



# 华中科技大学

## 数据库系统原理实践报告

综合设计题目：

姓 名： 龙际全  
学 院： 计算机科学与技术学院  
专 业： 计算机科学与技术  
班 级： CS1603  
学 号： U201614577  
指 导 教 师： 瞿彬彬

分数	
教师签名	

2019 年 06 月 19 日

## 任务书

### 题目 7：工资管理系统

#### 1、系统功能的基本要求：

- 员工每个工种基本工资的设置
- 加班津贴管理，根据加班时间和类型给予不同的加班津贴；
- 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；
- 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；
- 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：

- 员工考勤情况表；
- 员工工种情况表，反映员工的工种、等级，基本工资等信息；
- 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；
- 员工基本信息表
- 员工月工资表。

设计一个 B/S 或 C/S 模式的系统实现上述功能。

# 目录

<b>1</b>	<b>课程任务概述 .....</b>	<b>1</b>
<b>2</b>	<b>软件功能学习部分 .....</b>	<b>2</b>
2.1	任务要求.....	2
2.2	完成过程.....	2
2.2.1	脱机备份.....	2
2.2.2	系统备份.....	2
2.2.3	新增用户 .....	3
2.3	任务总结.....	4
<b>3</b>	<b>SQL 练习部分 .....</b>	<b>7</b>
3.1	任务要求.....	7
3.2	完成过程.....	7
3.2.1	建表.....	7
3.2.2	数据更新.....	11
3.2.3	查询.....	15
3.3	任务总结.....	23
<b>4</b>	<b>综合实践任务 .....</b>	<b>24</b>
4.1	系统设计目标.....	24
4.2	需求分析.....	24
4.3	总体设计.....	25
4.4	数据库设计.....	25
4.5	详细设计与实现.....	28
4.5.1	总体架构.....	28
4.5.2	后端设计.....	28
4.5.3	前端设计.....	31
4.6	系统测试.....	31
4.7	系统设计与实现总结.....	35
<b>5</b>	<b>课程总结 .....</b>	<b>36</b>

## 1 课程任务概述

实验主要分成三部分：软件功能学习部分、SQL 联系部分和综合实践部分。

- 1) 软件功能学习部分：实验指定使用 SQL server 数据库，该部分主要是熟悉该数据库的操作，包括创建用户，设置权限，备份数据和恢复数据等。
- 2) SQL 练习部分：用于熟悉 sql 语句的使用，所有的查询都基于与电影相关的关系，共 17 条查询语句，涵盖了大部分 sql 语法。
- 3) 综合实践部分：提供若干个题目供选择，如工资管理系统、航班管理系统等，每个系统都需要根据实际情况设计并创建若干个表，选用一种编程语言实现一个客户端/网页，要求数据库关系设计合理，客户端要有良好的人机交互界面。

## 2 软件功能学习部分

### 2.1 任务要求

- 1) 练习 sqlserver 的两种完全备份方式：数据和日志文件的脱机备份、系统的备份功能。
- 2) 练习在新增的数据库上增加用户并配置权限的操作。

### 2.2 完成过程

#### 2.2.1 脱机备份

- 1) 脱机指与服务器断开连接，脱机操作为“右键-任务-脱机”，如下图 2-1：

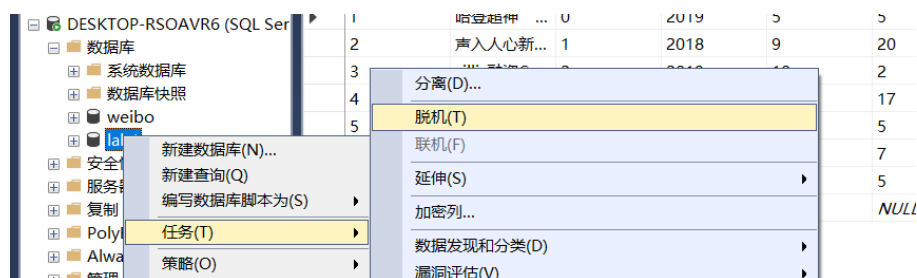


图 2-1 数据库脱机

- 2) 数据文件和日志文件可通过数据库属性页“属性-文件”查看，如下图 2-2，拷贝到想要拷贝的地方即实现了脱机备份。

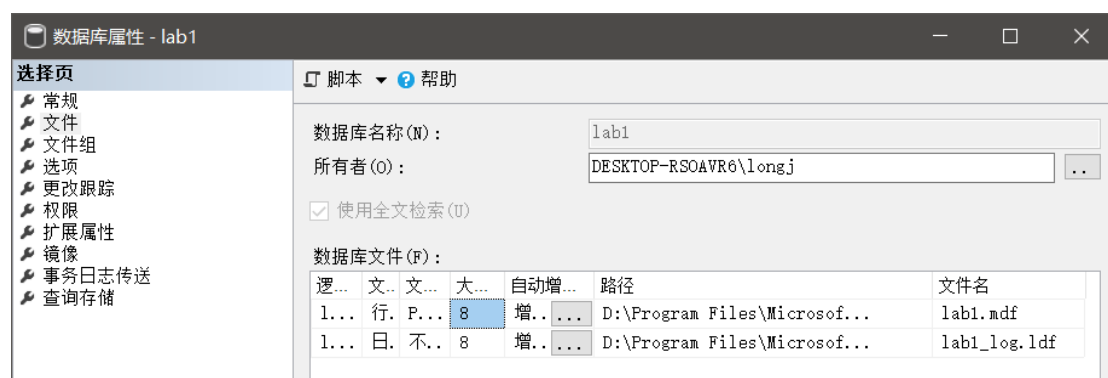


图 2-2 查看数据库文件位置

- 3) 通过“右键-附加”，选择刚刚拷贝的数据库数据文件即可实现脱机恢复，如下图 2-3；

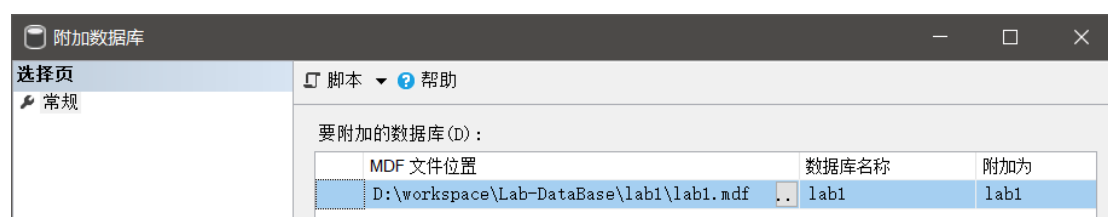


图 2-3 脱机恢复

#### 2.2.2 系统备份

- 1) 选中数据库“lab1”，通过“右键-备份”打开备份选项卡，如下图 2-4：

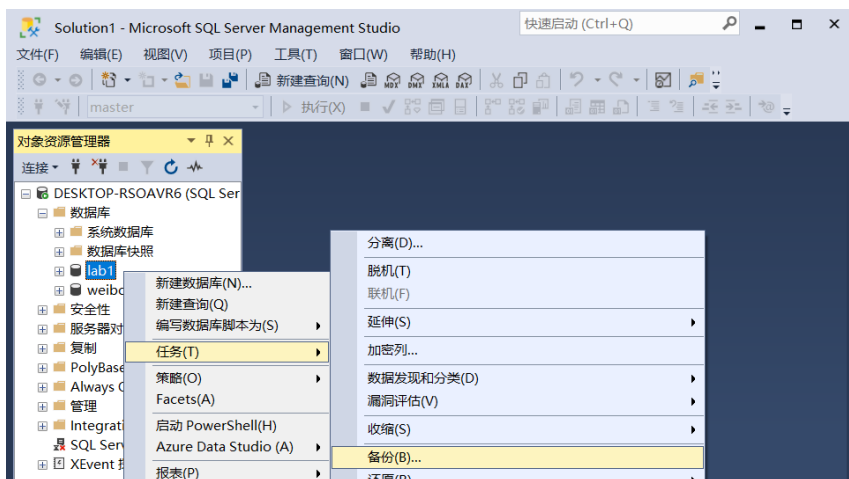


图 2-4 数据库系统备份

2) 备份到如下文件夹，如下图 2-5:

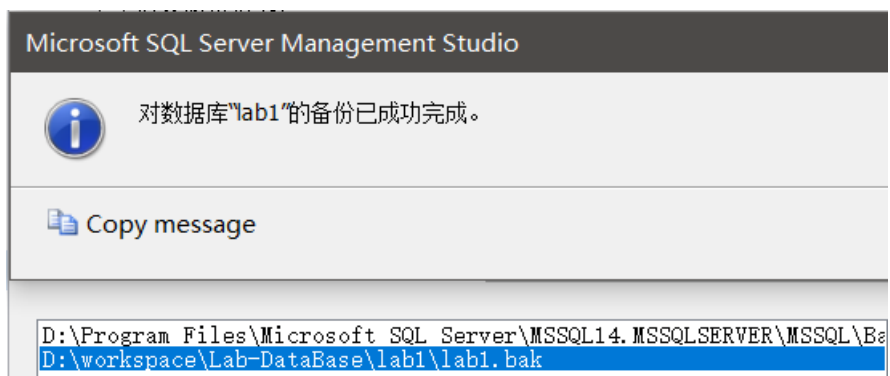


图 2-5 数据库备份位置

3) 选中数据库“lab1”，通过“右键-任务-还原-数据库”即可打开还原选项卡，选择刚才生成的.bak 文件即可还原。

### 2.2.3 新增用户

1) 选中“安全性”，“右键-新建-登录名”即可通过 SQL Server 身份验证创建一个新的用户，如下图 2-6;



图 2-6 新增用户

2) 给用户“ljq”授权如下图 2-7;

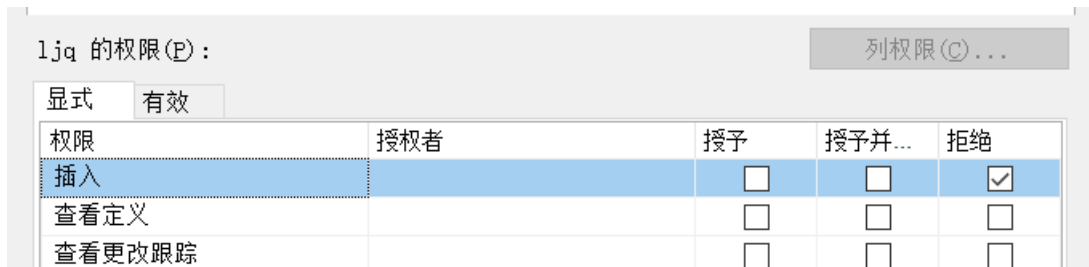


图 2-7 用户授权

3) 登录数据库，如下图 2-8;



图 2-8 登录数据库

## 2.3 任务总结

### 1) 脱机备份

- 现象：数据库 lab1 没有脱机时无法将数据库的数据文件 lab1.mdf 和日志文件 lab1\_log.mdf 拷贝到其他位置；
- 原因：数据库未进行脱机操作时，lab1.mdf 和 lab1\_log.mdf 被 SSMS 通过操作系统进行保护，无法进行拷贝；
- 解决方法：备份 lab1.mdf 和 lab1\_log.mdf 时先对数据库进行脱机操作即可。

### 2) 用户登录

- 现象：采用 SQL Server 身份验证登录时出现运行时错误，如图 2-9 下所示：

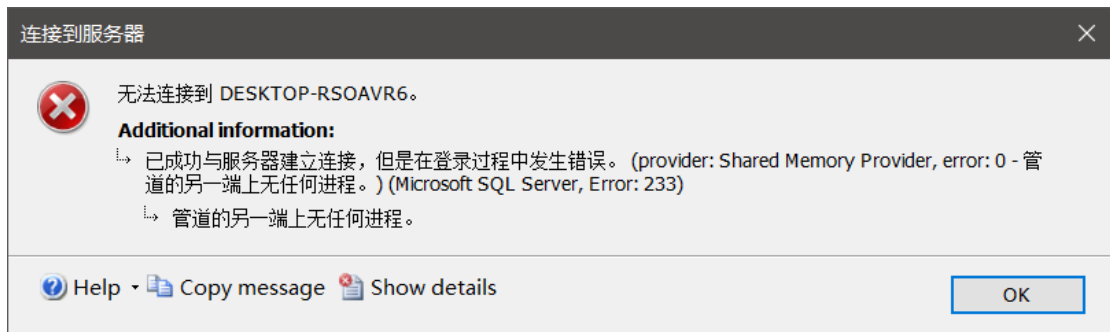


图 2-9 用户登录 Error233

- b) 原因：未开启 SQL Server 的 TCP/IP 协议和 Named Pipes 协议；
- c) 解决方法：可通过 Sql Server Configuration Manager 启用，如下图 2-10；

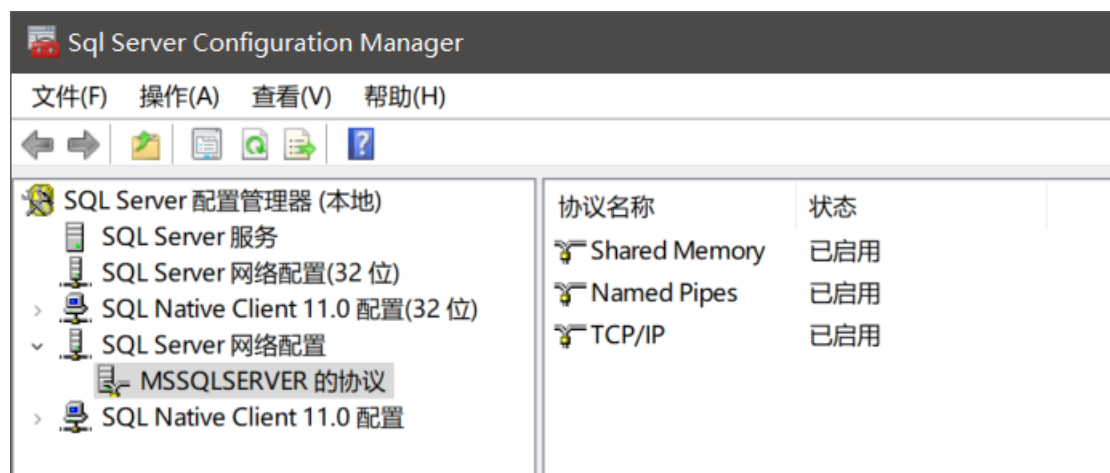


图 2-10 开启 MSSQLSERVER 协议

- 3) 身份验证
  - a) 现象：修改 SQL Server 登录协议后仍然无法连接到数据库，错误码提示 18456；
  - b) 原因：服务器配置为仅使用集成身份验证，SQL Server 运行日志如下图 2-11：

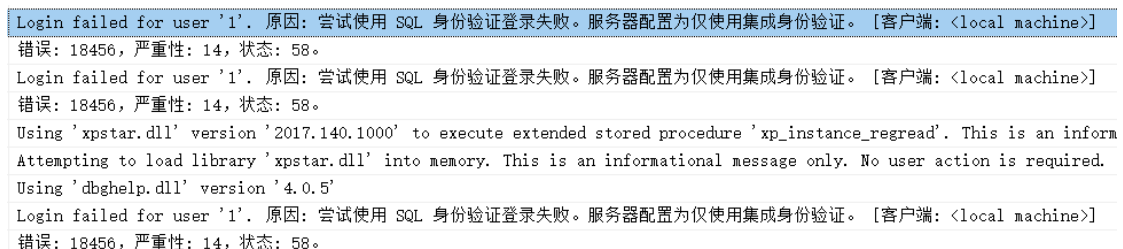


图 2-11 用户登录错误日志

- c) 解决方案：开启 SQL Server 用户验证，右键服务器，选择“属性-安全性”，选中“SQL Server 和 Windows 身份验证模式”即可，如下图 2-12：



