

**总体设计**

项目名称： 职业介绍信息管理系统

专业名称： 计算机应用技术

班 级： 17计专1班

开发小组组长： 杨钧升（17110546035）

开发组成员： 杨钧升（17110546035）

杨子慧（17110546037）

陈加楠（17110546002）

指导教师： 匡芳君、徐晓

**0、前言（一些必要的说明。）**

0.1 数据库说明

数据库名：JobIntroduce

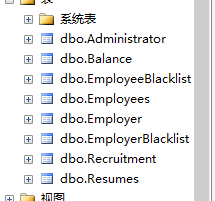
逻辑名称：职业介绍信息管理系统

数据文件：JobIntroduce.mdf

日志文件：JobIntroduce.ldf

登录名：sa,密码：123456

0.2表命名说明



0.3 系统功能模块图

余额管理

信息发布

公司查询

修改密码

黑名单管理

邀请

**职业介绍信息管理系统**

**求职者**

**企业**

**管理员**

用户管理

缴费管理

信息

发布

修改密码

简历查询

企业信息管理

求职者信息管理

修改密码

查询余额

修改余额

录用

黑名单

余额管理

充值

余额查询

充值

余额查询

投递简历

黑名单

1. 需求分析阶段

说明：职业介绍信息管理系统需要实现以下功能：

管理员可以审核管理删除招聘信息并且管理账户的缴费情况。黑名单将对违规操作用人单位及求职人员进行罚款级封号，如果一家企业或者求职者差评过多也将被加入黑名单。

一个企业可以发布多条招聘信息，一个岗位可以由多个求职者应聘。求职者可以投递简历给心仪企业，拉黑不喜欢不想去的企业，拉黑后该企业将无法邀请该求职者。管理员将信息审查通过后企业和求职者可以给账户充值。不同的用户根据身份不同拥有不同的权限。

1. 数据字典

（每个实体的详细说明）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号：JobIntroduce | | | | |
| 基表英文名称：Employer | | | | |
| 基表中文名称：企业表 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | EmployerID | 企业ID | Char(11) | 不为空 |
| 2 | Passwords | 密码 | Char(8) | 不为空 |
| 3 | EmpName | 企业姓名 | nvarchar（15） | 不为空 |
| 4 | Area | 地区 | nvarchar（7） | 不为空 |
| 5 | States | 缴费状态 | Int |  |
| 6 | EmpDescribes | 企业介绍 | Nvarchar(200) |  |
| 7 | JobDescribes | 岗位介绍 | Nvarchar(100) |  |
| 8 | Lincense | 营业执照 | Char（20） | 不为空 |
| 9 | Blacklist | 是否被列为黑名单 | Int |  |
| 10 | Email | 电子邮箱 | nvarchar（20） |  |
| 11 | Tel | 企业号码 | nvarchar（11） | 不为空 |
| 说明：缴费状态默认为0，黑名单状态默认为0 | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号：JobIntroduce | | | | |
| 基表英文名称：Administrator | | | | |
| 基表中文名称：管理员表 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | AdminID | 管理员账户 | Char(11) | 不为空 |
| 2 | Password | 密码 | nchar(16) | 不为空 |
| 说明： | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号：JobIntroduce | | | | |
| 基表英文名称：Resumes | | | | |
| 基表中文名称：简历表 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | ResumesID | 简历ID | Char(11) | 不为空 |
| 2 | EmployerID | 企业ID | Char(11) | 不为空 |
| 3 | EmpliyeeID | 求职者ID | Char(11) | 不为空 |
| 4 | EmpDescribe | 简历描述 | Nvarchar（50） |  |
| 说明： | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号：JobIntroduce | | | | |
| 基表英文名称：Recruitment | | | | |
| 基表中文名称：招聘信息表 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | RecruitmentID | 招聘id | char(11) | 不为空 |
| 2 | EmployerID | 企业ID | char(11) | 不为空 |
| 3 | Salary | 工资 | Money | 不为空 |
| 4 | JobDescribe | 岗位描述 | Nvarchar（50） | 不为空 |
| 5 | States | 审核状态 | int | 不为空 |
| 说明：kch参照课程表的kch。 | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号：JobIntroduce | | | | |
| 基表英文名称：Balance | | | | |
| 基表中文名称： 余额表 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | AccountID | 账户ID | Char(11) | 不为空 |
| 2 | Balance | 余额 | Char(3) | 不为空 |
| 说明： | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号JobIntroduce | | | | |
| 基表英文名称：EmployeeBlacklist | | | | |
| 基表中文名称： 求职黑名单 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | Employeeid | 求职ID | Char（11） | 不为空 |
| 2 | EmployerID | 企业ID | Char(11) | 不为空 |
| 3 | Balance | 余额 | Money | 不为空 |
| 说明： | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号JobIntroduce | | | | |
| 基表英文名称：EmployerBlacklist | | | | |
| 基表中文名称： 企业黑名单 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | EmployerID | 企业ID | Char(11) | 不为空 |
| 2 | Employeeid | 求职ID | Char（11） | 不为空 |
| 说明： | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 数据库编号JobIntroduce | | | | |
| 基表英文名称：score | | | | |
| 基表中文名称： 评分表 | | | | |
| 字段编号 | 英文字段名 | 中文字段名 | 字段类型 | 备注 |
| 1 | scoreID | 评分记录ID | Char(11) | 不为空 |
| 2 | Kind | 评分类型 | Char（2） | 不为空 |
| 3 | Objectid | 评分对象ID | Char（11） | 不为空 |
| 4 | Score | 分数（1-10） | Int | 不为空 |
| 说明： | | | | |

