

华侨大学

信息科学与工程学院

课程设计报告

课程名称	数据库技术课程设计
题 目	KTV 点歌系统
院(系)别	信息科学与工程学院
专 业	自动化
级 别	2017
学 号	1715321017
姓 名	孙佳伟
指导老师	黄彩虹 李平

2021 年 1 月 8 日

摘要

现如今随着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，每个人都处于忙碌烦乱的社会当中，不论是在家庭、工作场所或是学校中，无时无刻充满着各方面的压力。在闲时，找到一种能够缓解压力、释放疲劳的娱乐方式，已成为大家共同的愿望。由于受到工作条件和时间的限制，又由于去 KTV 既方便省事，又能很好地娱乐放松，所以越来越多的人选择去 KTV 唱歌跳舞。

本文便基于 SQL server 和 C# 设计了一套 KTV 点歌系统。本系统的使用 SQL server 创建了数据库，并建立了歌曲、歌手、会员表和管理员信息表，歌曲排行榜，后台操作记录等 KTV 点歌系统数据库关系表。使用 C# 窗体基本界面的构建，使用 C# 处理界面和数据库的接口，用户可以通过数字点歌、歌星点歌、拼音点歌和歌名点歌四种方式点播歌曲，也可以方便的查看歌曲排行榜信息。而管理员也可以很方便的对整个 KTV 系统的信息进行管理。

关键词：KTV 点歌系统，SQL Server，C#

目 录

引言.....	2
第一章 概述.....	3
1.1 数据库设计背景.....	3
1.1 数据库设计意义.....	3
1.3 数据库设计内容.....	3
第一章 KTV 点歌系统设计	5
2.1 需求分析.....	5
2.1.1 信息要求分析	5
2.1.3 处理要求分析	6
2.2 概念模式的设计.....	6
2.2.1 局部 E-R 图设计	7
2.2.2 全局 E-R 图设计	10
2.3 逻辑模式设计.....	11
2.3.1 逻辑模式的转换	11
2.3.2 逻辑模式的规范化	12
第三章 数据库的实施与 C#界面开发	14
3.1 数据库基本表设计.....	14
3.2 登录模块界面设计及程序分析.....	15
3.3 前台点歌模块界面设计及程序分析.....	19
3.4 后台管理模块界面设计及程序分析.....	28
3.4.1 用户信息管理	29
3.4.2 歌曲信息管理	33
3.4.3 管理员密码修改	31
3.4.4 管理员信息	34
总结.....	35
参考文献.....	37

引言

《数据库课程设计》是自动化专业的实践课程。在较为系统地学完《数据库技术及应用》后，通过此课程设计，一方面可以增强学生对数据库的理解；另一方面可以培养学生查找资料、分析问题和解决问题的能力。

关系数据库设计过程分为以下 5 个阶段：需求分析、概念模式设计、逻辑模式设计、数据库实施、数据库运行维护。也就是说，数据库的设计过程是先进行认真细致的需求分析，在清楚用户的要求后抽象出实体和实体之间的关系，用建模方法 E—R 表示出来，然后根据具体实施的 DBMS 平台将其转换为相应平台所支持的数据库逻辑模式并进行规范化处理，最后进行建库建标等物理模式的设计，从而完成整个数据库模式的设计过程。

本次课程设计是对 KTV 点歌系统的设计，设计过程中也严格遵循以上的 5 个阶段。在设计完成的数据库中，应该具备有两大功能：1、对于用户，数据库的设计应该满足查询歌曲信息、点歌、删除已点歌曲、查询个人积分、注册账号及修改密码；2、对于管理员，数据库的设计应该满足歌曲信息的添加、删除、修改、查询和对用户信息的添加、删除、修改、查询等操作。也就是，数据库设计应该具备基本的增、删、改、查功能。

在第一部分，我们将介绍 KTV 点歌系统的设计背景、设计目的、和设计内容。

第二部分是本报告的重点，基于 KTV 点歌系统，对数据库设计的 5 个阶段，即需求分析、概念模式设计、逻辑模式设计、数据库实施、数据库运行与维护。

数据库的设计对于本系统有事半功倍的效果，如何提高办事效率，一直是数据库工作者的研究课题！

第一章 概述

1.1 数据库设计背景

由于经济文化的高速发展，人们对生活质量及工作环境的要求也越来越高，追求也越来越高。同时，随着生活节奏的加快，每个人都处于忙碌的社会当中，不论是在家庭，工作场所，或是学校，无时无刻充满着生活和学习上的压力。工作之余，找到一种释放压力，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 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，KTV 点歌系统已经成为了娱乐场所必不可少的一部分。KTV 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 数据库设计意义

KTV 点歌系统是娱乐场所的一项管理措施，因此开发 KTV 点歌系统具有较大的社会意义，同时点歌系统是一娱乐软件的开端，它具有简单的软件特征，系统结构与现实生活紧密结合，具体直观，具有典范的便捷软件特点。KTV 点歌系统主要用于对音乐的各项查找，操作简单，直观。用起来极为方便。减少了手工操对工作人员带来的不便和繁琐，使每个用户都能根据自己爱好选择自己喜欢的音乐。因此，KTV 点歌系统是企业化、智能化、科学化、正规化不可缺少的管理软件。

1.3 数据库设计内容

运用所学的数据库技术理论知识，对 KTV 点歌系统进行数据库设计。设计应包含需求分析、概念模式设计、逻辑模式设计、数据库实施、数据库运行和维护五个阶段。本次 KTV 点歌系统的开发主要包含前台点歌系统和后台管理系统模块的开发。前台点歌系统包含以下功能：

1. 歌曲点播（包含歌星点歌、数字点歌、拼音点歌、歌名点歌、全部歌曲）
2. 歌曲排行榜
3. 将选中的歌曲添加至已点歌单
4. 用户登录与账号注册
5. 用户积分查询和密码修改

管理员系统包含以下功能：

1. 用户信息管理（增加、删除、更新、修改用户信息）
2. 歌曲信息管理（增加、删除、更新、修改歌曲信息）
4. 管理员信息查看
5. 管理员密码修改

第二章 KTV 点歌系统设计

本次对于 KTV 点歌系统的设计，应该包含以下 5 个阶段：需求分析、概念模式式设计、逻辑模式式设计、数据库实施、数据库运行维护。如下图所示：

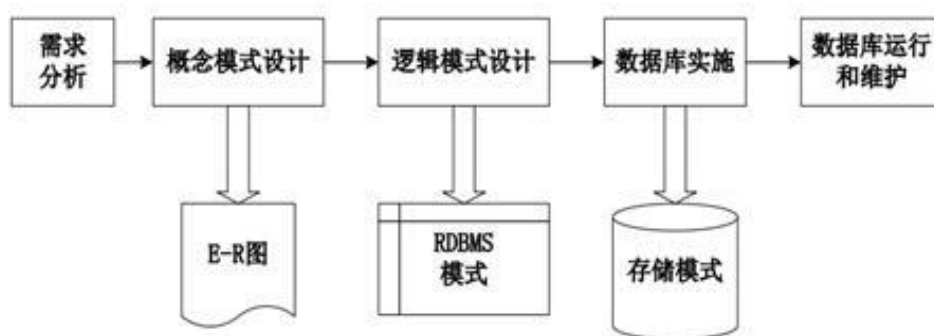


图 2.1 数据库设计过程

2.1 需求分析

2.1.1 信息要求分析

在此次《数据库课程设计》中，根据课程内容安排，所选题目为：KTV 点歌系统。经过对 KTV 点歌系统的结构化分析，得出信息要求如下：

- 1、歌曲信息：歌曲编号、名称、歌曲拼音缩写、歌手编号、歌手姓名、语种。
- 2、歌曲排行榜：歌曲编号、歌曲名称、总点击率。
- 3、我的歌单：歌曲编号、歌曲名称。
- 4、会员信息：会员编号、会员姓名、会员性别、出生日期、身份证号、注册时间、会员密码、积分。
- 5、管理员信息：管理员用户名、管理员密码。
- 6、后台操作记录：操作编号、操作内容、操作时间、操作用户

2.1.2 处理要求分析

根据以上 6 个表的属性，可以总结出，设计出来的数据库应该面向用户和管理员。对于用户，需要会员账号和密码登入前台点歌系统，进入后可以查询个人账户的积分以及修改账户密码。还可通过多种方式进行歌曲点播，如根据演唱歌星的名字、歌曲名称编号、歌曲名称拼音缩写以及歌曲名称等检索歌曲。同时，

可以查看歌曲排行榜信息。对于管理员，可使用管理员编号和密码进行登入后台操作系统，其进入后可对歌曲、歌手以及会员用户信息进行各类操作，如增加、删除、更新、修改等，也可以修改管理员账号的密码。

从上面表的结构关系较为混乱，如果没有进行必要的修改，将导致：（1）数据冗余；（2）修改异常；（3）删除异常；（4）插入异常等状况。所以接下来将根据需求分析所需要的功能做后续的分析 and 设计。

下图为简单的数据流程图：

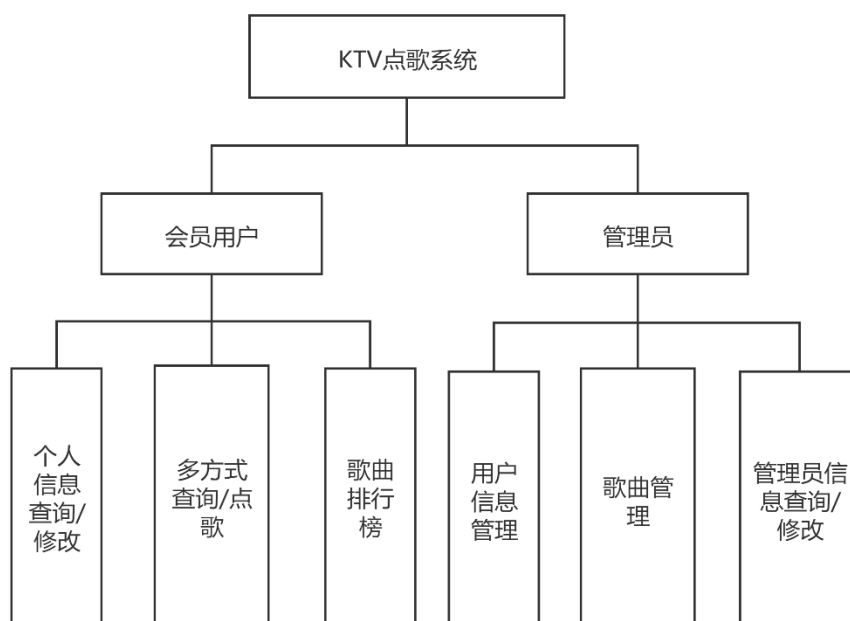


图 2-2 数据流程图

2.2 概念模式设计

数据库概念模式与计算机实现无关。它是对现实世界的第 1 层抽象。

直接把现实世界事物及其联系抽象为数据世界的数据库模型过于复杂，因此一般是先将现实世界的事物及其联系抽象为信息世界实体及其联系（概念模型），然后再将其转换为计算机世界的数据库模型（关系数据库模式）。

实体-联系模型（E-R 模型）是 1976 年美籍华人陈平山提出的。这个模型直接将现实世界中的事物及其之间的联系抽象为实体类型和实体间联系，然后用实体联系图表示数据库模型。

E-R 模型是用 E-R 图表示的。E-R 图中有下面四个基本成分：①矩形框：表示实体型；②菱形框：表示联系；③椭圆形框：表示实体或联系类型的属性；④

直线：联系类型与其所涉及的实体之间用直线连接，实体与实体之间用直线连接。

E-R 模型的基本元素由以下几个：(1) 实体：客观存在并可相互区别的事物称为实体。(2) 属性：实体所具有的某一特征称为实体的属性。(3) 键：也称关键字或码。唯一标识实体的最小的属性集称为实体的键。(4) 联系：现实世界的事物彼此是有联系的，反映在信息世界就是实体之间的联系。

实体间的相互关系称为联系。联系在数据库中的反映是实体集之间存在着这样和那样的联系，这种联系实际上表示了实体集之间的某种函数映射关系。可分为以下三类：

(1) 1:1 联系：已知实体集 A 和 B，若其中每个实体集中任何实体至多与另一实体集中的一个实体有联系，则称 A 和 B 的联系为“1 对 1 联系”，简记为 1:1 联系。