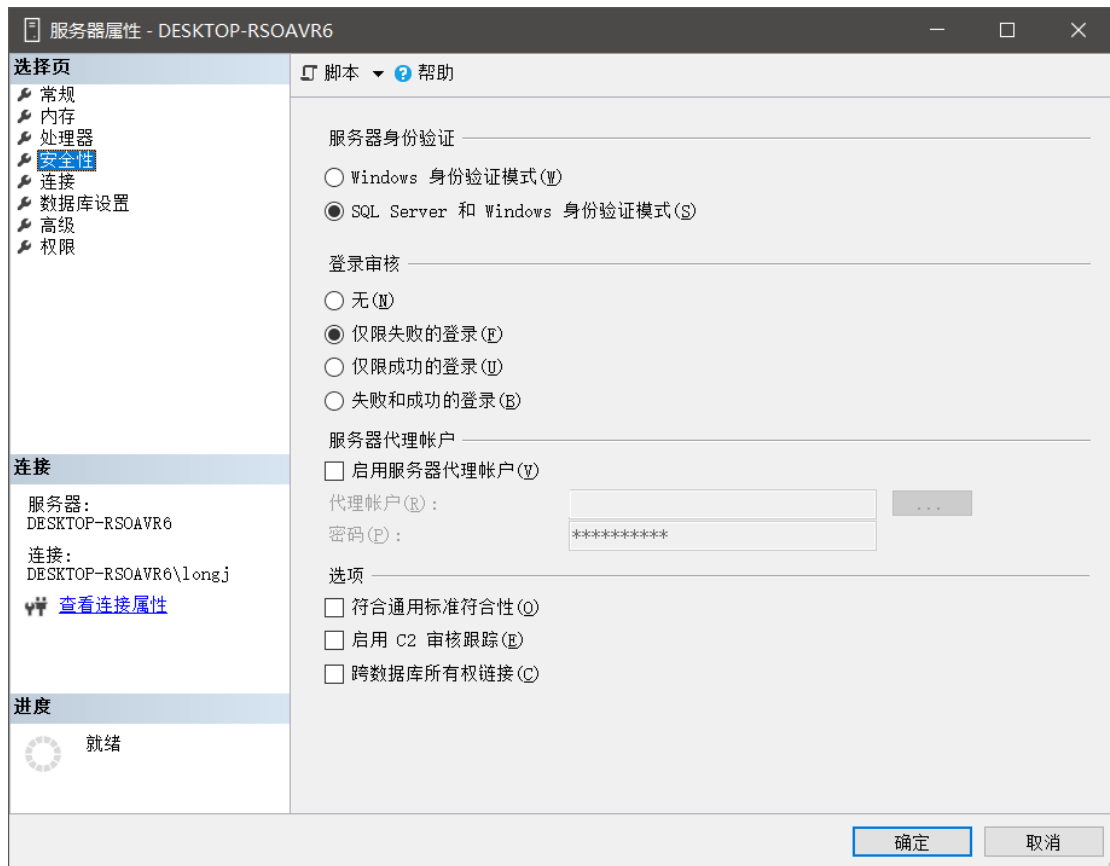


图 2-12 开启 SQL Server 身份验证

## 3 SQL 练习部分

### 3.1 任务要求

1. 建表：在 DBMS 中创建指定的关系表，包括主码和外码的说明，写出指定关系的建表 SQL 语句；
2. 数据更新：使用 SQL 语句对博文表增删改查、数据导入导出、编写触发器；
3. 查询：使用 SQL 语句完成相应的小题。

### 3.2 完成过程

#### 3.2.1 建表

- 1) 创建并使用数据库[weibo]:

```
create database weibo;
```

```
go
```

```
use weibo;
```

```
go
```

- 2) 其他关系表建表 SQL 语句类似，如下：

```
drop table if exists users;
```

```
create table users(
```

```
    uids int primary key,
```

```
    names char(30) not null,
```

```
    sex char(2) not null,
```

```
    byear int not null,
```

```
    city char(30) not null
```

```
);
```

```
drop table if exists label;
```

```
create table label(
```

```
    lid int primary key,
```

```
    lname char(30) not null
```

```
);
```

```
drop table if exists mblog;
```

```
create table mblog(
```

```
    bid int primary key,
```

```
    title char(30) not null,
```

```
    uids int not null,
```

```
    pyear int not null,
```

```
    pmonth int not null,
```

```
    pday int not null,
```

```
        cont text not null,  
        foreign key (uids) references users(uids)  
    );
```

```
drop table if exists b_l;  
create table b_l(  
    bid int not null,  
    lid int not null,  
    foreign key (bid) references mblog(bid),  
    foreign key (lid) references label(lid)  
);
```

```
drop table if exists follow;  
create table follow(  
    uids int not null,  
    uidfled int not null,  
    foreign key (uids) references users(uids),  
    foreign key (uidfled) references users(uids),  
    primary key (uids, uidfled)  
);
```

```
drop table if exists friends;  
create table friends(  
    uids int,  
    fuid int not null,  
    foreign key (uids) references users(uids),  
    foreign key (fuid) references users(fuid),  
    primary key (uids, fuid)  
);
```

```
drop table if exists sub;  
create table sub(  
    uids int not null,  
    lid int not null,  
    foreign key (uids) references users(uids),  
    foreign key (lid) references label(lid),  
    primary key (uids, lid)  
);
```

```
drop table if exists thumb;
```

```
create table thumb(
    uids int not null,
    bid int not null,
    foreign key (uids) references users(uids),
    foreign key (bid) references mblog(bid),
    primary key (uids, bid)
);
```

```
drop table if exists topday;
create table topday(
    tyear int not null,
    tmonth int not null,
    tday int not null,
    bid int not null,
    tno int not null,
    foreign key (bid) references mblog(bid)
);
```

### 3) 观察性实验

- a) 用户在订阅分类时一定要参考被参照关系的主码，如下图 3-1 byear 并不是 users 表的主码，SSMS 红线标错；

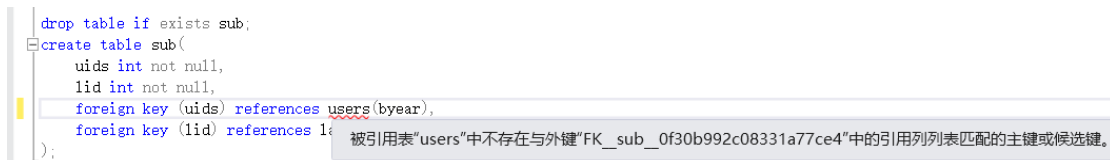


图 3-1 参照非主码

- b) 缺少参照，SSMS 仍然红线标错如下图 3-2。

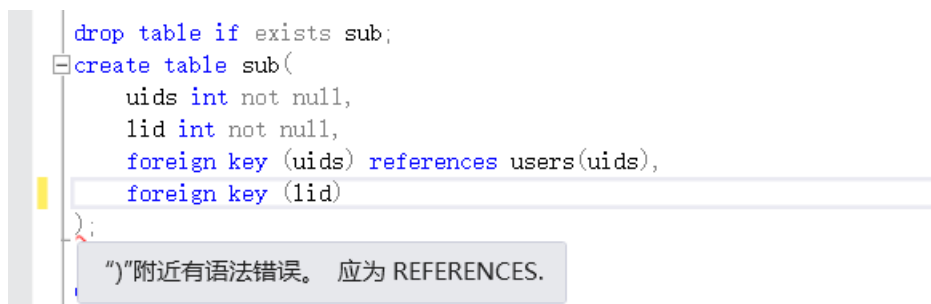


图 3-2 缺少参照

### 4) 数据准备（只展示了三组，皆为在 SSMS 中操作，其他类似）

- a) Users

DESKTOP-RSOAVR6...ibo - dbo.users						
	uids	names	sex	byear	city	
	1	赵一	男	1967	武汉	...
	2	钱二	女	1968	上海	...
	3	张三	男	1999	南京	...
	4	李四	女	2000	北京	...
	5	王五	男	2001	武汉	...
	6	孙六	女	2001	武汉	...
	7	周七	男	2002	武汉	...
	8	吴八	男	2002	杭州	...
	9	郑九	男	2003	南京	...
	10	冯十	女	2004	上海	...
	11	陈十一	男	2011	武汉	...
	12	诸十二	女	2012	北京	...
	13	卫十三	女	2011	上海	...
	14	蒋十四	女	2011	武汉	...
▶*	NULL	NULL	NULL	NULL	NULL	

图 3-3 user 数据

b) Label

DESKTOP-RSOAVR6...ibo - dbo.label		
	lid	lname
▶	1	文学
	2	艺术
	3	军事
	4	历史
	5	地理
	6	自然科学
	7	工程技术
	8	经济
	9	教育
	10	哲学
	11	音乐
*	NULL	NULL

图 3-4 label 数据

c) Mblog

DESKTOP-RSOAVR...ibo - dbo.mblog							
	bid	title	uids	pyear	pmonth	pday	cont
1	微博1	...	1	2019	4	18	"最多地铁站"和"华中科技大学"两个词
2	微博2	...	2	2019	4	18	华中科技大学是国家教育部直属重点综合性大学
3	微博3	...	3	2019	4	18	"最多地铁站"和"华中科技大学"两个词
4	微博4	...	4	2019	4	19	学校校园占地7000余亩
5	微博5	...	5	2019	4	19	校学科齐全、结构合理
6	微博6	...	6	2019	4	19	学校实施"人才兴校"战略
7	微博7	...	7	2019	4	19	"学生"的教育思想
8	微博8	...	8	2019	4	19	按照"应用领先"的科技发展方略
9	微博9	...	9	2019	4	19	学校坚持"贡献即发展"的办学思路
10	微博10	...	10	2019	4	19	学校坚持开放式办学理念
11	微博11	...	11	2019	4	19	《巢》："东欧文学的良心"流亡美国的坎坷人生
12	微博12	...	12	2019	4	19	派特·巴克代表作《重生三部曲》
13	微博13	...	13	2019	4	19	文学巨匠翻译的经典名著
14	微博14	...	14	2019	4	19	中国某新型核潜艇接受检阅
15	微博15	...	1	2019	4	20	中国核电装机容量升至全球第三
16	微博16	...	3	2019	4	20	海上阅兵活动开始 习近平检阅海上编队
17	微博17	...	5	2019	4	20	斯里兰卡警方：爆炸袭击案死亡人数升至321人
18	微博18	...	7	2019	4	20	辽宁舰升级改造 崭新亮相
19	微博19	...	9	2019	4	20	若是干掉慈禧荣禄，当年康党可赢
20	微博20	...	11	2019	4	20	三省制：唐代中书门下尚书职权之争