1. 概念设计阶段
2. 分ER图

（两个分ER图，1）学生和课程，2）教师，课程，班级）

**成绩**

**选课**

**N**

**M**

**课程**

**课程名称**

**开课系编号**

**课程号**

**学分**

**学生选课人数**

**班级名称**

**学生**

**性别**

**出生时间**

**姓名**

**学号**

**总学分**

**备注**

**联系方式**

**照片**

**专业**

**教师**

**教师姓名**

**专业**

**教师编号**

**分院**

**职称**

**职务**

**备注**

**课程**

**课程名称**

**开课系编号**

**课程号**

**学分**

**学生选课人数**

**班级**

M

**N**

**授课**

P

**时间**

**地点**

1. 总ER图

（由分ER图画出总ER图）

**教师**

**教师姓名**

**专业**

**教师编号**

**分院**

**职称**

**职务**

**备注**

**课程**

**课程名称**

**开课系编号**

**课程号**

**学分**

**学生选课人数**

M

**N**

**授课**

**时间**

**地点**

**学生**

**性别**

**出生时间**

**姓名**

**学号**

**总学分**

**备注**

**联系方式**

**照片**

**专业**

**班级名称**

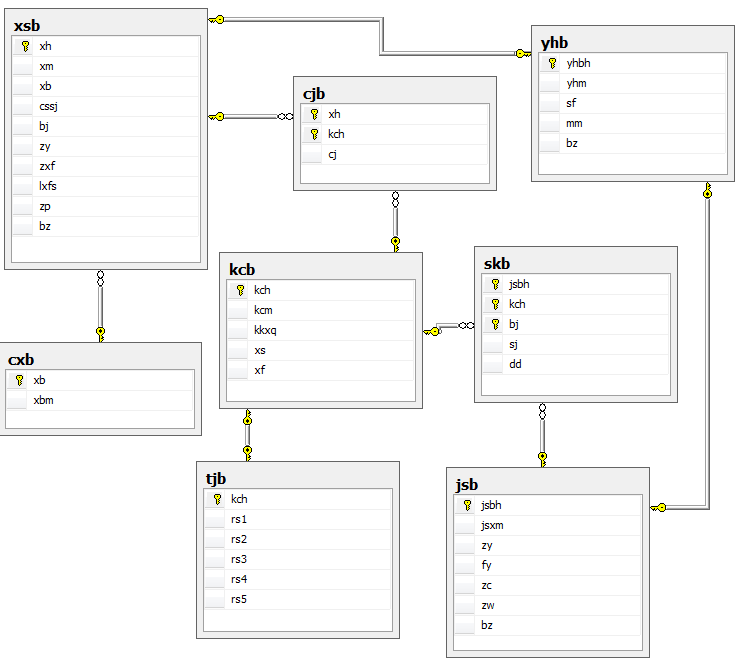
N

**M**

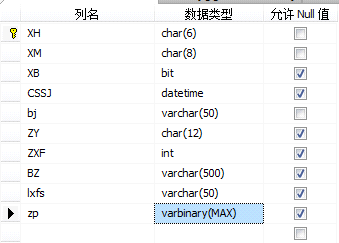
**选课**

**成绩**

1. 逻辑设计阶段
2. 表关系图



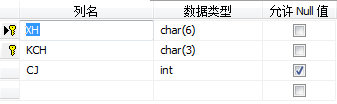
1. 表结构图



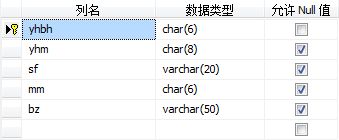
Xsb结构



Kcb结构



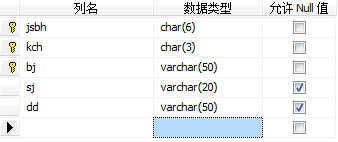
Cjb结构



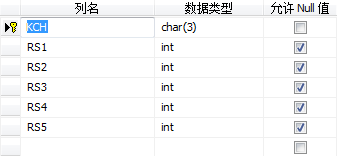
Yhb结构



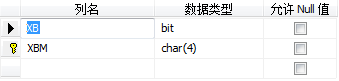
Jsb结构



Skb结构



Tjb结构



Cxb结构

1. 表优化（判断每个关系是否达到3NF要求，如果没有达到，则继续规范）

按照（1nf->2nf->3nf进行判断和优化）

Cjb（xh,kch,cj）

码：（xh,kch）

非主属性：cj

因为cj完全依赖于（xh,kch），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Cxb(xb,xbm)

码：（xbm）

非主属性：xb

因为xb完全依赖于（kch），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Jsb(jsbh,jsxm,zy,fy,zc,zw,bz)

码：（jsbh）

非主属性：jsxm,zy,fy,zc,zw,bz

因为jsxm,zy,fy,zc,zw,bz完全依赖于（kch），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Kcb(kch,kcm,kkxq,xs,xf)

码：（kch）

非主属性：kcm,kkxq,xs,xf