(2) 1:n 联系：已知实体集 A 和 B，若 A 中每个实体可与 B 中任意一个实体有联系，而 B 中每个实体至多与 A 中的一个实体有联系，则称 A 和 B 的联系为“1 对多联系”，简记为 1:n 联系。

(3) m:n 联系：已知实体集 A 和 B，若其中每个实体集中任何实体可与另一实体集中的多个实体有联系，则称 A 和 B 的联系为“多对多联系”，简记为 m:n 联系。

在下面的图形中我们首先将进行局部 E-R 图的分析设计，接着将进行全局 E-R 图设计。

2.2.1 局部 E-R 图设计

(1) 歌曲和歌手的局部 E-R 图

分析：图 2-3 为歌曲和歌手的局部 E-R 图。从图中可以看出：一首歌曲可以由多位歌手演唱，一位歌手可以演唱多首歌曲。所以二者是多对多的关系，即 m:n 的关系。其中歌曲的主键：歌曲编号；歌手的主键：歌手编号

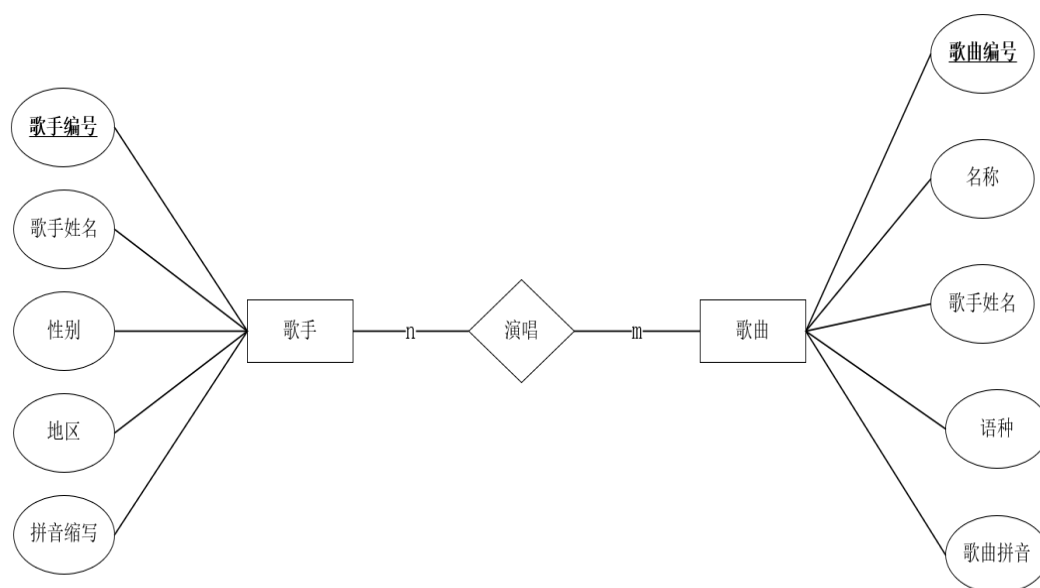


图 2-3 歌曲和歌手的局部 E-R 图

(2) 歌曲和用户的局部 E-R 图

分析：图 2-4 为歌曲和用户的局部 E-R 图。从图中可以看出：一首歌曲可以被多位用户点播，一位用户可以点播多首歌曲。所以二者是多对多的关系，即 $m:n$ 的关系。其中歌曲的主键：歌曲编号；用户的主键：用户名。

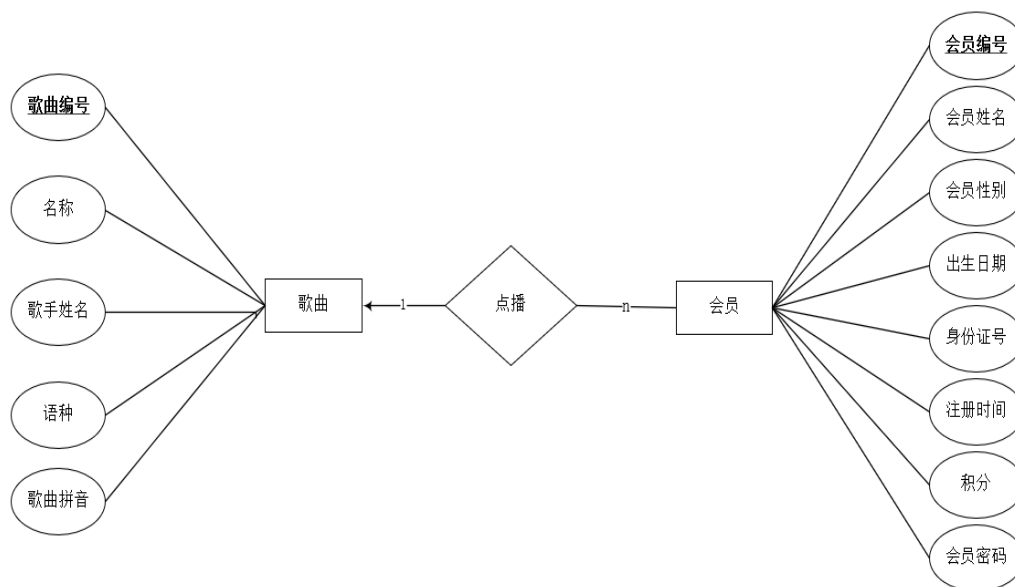


图 2-4 歌曲和用户的局部 E-R 图

(3) 歌曲和管理员的局部 E-R 图

分析：图 2-5 为歌曲和管理员的局部 E-R 图。从图中可以看出：一首歌曲可由多位管理员管理，一名管理员可以管理数据库中所有的歌曲。所以二者是多

对多的关系，即 $m:n$ 的关系。其中歌曲的主键：歌曲编号；管理员的主键：管理员编号。

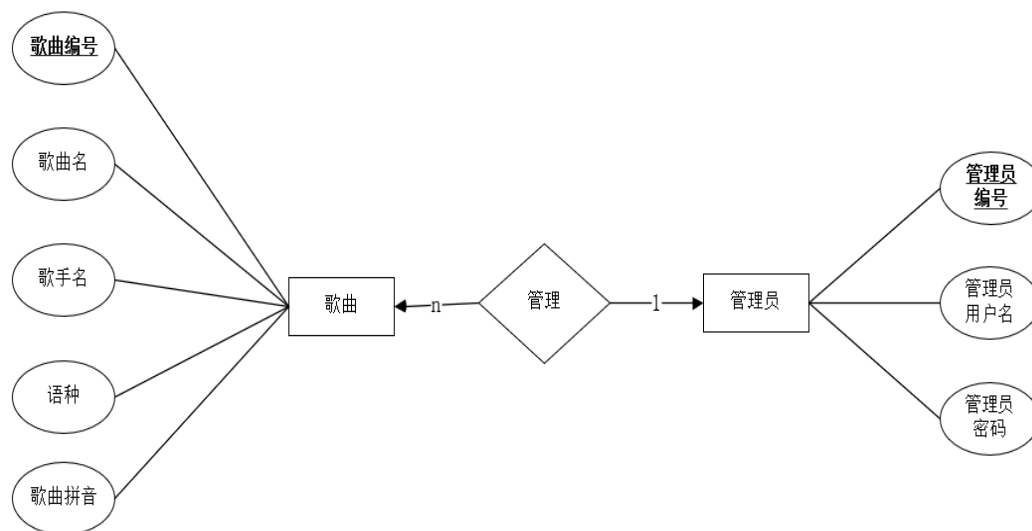


图 2-5 歌曲和管理员的局部 E-R 图

(4) 用户和管理员的局部 E-R 图 分析：图 2-6 为用户和管理员的局部 E-R 图。从图中可以看出：一名用户可由多位管理员管理，一名管理员可以管理数据库中所有的用户。所以二者是多对多的关系，即 $m:n$ 的关系。其中用户的主键：用户名；管理员的主键：管理员用户名。

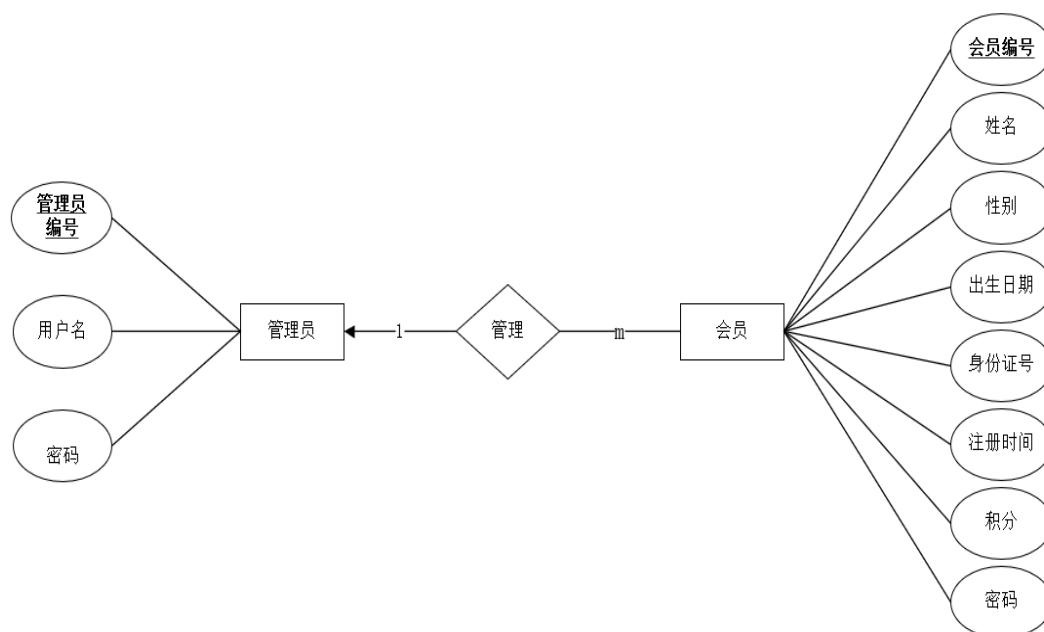


图 2-6 用户和管理员的局部 E-R 图

(5) 歌曲排行榜实体 E-R 图 分析：图 2-7 为歌曲排行榜实体 E-R 图。实体歌曲排行榜中一共有三个属性，分别是歌曲编号、歌曲名称和总点击率，每首歌都具有唯一的歌曲编号，因此将歌曲编号作为歌曲排行榜的主键。

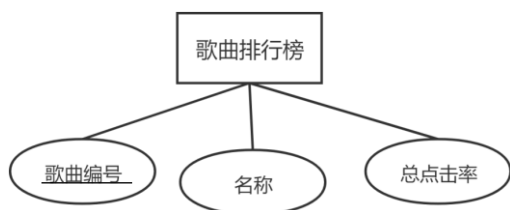


图 2-7 歌曲排行榜的实体 E-R 图

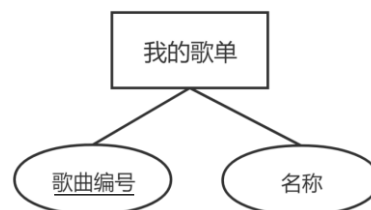


图 2-8 我的歌单的实体 E-R 图

(6) 我的歌单实体 E-R 图

分析：图 2-8 为我的歌单实体 E-R 图。实体我的歌单中一共两个属性，分别是歌曲编号和名称，每首歌都具有唯一的歌曲编号，因此将歌曲编号作为歌曲排行榜的主键。

(7) 后台操作记录实体 E-R 图

分析：图 2-9 为后台操作系统实体 E-R 图。实体后台操作系统中一共四个属性，分别是操作编号、操作内容、操作时间、操作人员，每次操作都有不同的操作编号，因此将操作编号作为后台操作系统的主键。

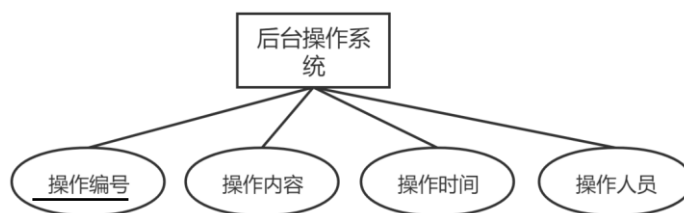


图 2-9 后台操作系统的实体 E-R 图

2.2.2 全局 E-R 图设计

在设计完所有实体之间的联系后，即局部 E-R 图设计，需将局部 E-R 图合并，在这个过程中需要去掉多个实体和属性之间的命名冲突和联系冲突等，从而生成一个完整的、满足应用需求的全局 E-R 图。图 2.2.8 即为 KTV 点歌系统的全局 E-R 图。通过图 2.2.8 可以清晰的看出各个实体之间的关系。

