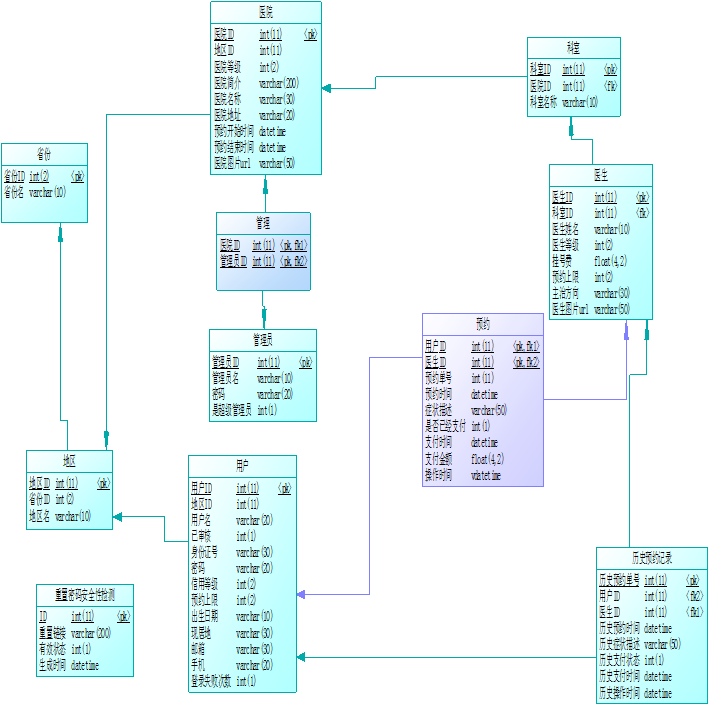
|  |  |
| --- | --- |
| 数据库 | Hospital\_Reservation\_DB |



表格清单

|  |  |
| --- | --- |
| 名称 | 代码 |
| 医生 | Doctor |
| 医院 | Hospital |
| 历史预约记录 | History\_Reservation |
| 地区 | Area |
| 用户 | User |
| 省份 | Province |
| 科室 | Depart |
| 管理 | Manage |
| 管理员 | Admin |
| 重置密码安全性检测 | ReSet\_Pwd\_Security |
| 预约 | Reservation |

Doctor

create table Doctor

(

Doctor\_ID int(11) not null,

Depart\_ID int(11),

Doctor\_Name varchar(10) not null,

Doctor\_Level int(2),

Doctor\_Fee float(4,2),

Doctor\_Limit int(2),

Doctor\_Major varchar(30),

Doctor\_Picture\_url varchar(50),

primary key (Doctor\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 医生ID | Doctor\_ID（主键） |
| 科室ID | Depart\_ID(外键) |
| 医生姓名 | Doctor\_Name（建立索引） |
| 医生等级 | Doctor\_Level |
| 挂号费 | Doctor\_Fee |
| 预约上限 | Doctor\_Limit |
| 主治方向 | Doctor\_Major |
| 医生图片url | Doctor\_Picture\_url |

Hospital

create table Hospital

(

Hospital\_ID int(11) not null,

Area\_ID int(11) not null,

Hospital\_Level int(2),

Hospital\_Introduction varchar(200),

Hospital\_Name varchar(30) not null,

Hospital\_Location varchar(20),

Reservation\_Start\_Time datetime,

Reservation\_End\_Time datetime,

Hospital\_Picture\_url varchar(50),

primary key (Hospital\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 医院ID | Hospital\_ID（主键） |
| 地区ID | Area\_ID（外键） |
| 医院等级 | Hospital\_Level |
| 医院简介 | Hospital\_Introduction |
| 医院名称 | Hospital\_Name（建立索引） |
| 医院地址 | Hospital\_Location |
| 预约开始时间 | Reservation\_Start\_Time（预约开始时间就是开放号源的时间，每个医院可以不同，管理员可以配置） |
| 预约结束时间 | Reservation\_End\_Time |
| 医院图片url | Hospital\_Picture\_url |

Depart

create table Depart

(

Depart\_ID int(11) not null,

Hospital\_ID int(11),

Depart\_Name varchar(10) not null,

primary key (Depart\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 科室ID | Depart\_ID（主键） |
| 医院ID | Hospital\_ID（外键） |
| 科室名称 | Depart\_Name（建立索引） |

之所以要把省份和地区拿出来是为了便于管理，医院、用户所关联的省份都是统一的，而医院表中的医院地址和用户表中的现居地则可以是他的具体位置

Area

create table Area

(

Area\_ID int(11) not null,

Province\_ID int(2) not null,

Area\_Name varchar(10) not null,

primary key (Area\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 地区ID | Area\_ID（主键） |
| 省份ID | Province\_ID（外键） |
| 地区名 | Area\_Name（建索引） |

Province

create table Province

(

Province\_ID int(2) not null,

Province\_Name varchar(10) not null,

primary key (Province\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 省份ID | Province\_ID（主键） |
| 省份名 | Province\_Name（建立索引） |