因为kcm,kkxq,xs,xf完全依赖于（kch），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Skb(jsbh,kch,bj,sj,dd)

码：（jsbh,kch,bj）

非主属性：sj,dd

因为sj,dd完全依赖于（jsbh,kch,bj），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Tjb(kch,rs1,rs2,rs3,rs4,rs5)

码：（kch）

非主属性：rs1,rs2,rs3,rs4,rs5

因为rs1,rs2,rs3,rs4,rs5完全依赖于（kch），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Xsb(xh,xm,xb,cssj,bj,zy,zxf,bz,lxfs,zp)

码：（xh）

非主属性：xm,xb,cssj,bj,zy,zxf,bz,lxfs,zp

因为xm,xb,cssj,bj,zy,zxf,bz,lxfs,zp完全依赖于（xh），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

Yhb(yhbh,yhm,mm,bz)

码：（yhbh）

非主属性：yhm,mm,bz

因为yhm,mm,bz完全依赖于（yhbh），所以属于2NF

因为不存在传递函数依赖，所以属于3NF

4、物理设计

选择合适的DBMS（要求用sql server 2008）

5、实施（把sql语句贴在下面）

（1）创建数据库

USE master --设置当前数据库为master，以便访问sysdatabases表

GO

IF EXISTS(SELECT \* FROM sysdatabases WHERE name ='PXSCJ')

DROP DATABASE PXSCJ

GO

create database PXSCJ

on primary

(

name=PXSCJ\_data,

filename='E:\DB\PXSCJ.mdf',

size=10,

maxsize=unlimited,

filegrowth=10%

)

log on

(

name=PXSCJ\_log,

filename='E:\DB\PXSCJ\_Log',

size=1,

maxsize=5,

filegrowth=1

)

Go

use PXSCJ

go

（2）创建表

--用户表

create table yhb

(

yhbh char(6) primary key not null,

yhm char(8) not null,

sf varchar(20) not null,

mm char(6) not null,

bz varchar(5)

)

go

--成绩表

create table cjb

(

xh char(6),

kch char(3),

cj int,

constraint c1 primary key(xh,kch)

)

go

--学生表

create table xsb

(

xh char(6) not null primary key,

xm char(8) not null,

xb bit not null,

cssj datetime,

bj varchar(50) not null,

zy char(12),

zxf int not null,

lxfs varchar(50),

zp varbinary(max),

bz varchar(500)