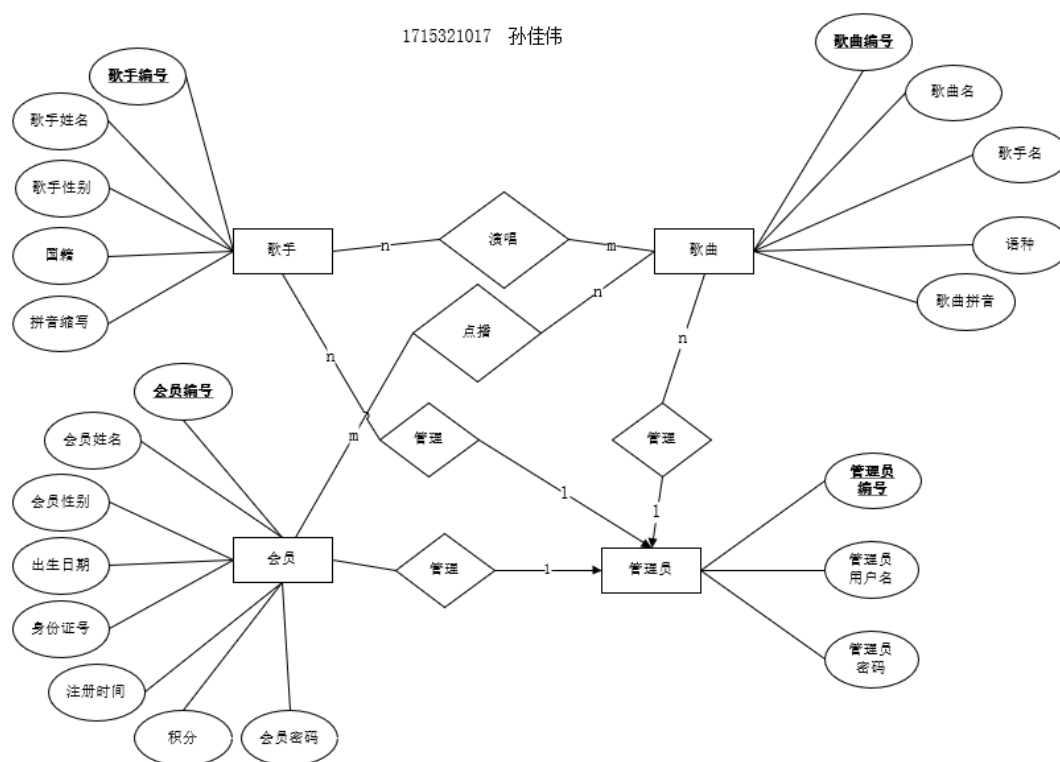


图 2-10 KTV 点歌系统的全局 E-R 图

2.3 逻辑模式设计

数据库概念模式设计完成后，数据库的设计进入到逻辑模式设计阶段。这时的数据库设计与具体的机器世界是相关联的。

逻辑模式设计阶段有两大任务：

- 1、按一定的规则将 E-R 图转换成关系模式。
- 2、关系模式的规范化处理。所谓关系模式的规范化处理，就是对关系式进行某种处理，使它满足一定的规范化要求。规范化的主要目的就是要尽可能地消除关系操作过程中的异常情况。

2.3.1 E-R 图到关系模式的转换

关系模型的逻辑结构是一组关系模式的集合。E-R 图则是由实体、实体的属性和实体之间的联系三个要素组成的。所以将 E-R 图转换为关系模型实际上就是要将实体、实体的属性和实体之间的联系转换为关系模式的集合。将 E-R 图转换为关系模式的转换规则如下：

规则 1：将 E-R 图中的一个实体集转换为一个同名关系模式。实体集的属性就是关系模式的属性。实体集的键码就是关系模式的主键。

规则 2：由于实体集之间的联系有 3 种类型：1:1、1:n 和 n:m，每一种类型转换都

不一样，现就每种类型予以说明。

若两实体集间的联系为 1:1，则可以在两个实体集转换成的两个关系模式中 的任意一个关系模式的属性集中加入另一个关系模式的主键和联系自身的属性，由此来完成 1:1 联系到关系模式的转换。

若两实体集间的联系为 1:n，则可以在多端实体集转换成的关系模式中加入 1 端实体集的键码和联系自身的属性，由此来完成 1:n 联系到关系模式的转换。

若两实体集间的联系为 m:n，则将联系转换成一个独立的关系模式，其属性 为两端实体集的键码加上联系自身的属性，联系关系模式的主键为复合键，由两 端实体集键码组合而成。

根据以上 E-R 模型向关系模型转换规则，对上述的 E-R 图进行转换，转换结果如下（主键用下划线标出，外键用波浪线标出）：

1) 歌手（歌手编号、姓名、性别、出生日期、地区）每位歌手的编号都是唯一的，通过属性歌手编号即可对应唯一的一位歌手，所以选择歌手编号作为主键。

2) 歌曲（歌曲编号、名称、歌曲拼音缩写、语种、歌手姓名、歌手编号）每首歌曲的编号都是唯一的，通过属性歌曲编号即可对应唯一的一首歌曲，所以选择歌曲编号作为主键，歌手编号是歌手信息表中的属性，所以在歌曲信息表中是外键。

3) 用户（会员用户名、会员性别、出生日期、身份证号、注册时间、会员密码、积分）每位用户的会员用户名都是唯一的，通过属性会员用户名即可对应唯一的一位用户，所以选择会员用户名作为主键。

4) 管理员（管理员编号、管理员用户名、管理员密码）每位管理员的管理员编号都是唯一的，通过属性管理员编号即可对应唯一的一位管理员，所以选择管理员编号作为主键。

5) 演唱（歌手编号、歌曲编号）演唱是歌曲和歌手这两个实体之间的联系，联系关系模式的属性是由两端实体集的键码和联系自身的属性构成，所以其属性为歌手编号和歌曲编号。同时，联系关系模式的键码为复合键码，由两端实体集键码以及相关属性组合而成，所以其主键为歌手编号和歌曲编号。以下六个联系关系模式集合的分析与此同理。

7) 点歌（歌曲编号、会员用户名）

8) 歌曲管理（歌曲编号、管理员编号）

9) 用户管理（会员用户名、管理员编号）

2.3.2 关系模式的规范化处理

在数据库逻辑模式设计阶段，完成 E-R 图到关系模式的转换后，就要进入关系模式的规范化处理阶段。这一过程主要研究关系模式内各属性之间的依赖关系，保持属性间好的数据依赖关系，消除导致异常的依赖关系，使得关系模式由低一级的规范模式上升到高一级的规范模式。我们设计的最终目的是使关系模

式规范化，所以最低应该达到第三范式。

首先，上述的关系中每一个属性都是不可再分的数据，所以均符合第一范式。其次，第二范式要求关系模式必须属于第一范式，且每个非主属性都是完全函数依赖于主键。在上述关系中，“歌手”、“歌曲”、“用户”、“管理员”都只有一个主键，所以不存在部分依赖关系，属于第二范式。“演唱”、“点歌”、“歌曲管理”和“用户管理”等四个关系中，均有两个主键，但是这四个关系均不存在非主属性，所以不存在部分函数依赖，也属于第二范式。最后，第三范式要求关系模式必须属于第一范式，且每个非主属性都不传递函数依赖于主键。在上述关系中，“歌手”、“歌曲”、“用户”、“管理员”都不存在传递函数依赖关系，属于第三范式。“演唱”、“点歌”、“歌曲管理”和“用户管理”等四个关系中，均有两个主键，但是这四个关系均不存在非主属性，所以不存在传递函数依赖，也属于第三范式。

第 3 章 数据库的实施与 C# 界面开发

数据库实施部分首先运用云服务器部署云数据库 SQL Server 建立 KTV 系统管理数据,再将各关系模型转化为实际的 SQL 数据表。C#界面开发部分主要是利用 C#窗体进行 KTV 前台点歌系统和后台管理系统人机交互界面的设计与制作。

3.1 数据库基本表设计

表 3-1 管理员信息表

编号	varchar(50)	Checked
管理名	varchar(50)	Unchecked
密码	varchar(50)	Checked

表 3-2 歌手信息表

歌手编号	varchar(50)	Unchecked
歌手姓名	varchar(50)	Checked
性别	varchar(50)	Checked
出生日期	varchar(50)	Checked
国籍	varchar(50)	Checked
歌曲类型	varchar(50)	Checked
拼音缩写	varchar(50)	Checked
地区	varchar(50)	Checked

表 3-3 歌曲信息表

歌曲编号	varchar(50)	Unchecked
名称	varchar(50)	Checked
歌手编号	varchar(50)	Checked
歌手姓名	varchar(50)	Checked
语种	varchar(50)	Checked
歌曲拼写	varchar(50)	Checked

表 3-4 会员信息表

会员编号	varchar(50)	Unchecked
会员用户名	varchar(50)	Unchecked
会员性别	varchar(50)	Checked
出生日期	varchar(50)	Checked
身份证号	varchar(50)	Checked
注册时间	datetime	Checked

积分	varchar(50) Checked
密码	varchar(50) Checked

表 3-5 歌曲排行榜

歌曲编号	varchar(50) Unchecked
名称	varchar(50) Checked
总点击率	varchar(50) Checked

表 3-6 后台操作记录

操作编号	varchar(50) Checked
操作内容	varchar(50) Checked
操作时间	datetime Checked
操作人员	varchar(50) Checked

表 3-7 我的歌单

歌曲编号	varchar(50) Checked
名称	varchar(50) Checked

3.2 登录模块界面设计及程序分析

登录模块由用户登录页面和管理员登录页面集合在一个窗口组成。登录页面在 KTV 点歌系统中起到了欢迎界面的作用,以构建一个美观舒适的人机交互环境,如图 3-1 所示。在该窗口有两个页面,分别是用户登录、管理员登录。其中点击用户下方的登录可以进入用户登录页面,登录成功后即进入前台点歌系统,如图 3-2 所示;点击管理员标签可以进入管理员登录页面,登录成功后即进入后台管理系统,如图 3-3 所示;点击退出,则系统会关闭。点击注册按键可以进入注册界面,用户可以注册账号和密码,如图 3-4。

The screenshot shows a window titled "KTV Song Request". It contains two main sections: "sign in as user" and "sign in as administrator". The "sign in as user" section has input fields for "username" and "passage", followed by a "Sign In" button. The "sign in as administrator" section has input fields for "account" and "passage", followed by a "Sign In" button. At the bottom right, there are "Sign Up" and "Exit" buttons.

图 3-1 用户与管理员登录界面

This screenshot shows the same login interface as Figure 3-1, but with a modal dialog box in the center. The dialog box has a close button (X) and contains the text "欢迎光临!" (Welcome!). Below the text is a "确定" (Confirm) button. The "Sign In" button for the user section is highlighted with a blue border, indicating it was the button clicked to trigger the dialog.

图 3-2 用户登录成功界面

This screenshot shows the same login interface as Figure 3-1, but with a modal dialog box in the center. The dialog box has a close button (X) and contains the text "欢迎光临!" (Welcome!). Below the text is a "确定" (Confirm) button. The "Sign In" button for the administrator section is highlighted with a blue border, indicating it was the button clicked to trigger the dialog.

