

《 软 件 工 程 》

课 程 设 计 论 文

题 目： 智能点歌系统

姓 名： 查琪乐、张桂兰、李成文

学 号： 121060100105、121060100208、121060100231

专业班级： 21计科1班、21计科2班

指导老师： 林 卫 中

完成日期： 2023.10.20

**前 言**

**摘 要：**本文档是智能点歌系统的软件文档，描述了智能点歌系统的功能、结构、设计、编码和测试等方面的详细信息。智能点歌系统是一个基于互联网的音乐服务平台，为用户提供方便、快捷、个性化的点歌体验。用户可以通过用户界面访问点歌系统，进行注册、登录、修改个人信息、查看历史记录和收藏歌单等操作。用户可以通过两种方式搜索和播放音乐：一是通过歌曲条件进行精确搜索，二是通过模糊语句进行智能搜索。用户还可以在推荐界面中查看和播放智能推荐的音乐，这些音乐是根据用户的喜好、场景、心情等因素进行个性化推荐的。操作员可以在系统后台界面对音乐库和用户信息进行管理和维护。本文档参考了智能点歌系统需求文档、数据流图绘制规范和软件工程教程等资料，为开发人员、测试人员、运维人员和用户提供参考和指导。

**关键词：**智能点歌系统 个性化推荐

**目 录**

[1 问题定义 1](#_Toc146716507)

[1.1 业务调查 1](#_Toc146716508)

[1.2 系统目标 1](#_Toc146716509)

[1.3 计算机配置方案 2](#_Toc146716510)

[2 可行性分析 3](#_Toc146716511)

[2.1 经济可行性 3](#_Toc146716512)

[2.2 技术可行性 3](#_Toc146716513)

[2.3 社会可行性 4](#_Toc146716514)

[2.4 风险与不确定性 4](#_Toc146716515)

[2 需求分析 5](#_Toc146716516)

[3.1 业务分析—业务模型 5](#_Toc146716517)

[3.2 功能建模—DFD 8](#_Toc146716518)

[3.3 数据建模—E-R图 8](#_Toc146716519)

[3.4 行为模型—状态转换图 8](#_Toc146716520)

[3.5 数据字典 8](#_Toc146716521)

[4 总体设计 9](#_Toc146716522)

[4.1 体系结构设计 9](#_Toc146716523)

[4.2 接口设计 9](#_Toc146716524)

[5 详细设计 10](#_Toc146716525)

[5.1 数据设计 10](#_Toc146716526)

[5.1.1 实体的映射 10](#_Toc146716527)

[5.1.2 联系的映射 10](#_Toc146716528)

[5.1.3 映射汇总 10](#_Toc146716529)

[5.1.4 设计视图 10](#_Toc146716530)

[5.2 过程设计 10](#_Toc146716531)

[5.2.1 算法设计 10](#_Toc146716532)

[5.2.2 数据结构细节和数据操作的设计 10](#_Toc146716533)

[6 代码实现 11](#_Toc146716534)

[7 测试与部署 12](#_Toc146716535)

[8 展望与致谢 13](#_Toc146716536)

# 1 问题定义

## 业务调查

音乐作为人类文化的重要组成部分，一直伴随着人类文明的发展和时代的变迁。适时地欣赏音乐，不仅有助于我们保持身心健康，提高工作效率，还能够提升我们的生活质量。因此，音乐已经成为我们生活中不可或缺的一种娱乐方式。随着社会和经济的进步，人们对音乐体验的需求也在不断变化。现在的用户不仅希望能够方便地找到自己喜欢的歌曲，而且还期待音乐平台能提供个性化的推荐服务，帮助他们发现更多可能喜欢的歌曲。

基于上述背景，本项目拟开发一款智能点歌系统，它主要有以下开发意义：能够准确、高效地帮助音乐消费者找到适合自己的音乐，提高用户满意度。以下是本系统包含的功能要求：

**（1）用户管理功能：**该系统应该能够让用户注册、登录、修改个人信息、查看历史点歌记录等。用户可以通过手机号、邮箱或第三方账号（如微信、QQ等）进行注册和登录。用户信息包括用户号、用户名、性别、年龄等。用户可以随时修改自己的个人信息，也可以注销自己的账号。

**（2）音乐库管理功能：**该系统应该能够维护一个丰富的音乐库，包括各种类型、风格、语言的音乐。音乐库应该定期更新，添加新的音乐或删除过时的音乐。音乐库中的每首音乐应该有相应的元数据，如歌名、歌手、发行时间、编号等。音乐库应该支持多种搜索方式，如按照歌名、歌手等进行搜索。

**（3）智能推荐功能：**该系统应该能够根据用户的个人信息和历史点歌记录，为用户推荐合适的音乐。推荐算法应该考虑用户的喜好、场景、心情等因素，动态调整推荐策略。推荐结果应该展示在用户的首页或专门的推荐页面，用户可以选择播放或收藏推荐的音乐。

