2：每当用户充值时触发，插入表 7 缴费记录。

触发的条件是插入了一条数据，进行的操作是从表 7 中把这个号码的最新纪录取出来，算出新的余额，然后插入表 7。

触发器 1 的 SQL 语句如下：

```
create trigger trg1
on dbo.phoneMFee
for insert
as
begin
```

```
 declare @num char(12),@date date,@total float,@prebalance float;
 select @num=pnumber,@date=mdate,@total=mtotal from inserted
 if(not exists(
 select *
 from phoneFeeHistory
 where phoneFeeHistory.pnumber=@num))
 select @prebalance=0;
 else begin
       select @prebalance=( select pbalance from phoneFeeHistory   where pnumber=@num
and pmonth = (
       select MAX(pmonth) from    phoneFeeHistory    where pnumber = @num
       ) )
 end
 insert into phoneFeeHistory
 values(@num, @date, @total, 0, @prebalance-@total)
end
```

触发器 2 的 SQL 语句如下：

```
create trigger trg2
on dbo.phonePaid
for insert
as
begin
 declare @num char(12),@date date,@total float,@prebalance float;
 select @num=pnumber,@date=pdate,@total=paid from inserted
 if(not exists(
 select *
 from phoneFeeHistory
 where phoneFeeHistory.pnumber=@num))
 select @prebalance=0;
 else begin
       select @prebalance=( select pbalance from phoneFeeHistory   where pnumber=@num
and pmonth = (
       select MAX(pmonth) from    phoneFeeHistory    where pnumber = @num
       ) )
 end
 insert into phoneFeeHistory
 values(@num, @date, @total, 1, @prebalance+@total)
end
```

检验触发器的执行效果的方法是查询表 7 的数据，查询到的数据如图 3.6 所示，对比插入的数据，可以发现触发器已经被正确触发。

pnumber	pmonth	fee	CType	pbalance
15122223333	2017-01-01	100	1	100
15122223333	2017-02-02	30	0	70
15122223333	2017-03-02	36	0	34
15144222233	2017-02-01	100	1	100
15144222233	2017-03-02	120	0	-20

图 3.6 触发器触发结果

5. 以不同用户的身份登录，完成以下查询任务：

查询的 SQL 代码如下，尝试以不同的用户去执行查看执行结果：

```
--revert
execute as user='oper1'
--查询手机销售信息
select *
from SaleRecord

--手机用户档案信息
select *
from cellUser

--个人手机话费信息；
select x.pnumber, username, pbalance
from cellUser, phoneFeeHistory x
where cellUser.pnumber=x.pnumber and x.pmonth=(
select MAX(pmonth) from phoneFeeHistory y where x.pnumber=y.pnumber)

--根据手机号码查询用户信息
select *
from cellUser
where pnumber='xxxx'

--每种机型的销售情况
SELECT [Ptype]
      ,COUNT(*) num
  FROM [MCPI].[dbo].[SaleRecord]
  group by Ptype
GO

--欠费用户查询；
select x.pnumber, username, pbalance
from cellUser, phoneFeeHistory x
where cellUser.pnumber=x.pnumber and x.pmonth=(
```

select MAX(pmonth) from phoneFeeHistory y where x.pnumber=y.pnumber) and pbalance<0

在 dbo 的权限下执行结果如图 3.7 所示：

	Sno	Ptype	Sdate	Salesman	Hno
1	001	s101	2011-01-01	小王	001
2	002	n302	2011-03-03	小李	001
3	003	n302	2011-03-05	小王	002

	pnumber	username	uid	email
1	15122223333	陈小兵	382325677895346782	ifxnbk@gmail.com
2	15144111333	陈秋鹤	382387657895346782	ifqqhe@gmail.com
3	15144222233	王世明	382326711111346782	whuimk@live.com
4	15144334433	李欣博	382326735895346782	lixbbo@live.com
5	15267367933	能世木	309785835895346782	nguimu@live.com

	pnumber	username	pbalance
1	15144222233	王世明	-20
2	15122223333	陈小兵	34

	pnumber	username	uid	email

	Ptype	num
1	n302	2
2	s101	1

	pnumber	username	pbalance
1	15144222233	王世明	-20

图 3.7dbo 查询结果

sales1 用户执行查询语句的结果和消息如图 3.8 和图 3.9 所示：

结果 消息

	Sno	Ptype	Sdate	Salesman	Hno
1	001	s101	2011-01-01	小王	001
2	002	n302	2011-03-03	小李	001
3	003	n302	2011-03-05	小王	002

	Ptype	num
1	n302	2
2	s101	1

图 3.8sales1 查询结果

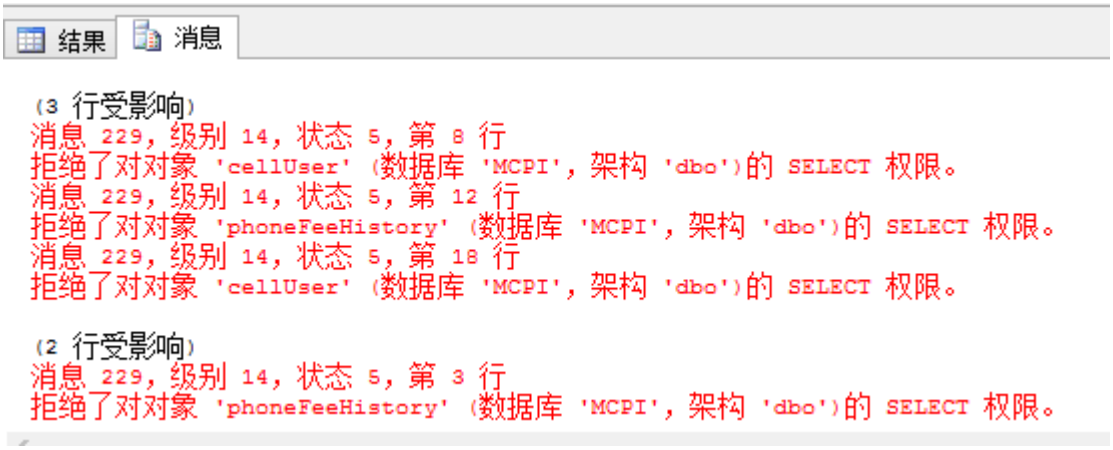


图 3.9sales1 查询的提示消息

可以看出不同权限的用户可以查询到的信息是不同的。

6. 在 3.的基础上进行权限的转授予与回收，并自行设计一些 SQL 语句，验证安全性操作的效果。

权限回收的 SQL 语句如下：

```
revoke select
on BHall from rolem cascade
```

执行后查询 rolem 的权限如图 3.10 所示，已经没有了 select 权限。

结果 消息

	Owner	Object	Grantee	Grantor	ProtectType	Action	Column
1	dbo	BHall	rolem	dbo	Grant_WGO	Delete	.
2	dbo	BHall	rolem	dbo	Grant_WGO	Insert	.
3	dbo	BHall	rolem	dbo	Grant_WGO	References	(All+New)
4	dbo	BHall	rolem	dbo	Grant_WGO	Update	(All+New)
5	dbo	cellphone	rolem	dbo	Grant_WGO	Delete	.
6	dbo	cellphone	rolem	dbo	Grant_WGO	Insert	.

图 3.10 收回权限结果

尝试插入违反约束条件的数据时，语句不能执行：

如以下语句：

```
insert into phonePaid
    values('15342223333','2017-01-01',100)
```

执行结果如图 3.11 所示，因为该号码是没有在表 celluser 中记录的。

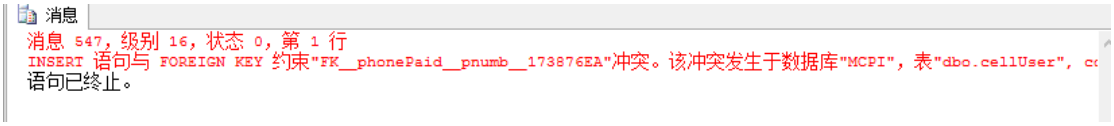


图 3.11 为不存在的号码插入付费记录

以下语句：

```
insert into SaleRecord
```

values('005','s101','2011-1-1','小王','011')

执行结果如图 3.12 所示，因为所依赖的 Hno 是没有在 BHall 中记录的。

图 3.12 为不存在营业厅插入销售记录

以下语句：

```
insert into cellUser
    values('15122335333','陈兵','382325655555346782','ifxnbcom')
```