)

go

--教师表

create table jsb

(

jsbh char(6) not null primary key,

jsxm char(8) not null,

zy varchar(30) not null,

fy varchar(30) not null,

zc varchar(20) not null,

zw varchar(20) not null,

bz varchar(50)

)

go

--课程表

create table kcb

(

kch char(3) not null primary key,

kcm char(16) not null,

kkxq int not null,

xs int not null,

xf int not null

)

go

--授课表

create table skb

(

jsbh char(6) not null,

kch char(3) not null,

bj varchar(50) not null,

sj varchar(20),

dd varchar(50),

constraint c2 primary key(jsbh,kch,bj)

)

go

--成绩统计表

create table tjb

(

kch char(3) primary key not null,

rs1 int,

rs2 int,

rs3 int,

rs4 int,

rs5 int

)

go

--性别查询表

create table cxb

(

xb bit not null primary key,

xbm char(4) not null

)

go

（3）数据库完整性

1）索引（如已先建主键，则主键就是唯一聚集索引）

--另如：为课程表的课程名称创建一个索引

Create unique Index Ix\_kcm

on KCB(kcm)

go

2）视图

--学生选课情况视图（学号，姓名，课程号，课程名，成绩，学分，总学分，教师）

create view xs\_xk\_view

as

select xsb.XH,XM,kcb.KCH,kcm,CJ,xf,zxf,jsb.jsbh,jsxm

from xsb,KCB,CJB,jsb,skb

where xsb.XH=cjb.XH and kcb.KCH=cjb.KCH

and cjb.KCH=skb.kch and skb.jsbh=jsb.jsbh

go

select \* from xs\_xk\_view

GO

3）自定义数据库类型

--如：学号，课程号可以考虑用自定义的数据类型。

exec sp\_addtype 'xh','char(6)'

go

exec sp\_addtype 'kch','char(3)'

go

4）默认值约束

Xsb:性别默认为1，总学分默认为0。

Kcb:学生人数默认为40，学分默认为2。

Jsb:zy默认为计算机，fy默认为信息，zc默认为讲师，zw默认为无。

Yhb:mm默认为’123456’，身份默认为学生。

--Xsb:性别默认为1。

alter table xsb

add constraint b1 DEFAULT 1 for xb

go

--总学分默认为0。

alter table xsb

add constraint b2 DEFAULT 0 for zxf

go

--Kcb:学生人数默认为40，

alter table kcb

add constraint b3 DEFAULT 40 for xs

go

--学分默认为2。

alter table kcb

add constraint b4 DEFAULT 2 for xf

go

--Jsb:zy默认为计算机，

alter table jsb

add constraint b5 DEFAULT '计算机' for zy

go

--fy默认为信息，

alter table jsb

add constraint b6 DEFAULT '信息' for fy

go

--zc默认为讲师，

alter table jsb

add constraint b7 DEFAULT '讲师' for zc

go

--zw默认为无。

alter table jsb

add constraint b8 DEFAULT '无' for zw

go

--Yhb:mm默认为’123456’，

alter table yhb

add constraint b9 DEFAULT '123456' for mm

go

--yhb：身份默认为学生

alter table yhb

add constraint b10 DEFAULT '学生' for sf

go

5）规则和check约束

--Xsb:zxf在-160之间。

alter table xsb

add constraint pk\_1 check (zxf between 0 and 160)

go

--Kcb:kkxq在-8之间

alter table kcb

add constraint pk\_2 check (kkxq between 1 and 8)

go

--Kcb:xf在-15之间

alter table kcb

add constraint pk\_3 check (xf between 1 and 15)

go

--Cjb:cj在-100之间

alter table cjb

add constraint pk\_4 check (cj between 0 and 100)

go

6)参照关系

/\*

xsb(xb)参照cxb(xb)

cjb(xh)参照xsb(xh)

cjb(kch)参照kcb(kch)

skb(jsbh)参照jsb(jsbh)

skb(kch)参照kcb(kch)

tjb(kch)参照kcb(kch)

skb(bj)参照xsb(bj) –-用触发器实现

yhb(yhbh)参照xsb(xh)和jsb(jsbh) –必须用触发器实现（当学生表或教师表有数据插入时在用户表中进行相应的用户插入操作）此内容在触发器设计中完成。