User

create table User

(

User\_ID int(11) not null,

Area\_ID int(11) not null,

UserName varchar(20) not null,

isChecked int(1) not null,

Identity\_ID varchar(30) not null,

Password varchar(20) not null,

Credit\_Rank int(2) not null,

Appointment\_Limit int(2) not null,

Birthday varchar(10) not null,

Location varchar(30) not null,

Mail varchar(30) not null,

Phone varchar(20) not null,

FailTime int(1) not null,

primary key (User\_ID)

);

之所以要加预约上限是为了具有更大的灵活性，一般地信用等级确定后上限就随之确定，但是管理员在管理用户的时候可以认为单独调整用户的预约上限或者信用等级。

|  |  |
| --- | --- |
| 名称 | 代码 |
| 用户ID | User\_ID（主键） |
| 地区ID | Area\_ID（外键） |
| 用户名 | UserName（建立索引） |
| 已审核 | isChecked（默认0） |
| 身份证号 | Identity\_ID |
| 密码 | Password |
| 信用等级 | Credit\_Rank（默认3） |
| 预约上限 | Appointment\_Limit（默认3） |
| 出生日期 | Birthday |
| 现居地 | Location |
| 邮箱 | Mail |
| 手机 | Phone |
| 登录失败次数 | FailTime（为了防止蛮力破解，当用户连续失败3次后应当冻结账户不允许继续登录或者锁定1小时后再允许登录） |

Admin

create table Admin

(

Admin\_ID int(11) not null,

Admin\_Name varchar(10),

Password varchar(20),

isSuper int(1),

primary key (Admin\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 管理员ID | Admin\_ID（主键） |
| 管理员名 | Admin\_Name（建立索引） |
| 密码 | Password |
| 是超级管理员 | isSuper（默认0） |

超级管理员的记录是内置的，ID 1，管理员名 SuperAdmin 是超级管理员 1

Manage

create table Manage

(

Hospital\_ID int(11) not null,

Admin\_ID int(11) not null,

primary key (Hospital\_ID, Admin\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 医院ID | Hospital\_ID |
| 管理员ID | Admin\_ID |

Reservation

create table Reservation

(

User\_ID int(11) not null,

Doctor\_ID int(11) not null,

Reservation\_ID int(11) not null,

Reservation\_Time datetime not null,

Reseration\_Symptom varchar(50),

Reservation\_Payed int(1) not null,

Reservation\_PayTime datetime,

Reservation\_PayAmount float(4,2),

Operation\_Time vdatetime not null,

primary key (User\_ID, Doctor\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 用户ID | User\_ID（外键） |
| 医生ID | Doctor\_ID（外键） |
| 预约单号 | Reservation\_ID（主键） |
| 预约时间 | Reservation\_Time（预约的时间） |
| 症状描述 | Reseration\_Symptom |
| 是否已经支付 | Reservation\_Payed（not null ，因为他决定下面两个字段是否有意义） |
| 支付时间 | Reservation\_PayTime |
| 支付金额 | Reservation\_PayAmount |
| 操作时间 | Operation\_Time（网上预约操作成功的时间） |

基本仿照预约表，作为一个历史备份，通过定时任务将过期的预约记录从预约表拿出导入历史预约记录表中。

History\_Reservation

create table History\_Reservation

(

History\_Reservation\_ID int(11) not null,

User\_ID int(11),

Doctor\_ID int(11),

History\_Reservation\_Time datetime not null,

History\_Reservation\_Symptom varchar(50),

History\_Reservation\_Paied int(1) not null,

History\_Pay\_Time datetime,

History\_Operation\_Time datetime not null,

primary key (History\_Reservation\_ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| 历史预约单号 | History\_Reservation\_ID（主键） |
| 用户ID | User\_ID（外键） |
| 医生ID | Doctor\_ID（外键） |
| 历史预约时间 | History\_Reservation\_Time |
| 历史症状描述 | History\_Reservation\_Symptom |
| 历史支付状态 | History\_Reservation\_Paied |
| 历史支付时间 | History\_Pay\_Time |
| 历史操作时间 | History\_Operation\_Time |

这个表主要是为了忘记密码的重置功能设计的，当用户忘记密码后，填写邮箱后，发送一封邮件到邮箱，内容是一个重置链接（包括一个100位的校验），这个链接会和他生成的时间会被存到表里，当用户点击这个链接时会先检查当前时间和这个genTime是否间隔超过了半个小时，若是，则把Status 更新为0，同时不允许用户重置密码。而且应当定时清除Status为0的记录

Reset\_Pwd\_Security

create table Reset\_Pwd\_Security

(

ID int(11) not null,

resetLink varchar(200) not null,

status int(1) not null,

genTime datetime not null,

primary key (ID)

);

|  |  |
| --- | --- |
| 名称 | 代码 |
| ID | ID(主键) |
| 重置链接 | resetLink |
| 有效状态 | Status（默认为1） |
| 生成时间 | genTime |