



**课 程 设 计**

课程名称 数据库系统课程设计

题目名称 卡拉OK点歌系统

专业班级 信息安全（1）班

学 号 3116005161 \_\_\_\_\_\_

学生姓名 张秉坤

指导教师 明俊峰

2019年 1 月 2日

# 目录

# 1.引言

## 1.1课题来源

## 1.2课程设计的目的

## 1.3本选题的使用技术

# 2.需求分析

## 2.1信息需求

## 2.2功能需求

## 2.3数据字典及安全性完整性需求

# 3.概念结构设计

## 3.1E-R图

# 4.逻辑结构设计

## 4.1关系模型

## 4.2用户子模式

## 4.3应用程序功能模块图

## 4.4安全性设计和实现说明

## 4.5完整性设计和实现说明

# 5.数据库物理设计

## 5.1索引

## 5.2数据的存放位置

## 5.3模块设计

# 6.数据库实施

# 7.数据库运行与维护

# 8.收获和体会

1. **引言**

**1.1课题来源**

**随着现如今经济文化水平的显著提高，人们对生活质量及工作环境的要求也越来越高。同时，随着生活节奏的加快，每个人都处于忙碌繁乱的社会的那个当中，不论是在家庭，工作场所，或是学校中，无时无刻充满着生活和学习上的压力。工作之余，找到一种能够换季压力，释放疲劳的娱乐方式，已成为大家共同的愿望。然而，受到工作条件和时间的限制，越来越多的人们选择了去KTV厅唱K休闲娱乐。唱歌，自然免不了点歌的繁琐。传统的点歌设备主要是CD和固放设备组成，虽然基本满足了人们的需求，但是也带来了设备成本高，点歌过程繁琐，更新麻烦等不少问题。随着科学的发展和信息时代的到来，点歌系统应运而生，卡拉OK点歌系统已经成为了娱乐场所必不可少的一部分。卡拉OK点歌系统不仅简化了用户操作，而且极大化满足了人们的需求和使用习惯**

**1.2课程设计的目的**

**研究该课题是为了通过对“卡拉OK点歌系统”的设计与实现，熟悉这类系统的开发原理和数据库的设计方法。在设计过程,能够深入地理解面向对象设计方法的优越性,获得这些方面的相关经验。对一个即将毕业的软件工程专业本科生来说,又想在软件行业领域里有所发展,研究并应用面向对象设计方法和数据库的设计方法是百益无一害的,可以使其走向社会后能够快速地适应行业需求。这正是本次设计的根本目的所在。**