执行结果如图 3.13 所示，因为 email 的格式错误。

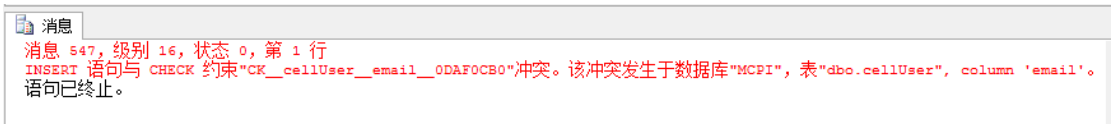


图 3.13 插入错误的 email 格式

7. 练习 DBMS 上的完全备份方式：数据和日志文件的脱机备份、系统的备份功能。

   在数据库名称上右击->任务，看到备份选项，然后就可以根据需要进行备份了。对 MCPI 数据库的备份执行结果如图 3.14 所示。

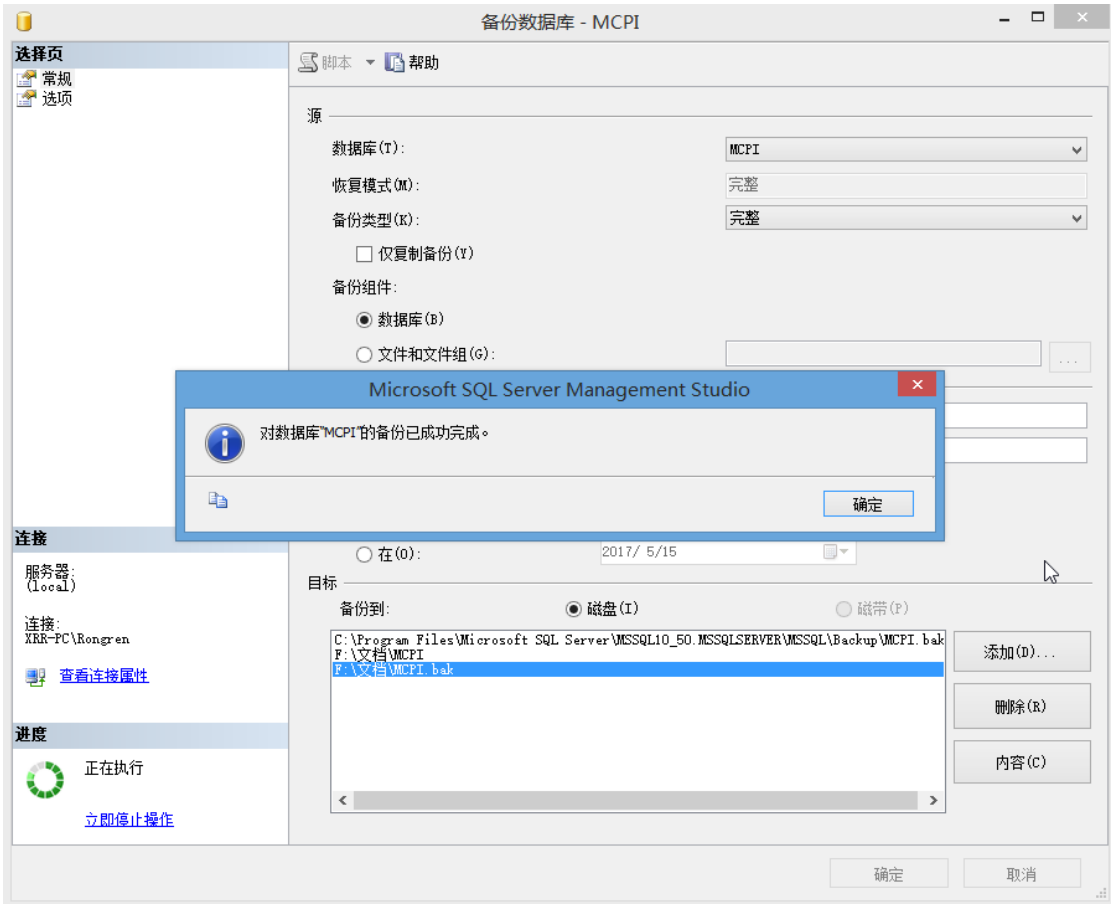


图 3.14 数据库备份

3.3 任务总结

实验中遇到了不少的问题，其中一部分记录如下：

1. 因为先看到第一个要求，创建三种不同的用户，只是简单的创建了用户并授权，没有意识到应该创建不同的角色，再把角色指派给用户，看到第三个要求才明白。
2. 创建触发器总是有问题。从网上查找有关 SQL server 的触发器的用法的博客，发现触发器的用法还是挺复杂的，要声明变量，select 赋值，与之前用过的编程语言有了比较相似的地方，最后还是向同学请教，用了嵌套的查询来实现了触发器。
3. 检验触发器的时候，因为一开始的插入语句是一条语句中插入多条数据，所以触发器的执行不完整，只对一部分数据作出了反应，后来修改出入语句，每次只插入一条记录，解决了问题。
4. 查询个人手机话费信息的时候，一开始用了 group by，根据 phoneFeeHistory 中号码进行分组，再按月份排序选出最新的数据就是话费信息，但执行的时

候发现有错误，因为逻辑不清楚，要选择的内容没有指定好，后来改变了策略，不需要分组，采用自身表的连接，选出与当前号码相同的记录中的最新日期，再根据这个日期和号码选出话费信息。

通过这次试验，自己对触发器，用户角色的权限控制有了直观的认识，提高了自己对 SQL 语言的使用能力。

4 小型数据库应用系统设计与开发

4.1 系统设计目标

设计一个工资管理系统，以多张表的形式保存数据，管理员登陆后可以增删查改所有的数据，没有其他的用户。

4.2 需求分析

1) 系统功能的基本要求：

（1） 员工每个工种基本工资的设定

（2） 加班津贴管理，根据加班时间和类型给予不同的加班津贴；

（3） 按照不同工种的基本工资情况、员工的考勤情况产生员工的每月的月工资；

（4） 员工年终奖金的生成，员工的年终奖金计算公式＝（员工本年度的工资总和＋津贴的总和）/12；

（5） 企业工资报表。能够查询单个员工的工资情况、每个部门的工资情况、按月的工资统计，并能够打印；

2) 数据库要求：在数据库中至少应该包含下列数据表：

（1） 员工考勤情况表；

（2） 员工工种情况表，反映员工的工种、等级，基本工资等信息；

（3） 员工津贴信息表，反映员工的加班时间、加班类别、加班天数、津贴情况等；

（4） 员工基本信息表

（5） 员工月工资表。

3) 数据完整性需求：

（1） 姓名，名称等用 char(10)表示，所有的金额用 float 型表示，编号用 char(10)表示。

（2） 总工资由基本工资、津贴和考勤信息计算得到，不需要主动插入，奖金信息同样如此。

4) 数据字典：

数据库中主要的数据及其类型如表 4.1 所示

表 4.1 数据字典

数据项	数据类型	说明
员工编号	char(10)	
员工姓名	char(10)	

员工性别	char（3）	“男”或“女”
员工电话	char(12)	
员工住址	char(30)	
年龄	int	[10,120]
工种编号	char(10)	
工种名称	char(10)	
工资	float	正数
月份	int	[1,12]
加班天数	int	[0,31]
缺勤次数	int	[0,31]

4.3 总体设计

系统分为 6 张表，分别是员工基本信息表，工种表，考勤表，津贴表，员工工资表和年终奖金表。

系统的功能模块如图 4.1 所示：

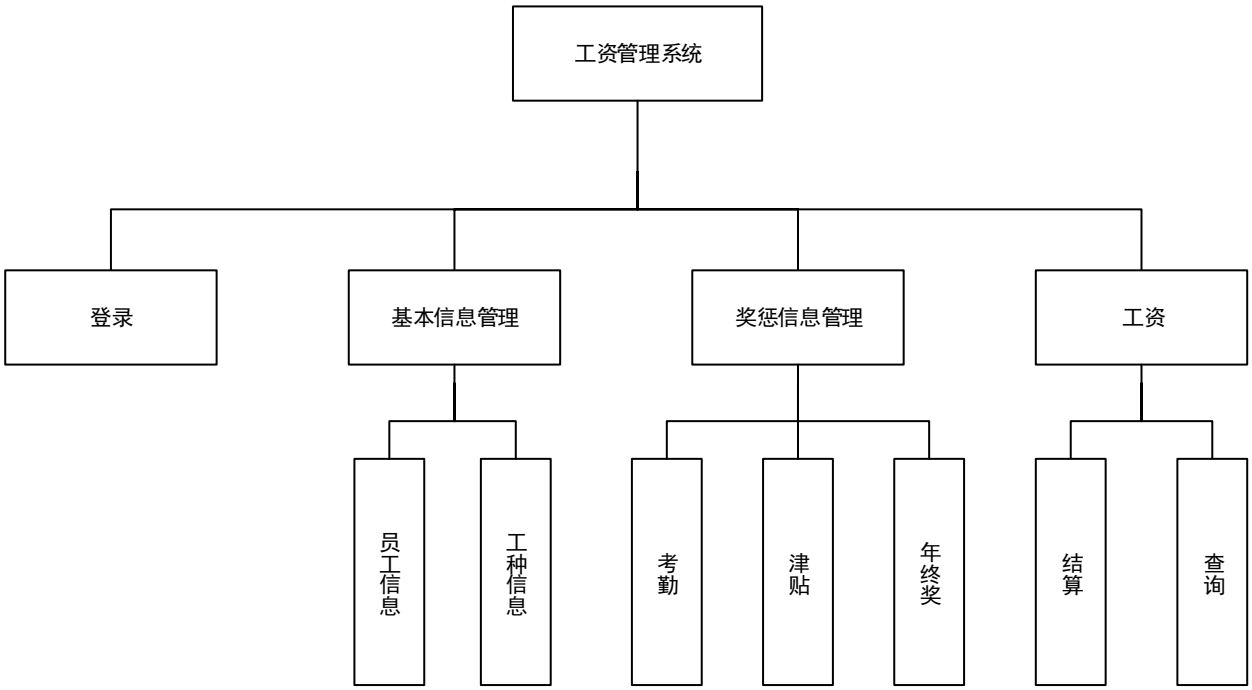


图 4.1 系统功能模块

4.4 数据库设计

数据库中主要有员工、工种、考勤、津贴、工资、年终奖几种实体，每种实体的属性如图 4.2、图 4.3、图 4.4、图 4.5、图 4.6 和图 4.7 所示，实体之间的 ER 图如图 4.8 所示，为了间接，只在 ER 图的实体上画出了主码。