\*/

--xsb(xb)--cxb(xb)

alter table xsb

add constraint fk\_0 foreign key (xb) references cxb (xb)

go

--cjb(xh)—xsb(xh)

alter table cjb

add constraint fk\_1 foreign key (xh) references xsb (xh)

go

--cjb(kch)—kcb(kch)

alter table cjb

add constraint fk\_2 foreign key (kch) references kcb (kch)

go

--skb(jsbh)—jsb(jsbh)

alter table skb

add constraint fk\_3 foreign key (jsbh) references jsb (jsbh)

go

--skb(kch)—kcb(kch)

alter table skb

add constraint fk\_4 foreign key (kch) references kcb (kch)

go

--tjb(kch)—kcb(kch)

alter table tjb

add constraint fk\_5 foreign key (kch) references kcb (kch)

go

--skb(bj)—xsb(bj) --用触发器实现

create trigger t1 on skb

for insert

as

if not exists(select \* from xsb where bj=(select bj from inserted))

begin

print '插入的班级不存在！'

rollback transaction

end

go

--yhb(yhbh)参照xsb(xh)和jsb(jsbh) –必须用触发器实现（当学生表或教师表有数据插入时在用户表中进行相应的用户插入操作）此内容在触发器设计中完成。

（4）用户自定义函数

--如：根据输入的课程号查询课程名。

create function bmrs(@kch as nvarchar(20))

returns nvarchar(20)

as

begin

declare @kcm nvarchar

select kcm from kcb where kch=@kch

return @kcm

end

go

（5）用户自定义存储过程

1）根据学号查询学生的选课情况

CREATE PROC xhcxxs @xh char(6)

as

if exists(select \* from cjb where xh=@xh)

select \* from cjb where xh=@xh

else

print '无此学生选课信息'

Go

2）根据课程号查询课程的选修情况

CREATE PROC kchcxxk @kch char(3)

as

if exists(select \* from cjb where kch=@kch)

select \* from cjb where kch=@kch

else

print '无此课程选课信息'

go

3）根据教师编号查询授课情况

CREATE PROC jsbhcxsk @jsbh char(6)

as

if exists(select \* from skb where jsbh=@jsbh)

select \* from skb where jsbh=@jsbh

else

print '无此教师授课信息'

go

-4）根据班级查询该班级的授课情况

CREATE PROC bjcxsk @bj varchar(50)

as

if exists(select \* from skb where bj=@bj)

select \* from skb where bj=@bj

else

print '无此班级授课信息'

go

5）根据课程号号查看授课情况

CREATE PROC kchcxsk @kch char(3)

as

if exists(select \* from skb where kch=@kch)

select \* from skb where kch=@kch

else

print '无此课程授课信息'

go

6）课程成绩分布统计。存储过程名称TJ\_CJ。参数：课程号（@kch）。实现功能：把成绩表（CJB）中指定课程按照分数段人数进行统计，放入统计表（TJB）中。

编写思路：（1）清空TJB表，插入一行所有分数段的人数都为的所要查找的课程的记录。（2）判断所查的课程号在CJB表中是否有记录，若有则查找出各个分数段的人数并且更新到TJB表中。

--方法一：

Create PROC TJ\_CJ @kch char(3)

as

--delete tjb

/\*if exists (select \* from CJB where KCH=@kch)

begin

if @kch not in (select kch from TJB)

insert into TJB values(@kch,0,0,0,0,0)

update TJB set RS1=(select count(\*) from CJB where CJ>=0 and CJ<60 and KCH=@kch)where KCH=@kch

update TJB set RS2=(select count(\*) from CJB where CJ>=60 and CJ<70 and KCH=@kch) where KCH=@kch

update TJB set RS3=(select count(\*) from CJB where CJ>=70 and CJ<80 and KCH=@kch) where KCH=@kch

update TJB set RS4=(select count(\*) from CJB where CJ>=80 and CJ<90 and KCH=@kch) where KCH=@kch

update TJB set RS5=(select count(\*) from CJB where CJ>=90 and CJ<=100 and KCH=@kch) where KCH=@kch

end

\*/

if exists(select kch from cjb where kch = @kch)

begin

if @kch not in (select kch from TJB)

insert into TJB values(@kch,0,0,0,0,0)