**1.3本选题的使用技术**

**系统开发平台：idea**

**系统开发语言：Java，SQL**

**数据库管理软件：Mysql**

**运行平台：Ubuntu**

1. **需求分析**

**2.1信息需求**

**经过对卡拉OK点歌系统的结构化分析，得出信息要求如下：**

**1.、歌星信息：包括了歌星编号、歌星姓名、歌星性别、国籍、歌星类型、地区、拼音缩写。**

**2、歌曲信息：包括了歌曲编号、歌曲名称、歌星编号、语种、歌曲类型、 路径、歌曲拼音缩写、歌曲时长**

**3、管理员信息：包括了管理员用户名 管理员密码**

**4、用户信息：包括了用户编号、用户登录名、用户密码、用户姓名、用户性别、出生日期 注册日期**

**2.2功能需求**

**卡拉OK点歌系统主要分为歌曲的查询与管理，管理主要包括歌曲的查询，增添，修改，删除；查询可以按照歌星查询，歌曲名称查询等方式查询。 系统功能需求：**

1. **点播歌曲**

**点歌是最基本也是最重要的功能，重在快捷方便。比如拼音点歌、指定歌星点歌，只需输入相应的拼音简写然后点击，就能检索到想要点播的歌曲。**

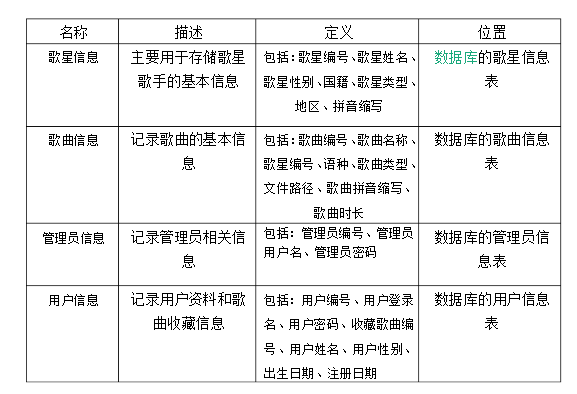
1. **添加歌曲**

**音乐是滚动的时尚元素，明天都可能有新的创作，所以添加新歌也是必不可少的功能。当然，经典老歌也是不可或缺的元素，适当添加经典老歌，增添怀旧的气氛。**

1. **歌曲管理**

**歌曲库十分庞大，给歌曲管理带来一定的难度。对歌曲信息的增添和删改同样需要相应的子功能模块管理。**

**2.3数据字典及安全性完整性需求**

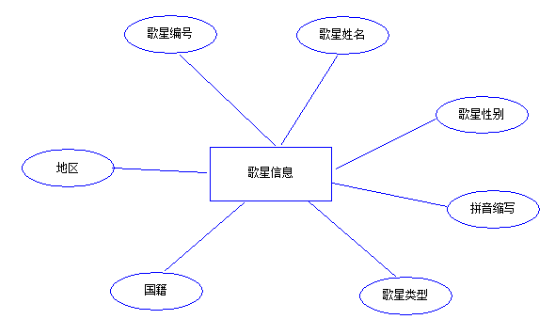


**表2.1**

1. **概念结构设计**

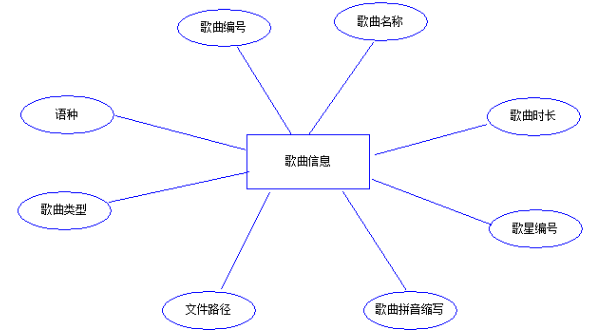
**3.1E-R图**

**3.1.1歌星信息实体**



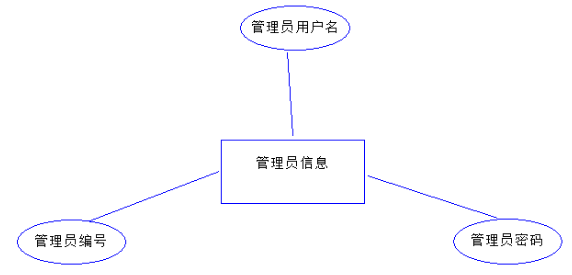
**图3.1.1**

**3.1.2歌曲信息实体**



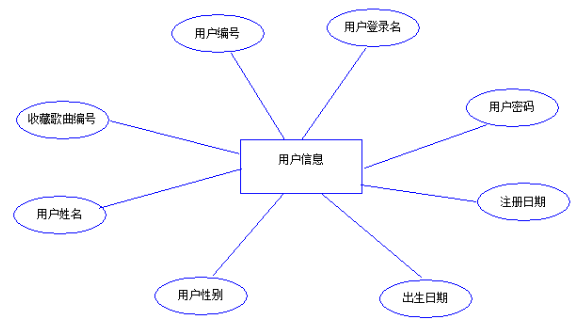
**图3.1.2**

**3.1.3管理员信息实体**



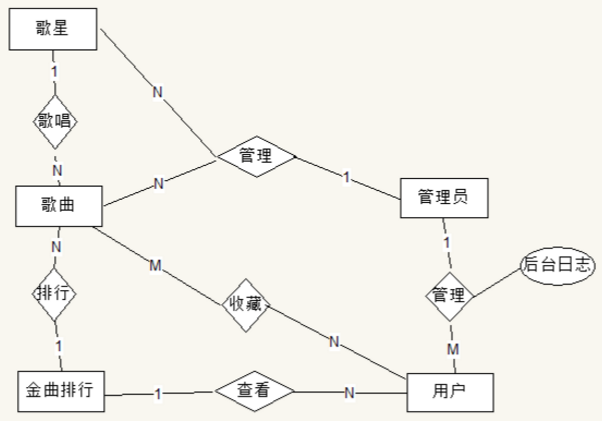
**图3.1.3**

**3.1.4用户信息实体**



**图3.1.4**

**3.2整体E-R图**

****

**图3.2**

**分ER图向整体ER图合并，合并过程中出现的问题和解决方案。**

**(1)结构冲突。歌星在歌星信息这个分E—R图中作为主体，而在歌曲信息分E—R图中则作为属性。歌星包含自己的属性，不能放在歌曲信息表的属性里，此时把歌曲表里面的歌星字段用歌星编号取代，而歌星表和歌曲表为一对多关系。**

**(2)命名冲突。歌曲信息与歌星信息都包含编号及名称，起名时需要区分开。分别命名为歌曲编号和歌星编号。**

**4.逻辑结构设计**

**4.1关系模型**

**为了完成需求分析中确定的基本功能和概念设计中模块的划分，通过基于3NF的数据库设计方法（通过分析，各数据表已达第三范式），构造了5张基本表。现在对这5张基本表划分基本关系，在已设计好的数据库中初步建立各表，用于存储歌曲信息和歌星信息等基本信息。此外，在基本表上定义了一些约束条件，包括基本表上数据类型和数据长度的限制以及是否允许为空或表内是否取值唯一的限定。**