图 3-5 mblog 数据

### 3.2.2 数据更新

#### 1) 博文表增删改

##### a) 增

Insert into mblog values

(‘44’, ‘微博 44’, ‘1’, ‘2019’, ‘5’, ‘9’, ‘水花回暖勇士险胜火箭取赛点’)

查看 mblog 表，如下图 3-6：

	43	微博43	...	2	2019	4	21	西班牙画家...
▶	44	微博44	...	1	2019	5	9	险胜火箭取赛点

图 3-6 博文表增

##### b) 改

Update mblog set cont = ‘我改了一条记录啊’ where bid = ‘44’

查看 mblo 表，如下图 3-7：

	42	微博42	...	1	2019	4	21	意大利画家达芬奇...
	43	微博43	...	2	2019	4	21	西班牙画家巴勃罗...
▶	44	微博44	...	1	2019	5	9	我改了一条记录啊

图 3-7 博文表改

##### c) 删

Delete from mblog where bid = ‘44’

查看 mblo 表，如下图 3-8：

	41	微博41 ...	1	2019	4	21	西晋王朝短暂的大...
	42	微博42 ...	1	2019	4	21	意大利画家达芬奇...
▶	43	微博43 ...	2	2019	4	21	西班牙画家巴勃罗...
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

图 3-8 博文表删

## 2) 批处理操作

### a) 代码如下:

```
use weibo;
go
drop table if exists fans_3;
create table fans_3(
    uids int primary key,
    names char(30) not null,
    sex char(2) not null,
    byear int not null,
    city char(30) not null
);
insert into fans_3
(uids, names, sex, byear, city)
select users.uids, users.names, users.sex, users.byear, users.city
from users, follow
where users.uids = follow.uids and follow.uidfled = 3
```

### b) 操作后的 fans\_3 表如下图 3-9:

	uids	names	sex	byear	city
1	4	李四	女	2000	北京
2	6	孙六	女	2001	武汉
3	8	吴八	男	2002	杭州
4	10	冯十	女	2004	上海
5	11	陈十一	男	2011	武汉

图 3-9 批处理后的 fans\_3 表

## 3) 数据导入导出

除第一部分“脱机备份”以及“系统备份”外,还可以选择数据库“weibo”,通过“右键-任务-生成脚本”生成数据库的架构(DDL)和数据(DML),如下图 3-10:



图 3-10 导出数据库脚本

#### 4) 观察性实验

##### a) 代码如下：

```
use weibo;
go
create table observe(
    obj1 int,
    obj2 text
);
insert into observe values
(1, 'hahaha'),
(1, 'hahaha'),
(1, 'hahaha')
```

##### b) 更新后的表如下图 3-11：

	obj1	obj2
1	1	hahaha
2	1	hahaha
3	1	hahaha

图 3-11 没有主键的表

##### c) 对重复元组进行修改或者删除报错，如下图 3-12：



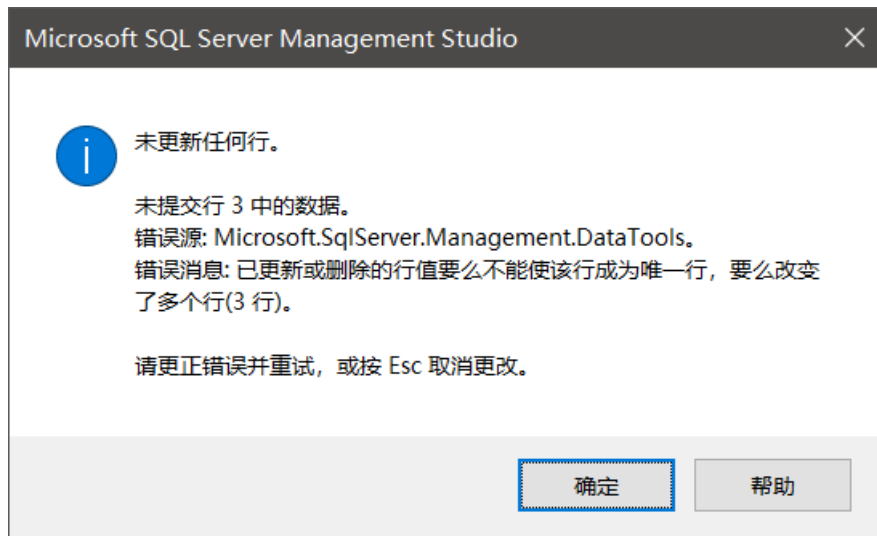


图 3-12 重复元组无法删除

## 5) 触发器实验

### a) 代码如下:

```
use weibo;
go
drop trigger if exists refuse_self_thumb
create trigger refuse_self_thumb on thumb
instead of insert, update
as
begin
    if exists (
        select *
        from inserted, mblog
        where inserted.bid = mblog.bid
        and inserted.uids = mblog.uids
    )
    begin
        print 'error, 不能给自己点赞'
    end
    else
    begin
        insert into thumb
        (uids, bid)
        select uids, bid
        from inserted
    end
end
go
```

b) 测试代码如下：

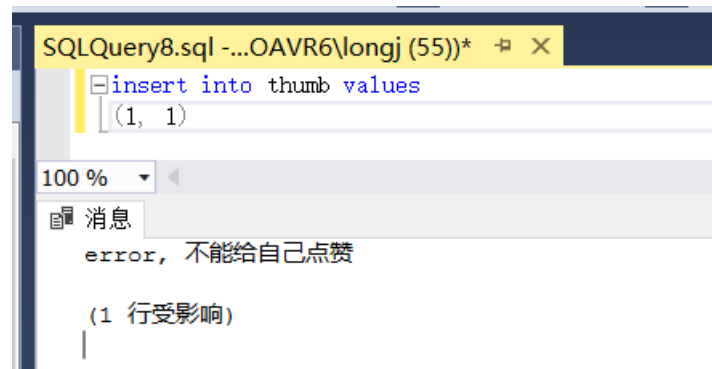


图 3-13 插入后触发器触发输出提示

### 3.2.3 查询

1) 查询“张三”用户关注的所有用户的 ID 号、姓名、性别、出生年份，所在城市，并且按照出生年份的降序排列，同一个年份的则按照用户 ID 号升序排列。

a) 代码：

```
select users.uids, users.names, users.sex, users.byear, users.city
from follow, users
where follow.uidfled = users.uids and follow.uids in(
    select users.uids
    from users
    where users.names = '张三'
)
order by users.byear desc, users.uids asc
```

b) 查询结果：

	uids	names	sex	byear	city
1	11	陈十一	男	2011	武汉
2	4	李四	女	2000	北京
3	2	钱二	女	1968	上海
4	1	赵一	男	1967	武汉

图 3-14

2) 查找没有被任何人点赞的博文 ID、标题以及发表者姓名，并将结果按照标题字符顺序排列。

a) 代码：

```
select mblog.bid, mblog.title, users.names
from mblog, users
where users.uids = mblog.uids and mblog.bid not in
(
    select bid from thumb
)
```

order by mblog.title asc

b) 查询结果:

	bid	title	names
1	1	微博1	赵一
2	2	微博2	钱二
3	3	微博3	张三
4	4	微博4	李四
5	5	微博5	王五
6	6	微博6	孙六

图 3-15

3) 查找 2000 年以后出生的武汉市用户发表的进入过头条的博文 ID;

a) 代码:

```
select mblog.bid
from mblog, users, topday
where  mblog.uids = users.uids and
       topday.bid = mblog.bid and
       users.byear > 2000 and
       users.city = '武汉'
```

b) 查询结果:

	bid
1	5
2	6
3	7
4	11
5	14
6	17
7	18
8	20
9	24
10	28
11	29
12	30
13	31
14	36
15	37

图 3-16

4) 查找订阅了所有分类的用户 ID;

a) 代码:

```
select sub.uids
from sub
group by uids
having count(*) = any(select count(*) from label);
```

b) 查询结果:

结果		消息
	uids	
1	1	

图 3-17

5) 查找出生年份小于 1970 年或者大于 2010 年的用户 ID、出生年份、所在城市，要求 where 子句中只能有一个条件表达式;

a) 代码:

```
select uids, byear, city
from users
where byear not between '1970' and '2010'
```

b) 查询结果:

	uids	byear	city
1	1	1967	武汉
2	2	1968	上海
3	11	2011	武汉
4	12	2012	北京
5	13	2011	上海
6	14	2011	武汉