图 3-3 管理员登录成功界面

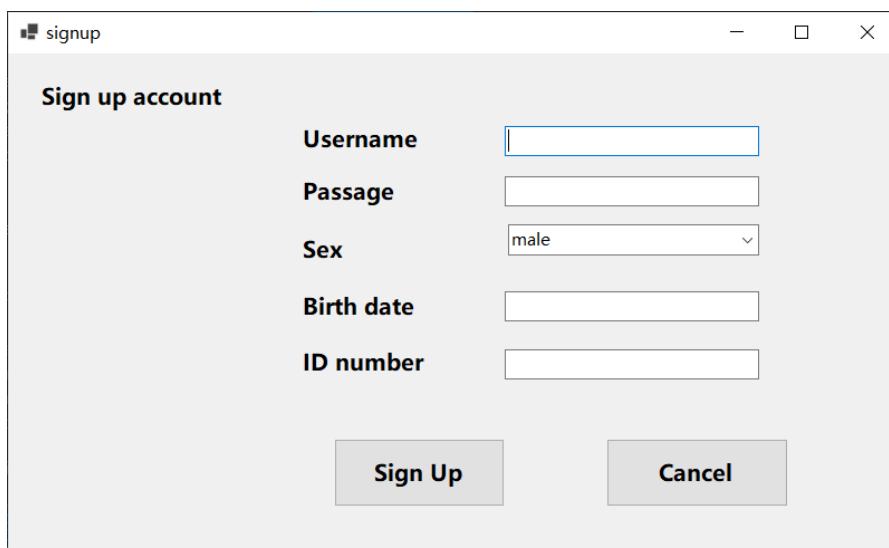


图 3-4 用户注册界面

在用户登录页面和管理员登录页面，用户和管理员必须输入正确用户名和密码才能成功登录前台点歌系统，图 3-5 所示。若点击退出则会退出程序。

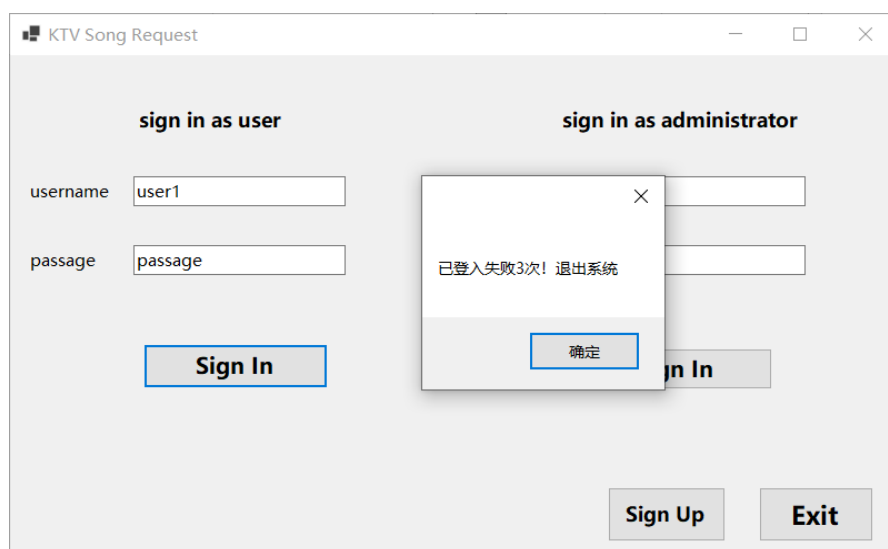


图 3-5 用户登录三次失败强制退出界面

为了提高系统的安全性，当用户或者管理员输入错误的用户名或密码的次数超过两次时，我们认为正在进行登录操作的人员不是我们的会员用户和系统管理员，系统执行退出指令，禁止该人员登录。如图 3-6 所示，管理员登录页面的设计在这点上与用户登录页面设计一致。

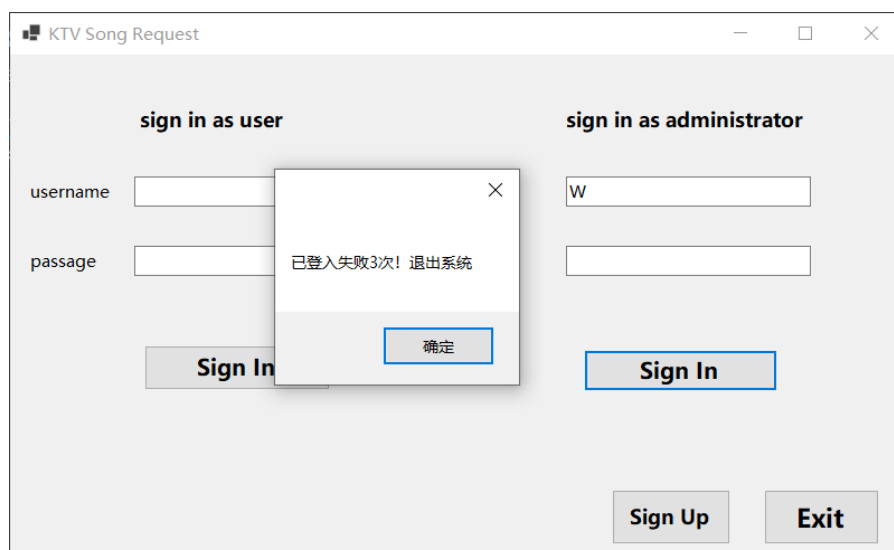


图 3-7 管理员登录失败三次退出系统

以下是 C#实现上述管理员登录功能，登录用户或者登录管理员输入的信息与数据库比对，实现登录功能。以及登录三次自动退出系统，提高系统安全性的部分代码

```
private void button3_Click(object sender, EventArgs e)
{
    try
    {
        // Build connection string
        SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
        builder.DataSource = "8.135.57.27"; // update me
        builder.UserID = "sa"; // update me
        builder.Password = "sun@5211314"; // update me
        builder.InitialCatalog = "KTV";
        using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
        {
            String sql = "select * from 用户 where 会员用户名=' " + TextBox_username.Text + "' and 密码=' " + TextBox2_passage.1
            SqlDataAdapter mydataadapter = new SqlDataAdapter(sql, connection);
            DataSet mydataset = new DataSet();
            mydataadapter.Fill(mydataset, "用户");
            if (mydataset.Tables["用户"].Rows.Count == 0)
            {
                n += 1;
                if (n < 3)
                {
                    MessageBox.Show("用户名或密码错误!");
                    TextBox_username.Text = "";
                    TextBox_username.Focus();
                }
                else
                {
                    MessageBox.Show("已登入失败3次! 退出系统");
                    this.Close();
                }
            }
            else
            {
                user = TextBox_username.Text;
                MessageBox.Show("欢迎光临!");
                this.Hide();

                n = 0;
                RequestSong requestSong = new RequestSong();
                requestSong.Show();
            }
        }
    }
}
```

```

private void button4_Click(object sender, EventArgs e)
{
    try
    {
        // Build connection string
        SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
        builder.DataSource = "8.135.57.27"; // update me
        builder.UserID = "sa"; // update me
        builder.Password = "sun@5211314"; // update me
        builder.InitialCatalog = "KTV";
        using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
        {
            String sql = "select * from 管理员 where 管理名='" + textBox_account.Text + "' and 密码='" + textBox4_passage.Text + "'";

            SqlDataAdapter mydataadapter = new SqlDataAdapter(sql, connection);
            DataSet mydataset = new DataSet();
            mydataadapter.Fill(mydataset, "管理员登录表");
            if (mydataset.Tables["管理员登录表"].Rows.Count == 0)
            {
                n += 1;
                if (n < 3)
                {
                    MessageBox.Show("用户名或密码错误!");
                }
                else if (n < 3)
                {
                    MessageBox.Show("用户名或密码错误!");
                    textBox_account.Text = "";
                    textBox4_passage.Text = "";
                    textBox_account.Focus();
                }
                else
                {
                    MessageBox.Show("已登入失败3次! 退出系统");
                    this.Close();
                }
            }
            else
            {
                MessageBox.Show("欢迎光临!");
                this.Hide();

                n = 0;
                AdministratorSystem administratorSystem = new AdministratorSystem();
                administratorSystem.Show();
            }
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("数据库连接失败!" + ex.Message);
    }
}

```

代码段 1 管理员登录代码

3.3 前台点歌模块界面设计及程序分析

会员用户在用户登录界面登录成功之后便可进入 KTV 点歌系统,如图 3-8 所示。在用户可以进行歌星点歌、数字点歌,如图、拼音点歌和歌名点歌四项点歌方式,。查询已点歌单和删除已点歌曲,如图 3-10 所示。查看歌曲排行榜,如图 3-11 所示。查看操作记录以及会员管理,如图 3-12 所示。

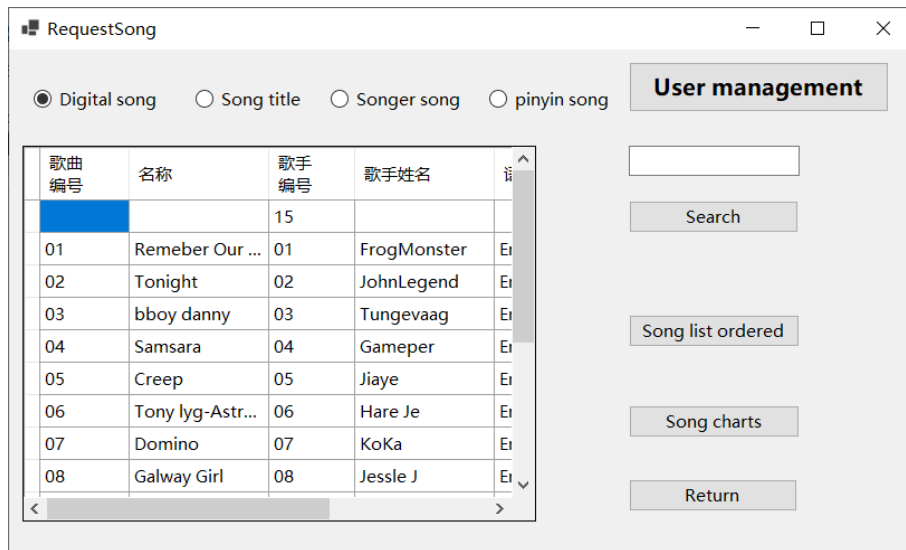


图 3-8 KTV 点歌界面

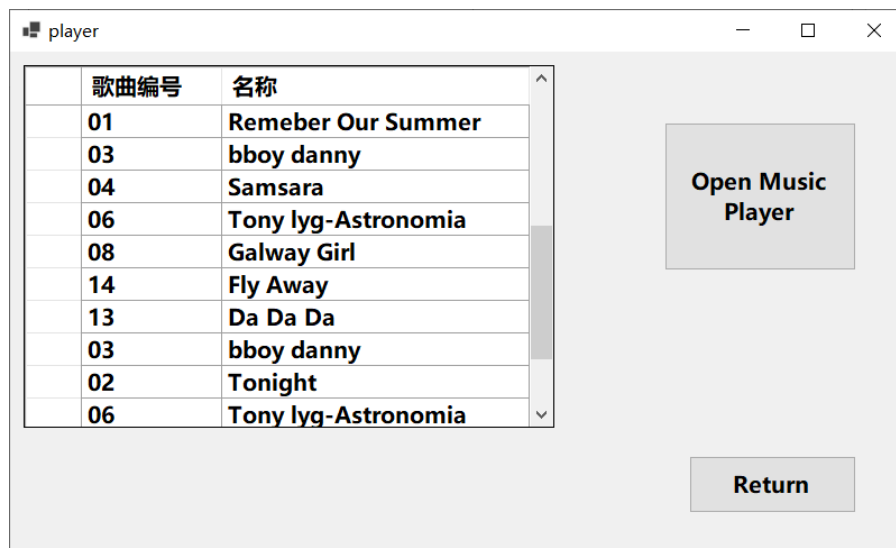


图 3-11 已点歌单界面

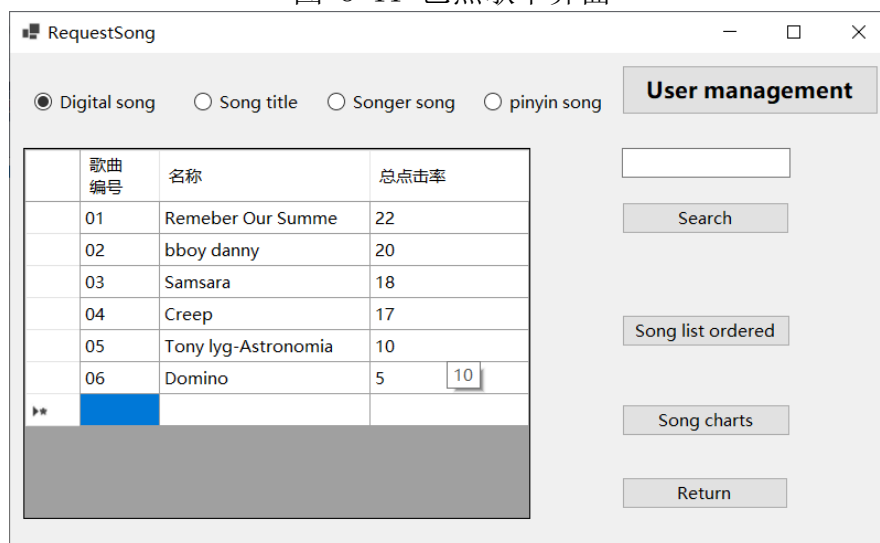


图 3-12 歌曲排行榜界面

(1) 歌曲搜索

在开始点歌窗口，用户可以通过输入关键字实现数字点歌（通过检索歌曲编号进行歌曲搜索）、歌星点歌（通过检索歌手姓名进行歌曲搜索）、拼音点歌（通过检索歌曲拼音缩写进行歌曲搜索）和歌名点歌（通过检索歌曲名称进行歌曲搜索）。同时，这四种歌曲搜索方式均可以实现模糊搜索。