**根据E-R图，可转化为如下关系模式：关系的主码用下划线标出，外码用删除线标出。 实体：**

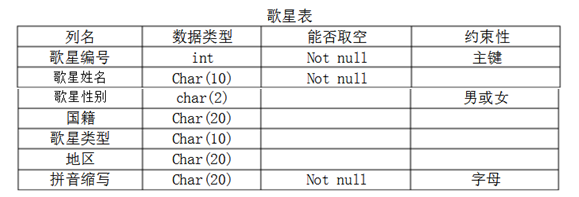
**歌星（歌星编号，歌星姓名，歌星性别，国籍，歌星类型，地区，拼音缩写）**

**歌曲（歌曲编号，歌曲名称，~~歌星编号~~，语种，歌曲类型，路径，歌曲拼音缩写，歌曲时长）**

**管理员（管理员用户名，管理员密码）**

**用户（用户编号，用户登录名，用户密码，用户姓名，用户性别，出生日期，注册日期）**

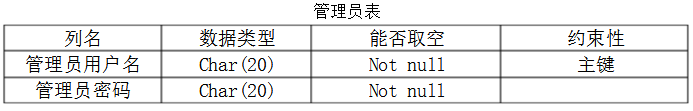
**联系： 收藏（~~用户编号，歌曲编号~~，日期）**



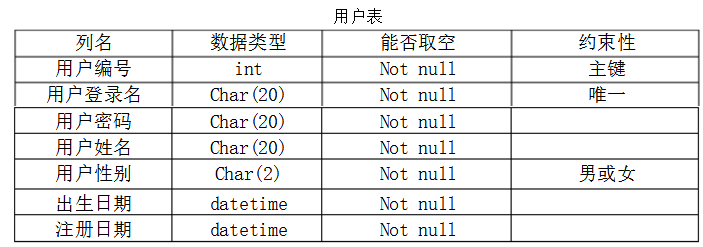
**表4.1.1 歌星表**



**表4.1.2 歌曲表**



**表4.1.3管理员表**



**表4.1.4用户表**

**4.2用户子模式(视图)**

**为了便于浏览经常查询的信息，提高查询效率，可以考虑建立相关视图，因为它不仅可提供了数据库安全性，还可以节省系统资源开销。**

**4.2.1 创建歌星表，歌曲表之间，建立一个视图**

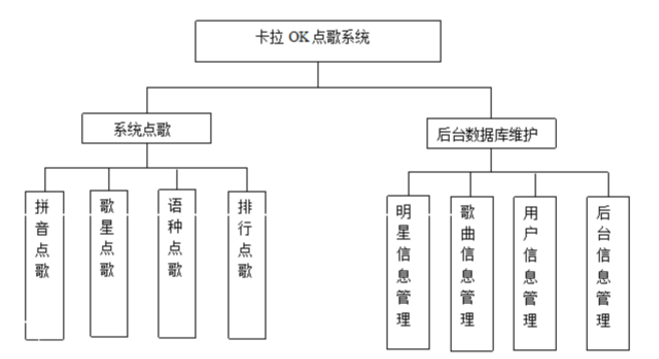
**create view VIEW\_歌曲歌星表**

**as select 歌曲编号,歌曲名称,歌星姓名**

**from 歌曲表,歌星表**

**where 歌曲表.歌星编号=歌星表.歌星编号；**

**4.3应用程序功能模块图**



**图4.3**

**4.4安全性设计和实现说明**

**在卡拉OK点歌系统中，设计了用户登录和管理员登录。其中用户只有检索和收藏歌曲的权限，管理员能够对歌曲信息、歌星信息、用户信息作增删改查操作。为了避免操作权限的交错，本系统设计规定只有一个管理员。**

**4.5完整性设计和实现说明**

**4.5.1数据库完整性概念**

**数据库完整性是指数据库中数据的正确性和相容性。数据库完整性由各种各样的完整性约束来保证，因此可以说数据库完整性设计就是数据库完整性约束的设计。 数据库的完整性包括：**

**(1)实体完整性**

**实体完整性将行定义为特定表的唯一实体。实体完整性通过索引、UNIQUE 约束、PRIMARY KEY 约束或 IDENTITY 属性强制表的标识符列或主键的完整性。**

**(2)参照完整性**

**参照完整性保证在主键（在被参考表中）和外键之间的关系总是得到维护（通过 FOREIGN KEY 和 CHECK 约束），确保键值在所有表中一致，要求不能引用不存在的值，如果键值更改了，那么在整个数据库中，对该键值的所有引用要进行一致的更改。**

**(3)用户定义的完整性**

**用户定义完整性使得用户得以定义不属于其它任何完整性分类的特定业务规则。所有的完整性类型都支持用户定义完整性（CREATE TABLE 中的所有列级和表级约束、存储过程和触发器）。**

**4.5.2违反数据库完整性**

**(1)违背实体完整性**

**本系统中有以下7个关系模式，带下划线为主码，删除线为外码。**

**歌星（歌星编号，歌星姓名，歌星性别，国籍，歌星类型，地区，拼音缩写）**

**歌曲（歌曲编号，歌曲名称，~~歌星编号~~，语种，歌曲类型，路径，歌曲拼音缩写，歌曲时长）**

**管理员（管理员用户名，管理员密码）**

**用户（用户编号，用户登录名，用户密码，用户姓名，用户性别，出生日期，注册日期）**

**收藏（~~用户编号，歌曲编号~~，日期）**

**对以上的基本表进行增加、删除和修改操作时，检测：**

**1、检查主码值是否唯一，如果不是唯一则拒绝插入。**

**2、检查主码的各个属性是否为空，只要有一个为空就拒绝插入。**

**(2)违反参照完整性**

**可能破坏参照完整性的情况及违约处理：**



**表4.5.2**

**(3)违反自定义完整性**

**1、性别只能取值男或女，对应Mysql设计CHECK(SEX IN '男'或'女')，**

**当性别插入值在可行域外的时候拒绝执行，并利用触发器提醒用户输入正确值。**

**2、要求用户登录名唯一，属性列定义需加上UNIQUE限制。**

**管理员添加用户的时候，若新增用户登录名已存在用户信息表中，回被系统拒绝执行。**

**5.数据库物理设计**

**数据库的物理设计就是为一个给定的逻辑数据模型选取一个最适合应用要求的物理结构的过程。物理结构设计阶段实现的是数据库系统的内模式，它的质量直接决定了整个系统的性能。因此在确定数据库的存储结构和存取方法之前，对数据库系统所支持的事务要进行仔细分析，获得优化数据库物理设计的参数。**

**数据库的物理设计通常分为两步：**

**一：确定数据库的物理结构，在关系数据库中主要指存取方法和存取结构；**

**二：对物理结构进行评价，评价的重点是时间和空间效率。**

**5.1索引**

**（1）建立3个唯一索引，分别由各表组成。**

**1.歌星表索引**

**create unique index 歌星索引 on 歌星表(歌星姓名)**

**2.歌曲表索引**

**create unique index 歌曲索引 on 歌曲表(歌曲名称)**

**3.用户索引**

**create unique index 用户索引 on 用户表(用户姓名)**

**5.2数据的存放位置**

**5.2.1 MySQL数据库文件介绍**

[**MySQL**](http://lib.csdn.net/base/14)的每个**[数据库](http://lib.csdn.net/base/14" \t "_blank" \o "MySQL知识库)**都对应存放在一个与数据库同名的文件夹中，MySQL数据库文件包括MySQL（server）所建数据库文件和MySQL（server）所用存储引擎创建的数据库文件。

1、MySQL（server）创建并管理的数据库文件：

.frm文件：存储数据表的框架结构，文件名与表名相同，每个表对应一个同名frm文件，与操作系统和存储引擎无关，即不管MySQL运行在何种操作系统上，使用何种存储引擎，都有这个文件。