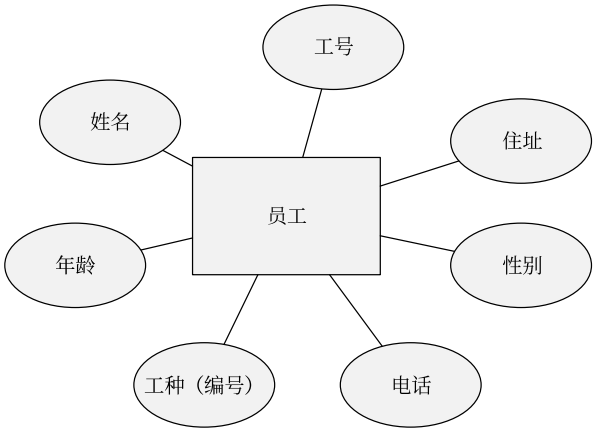


图 4.2 员工实体

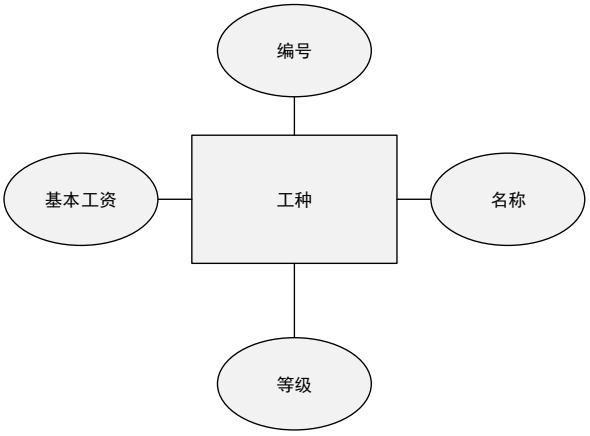


图 4.3 工种实体

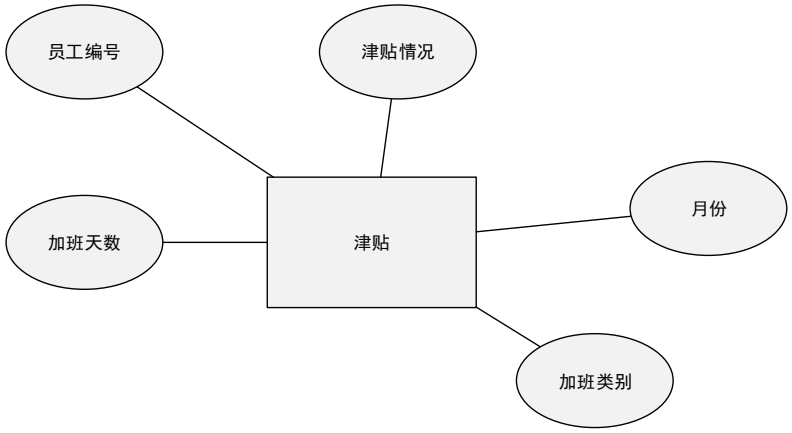


图 4.4 津贴实体

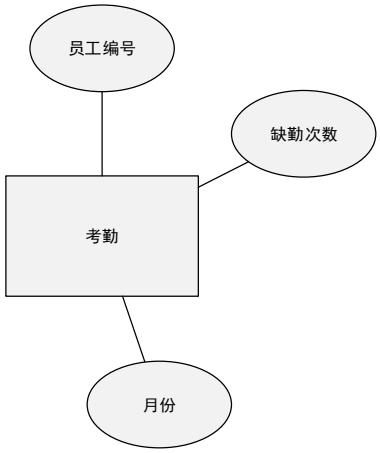


图 4.5 考勤实体

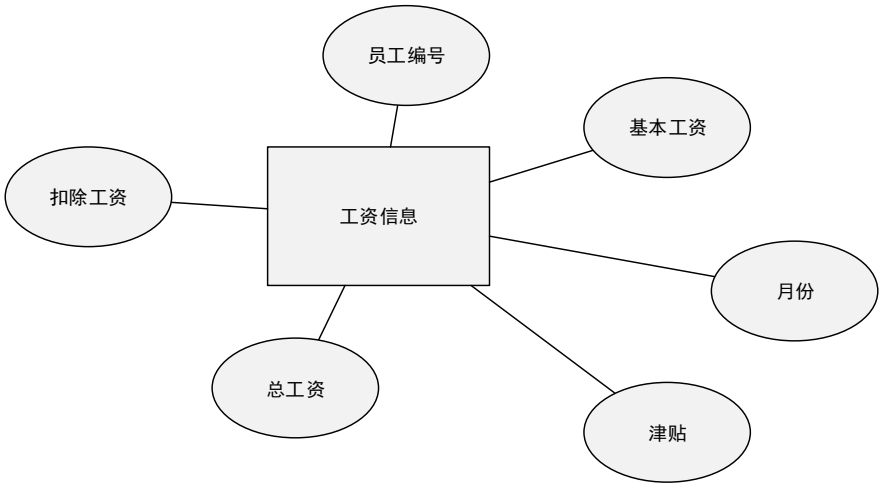


图 4.6 工资实体

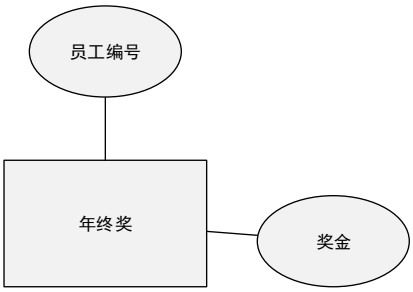


图 4.7 年终奖实体

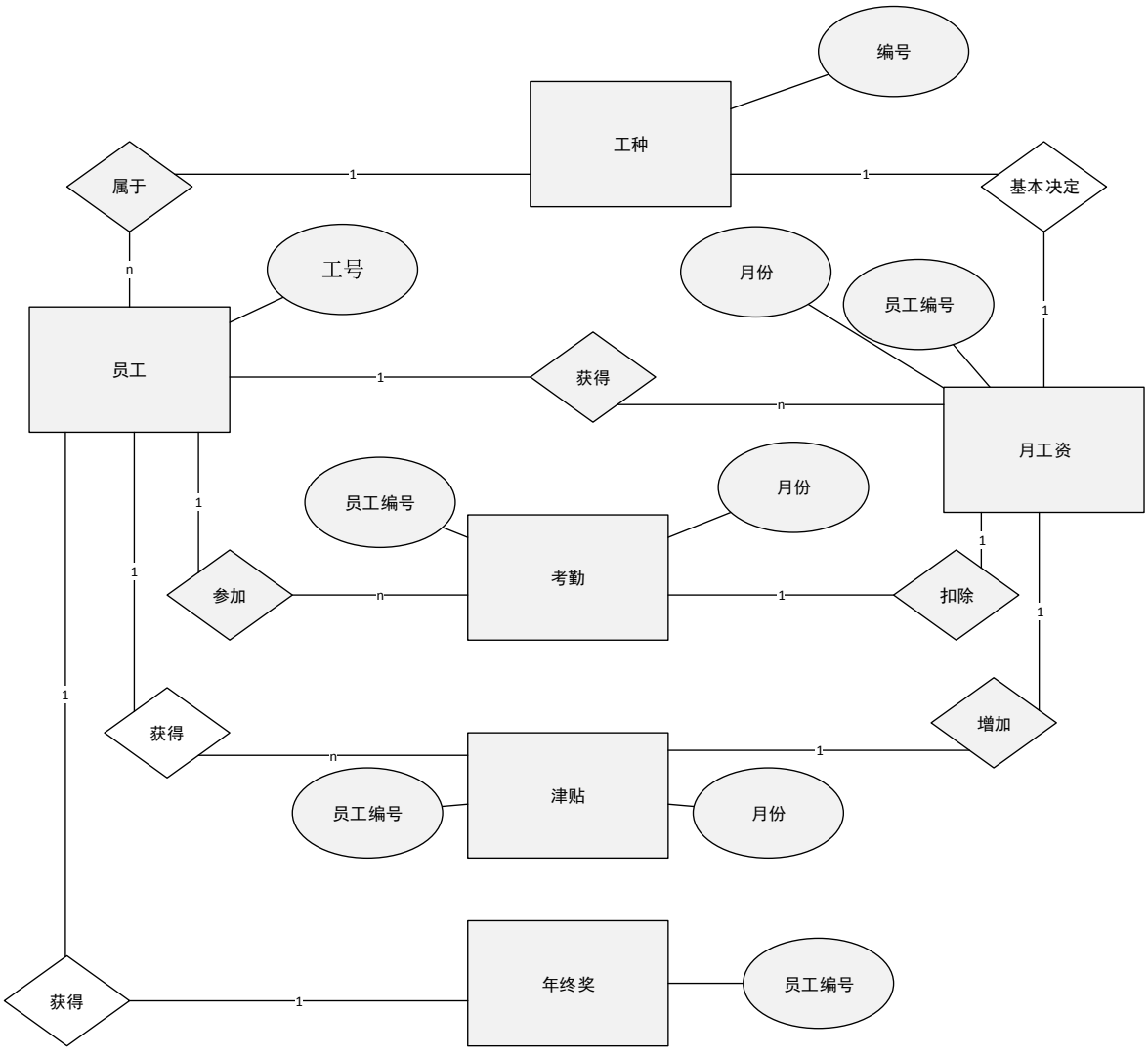


图 4.8 整体 ER 图

表说明：

员工表 Employee（工号 Eno，姓名 Ename，性别 Esex，年龄 Eage，工种，电话 Etel，住址 Eaddr）；

工种表 Branch（编号 Bno，名称 Bname，等级 Blevel，基本工资 BSalary）；

津贴表 Overtime（工号，月份 Month，加班类别 Otype，加班天数 Onum，津贴情况 Omoney）

Omoney=Onum*单次加班津贴

考勤表 Lost（工号，月份 Month，缺勤次数 Lnum，扣除工资 Lmoney）

Lmoney=Lnum*单次缺勤扣除工资

工资表 Salary(工号,月份 Month,基本工资,津贴,扣除工资,总工资 Tmoney)

Tmoney=BSalary+Omoney-Lmoney

年终奖表 YReward（工号，奖金 Reward）

Reward=（1~12 月份工资和）/12

带下划线的属性为表的主码，红色为外码。

4.5 详细设计与实现

阐述各主干功能的实现过程，包括主干功能的业务流程图、关键技术和算法说明、数据库事务的定义与实现、数据库函数和触发器的定义与实现等（不允许大段引用源码，如有必要引用必须加详细注释）。

系统使用的数据库是 web SQL（这是一个客户端的数据库，其语法与 sqlite 完全一致），使用的前端语言是 javascript。

系统分为了两类用户，分别是管理员和普通员工，用户身份和登录状态的判别是在登录页面实现的，使用了 sessionStorage 技术，如果是管理员用户则跳转到根管理页面，普通用户则跳转到普通用户的查询页面，并在跳转之前保存用户的状态；直接从地址栏打开除登陆页面以外的页面时，由于身份信息验证失败，所以会先给出错误（未登录）提示，然后自动跳转到登录页面，登陆后才可以使用相关的功能。关键技术是 sessionStorage，可以为每一个标签页保存单独的数据，在页面跳转时保留，并在标签页关闭后销毁。

系统管理员拥有对工种表、员工表、考勤表和津贴表的全部增删查改的权限，对工资表和年终奖表查询的权限，普通员工用户只能查询自己的工资、考勤、津贴和年终奖，没有其他权限；权限的控制是在前端完成的。

工种管理有 4 个功能，分别是增加工种，删除工种，修改现有工种和查询功能，删除工种。增加工种时需要输入该工种相关的完整信息才允许执行；修改工种时至少需要修改除工种编号外的一项内容，且工种编号为主键，不允许修改。

员工管理也是增删查改 4 个功能，增加员工时，工种应该先查询工种表的全部信息，然后交给用户选择；修改时同样不允许修改员工编号，修改至少一项其他信息。

考勤管理和津贴管理同样各有 4 个功能，增加记录时，让用户选择需要记录的员工和月份，然后填写其他信息；修改时选择编号和月份，至少修改一项其他信息。

工资和年终奖查询，管理员可以按特定员工、特定月份或两者皆指定或皆不指定（查询全部记录）的方式查询员工工资，由于年终奖的条目较少，所以只提供了查询全部员工的年终奖的功能，不需要指定员工。

对于普通用户来说，只能查询自己的考勤、津贴、工资和年终奖信息。

系统中拥有 13 个触发器，因为 web SQL 功能比较简单，不支持计算列和在触发器中声明变量等操作，所以使用了较多的触发器。

4.6 系统测试

包括对测试数据的说明、测试过程阐述、测试结果分析。

（1） 登陆测试

登录页面如图 4.9 所示，管理员登录成功会跳转到管理员页面，如图 4.10 所示，普通用户登录成功后会跳转到普通用户的页面，如图 4.11 所示，用户不需要选择身份，而登录失败后会有提示如果用户名不存在，则提示用户名错误，用户名存在而密码错误则提示密码错误，分别如图 4.12 和图 4.13 所示