declare @rs1 int,@rs2 int,@rs3 int,@rs4 int,@rs5 int

select @rs1=COUNT(kch) from cjb where cj<60 and kch = @kch

select @rs2=COUNT(kch) from cjb where cj<70 and cj>=60 and kch = @kch

select @rs3=COUNT(kch) from cjb where cj<80 and cj>=70 and kch = @kch

select @rs4=COUNT(kch) from cjb where cj<90 and cj>=80 and kch = @kch

select @rs5=COUNT(kch) from cjb where cj<=100 and cj>=90 and kch = @kch

update tjb

set rs1=@rs1,rs2=@rs2,rs3=@rs3,rs4=@rs4,rs5=@rs5

where kch = @kch

end

else

begin

print '统计的课程不存在！'

rollback transaction

end

go

exec TJ\_CJ '208'

go

Select \* From CJB

GO

Select \* From TJB

GO

方法二：

如果选择统计的课程在成绩表（CJB）已经存在，可以采用case代码完成统计功能，并且代码效率较高。但CJB表中没有所选择的课程记录，执行的结果是各个分数段的人数为null。

Alter procedure TJ\_CJ (@KCH varchar(3))

as

--delete TJB

if exists (select \* from CJB where KCH=@kch)

insert into TJB(KCH,RS1,RS2,RS3,RS4,RS5)

select @KCH,

sum(case when CJ < 60 then 1 else 0 end),

sum(case when CJ >= 60 and CJ <= 69 then 1 else 0 end),

sum(case when CJ >= 70 and CJ <= 79 then 1 else 0 end),

sum(case when CJ >= 80 and CJ <= 89 then 1 else 0 end),

sum(case when CJ >= 90 and CJ <= 100 then 1 else 0 end)

from CJB where KCH = @KCH

else

print '统计的课程不存在！'

rollback transaction --transaction可省略

go

exec TJ\_CJ '302'

go

（6）触发器

1）实现yhb(yhbh)—xsb(xh)和jsb(jsbh)参照关系–必须用触发器实现（当学生表或教师表有数据插入时在用户表中进行相应的用户插入操作）

create trigger yhxh\_tigger

on xsb for insert

as

declare @xh char(6),@xm varchar(10)

select @xh=xh,@xm=xm from inserted

insert yhb(yhbh,yhm,sf)

values(@xh,@xm,'学生')

go

create trigger yhjs\_tigger

on jsb for insert

as

declare @jsbh char(6),@jsxm varchar(10)

select @jsbh=jsbh,@jsxm=jsxm from inserted

insert yhb(yhbh,yhm,sf)

values(@jsbh,@jsxm,'教师')

go

2）实现更新学生信息和老师信息的时候用户表中的信息可以实现级联修改。

create trigger xsb\_tigger

on xsb for update

as

if update(xh)

begin

declare @new\_num varchar(6),@old\_num varchar(6)

select @new\_num=xh from inserted

select @old\_num=xh from deleted

update yhb

set yhbh=@new\_num

where yhbh=@old\_num

end

go

create trigger jsb\_tigger

on jsb for update

as

if update(jsbh)

begin

declare @new\_num varchar(6),@old\_num varchar(6)

select @new\_num=jsbh from inserted

select @old\_num=jsbh from deleted

update yhb

set yhbh=@new\_num

where yhbh=@old\_num

end

go

3）当删除学生信息的时候选课记录同时删除，同时用户表的记录也删除。

create trigger xsb\_d1 on xsb

for delete

as

declare @xh varchar(6)

select @xh=xh from deleted

delete from cjb where xh=@xh

delete from yhb where yhbh=@xh

go

4）当删除老师信息的时候授课记录也同时删除，同时用户表中的记录也删除。

create trigger jsb\_deletigger

on jsb for delete

as

declare @jsbh varchar(6)

select @jsbh=jsbh from deleted

delete from skb where jsbh=@jsbh

delete from yhb where yhbh=@jsbh

go

5）增加成绩时成绩大于应该给学生总学分加上这门课的学分。删除成绩的时候，如果成绩大于分，应该把学生的总学分去掉这门课的学分。

create trigger cjb\_insetigger

on cjb for insert

as

declare @xh varchar(6),@cj int,@xf int,@kch char(3)

select @xh=xh,@cj=cj,@kch=kch from inserted

select @xf=xf from kcb where kch=@kch

if(@cj>=60)