除了必有的.frm文件，根据MySQL所使用的存储引擎的不同（MySQL常用的两个存储引擎是MyISAM和InnoDB），存储引擎会创建各自不同的数据库文件。

2、MyISAM数据库表文件：

.MYD文件：即MY Data，表数据文件

.MYI文件：即MY Index，索引文件

.log文件：日志文件

3、InnoDB采用表空间（tablespace）来管理数据，存储表数据和索引，

InnoDB数据库文件（即InnoDB文件集，ib-file set）：

ibdata1、ibdata2等：系统表空间文件，存储InnoDB系统信息和用户数据库表数据和索引，所有表共用

.ibd文件：单表表空间文件，每个表使用一个表空间文件（file per table），存放用户数据库表数据和索引

日志文件： ib\_logfile1、ib\_logfile2

**5.2.2 MySQL数据库存放位置：**

1、MySQL如果使用MyISAM存储引擎，数据库文件类型就包括.frm、.MYD、.MYI，默认存放位置是C:\Documentsand Settings\All Users\Application Data\MySQL\MySQL Server 5.1\data

2、MySQL如果使用InnoDB存储引擎，数据库文件类型就包括.frm、ibdata1、.ibd，存放位置有两个，

.frm文件默认存放位置是C:\Documents and Settings\All Users\ApplicationData\MySQL\MySQL Server 5.1\data，ibdata1、.ibd文件默认存放位置是MySQL安装目录下的data文件夹

**5.2.3怎样查看MySql数据库物理文件存放位置**

想导出mysql中的数据库文件**,**使用如下命令：

|  |  |
| --- | --- |
|  | show global variables like "%datadir%"; |

**5.3模块设计**

**5.3.1用户系统功能描述**

**1.点歌功能描述：**

**进入卡拉OK点歌系统后，显示的是各种点歌模式：有拼音点歌、歌星点歌、分类点歌、语种点歌，全部歌曲，每个点歌模块都可以查询想要点的歌曲。主要模块功能如下：**

**拼音点歌：A~Z 26个字母自由组合，对应拼音模糊搜索。**

**歌星点歌：显示歌星列表，对应歌星个人专辑。**

**2.歌曲管理功能描述：**

**用户点歌完毕之后，可进入歌曲管理，查看自己收藏的歌曲，主要模块功能如下：**

**收藏歌曲：根据用户个人喜好收藏歌曲，进入已点歌曲列表点击收藏即可。**

**3.后台数据库维护功能描述**

**管理员可登录后台管理系统对歌曲、歌星、用户进行管理和维护。进入后台管理系统后，显示的是歌曲管理、歌星管理、用户管理大三模块。主要模块功能如下：**

**歌曲管理：管理员可根据歌曲信息添加歌曲，删除歌曲，查询歌曲。**

**歌星管理：管理员可根据歌星信息添加歌星，删除歌星，查询歌星。**

**用户管理：管理员可根据用户信息添加用户，删除用户，查询用户。**

**5.3.2 功能模块设计**

**对上述各项功能进行集中，分块，按照结构化程序设计的要求，分为系统点歌功能模块和后台数据库维护功能模块。**

**1.系统点歌模块**

**系统点歌模块由点歌和歌曲管理两大功能模块组成。**

**点歌功能模块包括了：拼音点歌、歌星点歌、分类点歌、语种点歌。**

**歌曲管理模块包括了：收藏歌曲。**

**6.数据库实施**

**数据库的实施主要是根据逻辑结构设计和物理结构设计的结果，在计算机系统上建立实际的数据库结构、导入数据并进行程序的调试。它相当于软件工程中的代码编写和程序调试的阶段。**

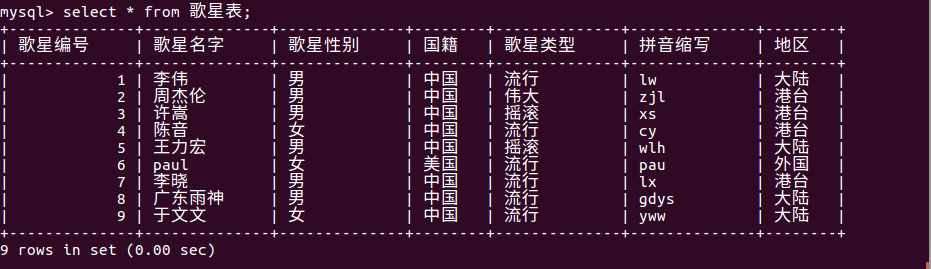
**创建数据库主要通过SQL语句来实现，分别输入以下语句来执行：**

**创建了卡拉OK点歌系统的数据库：**

|  |
| --- |
| **create database karaoke;** |

1. **创建歌星表**

|  |
| --- |
| **create table 歌星表 ( 歌星编号 int primary key, 歌星名字 char(10) not null, 歌星性别 enum('男','女'), 国籍 char(20), 歌星类型 char(10), 拼音缩写 char(20) not null, 地区 char(20) );**  **insert into 歌星表 values('1','李伟','男','中国','流行','lw','大陆');**  **insert into 歌星表 values('2','周杰伦','男','中国','伟大','zjl','港台');**  **insert into 歌星表 values('3','许嵩','男','中国','摇滚','xs','港台');**  **insert into 歌星表 values('4','陈音','女','中国','流行','cy','港台');**  **insert into 歌星表 values('5','王力宏','男','中国','摇滚','wlh','大陆');**  **insert into 歌星表 values('6','paul','女','美国','流行','pau','外国');**  **insert into 歌星表 values('7','李晓','男','中国','流行','lx','港台');**  **insert into 歌星表 values('8','广东雨神','男','中国','流行','gdys','大陆');**  **insert into 歌星表 values('9','于文文','女','中国','流行','yww','大陆');** |