图 4.9 登录页面

图 4.10 管理员页面

图 4.11 普通用户页面

图 4.12 登陆用户名错误

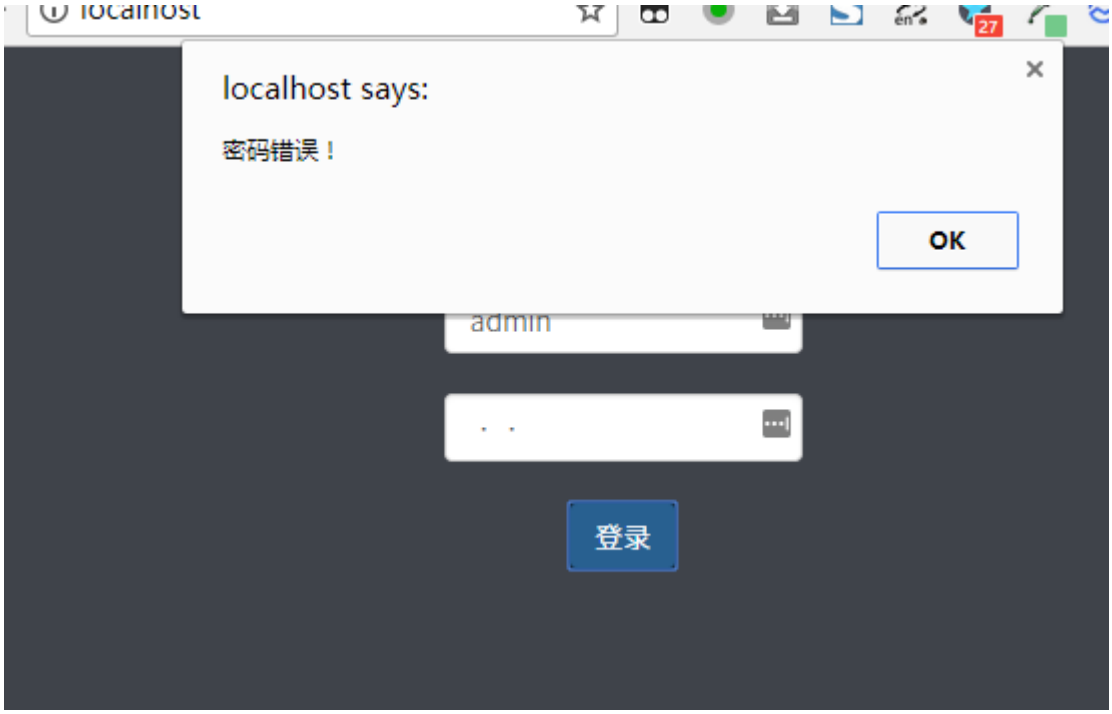


图 4.13 登录密码错误

（2） 工种管理测试

增加工种测试：输入完整信息后会提示增加完成，如图 4.14 所示，然后查询工种信息如图 4.15 所示，表明信息增加成功，输入的信息不完整则会提示，如图 4.16 所示；删除工种测试：输入存在的工种编号后会成功删除，前后对比如图 4.17 和图 4.18 所示，输入的编号不存在会提示表中没有该工种，如图 4.19 所示；更新工种测试：更新成功的和更新后的查询信息如图 4.20 和图 4.21 所示，更新失败的提示如图 4.22 所示。查询功能已经在上述步骤中体现。

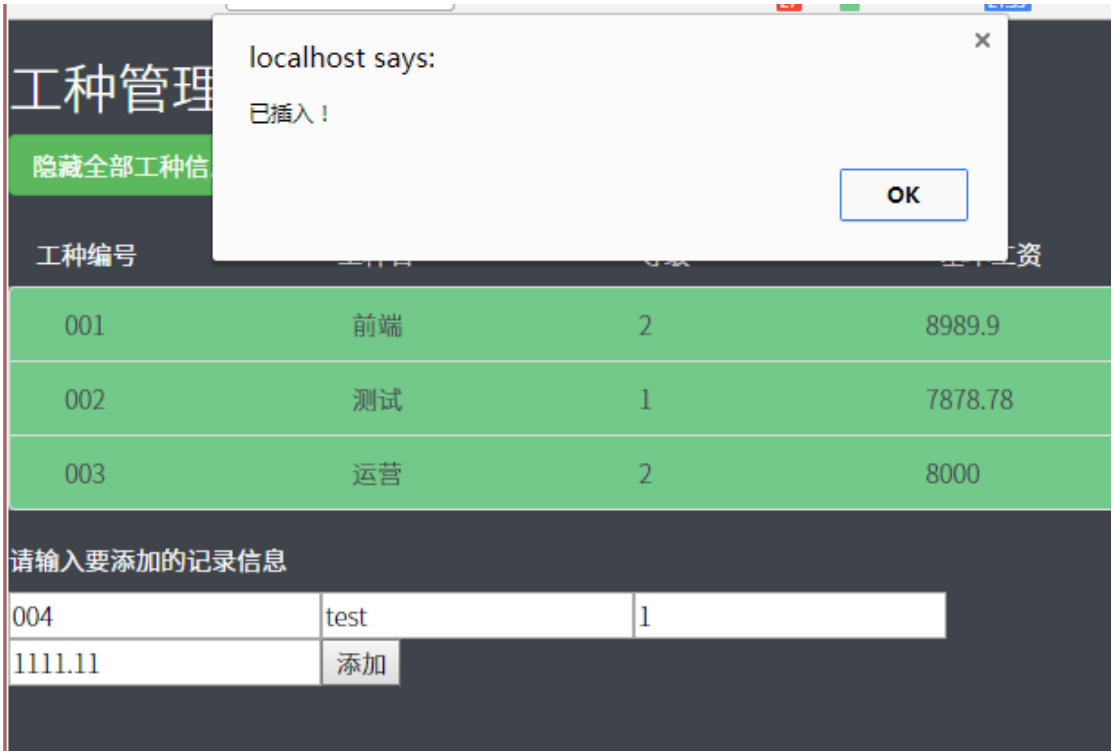


图 4.14 增加工种

图 4.15 增加工种后

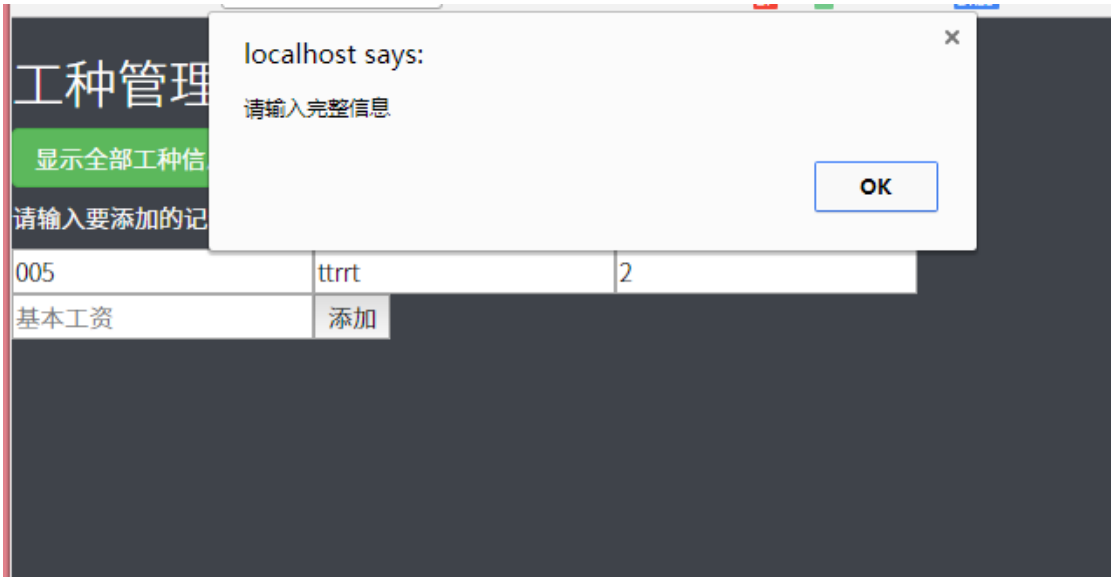


图 4.16 输入的新增信息不完整

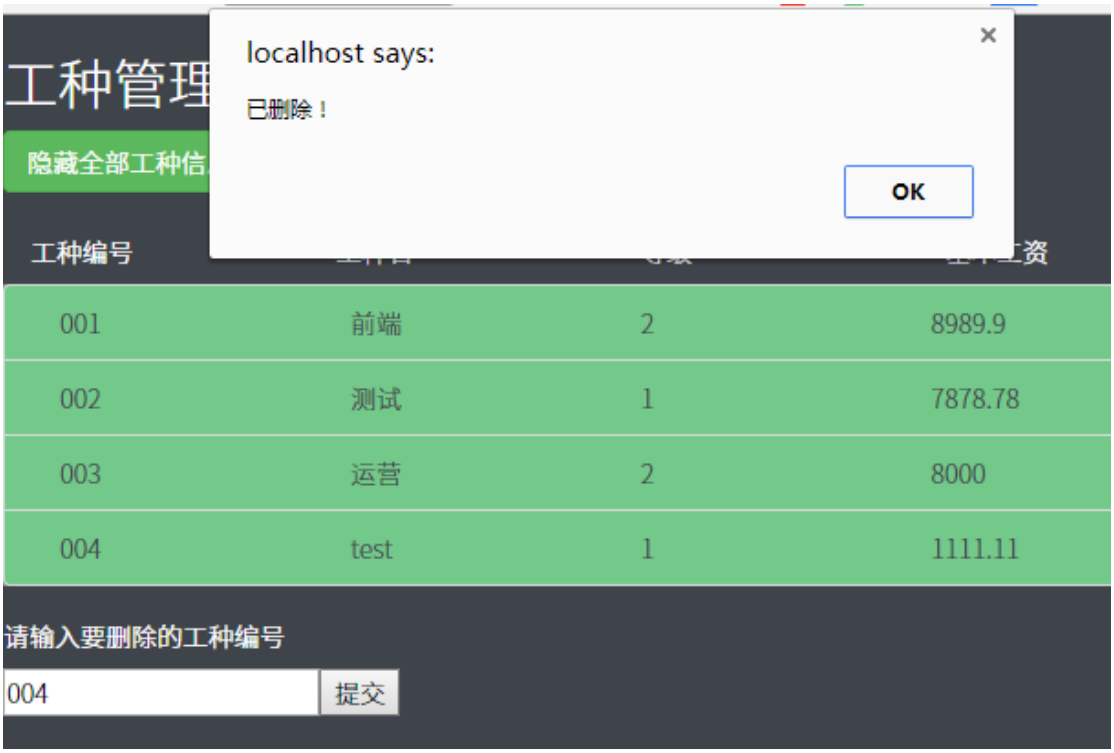


图 4.17 删除工种

图 4.18 删除工种后

图 4.19 删除不存在的工种

图 4.20 更新工种

图 4.21 更新工种后

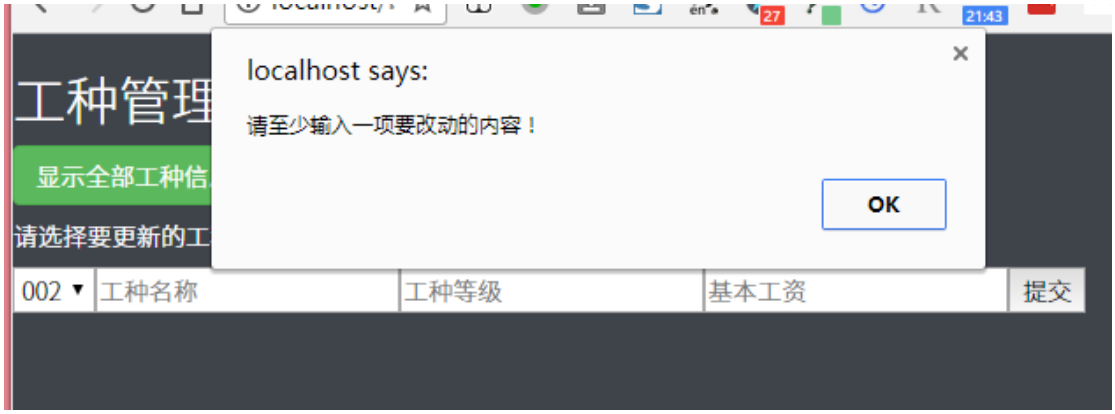


图 4.22 点击提交但未输入任何更新信息

（3） 员工管理测试

该模块的测试方法与之前模块的测试方法一致，先显示全部的员工信息，如图 4.23 所示，再增加一个员工，如图 4.24 所示，然后查看增加成功后的员工信息如图 4.25 所示，表示确实增加成功了，当尝试删除一个不存在的员工时会给出提示，如图 4.26 所示。