图 3-18

6) 统计每个城市的用户数;

a) 代码:

```
select city, count(uids) '用户数'
from users
group by city
```

b) 查询结果:

	city	用户数
1	北京	2
2	杭州	1
3	南京	2
4	上海	3
5	武汉	6

图 3-19

7) 统计每个城市的每个出生年份的用户数，并将结果按照城市的升序排列，同一个城市按照出生用户数的降序排列其相应的年份;

a) 代码:

```
select city, byear, count(uids)
from users
group by byear, city
order by city, byear desc;
```

b) 查询结果:

	city	byear	用户数
1	北京	2012	1
2	北京	2000	1
3	杭州	2002	1
4	南京	2003	1
5	南京	1999	1
6	上海	2011	1
7	上海	2004	1
8	上海	1968	1
9	武汉	2011	2
10	武汉	2002	1
11	武汉	2001	2
12	武汉	1967	1

图 3-20

8) 查找被点赞数超过 10 的博文 ID 号;

a) 代码:

```
select bid
from thumb
group by bid
having count(bid) > 1 -- 数据里只有超过 1 的
```

b) 查询结果:

	bid
1	9
2	10
3	11
4	12
5	13

图 3-21

9) 查找被 2000 年后出生的用户点赞数超过 10 的博文 ID 号;

a) 代码:

```
select bid
from thumb, users
where thumb.uids = users.uids and
      users.byear >= 2000
group by bid
having count(bid) > 1
```

b) 查询结果:

	bid
1	10
2	11
3	12
4	13

图 3-22

10) 查找被 2000 年后出生的用户点赞数超过 10 的每篇博文的进入头条的次数;

a) 代码:

```
select topday.bid, count(*) as '次数'
from topday
where topday.bid in (
    select bid
    from thumb, users
    where thumb.uids = users.uids and
          users.byear >= 2000
    group by bid
    having count(bid) > 1
)
group by topday.bid
```

b) 查询结果:

	bid	次数
1	10	1
2	11	1
3	12	1
4	13	1

图 3-23

11) 查找订阅了文学、艺术、哲学、音乐中至少一种分类的用户 ID, 要求不能使用嵌套查询, 且 where 子句中最多只能包含两个条件;

a) 代码:

```
select distinct uids
from sub, label
where sub.lid = label.lid and
      label.lname in ('文学','艺术','哲学','音乐')
```

b) 查询结果:

	uids
1	1
2	2
3	4
4	5
5	6
6	7
7	8
8	9
9	10
10	11
11	12
12	13

图 3-24

12) 查找标题中包含了“最多地铁站”和“\_华中科技大学”两个词的博文基本信息；

a) 代码：

```
select bid, title, uids, pyear, pmonth, pday
from mblog
where cont like '%最多地铁站%' or
      cont like '%_华中科技大学%'
```

b) 查询结果：

	bid	title	uids	pyear	pmonth	pday
1	1	微博1	1	2019	4	18
2	3	微博3	3	2019	4	18

图 3-25

13) 查找所有相互关注的用户对的两个 ID 号，要求不能使用嵌套查询；

a) 代码：

```
select one.uids, two.uids
from follow one, follow two
where one.uids = two.uidfled and
      one.uidfled = two.uids and
      one.uids > two.uids
```

b) 查询结果：

	uids	uids
1	2	1
2	4	3
3	11	3

图 3-26

14) 查找好友圈包含了 5 号用户好友圈的用户 ID；

a) 代码：

```

select uids
from users
where not exists(
    select *
    from friends one
    where one.uids='5'and
        not exists(
            select *
            from friends two
            where two.uids = users.uids and
                one.fuid = two.fuid
        )
    )

```

b) 查询结果:

	uids
1	5

图 3-27

15) 查找 2019 年 4 月 20 日每一篇头条博文的 ID 号、标题以及该博文的每一个分类 ID，要求即使该博文没有任何分类 ID 也要输出其 ID 号、标题；

a) 代码:

```

select mblog.bid, mblog.title, b_l.lid
from topday, mblog, b_l
where topday.bid = mblog.bid and
    mblog.bid = b_l.bid and
    pyear = '2019' and
    pmonth = '4' and
    pday = '20'

```

b) 查询结果:

	bid	title	lid
1	15	微博15	3
2	16	微博16	3
3	17	微博17	3
4	18	微博18	3
5	19	微博19	4
6	20	微博20	4
7	21	微博21	6
8	22	微博22	6
9	23	微博23	6

图 3-28



16) 查找至少有 3 名共同好友的所有用户对的两个 ID 号。

a) 代码:

```
select one.uids, two.uids
from friends one, friends two
where one.uids <> two.uids and
      one.fuid = two.fuid and
      one.uids < two.uids
group by one.uids, two.uids
having count(*) > 1
```

b) 查询结果:

	uids	uids
1	4	12
2	5	13

图 3-29

17) 创建视图: 查阅 DBMS 内部函数, 创建一个显示当日热度排名前十的微博信息的视图, 其中的属性包括: 博文 ID、博文标题、发表者 ID、发表者姓名、被点赞数。

a) 代码:

```
drop view if exists topten;
go
create view topten as
select distinct topday.bid, mblog.title, users.uids, users.names, count(*)
as liking
from topday, mblog, users, thumb
where topday.tyear = (select DATENAME(YYYY, GETDATE()))
      and topday.tmonth = (select DATENAME(MM, GETDATE()))
      and topday.tday = (select DATENAME(DAY, GETDATE()))
      and topday.bid = mblog.bid
      and mblog.uids = users.uids
      and thumb.bid = mblog.bid
group by topday.bid, mblog.title, users.uids, users.names
go
```

b) 结果如下图 3-30:

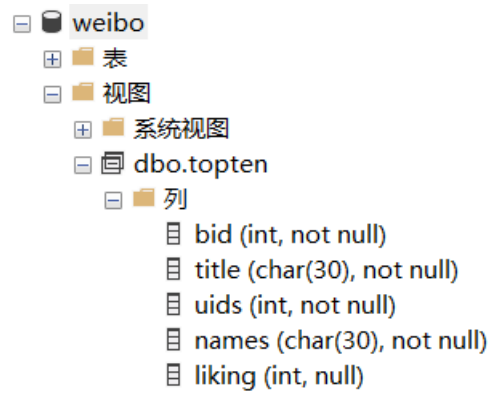


图 3-30 视图界面

c) 视图前 1000 行如下图 3-31:

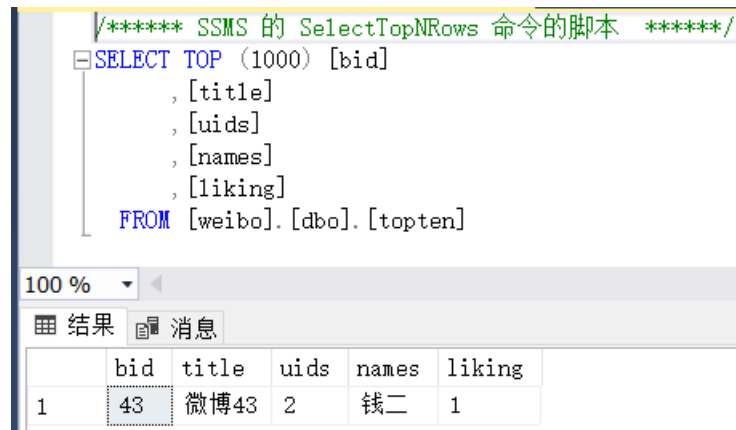


图 3-31 视图结果

### 3.3 任务总结

1. 我学会了如何在 SQL Server 中创建数据库和数据表；
2. 我学会了如何创建触发器来保证数据表之间的约束完整性；
3. 基本掌握了在数据库中检索数据的方法。

## 4 综合实践任务

### 4.1 系统设计目标

#### 1) 应用背景:

随着经济的发展,企业正在向着大型化、规模化发展,而对于大中型企业,员工、职称等跟工资管理相关的信息随着急剧增加。在这种情况下单靠人工来处理员工的工资不但显得力不从心,而且极容易出错。该系统就是设计一个小型企业工资的数据库管理系统,由计算机代替人工执行一系列诸如增加新员工,删除旧员工,工资查询,统计等操作。这样就使办公人员可以轻松快捷地完成工资管理的任务。

#### 2) 总体目标:

实现工资的集中管理。可供财务人员对本单位的人员以及工资进行增加、删除、修改、查询,对人事的管理及工资发放中的应发工资合计等项目由系统自动进行计算;同事本系统还可对人事及工资管理情况进行多角度查询。