**图6.1**

1. **创建歌曲表**

|  |
| --- |
| **create table 歌曲表 ( 歌曲编号 int primary key, 歌曲名称 char(10) not null, 歌星编号 int not null, 语种 char(10), 歌曲类型 char(10), 路径 char(20), 歌曲拼音缩写 char(20) not null, 歌曲时长 datetime, foreign key (歌星编号) references 歌星表(歌星编号) );**  **insert into 歌曲表 values('1','七里香','2','国语','爱情','D盘','qlx','06:00');**  **insert into 歌曲表 values('2','千里之外','2','国语','悲伤','D盘','qlzw','03:51');**  **insert into 歌曲表 values('3','告白气球','2','国语','爱情','D盘','gbqq','04:00');**  **insert into 歌曲表 values('4','大城小爱','5','国语','爱情','D盘','dcxa','04:21');**  **insert into 歌曲表 values('5','广东爱情故事','8','粤语','伤感','D盘','gdaqgs','04:00');** |

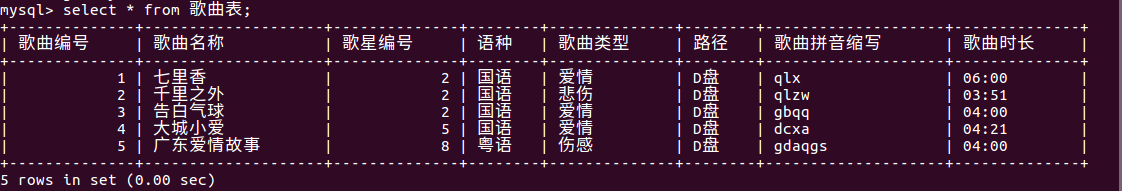
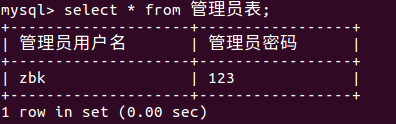


图 6.2

1. **创建管理员表**

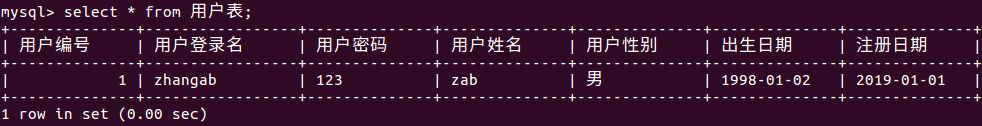
|  |
| --- |
| **create table 管理员表 (管理员用户名 char(20) primary key, 管理员密码 char(20) not null);**  **insert into 管理员表 values ('zbk','123');** |



**图6.3**

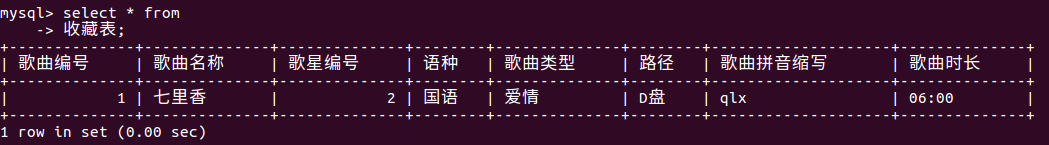
1. **创建用户表**

|  |
| --- |
| **create table 用户表 ( 用户编号 int primary key, 用户登录名 char(20) not null unique, 用户密码 char(20) not null, 用户姓名 char(20) not null, 用户性别 enum('男','女'), 出生日期 date not null, 注册日期 date not null );**  **insert into 用户表 values(1, 'zhangab', '123', 'zab','男','1998-01-02', '2019-01-01');** |

 图6.4

1. **创建收藏表**

|  |
| --- |
| **create table 收藏表 ( 歌曲编号 int primary key, 歌曲名称 char(10) not null, 歌星编号 int not null, 语种 char(10), 歌曲类型 char(10), 路径 char(20), 歌曲拼音缩写 char(20) not null, 歌曲时长 char(20) , foreign key (歌星编号) references 歌星表(歌星编号) );**  **insert into 收藏表 values (1, '七里香',2, '国语','爱情','D盘','qlx', '06:00');** |

 图6.5

**7.数据库运行与维护**

**7.1系统功能测试**

**7.1.1 登录界面测试**

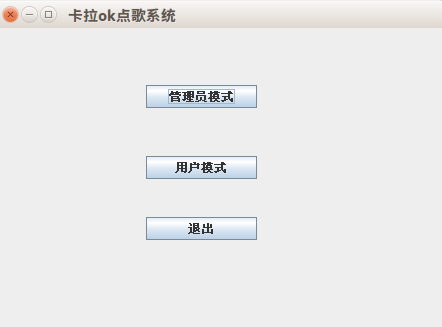


图7.1

**1管理员登录界面**



图7.2

**2用户登录界面**



图7.3

**3登录名错误或者密码错误**

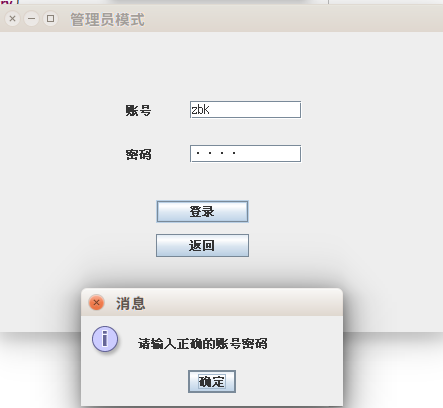


图7.4

**7.1.2点歌测试**

**1拼音点歌**



图7.5



图7.6

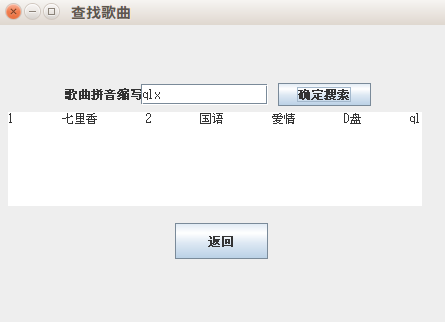
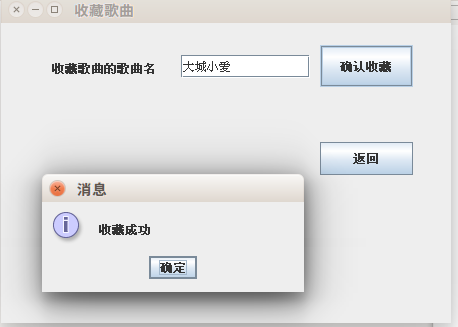