图 4.23 显示全部员工信息

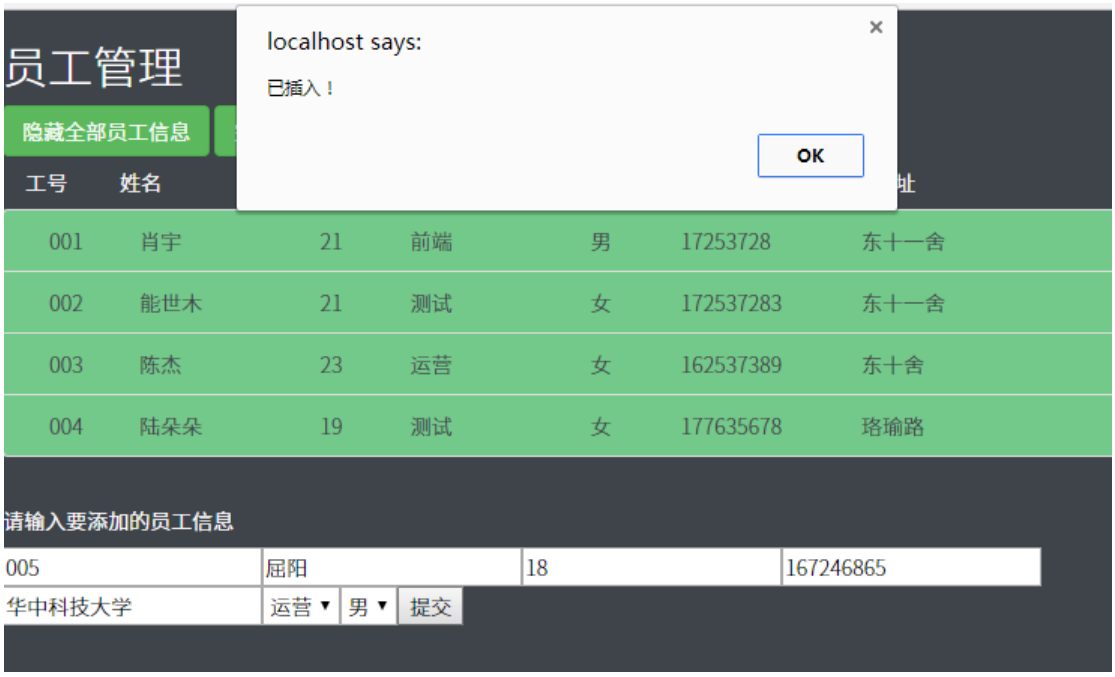


图 4.24 插入新的员工信息

图 4.25 插入后的员工信息

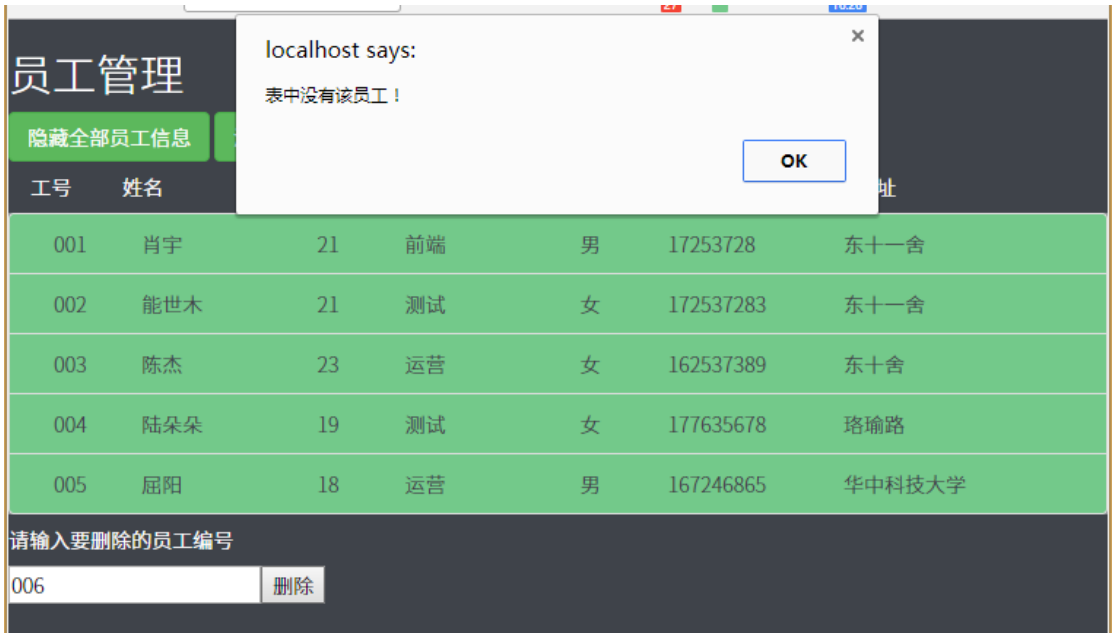


图 4.26 删除不存在的员工

（4） 考勤管理测试

该模块的测试方法仍与之前类似，先显示全部考勤信息，如图 4.27 所示，插入新的考勤信息如图 4.28 所示，插入成功后如图 4.29 所示，删除记录如图 4.30 所示，修改记录如图 4.31 所示，显示修改后的信息如图 4.32 所示。

图 4.27 显示考勤信息

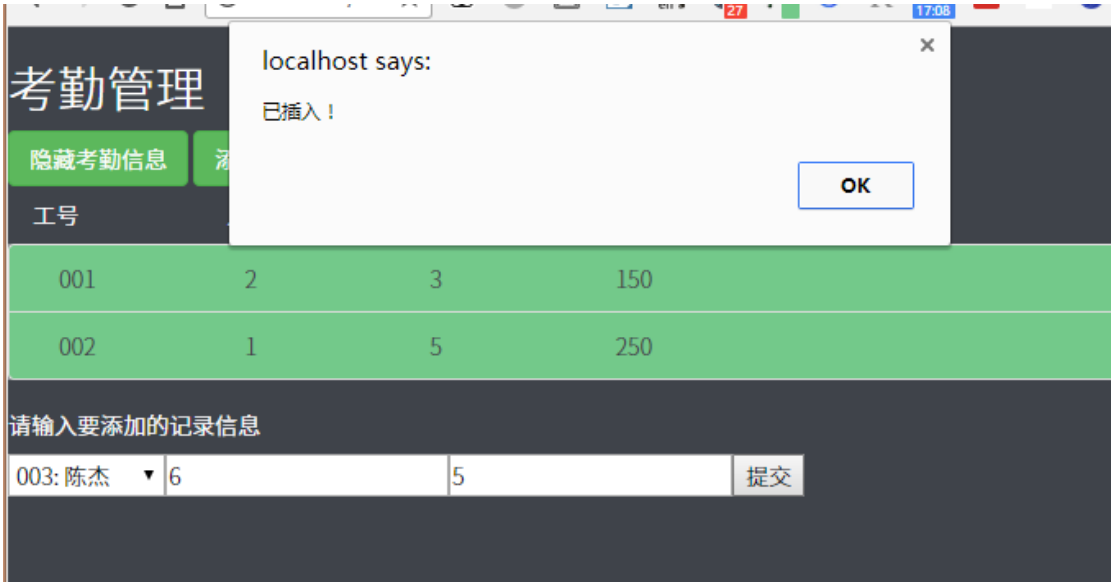


图 4.28 插入考勤记录

图 4.29 插入后的信息

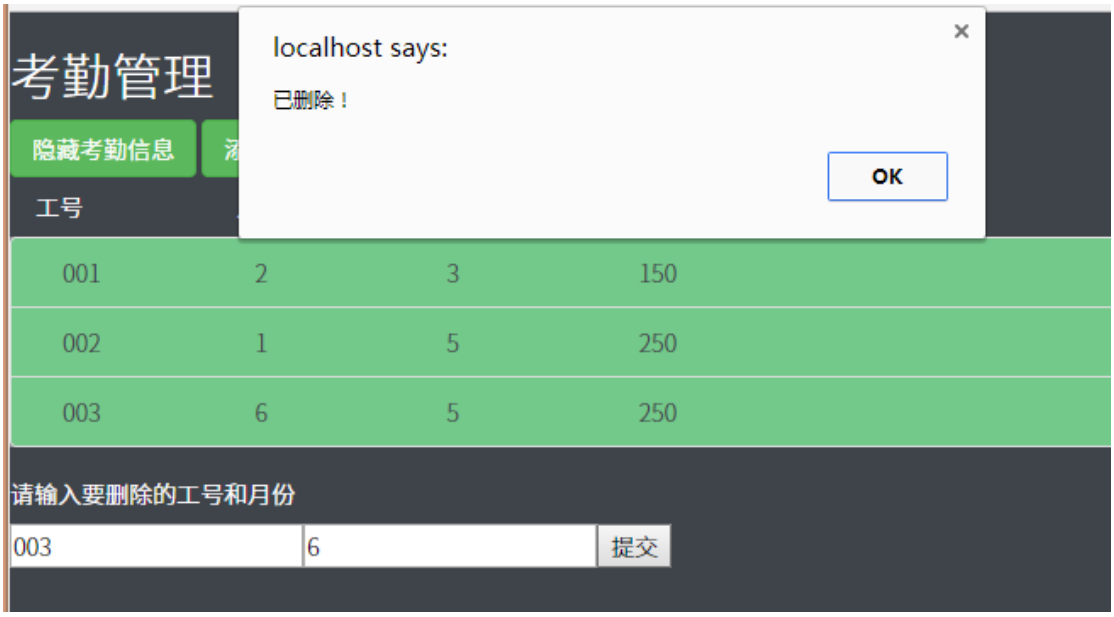


图 4.30 删除考勤记录

图 4.31 修改考勤记录

图 4.32 修改后的考勤信息

（5） 津贴管理测试

津贴管理模块与考勤管理模块的功能和实现即为相似，测试方法同样几乎完全相同，显示信息如图 4.33 所示，试图插入不完整的信息提示如图 4.34 所示，插入成功如图 4.35 所示，删除不存在的记录如图 4.36 所示，点击修改按钮但未输入修改信息如图 4.37 所示。