1) 歌星搜索

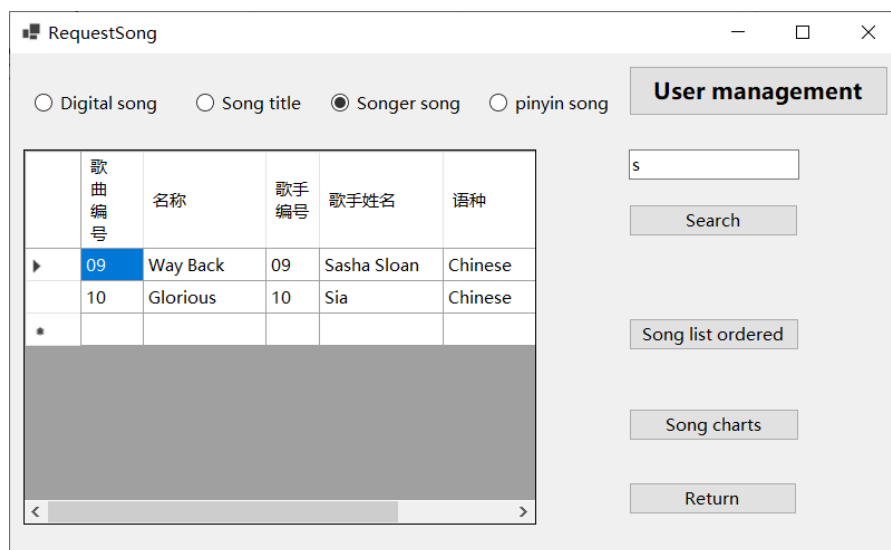


图 3-13 歌星搜索

实现通过歌手名字搜索歌曲，部分 C#代码如下

```
else if (radioButton3.Checked == true)
{
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = " select * From 歌曲 Where 歌手姓名 like '" + textBox1.Text + "%'" + " ";
        SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
        DataSet dataSet = new DataSet();
        mysqlcommand.Fill(dataSet, "数字点歌");
        dataGridView1.DataSource = dataSet.Tables["数字点歌"];
    }
}
```

代码段 2 歌手名称搜索代码

2) 数字搜索

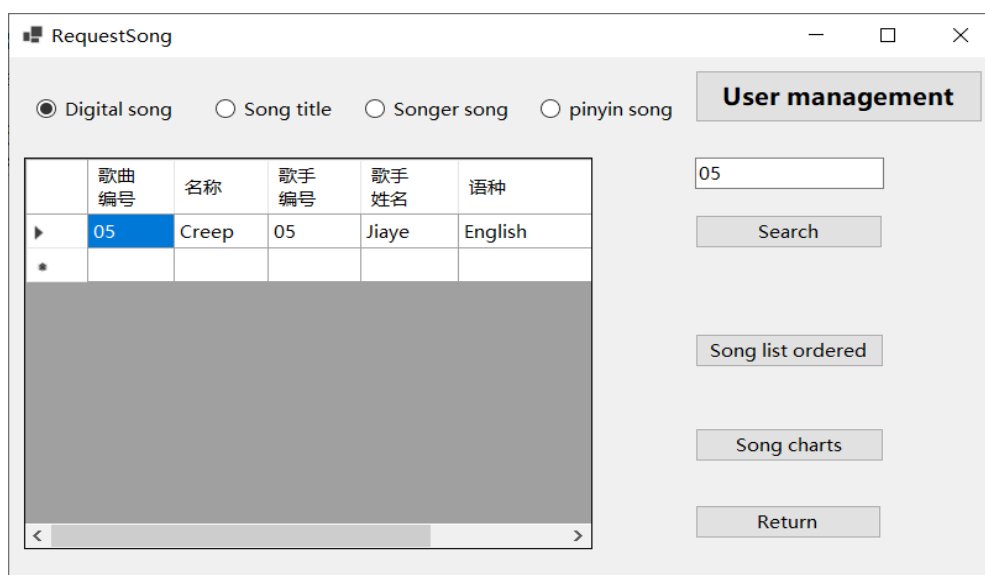


图 3-14 数字搜索

实现通过歌曲编号搜索歌曲，部分 C#代码如下

```
if (radioButton1.Checked == true)
{
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = " select * From 歌曲 Where 歌曲编号 like '%" + textBox1.Text + "%' " + " ";
        SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
        DataSet dataSet = new DataSet();
        mysqlcommand.Fill(dataSet, "数字点歌");
        dataGridView1.DataSource = dataSet.Tables["数字点歌"];
    }
}
```

代码段 3 歌曲编号搜索代码

3) 拼音搜索

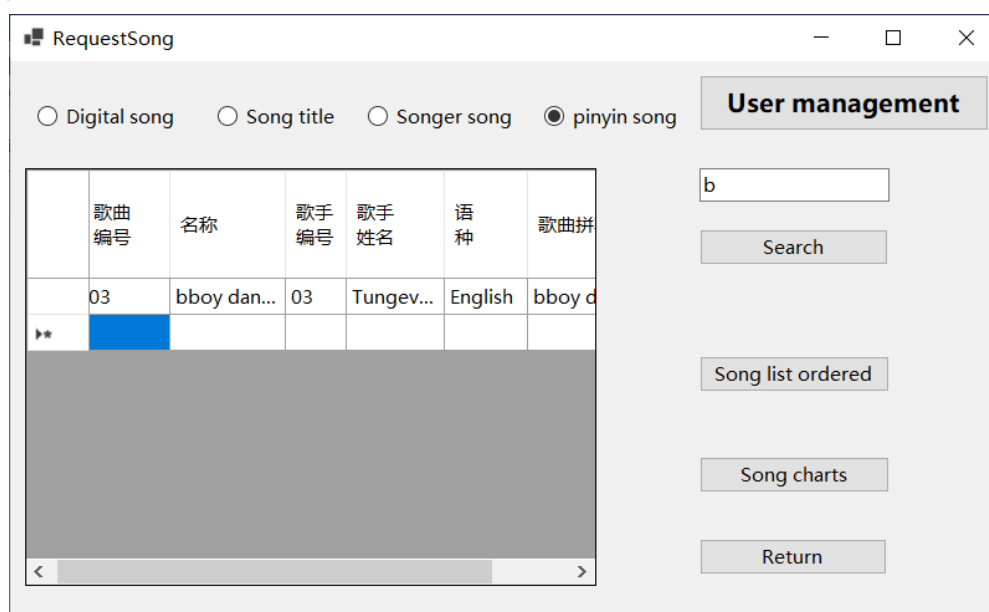


图 3-15 拼音搜索

实现通过歌曲拼写搜索歌曲，部分 C#代码如下

```

else if (radioButton4.Checked == true)
{
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = " select * From 歌曲 Where 歌曲拼写 like '" + textBox1.Text + "%' " + " ";
        SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
        DataSet dataSet = new DataSet();
        mysqlcommand.Fill(dataSet, "数字点歌");
        dataGridView1.DataSource = dataSet.Tables["数字点歌"];
    }
}

```

代码段 4 歌曲拼写搜索代码

4) 歌名搜索

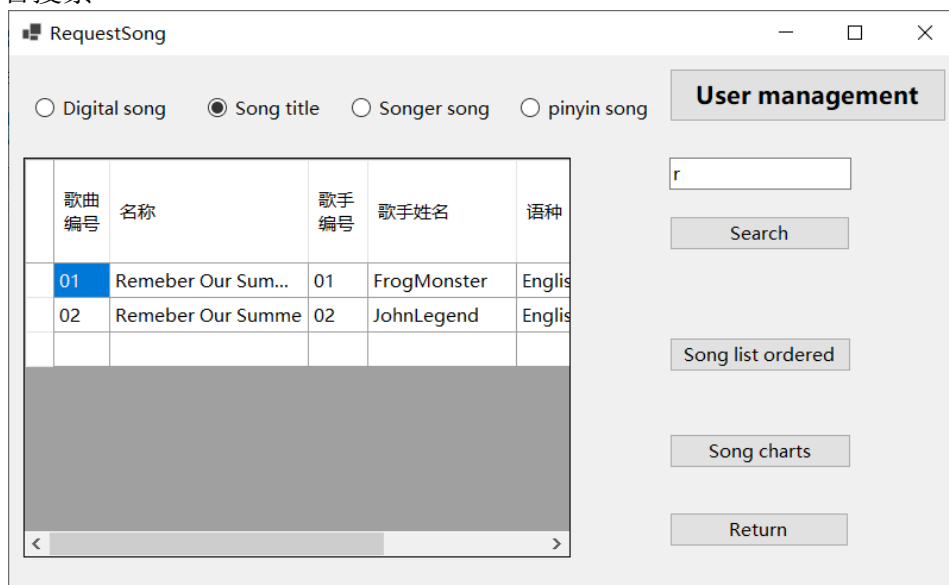


图 3-16 歌名搜索

```

else if (radioButton2.Checked == true)
{
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = " select * From 歌曲 Where 名称 like '" + textBox1.Text + "%' " + " ";
        SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
        DataSet dataSet = new DataSet();
        mysqlcommand.Fill(dataSet, "数字点歌");
        dataGridView1.DataSource = dataSet.Tables["数字点歌"];
    }
}