图7.7

**2收藏歌曲**



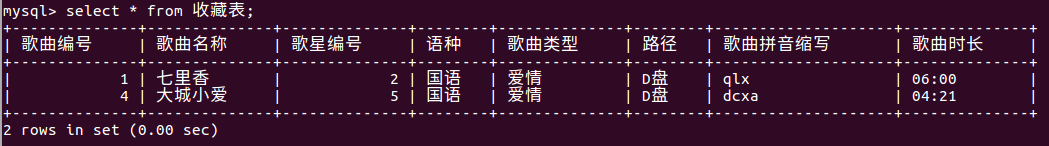
 图7.8

图7.9

**7.2后台管理测试**

**本系统规定后台管理员只有一个，帐号zbk密码123下面开始测试：**

**7.2.1添加歌曲**



图7.10



图7.11



图7.12



图7.13

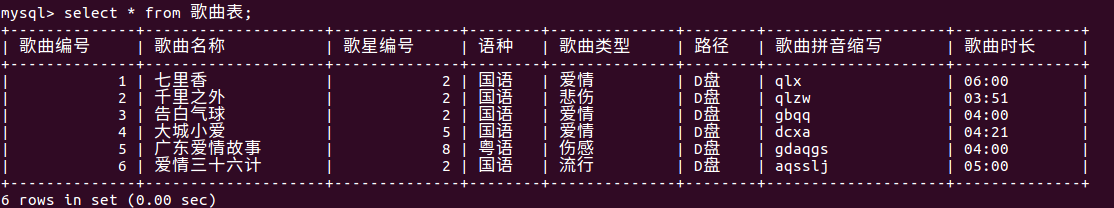


图7.14

**7.2.2删除歌曲**

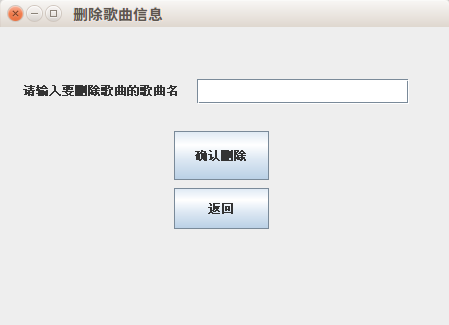


图7.15



图7.16

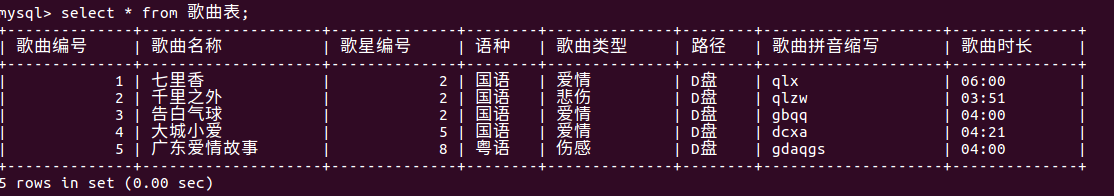
**查看是否从表中删除**

图7.17

**歌曲已经从表中删除**

**7.2.3修改歌曲信息**

**修改广东爱情故事的歌曲时长和类型**

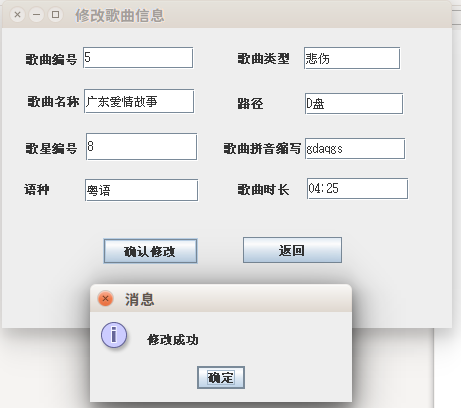


图7.18

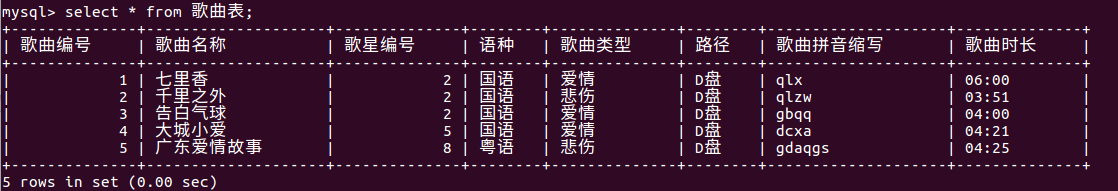
**查看是否修改成功**

图7.19

**歌曲类型和歌曲时长的属性修改成功**

**7.2.4添加歌星**



图7.20



图7.21

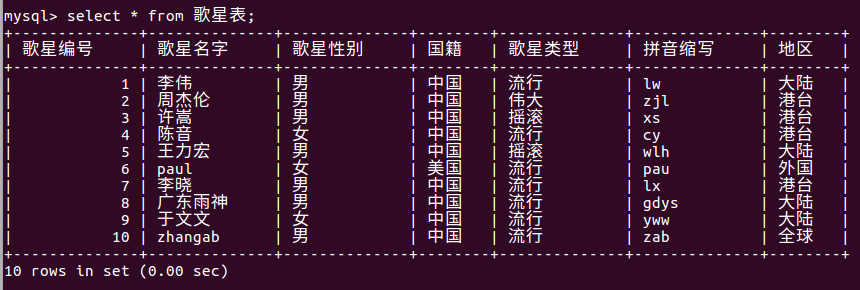
**查看是否添加成功**

图7.22

**已经添加成功**

**7.2.5删除歌星**



图7.23

**查看是否删除成功**

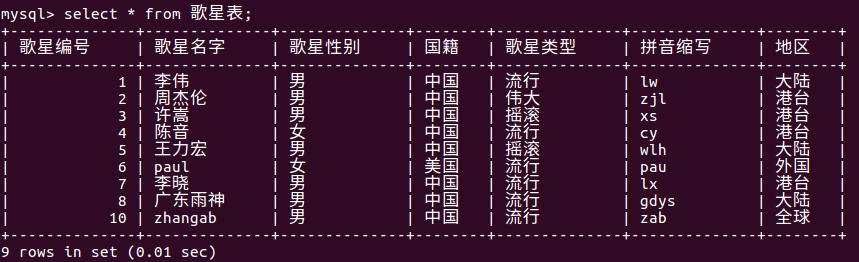


图7.24

**已经删除成功.**

**7.2.6修改歌星信息**

**把zhangab的地区修改为全宇宙**

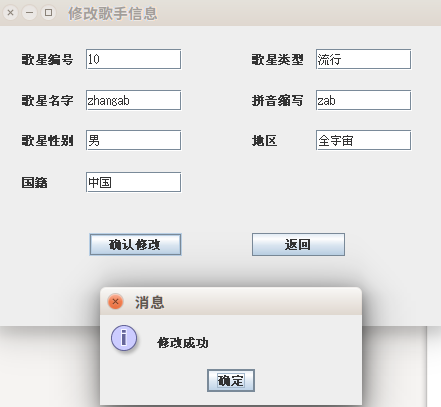


图7.25

**查看是否修改成功**

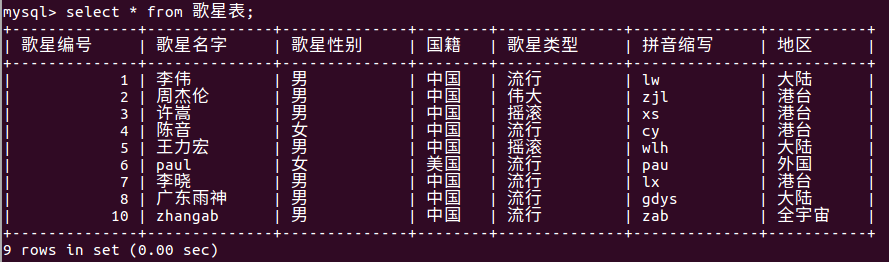


图7.26

**修改成功**

**7.2.7查找歌星信息**

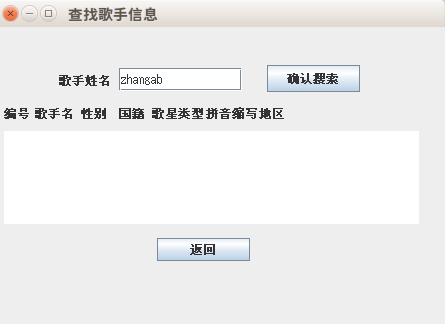


图7.27

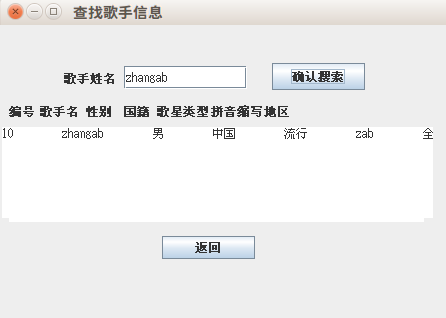


图7.28

**7.2.8添加用户**



图7.29



图7.30

**查看是否添加成功**



图7.31