图 4.33 显示津贴信息

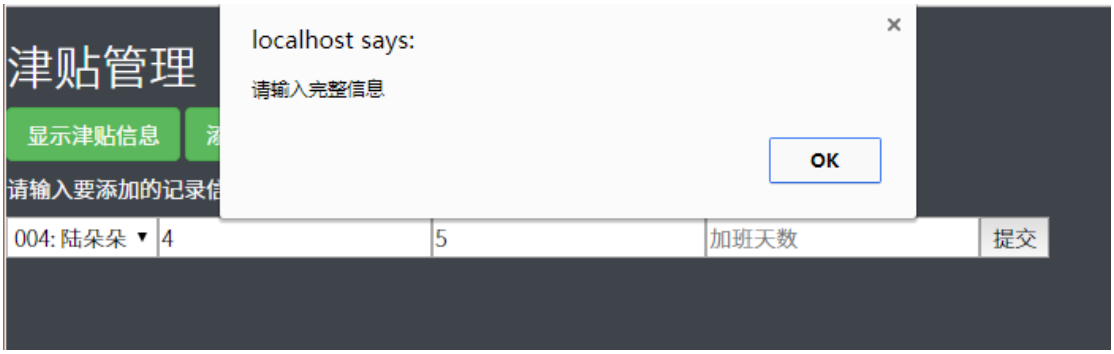


图 4.34 未插入

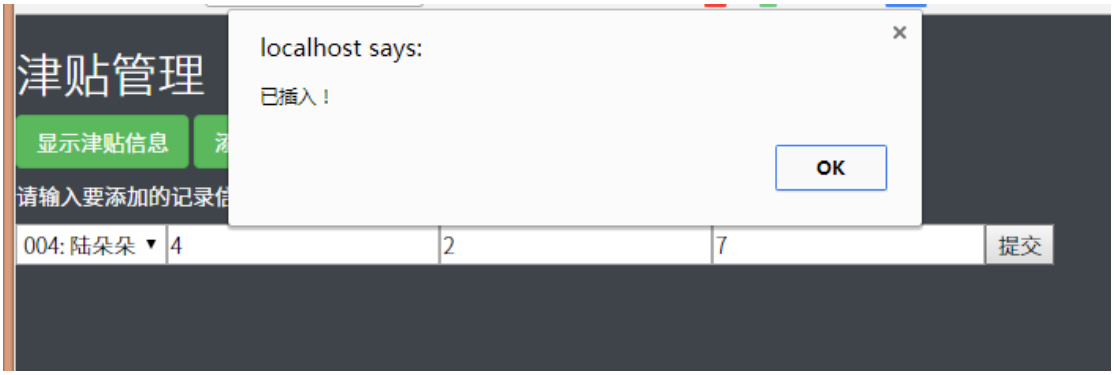


图 4.35 已插入

图 4.36 要删除的记录不存在

图 4.37 未输入任何修改信息

（6） 工资与年终奖查询测试

管理员可以按照多种方式查询工资信息，分别是全部、按员工、按月份、和按员工与月份，如图 4.38、图 4.39、图 4.40 和图 4.41 所示；因为每个员工只有一条年终奖记录，记录条数不是很多，所以年终奖的查询只提供了一种方式，显示全部员工的年终奖信息，如图 4.42 所示。

图 4.38 全部工资信息

图 4.39 某员工的工资信息

图 4.40 某月份的工资信息

图 4.41 某员工在某月的工资信息

图 4.42 全部年终奖信息

（7） 员工自助查询测试

首先是员工登录，登录页面如图 4.43 所示，与管理员共用一个登录页

面，以员工的工号作为登录名，登陆成功如图 4.44 所示，会根据用户显示对应的欢迎信息，然后测试查询考勤与津贴记录如图 4.45 所示，工资信息和年终奖如图 4.46 所示。

图 4.43 登录页面

图 4.44 登陆成功后

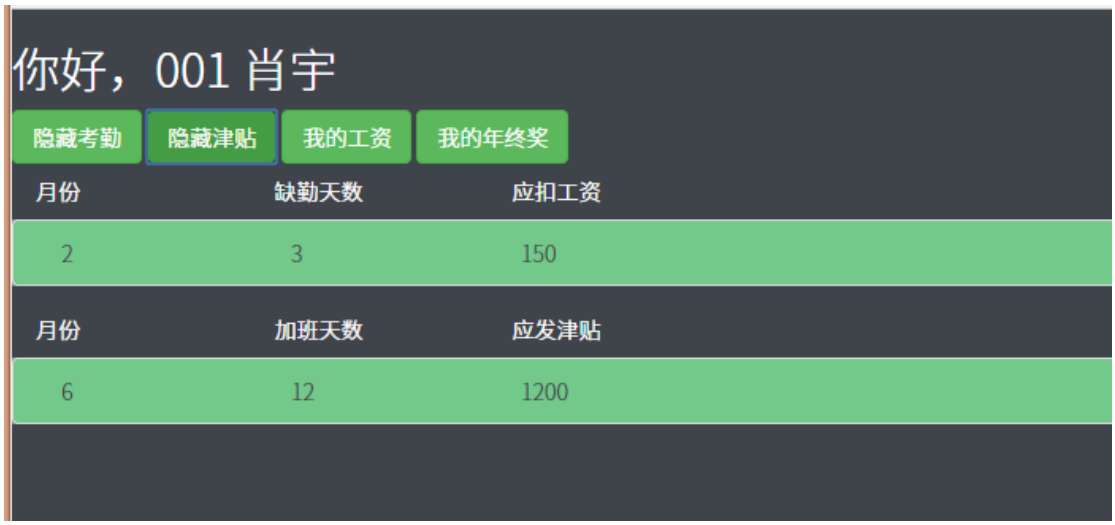


图 4.45 我的考勤与津贴记录

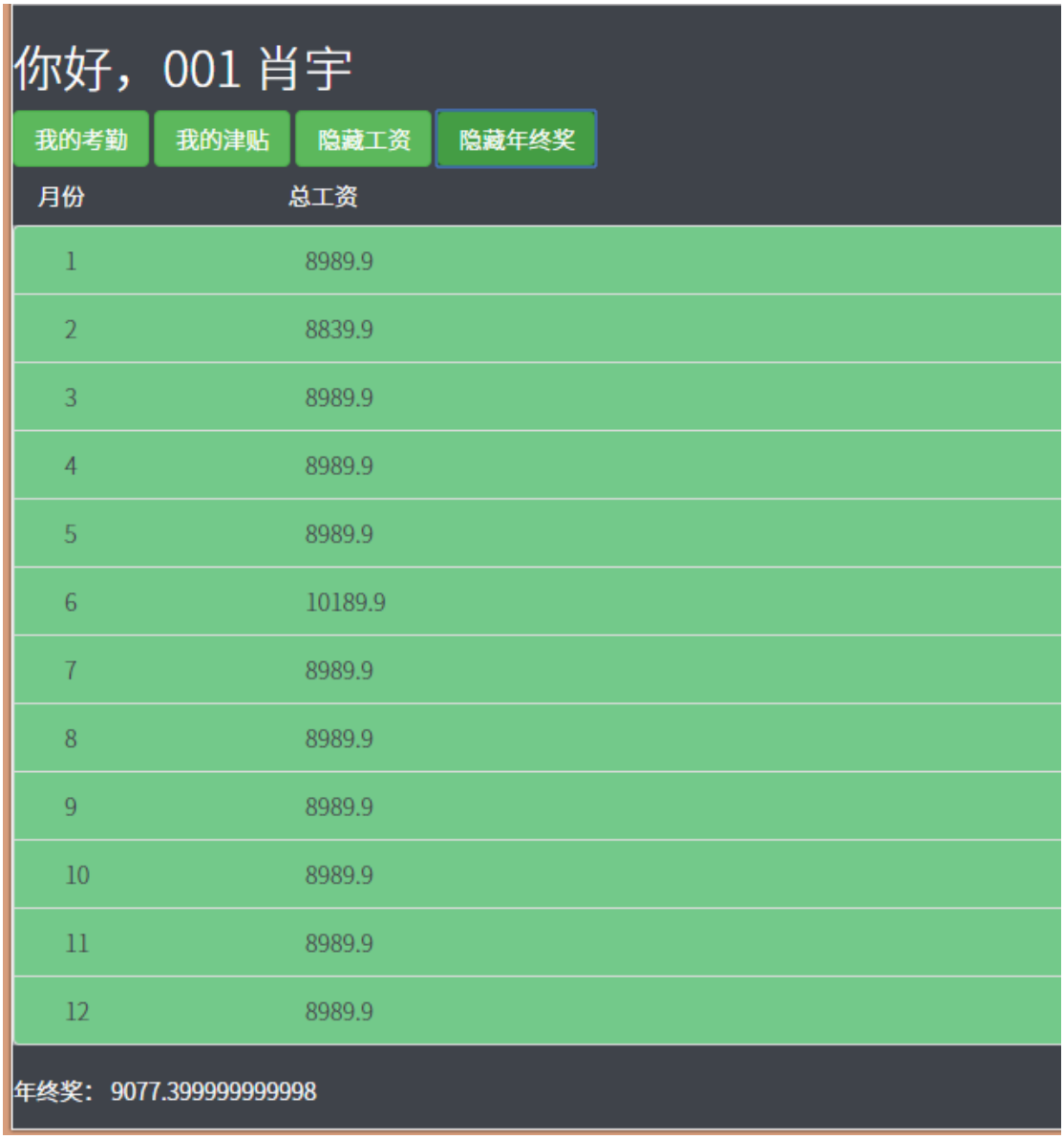


图 4.46 我的工资与年终奖信息

（8） 安全测试

系统的主要是对登录做了验证，确保用户无法在未登录的条件下获得使用权限，当尝试直接从地址栏打开管理页面而又没有以管理员身份登陆时，会提示错误并自动跳转到登录页面，如图 4.47 所示；当尝试直接从地址栏打开员工查询页面而没有以员工的身份登陆时，也会提示错误并自动跳转到登录页面，如图 4.48 所示。

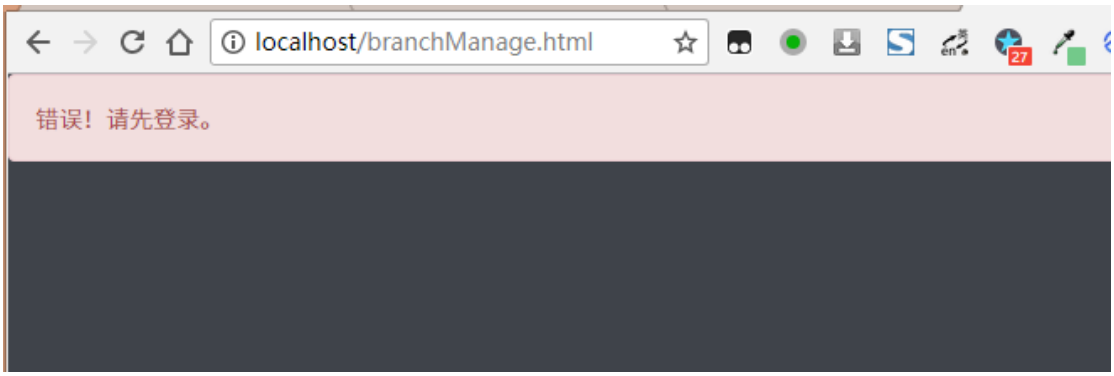


图 4.47 工种管理（尚未登录）

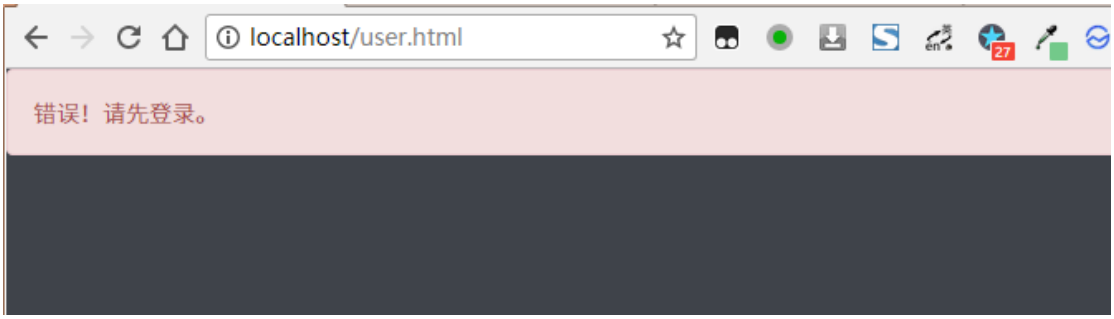


图 4.48 员工自主查询（尚未登录）

（9） 其他测试

4.7 系统设计与实现总结

该系统主要分了两大部分的工作，一是后台数据库的设计和实现，二是前端界面的实现。

数据库的设计实现方法比较传统，就是创建数据库，建表，规范约束，建触发器等操作。具体的工作是创建了 1 个数据库，7 张表（其中 1 张 user 表与逻辑功能无关，用于存放用户名和密码），13 个触发器。