update xsb

set zxf=zxf+@xf

where xh=@xh

go

create trigger cjb\_deletigger

on cjb for delete

as

declare @xh varchar(6),@cj int,@xf int,@kch char(3)

select @xh=xh,@cj=cj,@kch=kch from deleted

select @xf=xf from kcb where kch=@kch

if(@cj>=60)

update xsb

set zxf=zxf-@xf

where xh=@xh

go

---或者创建一个插入与删除触发器，将上面两触发器合而为一。代码如下：

create trigger cjb\_ins\_del\_igger

on cjb for insert,delete

as

declare @xh varchar(6),@cj int,@xf int,@kch char(3)

if exists(select \* from inserted)

begin

select @xh=xh,@cj=cj,@kch=kch from inserted

select @xf=xf from kcb where kch=@kch

if(@cj>=60)

update xsb

set zxf=zxf+@xf

where xh=@xh

end

if exists(select \* from deleted)

begin

select @xh=xh,@cj=cj,@kch=kch from deleted

select @xf=xf from kcb where kch=@kch

if(@cj>=60)

update xsb

set zxf=zxf-@xf

where xh=@xh

end

go

6）通过instead of触发器实现对学生选课情况视图（学号，姓名，课程号，课程名，成绩，学分，总学分，教师）的插入数据操作、更新数据操作、删除数据操作。

create view v1

as

select cjb.xh,xm,xsb.bj,cjb.kch,kcm,CJ,xf,zxf,jsb.jsbh,jsxm

from XSB,KCB,CJB,jsb,skb

where xsb.XH=CJB.XH and kcb.KCH=cjb.KCH and jsb.jsbh=skb.jsbh and skb.kch=CJB.KCH

go

create trigger v1\_t

on v1

instead of insert,update,delete

as

declare @xh char(6),@xm char(8),@kch char(3),@kcm char(16),@cj int,@jsbh char(6),@jsxm char(8),@bj varchar(50)

select @xh=xh,@xm=XM,@kch= kch,@kcm= kcm,@cj=CJ,@jsbh= jsbh, @jsxm= jsxm ,@bj= bj from inserted

if not exists(select \* from XSB where xh=@xh)

begin

insert into XSB(XH,XM,bj)

values(@xh,@xm,@bj)

end

if not exists(select \* from KCB where KCH=@kch)

begin

insert into KCB(KCH,KCM)

values(@kch,@kcm)

end

if not exists(select \* from CJB where XH=@xh and KCH=@kch)

begin

insert into CJB(XH,KCH,CJ)

values(@xh,@kch,@cj)

end

if not exists(select \* from jsb where jsbh=@jsbh)

begin

insert into jsb(jsbh,jsxm)

values(@jsbh,@jsxm)

end

if not exists(select \* from skb where jsbh=@jsbh and kch=@kch and bj=@bj)

begin

insert into skb(jsbh,kch,bj)

values(@jsbh,@kch,@bj)

end

if exists(select \* from Deleted)

begin

select @xh=xh, @kch=kch from Deleted

delete cjb where xh=@xh and kch=@kch

end

go

（7）权限设置

数据库的用户和密码

超级管理员（DBA）:admin/123456

查询用户（selectuser）:selectuser/123456

学生用户：最好添加角色，然后对角色设置权限。

教师用户：最好添加角色，然后对角色设置权限。

--超级管理员

create login admin with password='123456'

go

create user admin

go

grant all to admin

go

--查询用户（user）

create login selectuser with password='123456'

go

create user selectuser

go

Grant Select on cjb to selectuser

GO

--其他略。

（8）备份与恢复

定期对系统备份，并可以实现恢复数据库。

--（8）备份与恢复

--定期对系统备份，并可以实现恢复数据库。

USE master

GO

--创建备份文件(备份设备)

EXEC sp\_addumpdevice 'disk', 'pxscj20161122','E:\DB\pxscj20161122' --E:\DB是文件夹，xk\_bak3是备份的文件名

-- 使用BACKUP DATABASE备份数据库

BACKUP DATABASE PXSCJ TO pxscj20161122

GO

--还原数据库

USE master

GO

RESTORE DATABASE PXSCJ FROM DISK='E:\DB\pxscj20161122'

WITH REPLACE

GO

（9）其它：系统全局变量……

比如调用@@servername查看服务器名称。

select @@servername

6、运行与维护

用java或c#和sql server 2008把系统功能实现。