```

代码段 5 歌曲名称代码

5) 所有歌曲

点击搜索窗口的刷新按键，在信息显示窗口将会显示所有歌曲的信息，执行效果如图 3-17 所示。

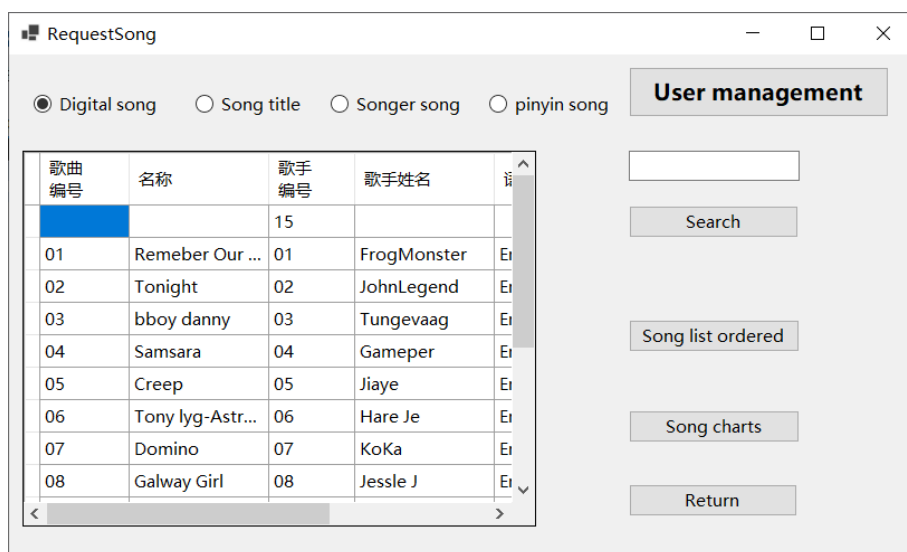


图 3-17 全部信息

6) 点歌

点击歌曲信息栏中的歌曲所在的行，会弹出添加确认的窗口，若选择是则将选择的曲目加入到已点歌单。如图 3-18，3-19。

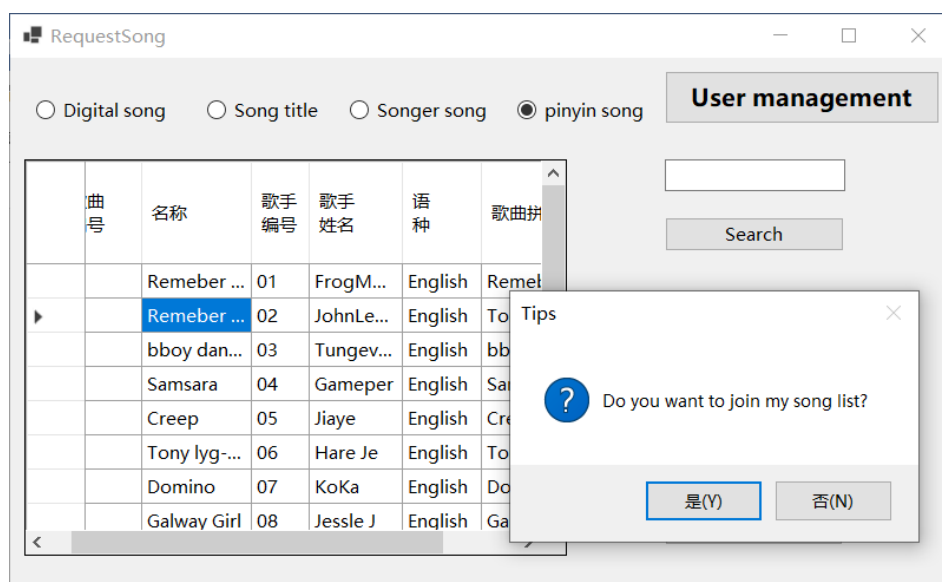


图 3-18 加入歌单界面

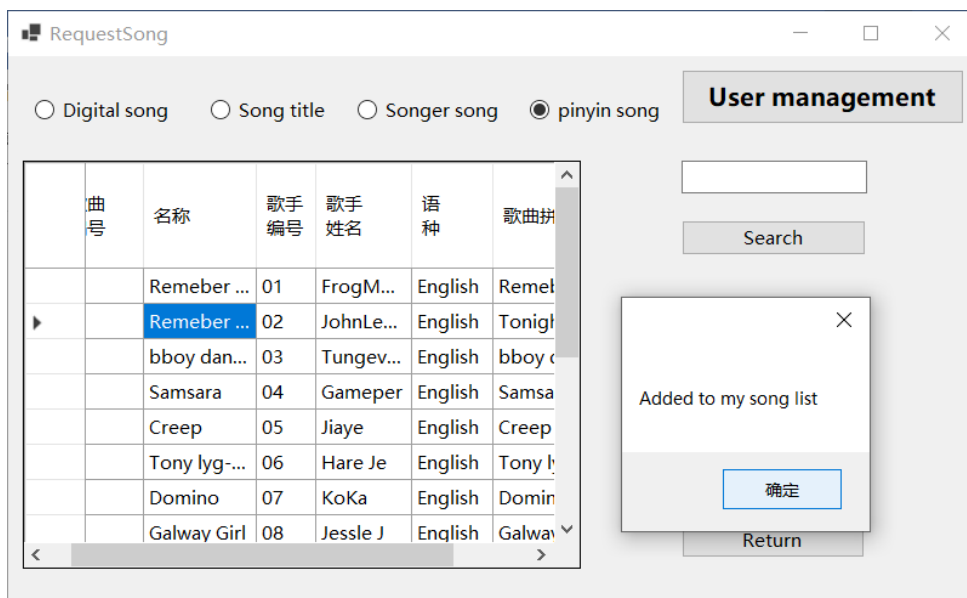


图 3-19 加入歌单成功界面

实现在 dataGridView 窗口直接进行双击点歌 C#代码如下

```
private void dataGridView1_CellContentClick(object sender, DataGridViewCellEventArgs e)
{
    int count = 0;
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        if (e.RowIndex < dataGridView1.RowCount - 1)
        {
            count += 1;
            string sql = "INSERT INTO 我的歌单 SELECT 歌曲编号, 名称 FROM 歌曲 WHERE 歌曲编号 = '" + dataGridView1.Rows[e.RowIndex].Cells[1].Text + "'";
            SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
            DataSet dataSet = new DataSet();
            if (MessageBox.Show("Do you want to join my song list?", "Tips", MessageBoxButtons.YesNo, MessageBoxIcon.Question) == DialogResult.Yes)
            {
                mysqlcommand.Fill(dataSet, "歌曲信息表");
                MessageBox.Show("Added to my song list");
            }
            string sql1 = "UPDATE 歌曲排行榜 SET 总点击率 += " + count + " WHERE 歌曲编号 = '" + dataGridView1.Rows[e.RowIndex].Cells[1].Text + "'";
            SqlDataAdapter mysqlcommand1 = new SqlDataAdapter(sql1, connection);
            DataSet dataSet1 = new DataSet();
            mysqlcommand1.Fill(dataSet1, "歌曲排行榜");
            signin signin = new signin();
            string sql2 = "UPDATE 用户 SET 积分 += " + count + " WHERE 会员用户名 = '" + signin.user + "'";
            SqlDataAdapter mysqlcommand2 = new SqlDataAdapter(sql2, connection);
            DataSet dataSet2 = new DataSet();
            mysqlcommand2.Fill(dataSet2, "歌曲排行榜");
        }
    }
}
```