因为使用的数据库比较简单，所以前端的工作就复杂了一些，比如 web SQL 中没有用户和角色的概念，打开数据库时也不需要登录，所以用户登录状态，身份识别等功能就要在前端完成。具体的工作有：

1) 连接数据库；
2) 用户身份与登录状态的识别；
3) 8 个静态 html 页面，分别是登录，管理员主功能，工种管理，员工管理，考勤管理，津贴管理，管理员查询工资与年终奖，以及普通员工查询页面。其中每个页面分别对应一个 javascript 脚本文件，用于实现全部的逻辑功能。

5 课程总结

这次课程实践完成了一个工资管理系统，实现了基本的工种、员工、考勤与津贴管理，以及多种查询功能，但是仍然有一些缺点，包括数据库的设计上还有不合理的地方，比如数据冗余，前端的样式比较简单，虽然实现了基本的功能但界面上不够友好和美观，再就是仍然使用了多个静态页面，做成单页应用或者动态页面会更好。

通过这次实践，自己也学到了很多东西，最直接的是学习了 web SQL 这一在 web 客户端的数据库，了解到了不同的数据库之间功能与语法上的一些差异，比如 web SQL 完全使用了 sqlite 的语法，与前两次实验用到的 sql server 还是存在一些差别，在功能上也存在这许多差异，主要还是这个数据库太轻量，没有太复杂的功能；因为前端的实现中编程比较多，也提高了自己的编程能力，而且遇到的许多问题也学会了如何去解决或者换一种思路，使用变通的方法，比如因为 web SQL 提供的是异步 API 的原因，许多操作的执行不按编程的顺序来，要通过回调来解决，而在循环中，回调也不容易实现，最终是换了一种操作，把功能的实现从前端分离出来，放到了数据库的触发器中，挣扎许久，提高了自己写 bug 和调 bug 的能力；最后是对数据库本身的认识，在管理数据上有着极大的优势（又想起了之前做 C 语言课程设计时使用链表和文件存取时被支配的恐惧），不用再关心怎样把数据保存下来和读取出来，只要跟数据库管理系统操作就可以了。

附录

实践报告要统一有附录，包含第 3 部分的核心功能的源代码。

建表操作的 SQL 代码：

```
CREATE TABLE `Branch` (
 `Bno`      char(10) UNIQUE,
 `Bname`  char(10) UNIQUE,
 `Blevel`   int CHECK(Blevel in ( 1 , 2 , 3 , 4 )),
 `Bsalary` float CHECK(Bsalary > 0),
 PRIMARY KEY(`Bno`)
);
CREATE TABLE `Employee` (
 `Eno`      char(10),
 `Ename`  char(10) NOT NULL,
 `Esex`     char(3) CHECK(Esex in ( '男' , '女' )),
 `Eage`     int CHECK(Eage >= 10 and Eage <= 120),
 `Bno`      char(10),
 `Etel`     char(12),
 `Eaddr`    char(30),
 PRIMARY KEY(`Eno`),
 FOREIGN KEY(`Bno`) REFERENCES `Branch`(`Bname`)
);
CREATE TABLE `Lost` (
 `Eno`      char(10),
 `Lmonth` int CHECK(Lmonth >= 1 and Lmonth <= 12),
 `Lnum`    int CHECK(Lnum >= 0 and Lnum <= 31),
 `Lmoney`float,
 PRIMARY KEY(`Eno`,`Lmonth`),
 FOREIGN KEY(`Eno`) REFERENCES `Employee`(`Eno`)
);
CREATE TABLE `Overtime` (
 `Eno`      char(10),
 `Omonth`int CHECK(Omonth >= 1 and Omonth <= 12),
 `Otype`    int,
 `Onum`    int CHECK(Onum >= 0 and Onum <= 31),
 `Omoney`       float,
 PRIMARY KEY(`Eno`,`Omonth`),
 FOREIGN KEY(`Eno`) REFERENCES `Employee`(`Eno`)
);
CREATE TABLE `Salary` (
 `Eno`      char(10),
```

```
 `Ename`  char(10),
 `Smonth` int CHECK(Smonth >= 1 and Smonth <= 12),
 `Bsalary` float CHECK(Bsalary > 0),
 `Omoney`      float DEFAULT 0 CHECK(Omoney >= 0),
 `Lmoney`float DEFAULT 0 CHECK(Lmoney >= 0),
 `Tmoney`float,
 FOREIGN KEY(`Eno`) REFERENCES `Employee`(`Eno`),
 FOREIGN KEY(`Bsalary`) REFERENCES `Branch`(`Bsalary`),
 FOREIGN KEY(`Omoney`) REFERENCES `Overtime`(`Omoney`),
 FOREIGN KEY(`Lmoney`) REFERENCES `Lost`(`Lmoney`)
);
CREATE TABLE `Reward` (
 `Eno`      char(10),
 `Reward` float,
 FOREIGN KEY(`Eno`) REFERENCES `Employee`(`Eno`)
);
```

部分触发器的 SQL 语句：

```
CREATE TRIGGER addLost after insert on Lost begin update Salary set Lmoney=new.Lmoney, Tmoney=(select Tmoney from Salary where Eno=new.Eno and Smonth=new.Lmonth)-new.Lmoney where Eno=new.Eno and Smonth=new.Lmonth; end
CREATE TRIGGER addOver after insert on Overtime begin update Salary set Omoney=new.Omoney, Tmoney=(select Tmoney from Salary where Eno=new.Eno and Smonth=new.Omonth)+new.Omoney where Eno=new.Eno and Smonth=new.Omonth; end
CREATE TRIGGER delBranch after delete on Branch begin delete from Employee where Bno=old.Bname; end
CREATE TRIGGER delLost after delete on Lost begin update Salary set Lmoney=0, Tmoney=(select Tmoney from Salary where Eno=old.Eno and Smonth=old.Lmonth)+old.Lmoney where Eno=old.Eno and Smonth=old.Lmonth; end
CREATE TRIGGER delOver after delete on Overtime begin update Salary set Omoney=0, Tmoney=(select Tmoney from Salary where Eno=old.Eno and Smonth=old.Omonth)-old.Omoney where Eno=old.Eno and Smonth=old.Omonth; end
CREATE TRIGGER deleteEmployee after delete on Employee begin delete from Lost where Eno=old.Eno; delete from Overtime where Eno=old.Eno; delete from Salary where Eno=old.Eno; delete from Reward where Eno=old.Eno; end
CREATE TRIGGER updateBranch after update on Branch when old.Bsalary!=new.Bsalary begin update Salary set Bsalary=new.Bsalary where Eno in (select Eno from Employee where Bno= (select Bname from Branch where Bno=new.Bno) ); end
CREATE TRIGGER updateSalaryB after update on Salary when old.Tmoney!=new.Tmoney begin update Reward set Reward=(select avg(Tmoney) from Salary where Eno=old.Eno) where Eno=new.Eno; end
CREATE TRIGGER updateSalaryT after update on Salary when old.Bsalary!=new.Bsalary begin update Salary set Tmoney=(select Tmoney from Salary where Eno=old.Eno and Smonth=old.Smonth)-old.Bsalary+new.Bsalary; end
CREATE TRIGGER updateemployee after update on Employee begin update Salary set
```

```
Ename=new.Ename, Bsalary=(select Bsalary from Branch where Bname=new.Bno) where
Eno=new.Eno; end
```

登录操作的主要代码：

```
function login(){
  var username=document.getElementById("username").value,
    password=document.getElementById("password").value;
    db.transaction(function (tx) {
        tx.executeSql('select psword from user where username=?',[username],
        function (tx, results) {
          tmp=results;
          if(tmp.rows.length==0){
            alert("用户名错误！")
          } else if(password==tmp.rows.item(0).psword){
            if (username=="admin") {
              sessionStorage.setItem('state',true)
              window.location="admin.html";
            } else {
              sessionStorage.setItem('user',username)
              window.location="user.html"
            }
          } else{
            alert("密码错误！");
          }
        },
        function (tx, errmeg) {
          tmp=errmeg;
          alert(errmeg);
        });
    });
}
```

管理员主模块的主要代码：

```
    var db = openDatabase('testDB', '1.0', 'Test DB', 32 * 1024 * 1024)
    var arr = document.getElementsByTagName('button');
    for(var i = 0;i<arr.length;i++){
        arr[i].onclick = function(){
            var sid = this.id;
          window.location=sid+".html";
        }
    }
```

显示工种信息的主要代码：

```
function getInfo(){
  var showtitle=$('<h5 class="new"><br><div class="col-xs-3" id="Eno">工种编号</div>\
    <div class="col-xs-3" id="Omonth">工种名</div>\
```

```
        <div class="col-xs-3" id="Otype">等级</div>\
        <div class="col-xs-3" id="Onum">基本工资</div><br></h5>')
    $("body").append(showtitle)
    var showlist=$('<ul class="new list-group" id="listtitle"></ul>')
    $("body").append(showlist)
    db.transaction(function(tx){
        tx.executeSql('select * from Branch',[],
        function(tx,results){
            tmp=results
            resultToLists()
        },
        function(tx,errmeg){
            tmp=errmeg
            alert(tmp.message)
        })
    })
}
function resultToLists(){
    if(tmp.rows.length==0) {
        //空
    } else {
        var bno, bname, blevel, bsalary, len=tmp.rows.length, i;
        rows.length=len;
        for(i=0; i<len; i++) {
            bname = $('<div class = "col-xs-3"></div>').text(tmp.rows.item(i).Bname);
            blevel = $('<div class = "col-xs-3"></div>').text(tmp.rows.item(i).Blevel);
            bsalary = $('<div class = "col-xs-3"></div>').text(tmp.rows.item(i).Bsalary);
            bno = $('<div class = "col-xs-3 new"></div>').text(tmp.rows.item(i).Bno);
            var clrflt = $('<div class = "clear-float"></div>');
            var newli = $('<li class = "new list-group-item"></li>').append(bno, bname, blevel,
bsalary, clrflt);
            $("#listtitle").append(newli);
        }
    }
    hide(".new","emp0","显示全部工种信息","隐藏全部工种信息")
}
function hide(clsname,opid,distext,hidtext){
    document.getElementById(opid).innerHTML=hidtext
    document.getElementById(opid).onclick=function(){
        $(clsname).remove()
        document.getElementById(opid).innerHTML=distext
        document.getElementById(opid).onclick=function(){operation(opid)}
    }
}
```