### 4.2 需求分析

#### 1) 任务文档要求如下:

##### 1、系统功能的基本要求:

- 员工每个工种基本工资的设置
- 加班津贴管理,根据加班时间和类型给予不同的加班津贴;
- 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;
- 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;
- 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:

- 员工考勤情况表;
- 员工工种情况表,反映员工的工种、等级,基本工资等信息;
- 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;
- 员工基本信息表
- 员工月工资表。

设计一个 B/S 或 C/S 模式的系统实现上述功能。

#### 2) 总体功能需求列表如下所示:

功能编号	功能名称
1	增加考勤信息
2	删除考勤信息
3	修改考勤信息

4	查询考勤信息
5	增加工种信息
6	删除工种信息
7	修改工种信息
8	查询工种信息
9	增加津贴信息
10	删除津贴信息
11	修改津贴信息
12	查询津贴信息
13	增加员工基本信息
14	删除员工基本信息
15	修改员工基本信息
16	查询员工基本信息
17	员工年终奖生成
18	企业工资报表

### 4.3 总体设计

总体结构设计如下图 2-1 所示：

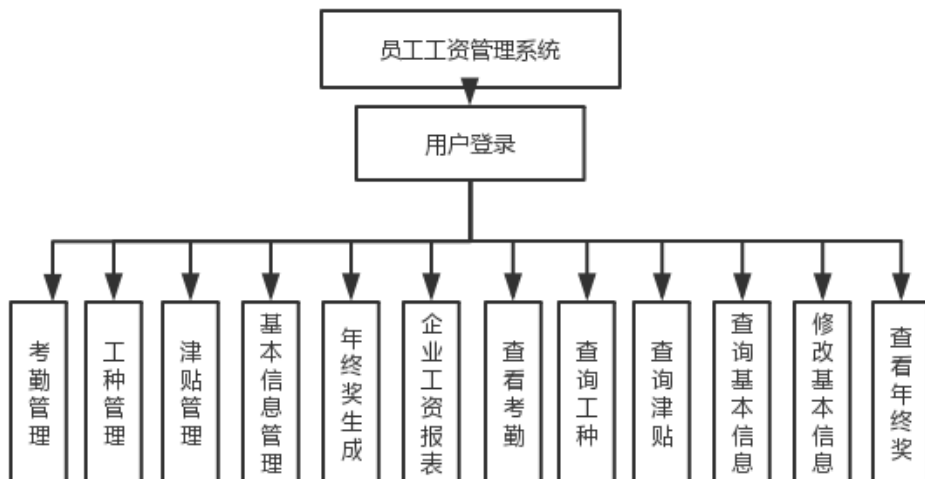


图 4-1 总体结构设计

### 4.4 数据库设计

1) ER 图如下图 4-2 所示：

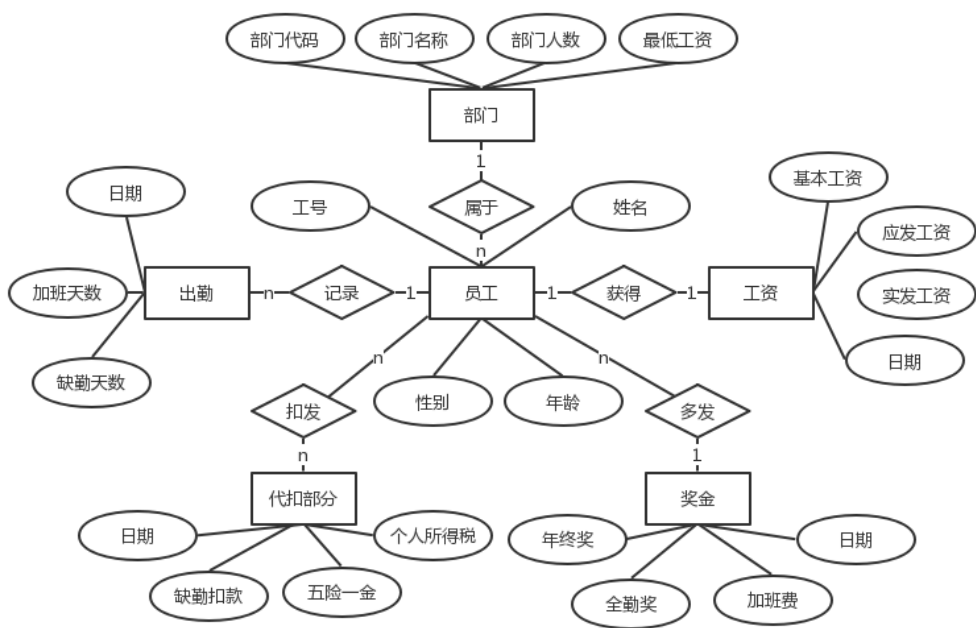


图 4-2 工资管理系统 ER 图

## 2) 数据库逻辑结构设计

- 员工基本信息表：员工姓名、员工密码、员工工号、员工性别、员工年龄、员工所属部门；
- 部门信息表：部门名称、部门代码、部门最低工资、部门人数；
- 员工工资信息表：员工工号、日期、基本工资、奖金、应发工资、代扣款、实发工资；
- 员工奖金信息表：员工工号、日期、全勤奖、加班费、年终奖；
- 员工扣发工资信息表：员工工号、日期、缺勤扣款、五险一金、税费；
- 出勤信息表：员工工号、日期、加班天数、缺勤天数；
- 账号信息表：用户 ID、用户名、用户密码、是否为管理员。

## 3) 数据库物理结构设计

表 4.1 员工基本信息表

字段名	类型	主键/外键	为空	索引	备注
员工姓名	char(20)		否	是	
员工工号	int	主键	否	是	
员工性别	char(2)				
员工年龄	int				
所属部门	int	外键	否	是	

表 4.2 部门信息表

字段名	类型	主键/外键	为空	索引	备注
部门名称	char(20)		否	是	
部门代码	int	主键	否	是	
最低工资	decimal(12,3)				
部门人数	int				

表 4.3 工资信息表

字段名	类型	主键/外键	为空	索引	备注
员工工号	int	外键	否	是	
日期	datetime		否		
基本工资	decimal(12,3)				
奖金	decimal(12,3)				
应发工资	decimal(12,3)				
代扣款	decimal(12,3)				
实发工资	decimal(12,3)				

表 4.4 奖金信息表

字段名	类型	主键/外键	为空	索引	备注
员工工号	int	外键	否	是	
日期	datetime			是	
全勤奖	decimal(12,3)				
加班费	decimal(12,3)				
年终奖	decimal(12,3)				

表 4.5 扣工资信息表

字段名	类型	主键/外键	为空	索引	备注
员工工号	int	外键	否	是	
日期	datetime			是	
缺勤扣款	decimal(12,3)				
五险一金	decimal(12,3)				
税费	decimal(12,3)				

表 4.6 出勤信息表

字段名	类型	主键/外键	为空	索引	备注
员工工号	int	外键	否	是	

日期	datetime			是	
加班天数	int				
缺勤天数	int				

表 4.7 账户信息表

字段名	类型	主键/外键	为空	索引	备注
用户 ID	int	主键	否	是	
用户名	char(128)		否	是	
用户密码	char(128)		否		
是否为管理员	bool		否		false-不是 true-是

## 4.5 详细设计与实现

### 4.5.1 总体架构

该系统采用 B/S 架构，采用前后端分离的方式进行开发，后端使用 Stack Overflow 网站上非常火的 Django 框架，Django 是一个 Python 的网络编程库，主要面向网站开发人员，值得一提的是，由于采用 Django 框架，该框架屏蔽了对底层数据库的连接细节，而是提供了对数据库增删改查的统一 API。

### 4.5.2 后端设计

系统后端设计主要分为以下四个部分：

- a) 路由分发；
- b) 表单设计；
- c) 请求处理；
- d) 数据库处理。

这四个部分也分别对应系统中几个不同的文件：

- a) url.py;
- b) forms.py;
- c) views.py;
- d) models.py。

路由分发指将不同的网址 URL 发送到对应的函数去进行处理，这部分功能由 url.py 实现，目前系统中支持的 URL 如下（path 中第一个参数）：

```
urlpatterns = [
    path('login', views.login, name = 'login'),
    path("", RedirectView.as_view(url = 'login')),
    path('index', views.index, name = 'index'),
```

```

    path('department', views.department, name = 'department'),
    path('department-create', views.department_create, name = 'department-
create'),
    path('department-update', views.department_update, name = 'department-
update'),
    path('department-delete', views.department_delete, name = 'department-
delete'),
    path('department-search', views.department_search, name = 'department-
search'),
    path('department-report', views.department_report, name = 'department-
report'),
    path('staff', views.staff, name = 'staff'),
    path('staff-create', views.staff_create, name = 'staff-create'),
    path('staff-update', views.staff_update, name = 'staff-update'),
    path('staff-delete', views.staff_delete, name = 'staff-delete'),
    path('staff-search', views.staff_search, name = 'staff-search'),
    path('staff-report', views.staff_report, name = 'staff-report'),
    path('attendance', views.attendance, name = 'attendance'),
    path('attendance-create', views.attendance_create, name = 'attendance-create'),
    path('attendance-update', views.attendance_update, name = 'attendance-
update'),
    path('attendance-delete', views.attendance_delete, name = 'attendance-delete'),
    path('attendance-search', views.attendance_search, name = 'attendance-search'),
    path('attendance-report', views.attendance_report, name = 'attendance-report'),
    path('wage', views.attendance, name = 'attendance'),
    path('yearending', views.yearending, name = 'yearending'),
    path('wage-report', views.wage_report, name = 'wage-report'),
]