代码段 6 点歌实现代码

(2) 已点歌单

点击已点歌单会在歌曲信息显示窗口显示出已点歌单的所有信息,如图 3-20 所示。选择显示窗口中的歌曲所在的行, 点击删除按钮即可删除该曲目。

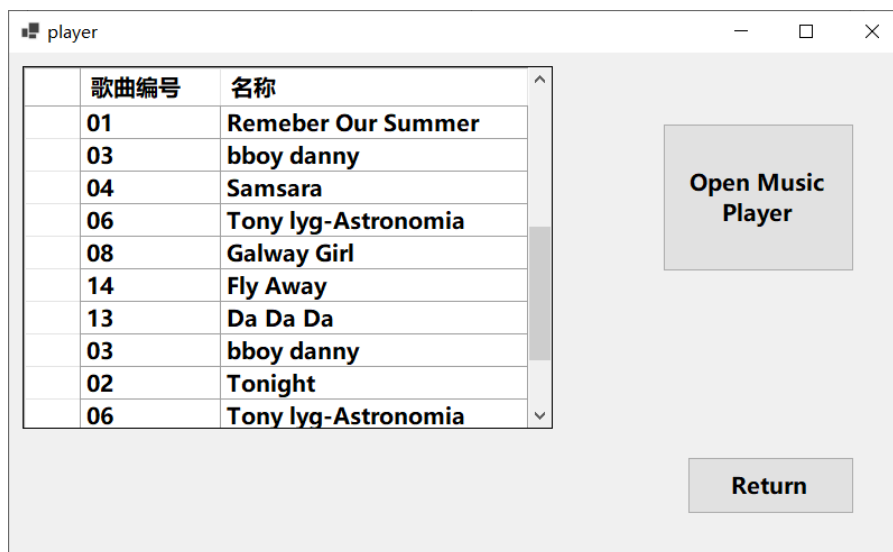


图 3-20 已点歌单界面

(3) 歌曲排行榜

点歌成功后会累加总点击率，以表示歌曲的热度，总歌曲表由最高点击率依次递减排列，如图 3-22 所示。

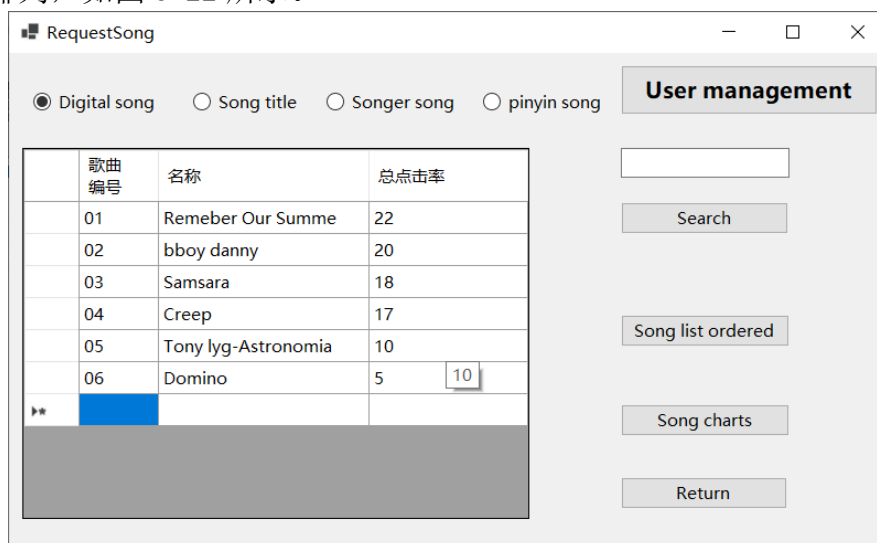


图 3-22 歌曲排行榜界面

(4) 会员管理

会员管理窗口可以查询本用户的积分，更改密码以及退出账号，如图 3-24 所示。

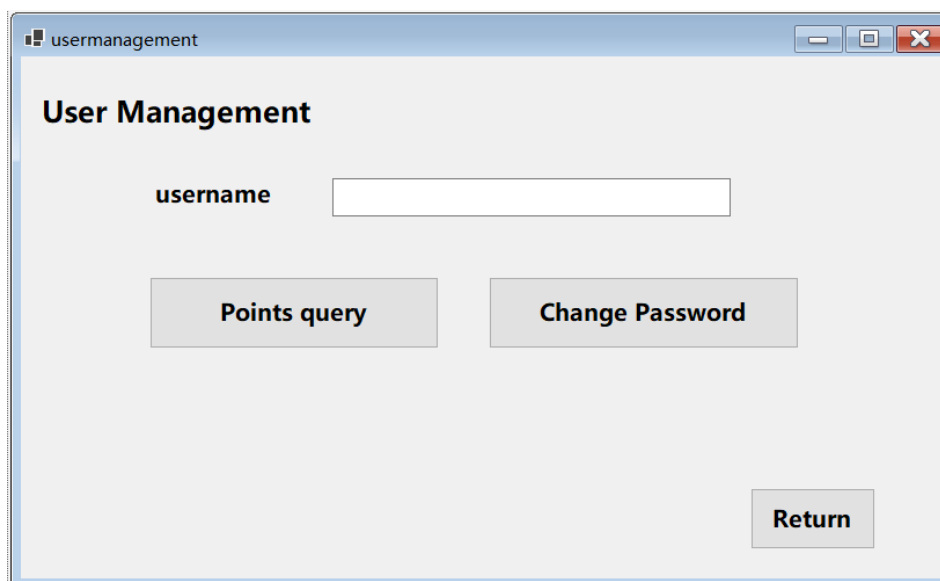


图 3-24 用户自我管理界面

查询本用户积分，用户每点一首歌曲，用户积分都增加 1，不同用户积分不同，表示用户点歌次数和唱歌频率。如图 3-25 所示。

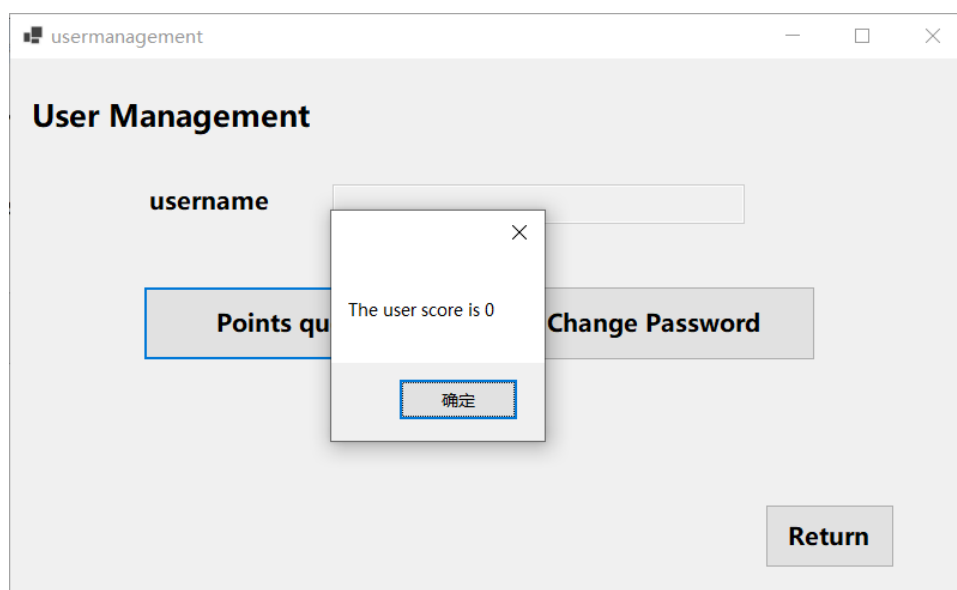
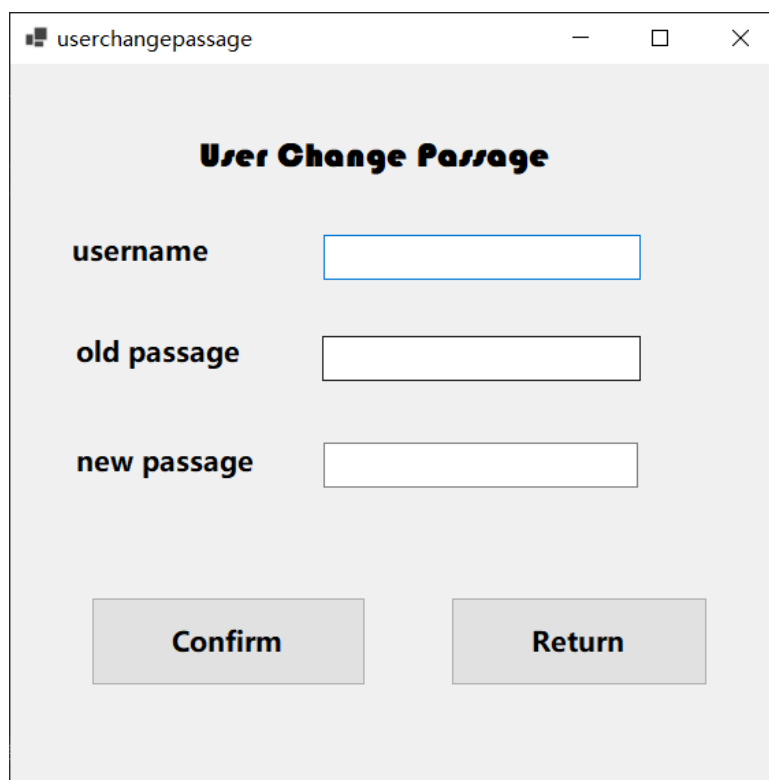


图 3-25 用户查询积分提示窗显示

(5) 更改密码

用户输入用户名和旧密码，再输入新密码，若旧密码正确，则改为新密码。如图 3-26 所示。

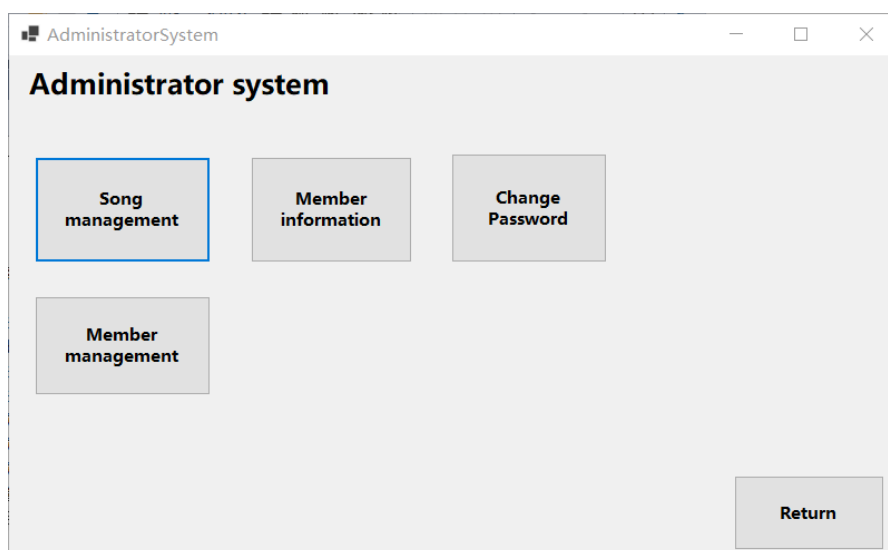


The image shows a web browser window titled "userchangeassage". The main heading is "User Change Passage". Below the heading, there are three input fields: "username", "old passage", and "new passage". At the bottom of the form, there are two buttons: "Confirm" and "Return".

图 3-26 用户修改密码界面

3.4 后台管理模块界面设计及程序分析

在登录页面选择管理员登录，并使用正确的管理员用户名和密码登录后便可进入 KTV 后台管理系统。在后台管理系统中，管理员可以对用户信息、歌曲信息进行管理，也可以修改管理员账号的密码。KTV 后台管理系统操作界面如图 3-27 所示。



The image shows a web browser window titled "AdministratorSystem". The main heading is "Administrator system". Below the heading, there are four buttons: "Song management", "Member information", "Change Password", and "Member management". At the bottom right, there is a "Return" button.

图 3-27 管理员管理界面

3.4.1 用户信息管理

用户信息管理操作界面如图 3-28 所示。在不输入任何用户信息时，管理员可以通过点击查询用户信息查询所有用户的信息。填入会员编号或者会员姓名或者身份证号，可以查询指定用户的信息；在信息表内新行所有信息格填写好信息后，可以点击增加用户将新的用户添加进会员信息表，如图 3-29 所示；若要删除某用户的信息时可以选择表格中想要删除的用户所在行，再点击删除用户即可，如图 3-30 所示；修改某用户时只要修改所选行的所有信息再点击修改用户，即可对用户信息进行更改。

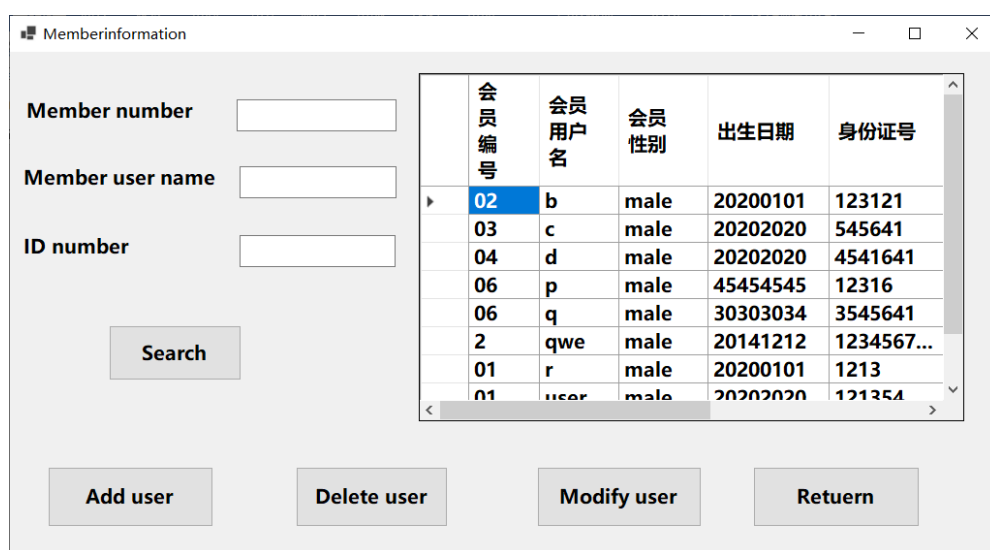


图 3-28 管理员用户信息管理界面

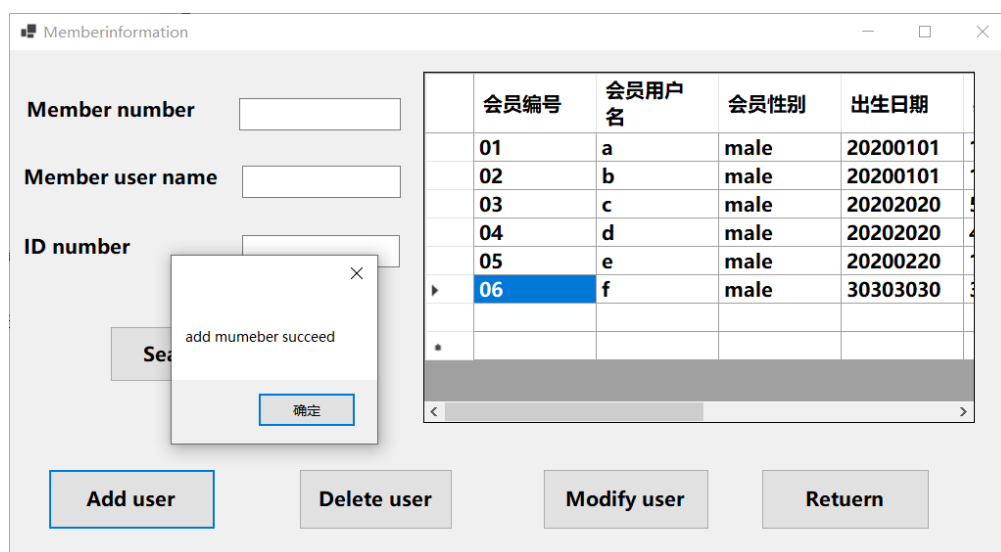


图 3-29 管理员增加用户功能

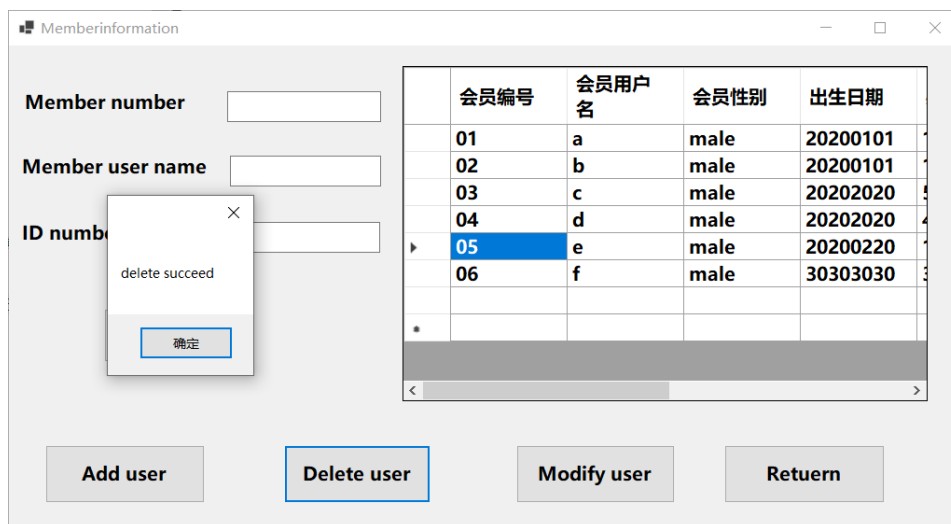


图 3-30 管理员删除用户功能

```
private void button2_Click(object sender, EventArgs e)
{
    // Build connection string
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = "SELECT * FROM 用户";
        SqlDataAdapter myadapter = new SqlDataAdapter(sql, connection);
        DataSet mydataset = new DataSet();
        myadapter.Fill(mydataset, "用户");
        DataRow _row = mydataset.Tables["用户"].NewRow();
        _row[0] = dataGridView1.CurrentRow.Cells[0].Value;
        _row[1] = dataGridView1.CurrentRow.Cells[1].Value;
        _row[2] = dataGridView1.CurrentRow.Cells[2].Value;
        _row[3] = dataGridView1.CurrentRow.Cells[3].Value;
        _row[4] = dataGridView1.CurrentRow.Cells[4].Value;
        _row[5] = dataGridView1.CurrentRow.Cells[5].Value;
        _row[6] = dataGridView1.CurrentRow.Cells[6].Value;
        _row[7] = dataGridView1.CurrentRow.Cells[7].Value;
        string sql1 = "select * from 用户 where 会员用户名='" + dataGridView1.CurrentRow.Cells[1].Value + "'";
        SqlDataAdapter myadapt1 = new SqlDataAdapter(sql1, connection);

        DataSet mydataset1 = new DataSet();
        myadapt1.Fill(mydataset1, "用户重复");
        if (mydataset1.Tables["用户重复"].Rows.Count <= 0)
        {
            if (dataGridView1.CurrentRow.Cells[1].Value == null && dataGridView1.CurrentRow.Cells[0].Value == null)
            {
                MessageBox.Show("add mumber succeed");
                mydataset.Tables["用户"].Rows.Add(_row);
                SqlCommandBuilder sqlCommandBuilder = new SqlCommandBuilder(myadapter);
                myadapter.Update(mydataset, "用户");
            }
            else
            {
                MessageBox.Show("The number and user name cannot be empty");
            }
        }
        else
        {
            MessageBox.Show("Failed to add repeatedly");
        }
    }
}
```