删除考勤信息的主要代码：

```
function delOper(){
   var inputsdel = $('<div class="newdel" id="newdel"></div>'),
      deltip=$("<h5></h5>").text("请输入要删除的工号和月份")
   $("body").append(inputsdel)
   $("#newdel").append(deltip)
   var inputdel=$('<input placeholder="工号" class="newdel" id="delEno"></input><input
placeholder="月份" class="newdel" id="delLmon"></input>')
   $("#newdel").append(inputdel)
   var execdel = $('<button id="subdel"></button>').text("提交")
   $("#newdel").append(execdel)
   document.getElementById('subdel').onclick=function(){sqlDelete()}
   hide(".newdel","emp2","删除记录","删除完成")
}
function sqlDelete(){
   var Eno=document.getElementById("delEno").value,
        Lmon=document.getElementById('delLmon').value
   if (Eno==""||Lmon=="") {
      alert("请输入完整信息！")
   } else {
      db.transaction(function (tx) {
         tx.executeSql('delete from Lost where Eno=? and Lmonth=?',[Eno,Lmon],
         function (tx, results) {
            tmp=results;
            if(tmp.rowsAffected==0){
               alert("表中没有该记录！")
            } else {
               alert("已删除！");
            }
         },
         function (tx, errmeg) {
            tmp=errmeg;
            alert(errmeg);
         });
      });
   }
}
```

添加考勤记录的主要代码：

```
function addOper(){
   var addtip=$("<h5></h5>").text("请输入要添加的记录信息")
   var inputsadd = $('<div class="newadd" id="newadd"></div>')
   $("body").append(inputsadd)
   addtextarea(3,"newadd","newadd")
   $("#add0").remove()
```

```
    var selt=$('<select id="add0"></select>')
    $("#newadd").prepend(selt)
    for(var i=0;i<Enames.length;i++){
      var opt=$('<option></option>').text(Enos[i]+": "+Enames[i])
      $("#add0").append(opt)
    }
    var execadd = $('<button id="sub"></button>').text("提交")
    $("#newadd").append(execadd)
    $("#newadd").prepend(addtip)
    document.getElementById('sub').onclick=function(){sqlAdd()}
    hide(".newadd","emp1","添加纪录","添加完成")
  }
  function sqlAdd(){
    var addvalue=[], Id
    for(var i=0;i<3;i++){
      Id="add"+i
      addvalue[i]=document.getElementById(Id).value
      if (addvalue[i]=="") {
        alert("请输入完整信息")
        return
      }
    }
    var noname=addvalue[0].split(": ")
    db.transaction(function(tx){
      tx.executeSql('insert into Lost
  values(?,?,?,?)',[noname[0],addvalue[1],addvalue[2],addvalue[2]*50],
      function(tx,results){
        tmp=results
        if(tmp.rowsAffected==0){
          alert("插入失败！")
        } else {
          alert("已插入！")
        }
      },
      function(tx,errmeg){
        tmp=errmeg
        alert(tmp.message)
      })
    })
  }
```

更新津贴记录的主要代码：

```
  function getAllEnoOmons(){
    db.transaction(function(tx){
      tx.executeSql('select Eno, Omonth from Overtime',[],
```

```
        function(tx,results){
            tmp=results
            var len=tmp.rows.length
            for(var i=0; i<len; i++){
                Enos[i]=tmp.rows.item(i).Eno
                Omons[i]=tmp.rows.item(i).Omonth
            }
            upOper()
        },
        function(tx,errmeg){
            tmp=errmeg
            alert(tmp.message)
        })
    })
}
function upOper(){
    var inputsup = $('<div class="newup" id="newup"></div>'),
        uptip=$("<h5></h5>").text("请选择要更新的工号和月份，并输入正确的加班类型和
天数")
    $("body").append(inputsup)
    var enoomon=$('<select id="enovse"></select>')
    var inputtype=$('<input id="uptype" placeholder="加班类型"></input>')
    var inputnum=$('<input id="upnum" placeholder="加班天数"></input>')
    var execup=$("<button id='subup'></button>").text("提交")

$("#newup").append(uptip).append(enoomon).append(inputtype).append(inputnum).append(e
xecup)
    for(var i=0;i<tmp.rows.length;i++){
        var list=$('<option></option>').text(Enos[i]+"号  at "+Omons[i]+"月")
        $('#enovse').append(list)
    }
    document.getElementById('subup').onclick=function(){sqlUpdate()}
    hide(".newup","emp3","修改记录","修改完成")
}
function sqlUpdate(){
    var emstr=document.getElementById("enovse").value,
        onum=document.getElementById("upnum").value,
        otype=document.getElementById("uptype").value
    var emvalues=emstr.split("号  at ")
    var tttp=emvalues[1].split("月")
    emvalues[1]=tttp[0]
    if (onum==""&&otype=="") {
        alert("请至少输入一项要修改的信息！ ")
        return
```

```
    } else {
        var uupsql, args=[]
        if (onum=="") {
            uupsql='update Overtime set Otype=? where Eno=? and Omonth=?'
            args=[otype,emvalues[0],emvalues[1]]
        }else if (otype=="") {
            uupsql='update Overtime set Onum=?,Omoney=? where Eno=? and Omonth=?'
            args=[onum,onum*50,emvalues[0],emvalues[1]]
        } else {
            uupsql='update Overtime set Otype=?, Onum=?,Omoney=? where Eno=? and
    Omonth=?'
            args=[otype,onum,onum*50,emvalues[0],emvalues[1]]
        }
        db.transaction(function(tx){
            tx.executeSql(uupsql,args,
            function(tx,results){
                tmp=results
                if (tmp.rowsAffected==0) {
                    alert("修改失败！")
                } else {
                    alert("修改成功！")
                }
            },
            function(tx,errmeg){
                alert("tmp.message")
            })
        })
    }
}
```

管理员查询工资信息的主要代码：

```
function viewsalary(){
    var showtitle=$('<h5 class="new"><div class="col-xs-1" id="Eno">工号</div>\
        <div class="col-xs-1" id="Omonth">姓名</div>\
        <div class="col-xs-1" id="Otype">月份</div>\
        <div class="col-xs-2" id="Onum">基本工资</div>\
        <div class="col-xs-2" id="Omoney">扣除工资</div>\
        <div class="col-xs-2" id="Omoney">加班津贴</div>\
        <div class="col-xs-2" id="Omoney">总工资</div><br></h5>')

    $('body').append(showtitle)
    var showlist=$('<ul class="new list-group" id="listtitle"></ul>')
    $("body").append(showlist)
    db.transaction(function(tx){
        var name=document.getElementById('ename').value
```

```
    var month=document.getElementById('month').value
    var sqlstr, args=[]
    if (name=="全部员工" && month=="全部月份") {
      args=[]
      sqlstr='select Salary.Eno, Employee.Ename, Smonth, Bsalary, Lmoney, Omoney, Tmoney from Salary,Employee where Salary.Eno=Employee.Eno'
    } else if (name=="全部员工") {
      sqlstr='select Salary.Eno, Employee.Ename, Smonth, Bsalary, Lmoney, Omoney, Tmoney from Salary,Employee where Salary.Eno=Employee.Eno and Smonth=?'
      args[0]=month.split("月")[0]
    } else if (month=="全部月份") {
      sqlstr='select Salary.Eno, Employee.Ename, Smonth, Bsalary, Lmoney, Omoney, Tmoney from Salary,Employee where Salary.Eno=Employee.Eno and Salary.Eno=?'
      args[0]=name.split(":")[0]
    } else {
      sqlstr='select Salary.Eno, Employee.Ename, Smonth, Bsalary, Lmoney, Omoney, Tmoney from Salary,Employee where Salary.Eno=Employee.Eno and Smonth=? and Salary.Eno=?'
      args[0]=month.split("月")[0]
      args[1]=name.split(":")[0]
    }
    tx.executeSql(sqlstr,args,
    function(tx,results){
      tmp=results
      resultToLists()
    },
    function(tx,errmeg){

    })
  })
}
function resultToLists(){
  var len=tmp.rows.length
  if (len==0) {
    //空
  } else {
    var eno, ename, smonth, bmoney, lmoney, omoney, tmoney
    for(i=0;i<len;i++){
      eno=$('<div class = "col-xs-1"></div>').text(tmp.rows.item(i).Eno)
      ename=$('<div class = "col-xs-1"></div>').text(tmp.rows.item(i).Ename)
      smonth=$('<div class = "col-xs-1"></div>').text(tmp.rows.item(i).Smonth)
      bmoney=$('<div class = "col-xs-2"></div>').text(tmp.rows.item(i).Bsalary)
      lmoney=$('<div class = "col-xs-2"></div>').text(tmp.rows.item(i).Lmoney)
      omoney=$('<div class = "col-xs-2"></div>').text(tmp.rows.item(i).Omoney)
```

```
            tmoney=$('<div class = "col-xs-2"></div>').text(tmp.rows.item(i).Tmoney)
            var clrflt=$('<div class="clear-float"></div>')
            newli=$('<li  class="list-group-item"></li>').append(eno,  ename,  smonth,  bmoney,
    lmoney, omoney, tmoney, clrflt)
            $("#listtitle").append(newli)
        }
    }
    hide(".new","emp0","显示工资信息","隐藏工资信息")
}
```

员工自主查询工资信息的主要代码：

```
function showSalary(){
    var Salarytitle=$('<h5 class="newS"><div class="col-xs-2">月份</div>\
      <div class="col-xs-2">总工资</div><br></h5>')
    $("body").append( Salarytitle)
    var Salarylist=$('<ul class="newS list-group" id="Stitle"></ul>')
    $("body").append(Salarylist)
    db.transaction(function(tx){
      tx.executeSql('select * from Salary where Eno=?',[UID],
      function(tx,results){
        tmp=results
        showS()
      },
      function(tx,errmeg){
        tmp=errmeg
        alert(tmp.message)
      })
    })
}

function showS(){
    var len=tmp.rows.length
    if (len==0) {
      //空
    } else {
      var Smonth, Smoney
      for(i=0;i<len;i++){
        Smonth=$('<div class = "col-xs-2"></div>').text(tmp.rows.item(i).Smonth)
        Smoney=$('<div class = "col-xs-2"></div>').text(tmp.rows.item(i).Tmoney)
        var clrflt=$('<div class="clear-float"></div>')
        newli=$('<li class="list-group-item"></li>').append(Smonth,Smoney,clrflt)
        $("#Stitle").append(newli)
      }
    }
    hide(".newS","emp2","我的工资","隐藏工资")
```

}