```

表单设计部分对应文件 `forms.py`，为前后端交互提供接口，由于该系统较为简单，因此只设计了基本的登录表单、搜索表单以及创建对象表单，该部分具体介绍将在 4.6 系统测试中说明。

请求处理模块对应文件 `views.py`，处理 `url.py` 部分分发过来的 URL 网址请求，负责解析 URL，获取请求数据，生成相应数据并包装成网页发送到浏览器前端，以登录 URL——“login”为例，`views.login` 函数获取用户发来的表单数据中的用户名和密码，通过用户名去数据库中检索密码是否匹配，如果匹配，则跳转到系统主页，如果不匹配，输出错误信息，该功能实现如下：

```

def login(request):
    if 'wms_name' in request.POST and 'wms_password' in request.POST:

```



```

wms_name = request.POST['wms_name']
wms_password = request.POST['wms_password']
try:
    wms_acc = TAccount.objects.get(name = wms_name)
    if wms_acc.password != wms_password:
        messages.error(request, "User or Password error!")
        return render(request, 'wms/login.html')
    else:
        return render(request, 'wms/index.html')
except:
    messages.error(request, "Sorry, but this User does not exist!")
    return render(request, 'wms/login.html')
else:
    return render(request, 'wms/index.html')

```

数据库部分使用了 Django 框架的 `django.db.model`，每一个表都继承自 `django.db.models.Model`，这样做的好处是可以使用 Django 提供的 API 来处理底层数据库，而不必为专门的不同数据库编写查询脚本，以表 4.1 为例，为该数据表简历模型的语句如下：

```

class TStaff(models.Model):
    scode = models.AutoField(primary_key = True)
    sname = models.CharField(max_length = 20, blank = False)
    sgender = models.CharField(max_length = 2)
    sage = models.IntegerField()
    dcode = models.ForeignKey(TDepartment, on_delete = models.PROTECT)

```

查询文档可知对应的建表 SQL 语句为：

```

BEGIN;
--
-- Create model TStaff
--
CREATE TABLE "wms_tstaff" (
    "scode" integer NOT NULL PRIMARY KEY,
    "sname" varchar(20) NOT NULL,
    "spassword" varchar(20) NOT NULL,
    "sgender" varchar(2) NOT NULL,
    "dcode_id" integer NOT NULL REFERENCES "wms_tdepartment" ("dcode")
    DEFERRABLE INITIALLY DEFERRED
);

```

```
--  
CREATE INDEX "wms_tstaff_dcode_id_b3b68a8f" ON "wms_tstaff" ("dcode_id")  
COMMIT
```

要通过员工代码及外键部门代码检索员工所在部门的最低薪资的 API 为：

```
w = TStaff.objects.get(scode = 2).dcode.minimum_wage
```

### 4.5.3 前端设计

由于本人之前没有接触过前端设计，系统前端样式全部通过网络素材拼凑而成，因此整个系统样式不统一，重新修改前端样式较为繁琐加上时间精力有限，系统完成后也没有在对样式进行整改，由于不是本人设计，因此在这里也不再详细说明。

不过值得一提的是，我在系统中加入了打印报表功能，该功能可以打印丰富多彩的饼状图和柱状图等，并将之封装成了可重用的 js 库函数，这使得我以后可以有机会再次使用它，该部分将在系统测试中详细介绍。