**已经添加成功**

**7.2.9删除用户**

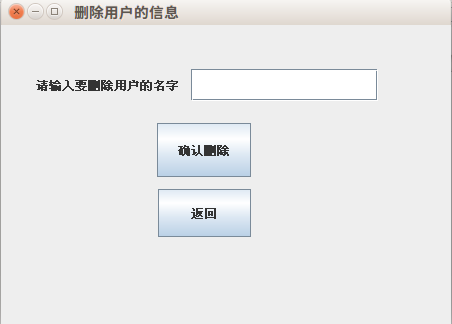


图7.32



图7.33

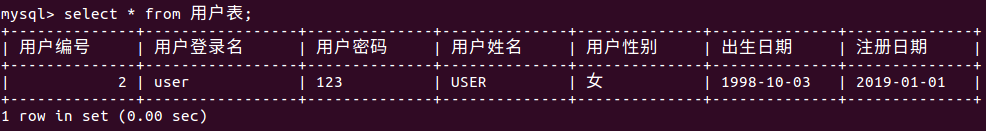
**查看是否删除成功**

图7.34

**已经从用户表中删除.**

**7.2.10修改用户信息**

**修改user的注册日期为2019-1-2**



图7.35

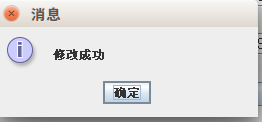


图7.36

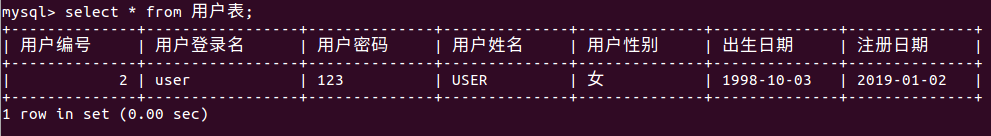
**查看用户信息是否修改**

图7.37

**修改成功**

**7.2.11查找用户信息**

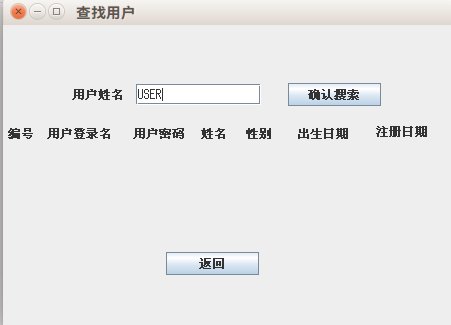
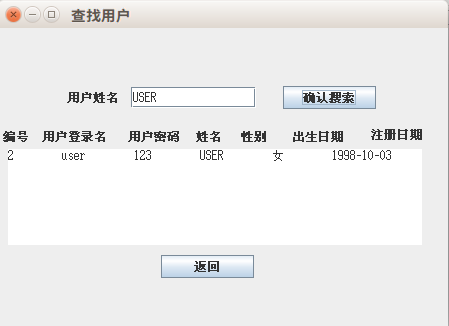


图7.38

 图7.39

**7.2.12其他功能**

**功能可以在下面界面中对应进行使用**



图7.40

**7.3数据库的维护**

**管理员可登录后台管理系统对歌曲、歌星、用户进行管理和维护。进入后台管理系统后，显示的是歌曲管理、歌星管理、用户管理大三模块。主要模块功能如下：**

**歌曲管理：管理员可根据歌曲信息添加歌曲，删除歌曲，查询歌曲。**

**歌星管理：管理员可根据歌星信息添加歌星，删除歌星，查询歌星。    用户管理管理员可根据用户信息添加用户，删除用户，查询用户**

**后从事这方面的工作都需要不断地去学习去实践，这次实践可以给我们敲一个警钟，我们面临毕业，面临择业，需要这些实践经验，在困难面前要勇于尝试，这是这次课程设计给我的最大感想！通过这次毕业设计的完成，使我对自己所学知识进一步的了解，同时通过实践发现自己的不足，对知识的掌握及运用存在诸多的不完善。本系统也存在着很多不足，如系统十分简单，实现功能也不够完善，安全性问题也没有很好的解决。所以在今后的学习中我会继续努力，完善自我，同学的帮助、指导老师的用心辅导和学校的精心安排使这个课题能够顺利进行。但是，由于时间仓促以及本人水平有限，本系统还有很多不完善之处，也难免存在错漏，恳请老师谅解。**

**8收获和体会**

**通过这次课程设计发现这其中需要的很多知识我们没有接触过，上网查找资料的时候发现我们以前所学到的仅仅是皮毛，还有很多需要我们掌握的东西我们根本不知道。同时也发现有很多已经学过的东西我们没有理解到位，不能灵活运用于实际，不能很好的用来解决问题，这就需要自己不断的大量的实践，通过不断的自学，不断地发现问题，思考问题，进而解决问题。