代码段 7 窗体搜索用户代码


```

private void button4_Click(object sender, EventArgs e)
{
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = "update 用户 set 会员用户名=' " + dataGridView1.CurrentRow.Cells[1].Value + "', 会员性别=' " + dataGridView1.CurrentRow.Cells[2].Value + "'";
        SqlCommand sqlCommand = new SqlCommand(sql, connection);
        connection.Open();
        sqlCommand.ExecuteNonQuery();
        MessageBox.Show("modify succeed");
        connection.Close();
        sql = "select * from 用户";
        SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sql, connection);
        DataSet dataSet = new DataSet();
        sqlDataAdapter.Fill(dataSet);
        dataGridView1.DataSource = dataSet.Tables["用户"];
    }
}

```

代码段 8 窗体修改用户信息代码

```

if (textBox1_numbermember.Text == "" && textBox2_numbermembername.Text == "" && idnumber.Text == "")
{
    // Build connection string
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = "SELECT * FROM 用户";
        SqlDataAdapter myadapter = new SqlDataAdapter(sql, connection);
        DataSet mydataset = new DataSet();
        myadapter.Fill(mydataset, "用户");
        dataGridView1.DataSource = mydataset.Tables["用户"];
    }
}
else {
    // Build connection string
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        String sql = "SELECT * FROM 用户 where 会员编号=' " + textBox1_numbermember.Text + "' or 会员用户名=' " + textBox2_numbermembername.Text + "'";
        SqlDataAdapter myadapter = new SqlDataAdapter(sql, connection);
        DataSet mydataset = new DataSet();
        myadapter.Fill(mydataset, "用户");
        dataGridView1.DataSource = mydataset.Tables["用户"];
    }
}

```

代码段 9 窗体修改用户信息代码

3.4.2 歌曲信息管理

歌曲信息管理在实现上与用户信息管理大致相同。所要实现的功能是对歌曲信息进行增删改查。如图 3-31

歌曲编号	名称	歌手编号	歌手姓名
05	Creep	05	Jiaye
06	Tony Iyg-As...	06	Hare
07	Domino	07	KoKa
08	Galway Girl	08	Jessle
09	Way Back	09	Sasha
10	Glorious	10	Sia
11	Sold Out	11	Mia V
12	Ido	12	Behm

图 3-31 歌曲信息管理界面

```
private void button3_Click(object sender, EventArgs e)
{
    SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
    builder.DataSource = "8.135.57.27"; // update me
    builder.UserID = "sa"; // update me
    builder.Password = "sun@5211314"; // update me
    builder.InitialCatalog = "KTV";
    using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
    {
        namespace ktvsystem
        {
            public partial class SongManagement
            {
                private void button3_Click(object sender, EventArgs e)
                {
                    String sql = "SELECT * FROM 歌曲";
                    SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
                    DataSet dataSet = new DataSet();
                    mysqlcommand.Fill(dataSet, "歌曲排行榜");
                    dataGridView1.DataSource = dataSet.Tables["歌曲排行榜"];
                }
            }
        }
    }
}
```

代码段 10 刷新歌曲代码

```
private void button1_Click(object sender, EventArgs e)
{
    if (textBox1_songnumber.Text == "")
    {
        MessageBox.Show("Fill in the song number first");
    }
    else {
        SqlConnectionStringBuilder builder = new SqlConnectionStringBuilder();
        builder.DataSource = "8.135.57.27"; // update me
        builder.UserID = "sa"; // update me
        builder.Password = "sun@5211314"; // update me
        builder.InitialCatalog = "KTV";
        using (SqlConnection connection = new SqlConnection(builder.ConnectionString))
        {
            String sql = "SELECT * FROM 歌曲";
            SqlDataAdapter mysqlcommand = new SqlDataAdapter(sql, connection);
            DataSet dataSet = new DataSet();
            mysqlcommand.Fill(dataSet, "歌曲信息表");
            DataRow _row = dataSet.Tables["歌曲信息表"].NewRow();
        }
    }
}
```

```

DataRow _row = dataSet.Tables["歌曲信息表"].NewRow();
_row[0] = textBox1_songnumber.Text.ToString();
_row[1] = textBox2_songname.Text.ToString();
_row[2] = textBox3_songnumber.Text;
_row[3] = textBox4_1.Text.ToString();
_row[4] = textBox5_language1.Text;
_row[5] = textBox6_renring1.Text;

dataSet.Tables["歌曲信息表"].Rows.Add(_row);
SqlCommandBuilder sqlCommandBuilder = new SqlCommandBuilder(mysqlcommand);
mysqlcommand.Update(dataSet, "歌曲信息表");

```

代码段 11 增加歌曲代码

3.4.3 管理员密码修改

与用户修改密码原理一样，如图 3-32 所示。

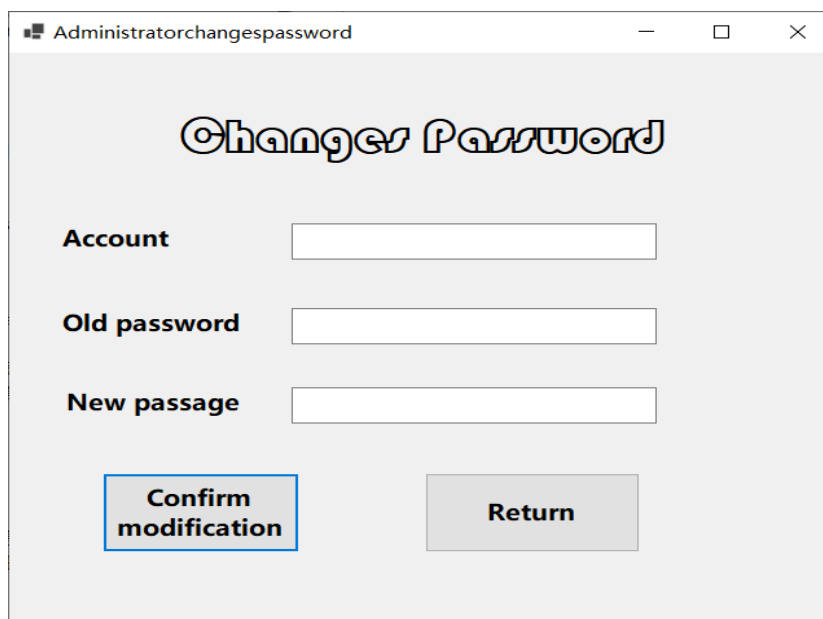


图 3-32 管理员修改密码界面

```

Setting sql = "select * from users where username = " + textBox1_account.Text + " and password = " + textBox2_old.Text + " and newpassword = " + textBox3_new.Text;
SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sql, connection);
DataSet mydataset = new DataSet();
sqlDataAdapter.Fill(mydataset, "密码更改1");
if (mydataset.Tables["密码更改1"].Rows.Count == 0)
{
    n += 1;
    if (n < 3)
    {
        MessageBox.Show("Wrong user name or password! ");
        textBox1_account.Text = "";
        textBox2_old.Text = "";
        textBox3_new.Text = "";
        textBox1_account.Focus();
    }
}

```

```
    {  
        MessageBox.Show("Login failed 3 times! Exit the system");  
        this.Close();  
    }  
}  
else  
{  
    sql = "update 管理员 set 密码=' " + textBox3_new.Text + "' where 管理名=' " + textBox1_account.Te  
    SqlCommand sqlCommand = new SqlCommand(sql, connection);  
    connection.Open();  
    sqlCommand.ExecuteNonQuery();  
  
    connection.Close();  
  
    MessageBox.Show("modification succeed");  
}
```

代码段 12 管理员修改密码代码

3.4.4 管理人员信息

通过表格显示管理人员的信息，因为没有设置增加管理员功能，故该界面只能简单的显示该管理员的信息。如图 3-33 所示。



图 3-33 管理员信息查看界面

总结

通过一周的课程设计，基本完成了 KTV 点歌系统的设计。前台点歌可以实现按数字点歌、拼音点歌、歌星点歌和歌名点歌，同时也可以查看歌曲排行榜和后台操作记录，查看用户积分。后台信息管理可以实现用户、歌曲、管理员的信息管理。但与此同时，此次的设计仍存在许多的不足，比如无法播放歌曲、添加管理员，歌曲非主属性信息不完整也可以添加曲目。同时，通过这次课程设计我巩固了 SQL Server 数据库和 VB.NET 界面开发的知识，通过实践学习到很多课本上学不到的知识，学习到了很多解决实际问题的经验。也让我深刻的意识到在做数据库设计时，必须包含需求分析、概念模式设计、逻辑模式设计、数据库实施、数据库运行和维护 5 个阶段。此次设计的系统很简洁也很微小，在设计过程中简化了非常多的步骤，在调试过程中，总会想到更多符合实际需求的条件去充实完善系统，也知道自己当初设计的时候考虑略欠周全，也明白了实际应用的系统是需要不断更新、维护，是需要许多人的智慧和努力！

参考文献

- [1]sql server 数据库在实践教学中的改革[J]. 刘芳芳. 报刊荟萃. 2017(12)
- [2]浅谈 SQL Server 数据库的特点和基本功能[J]. 闫旭. 价值工程. 2012(22)
- [3]做好数据库 SQL Server 的数据安全防护[J]. 孙英巍. 科技创新与应用. 2017(09)
- [4]SQL Server 数据库的证据收集与分析[J]. 徐坤. 科技传播. 2016(02)
- [5]《SQL Server 数据库》理实一体化课程的开发管窥[J]. 林忠会,符啸威. 科技经济市场. 2016(05)
- [6]SQL server 数据库的安全和管理策略探讨[J]. 杨娜. 计算机光盘软件与应用. 2012(23)
- [7]基于 SQL Server 链接服务器实现 Proficy Historian 数据库的访问[J]. 张海洲,尹龙,李玺. 现代食品. 2017(14)
- [8]浅析 SQL Server 数据库的安全问题[J]. 陈红艳. 科技创新导报. 2016(35)
- [9]基于 SQL Server 数据库的安全设计研究分析[J]. 李鹏. 无线互联科技. 2015(01)
- [10]论 SQL Server 数据库的安全策略[J]. 方程. 电脑编程技巧与维护. 2014(14)
- [11]邓英伟,袁晓红,张小琳,谭艳,彭伟. 智能在线考试系统研究[J]. 海峡科技与产业, 2017(10):67-68+74.
- [12]魏国利,张成刚. 基于 ASP.NET 框架的在线考试管理系统设计与实现[J]. 信息与电脑(理论版), 2017(23):121-122+125.
- [13]蒋玉芳. 基于 ASP.NET 的在线考试系统的设计与实现[J]. 科技广场, 2017(08):35-38.
- [14]刘晓婷,刘丰恕,朱斌. 基于 ASP.NET 的机械 CAD/CAM 课程在线考试系统的设计与实现[J]. 软件导刊(教育技术), 2018, 17(01):86-88.
- [15]张东圆,袁同山. 基于 Drupal 的在线考试系统的研究与应用[J]. 信息系统工程, 2018(01):92+94.
- [16]马生骏. 基于 AJAX 技术在线远程考试系统的设计与实现[J]. 甘肃科技纵横, 2018, 47(01):1-3.
- [17]林志灿. Struts 与 Hibernate 框架下在线考试系统的设计与实现[J]. 信息技术与信息 7 化, 2018(01):79-83.
- [18]伍四军. 一种基于 WEB 的在线考试系统设计[J]. 科技广场, 2017(11):47-50.
- [19]刘冬,冉崇善. 基于 C/S 模式的计算机应用在线考试系统分析[J]. 计算机产品与流通, 2017(08):31.
- [20]黄春. 基于 JSP 的在线考试系统的开发与实现[J]. 信息通信, 2018(02):163-164.
- [21]金圣道. 在线考试及试卷分析系统的设计与实现[J]. 电子技术与软件工程, 2018(07):170-171.

- [22] 马万强. 计算机在线考试系统的设计与实现[J]. 信息与电脑(理论版), 2018(05):74-75+78.
- [23] 金誉华, 周蕾. 基于可变比例的模块化试题生成的在线考试系统的设计与实现[J]. 安徽电子信息职业技术学院学报, 2018, 17(02):14-17.
- [24] 郭子文, 刘平. 基于 MVC 模式的在线考试系统设计与实现[J]. 电脑知识与技术, 2018, 14(05):71-72.
- [25] 吴光成. 基于.NET 的在线考试系统的设计与实现[J]. 教育教学论坛, 2018(22):94-95.