## 4.6 系统测试

登录。

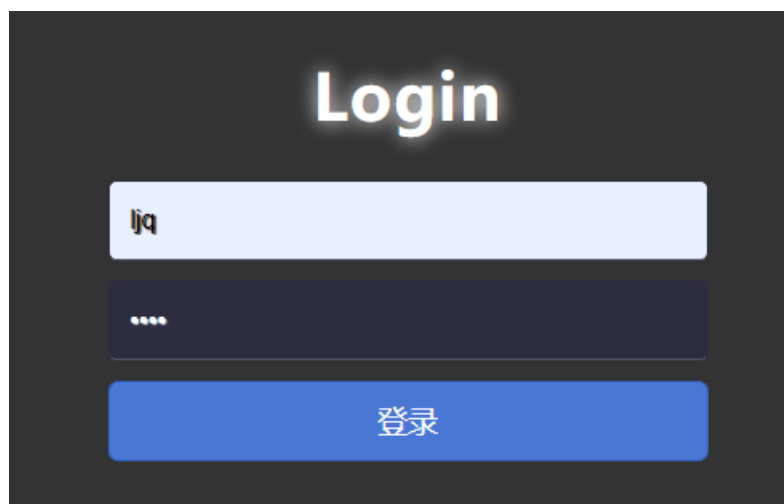


图 4-3 登录

首页。

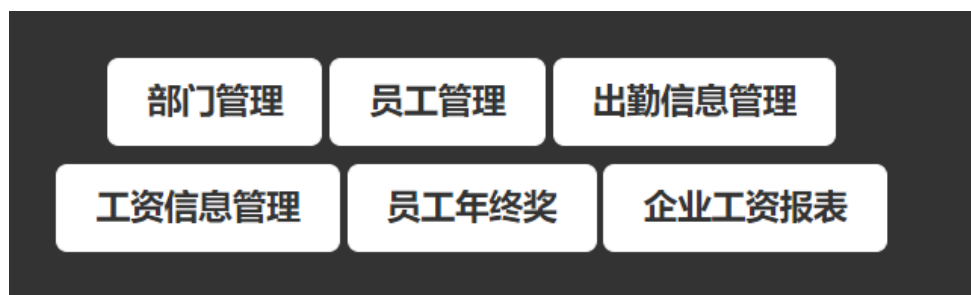


图 4-4 首页

部门管理首页（使用开源库 layui 其他数据表管理页面样式类似）。

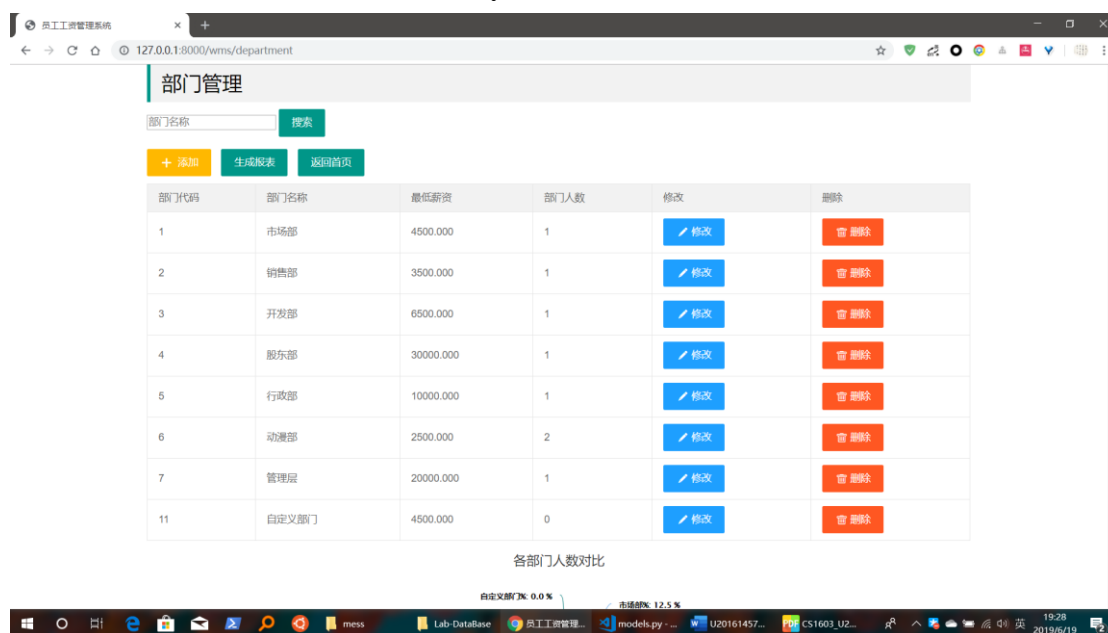


图 4-5 部门管理首页

添加部门。

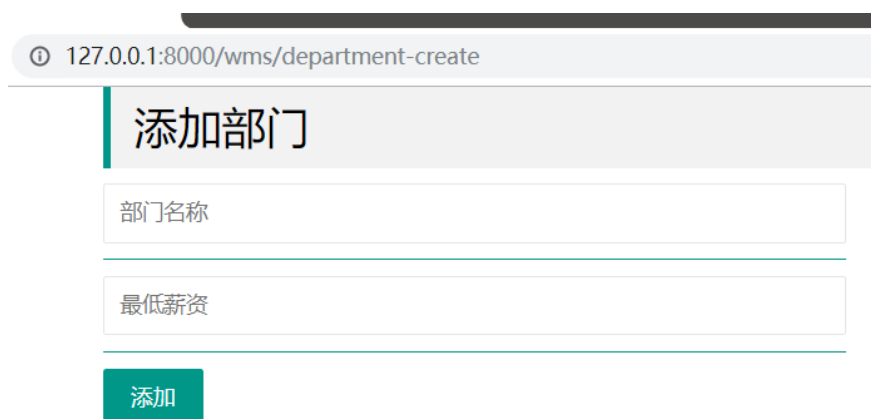


图 4-6 添加部门

修改刚刚创建的部门“数据库测试部门”，该网址 URL 中向网站后端发送“wms\_dcode”字段用以在数据库中检索。

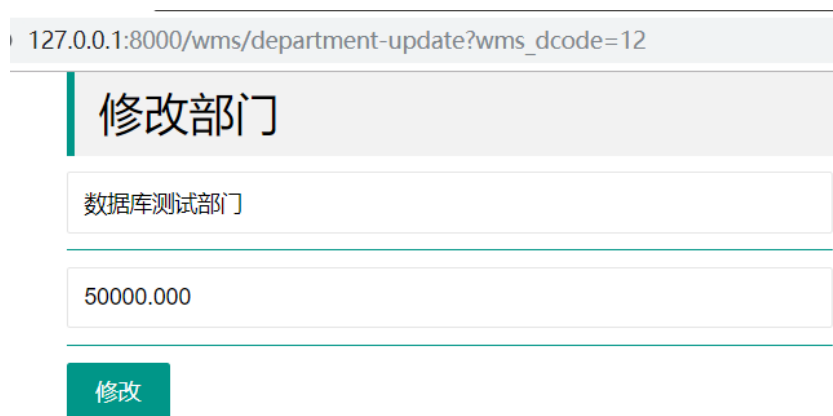


图 4-7 修改部门

各部门人数对比，可以看到，刚创建的部门“数据库测试部门”人数为 0，该界面使用优秀的 js 库 Highcharts。

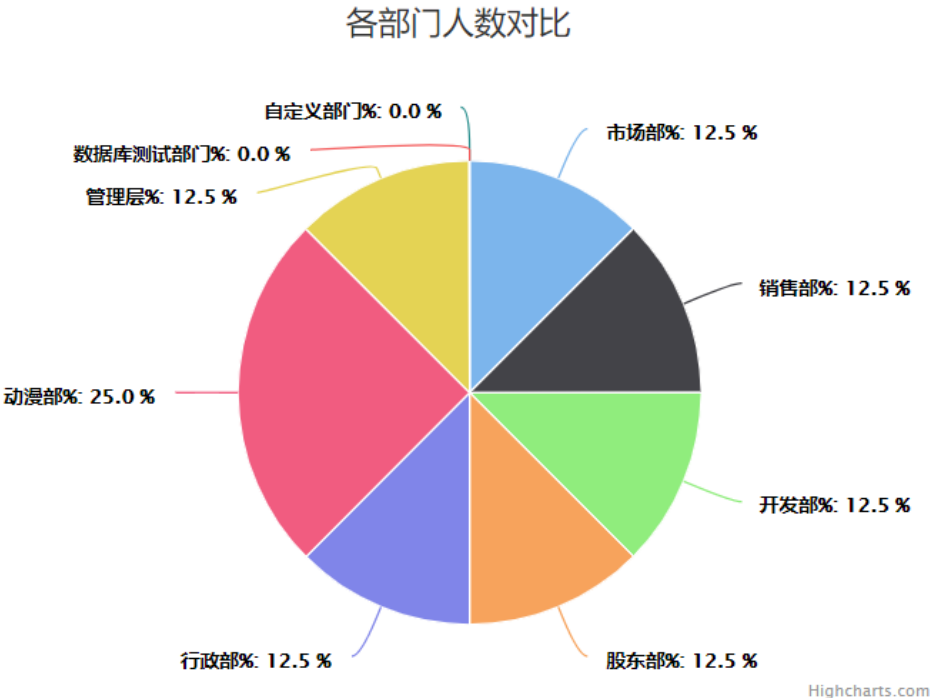


图 4-8 各部门人数对比

各部门最低薪资对比。

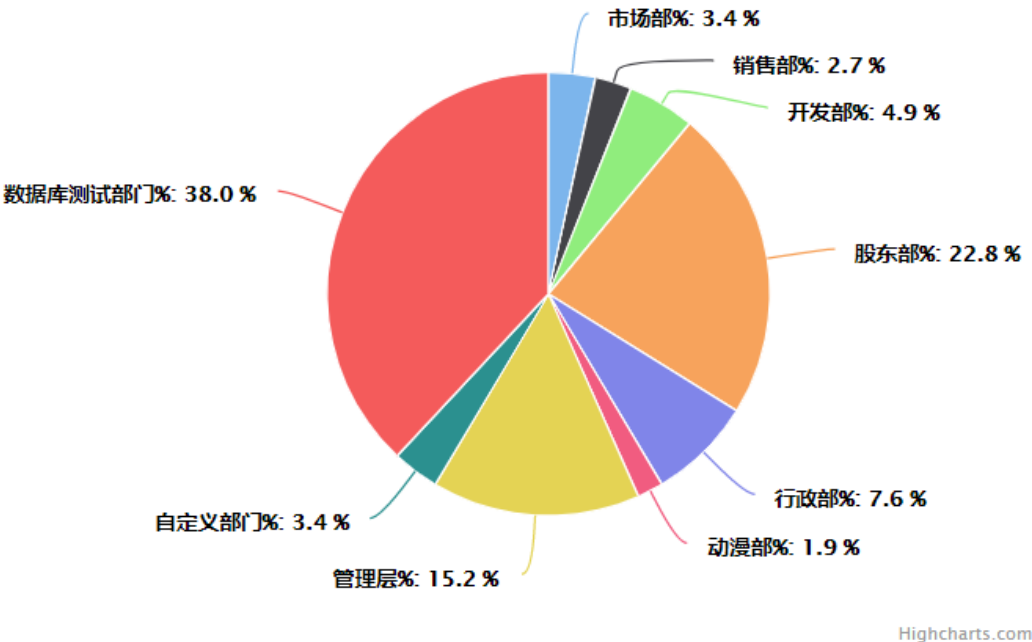


图 4-9 各部门最低薪资对比

PDF 报表生成。

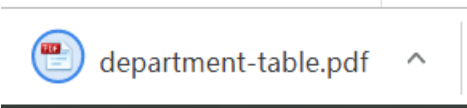


图 4-10 chrome 浏览器下载

报表内容如下（报表内容使用 reportlab 生成）。

部门代码	部门名称	最低薪资	部门人数
1	市场部	4500.000	1
2	销售部	3500.000	1
3	开发部	6500.000	1
4	股东部	30000.000	1
5	行政部	10000.000	1
6	动漫部	2500.000	2
7	管理层	20000.000	1
11	自定义部门	4500.000	0
12	数据库测试部门	50000.000	0

图 4-11 部门报表

查找对应的部门。



图 4-12 部门管理搜索功能

其他数据表功能、实现逻辑和界面基本类似，不再赘述。

## 4.7 系统设计与实现总结

- 1) 后端: django + sqlite + reportlab;
- 2) 前端: layui + highcharts;
- 3) 路由分发: url.py;
- 4) 请求处理: views.py;
- 5) 表单设计: forms.py;
- 6) 数据库设计: models.py;
- 7) 系统所有代码和 HTML 文件已上传到本人 github, 具体网址为  
<https://github.com/DragonDriver/Lab-DataBase/tree/master/lab3-wms>

## 5 课程总结

这次课程设计让我熟悉了具体的数据库应用系统设计，了解了 B/S 系统中前后端的开发，于此同时，我接触了很多优秀的开源库，这对我以后参加工作会有很大的好处。

这次课程设计也锻炼了我设计复杂与实现复杂系统的能力，我现在已经能够熟练地使用 `django` 框架开发一个小型的管理网站。