在这个过程中我们将深刻理解所学知识，同时也可以学到不少很实用的东西。做完卡拉OK点歌系统的时候，有种如释负重的感觉。数据库修改了许多次，界面也是调试了很久。回顾整个项目的开发，连接数据库比较简单，系统界面也不难，关键在按钮的“监听”和对数据库的数据处理，与学JAVA的师兄交流了好几天，刚开始做还是比较麻烦的，经常遇到许多未知的BUG，也有无法读取结果的时候。下苦心调试好前台的点歌模块，后续编程思路就清晰许多了，完成了添加歌曲功能模块后，添加用户和歌星参照设计模版，继而把后台管理系统开发出来，最后也总算把整个卡拉OK系统完成了。很高兴能够达到预期的目标和积累到数据库项目开发的经验，更加坚定了自己以后的编程之路。比较遗憾的是，由于时间的限制和对JAVA界面设计进阶了解的局限性，系统做得并不是十分精美，还有点播歌曲这个功能模块尚未实现。但课程设计并不是学习项目开发的终点，寒假也有充裕的时间去了解点播歌曲的实现，所以最后决定在后几天继续完成对音乐点播模块的开发设计和对系统界面的进一步美化。**

**通过本次课程设计，对SQL语言，数据库的创建、修改、删除方法有了一定的了解，通过导入表和删除表、更改表，学会了数据库的基本操作。 很多事情不是想象中的那么简单的，它涉及到的各种实体、属性、数据流程、数据处理等等。很多时候感觉后面的设计根本无法继续，感觉像是被前面做的各种图限制了。在做关系模型转换的时候碰到有些实体即可以认为是实体又可以作为属性，为了避免冗余，尽量按照属性处理了。 通过本次课程设计，我觉得不管做什么，始终都要相信自己。不要畏惧，不要怕遇到困难，什么都要去尝试，有些你开始认为很难的事在你尝试之后你可能会发现原来它并没有你你想象的那么难。如果没有自信，没有目标，没有信心就很难把事情做好，当其他人都在迷茫的时候，自己一定要坚信目标。从学习这个专业，到以后从事这方面的工作都需要不断地去学习去实践，这次实践可以给我们敲一个警钟，我们面临毕业，面临择业，需要这些实践经验，在困难面前要勇于尝试，这是这次课程设计给我的最大感想！通过这次毕业设计的完成，使我对自己所学知识进一步的了解，同时通过实践发现自己的不足，对知识的掌握及运用存在诸多的不完善。本系统也存在着很多不足，如系统十分简单，实现功能也不够完善，安全性问题也没有很好的解决。所以在今后的学习中我会继续努力，完善自我，同学的帮助、指导老师的用心辅导和学校的精心安排使这个课题能够顺利进行。但是，由于时间仓促以及本人水平有限，本系统还有很多不完善之处，也难免存在错漏，恳请老师谅解。**