**（4）智能点歌功能：**该系统应该能够接受用户的自然语言输入，如“我想听一首轻松的英文歌”、“给我来一首周杰伦的快歌”等，然后根据输入内容和音乐库中的数据，为用户点播一首符合要求的音乐。点播结果应该展示在用户的播放页面，用户可以控制播放、暂停、上一首、下一首等操作。点播结果也应该保存在用户的历史点歌记录中。

## 1.2 系统目标

（1）建立一个功能全面的智能点歌系统，满足用户的基本需求和期望。

（2）系统应该提供友好的用户界面，方便用户进行操作和交互，并支持多种语言，适应不同地区和文化的用户。

（3）系统应该保证高可靠性和安全性，防止数据丢失、错误或泄露。

（4）系统应该保证高准确度的推荐和点歌结果，符合用户的要求和喜好。

（5）系统应该具有良好的可扩展性和可维护性，能够适应音乐市场和用户需求的变化，添加新的功能或优化现有的功能。

（6）系统应具有良好的兼容性和可移植性，能够在不同的平台和设备上运行，如网页、手机、电视等。

## 1.3 计算机配置方案

确立计算机配置方案需要综合考虑系统的客观约束条件、新系统的处理方式、联机存储量、系统所需硬件资源以及系统所需软件。针对一般的电子商务业务，同时考虑到开发成本,系统设计成B/S两层结构。综合考虑以上问题,本系统配置方案如下：

1. **分布方案**

本系统采用浏览器/服务器的运行方式，数据和程序集中存储在服务器上。对服务器硬件的最低要求如下。

·处理器：Inter i5-11400F或者更高

·内存：16G或者更高

·磁盘空间：12G或者更高

1. **软件环境**

·服务器端操作系统：Windows 7 LTS

·数据库服务器：MySQL 8.0

·浏览器端操作系统：Windows 10或更高版本

·浏览器：IE11及其以上版本（或者其他浏览器，如Chrome\Firefox等）

# 可行性分析

## 2.1 经济可行性

经济可行性的目的是估算开发成本，确定项目值得投资。这里从以下三个方面来讨论智能点歌系统系统开发的经济可行性。

（1）随着个人计算机的迅速崛起，线上点歌等新兴娱乐项目越来越为人们所接受， 它们有着传统模式无可比拟的优点：更少的材料需求、更丰富的歌曲选择、更灵活的点歌方式、更快速的点播响应、数据统计和分析等。随着生活水平的发展，人们对于精神层面的满足和放松需求也越来越强烈。在这样的背景下，一个优秀的智能点歌系统可以为人们提供一种舒缓生活节奏、缓解压力的方式，同时能够丰富身心。可见此系统的前景良好。

（2）本系统奉行界面简约高效、功能逻辑严格直接的原则，故而开发此系统所花费的设计与编码时间是可控的，开发周期预估为两至三个月。

（3）由于开发此系统的相关技术已经较为成熟，软件整体功能设计并不复杂，体量较小，是纯线上的软件，故而资金投入不多且可控，总体费用大约在20000块钱左右，主要的收入来源可以结合用户付费订阅、广告投放、音乐版权分成等方式，增加直接收益；并提高用户的满意度和忠诚度，通过用户口碑传播、用户留存率提高、用户消费能力增强等方式，增加间接收益。预计投资回收期在5年左右。

综上所述，开发智能点歌系统在经济上是可行的。

## 2.2 技术可行性

技术可行性要对项目的功能和限制条件等进行分析，目的是确定项目能否实现。一般包括开发风险和技术水平。

在开发智能点歌系统时，虽然市场上已经存在相似的同类产品，但这也说明了该领域有一定的市场需求和潜力。通过对现有产品的分析和改进，可以提供更好的用户体验和功能，从而吸引更多的用户。另外，本系统不涉及复杂的物理仿真或科学计算，而是基于现有的浏览器/服务器WEB开发技术进行实现。这些技术已经非常成熟，并且有很多开发资源和工具可供使用。因此，本系统的开发风险较低。

开发技术方面，本系统选择生命周期法进行分析和设计，实现过程基于Java + Web + MySQL，在技术上是可行的；而且笔者在之前的课程中，在前端领域已经学习了HTML5、CSS3和JavaScript,后端学习了Java、Python和MySql，都能为系统的开发提供技术支持，其他有关服务器端的连接配置将在之后继续学习，故认为本系统开发技术可行。

综上所述，开发智能点歌系统在技术上是可行的。

## 2.3 社会可行性

社会可行性是在特定环境下对项目的开发与实施，受多方面社会因素的影响。本系统主要对以下几个方面的社会影响因素进行分析：

在法律方面，本系统遵守相关的版权法律，保证所使用的歌曲都经过了合法授权，避免侵犯他人的知识产权。同时，本系统也对用户的隐私和个人信息进行保密，坚守不泄露或滥用用户的数据的原则。

在伦理方面，本系统尊重用户的喜好和选择，不强制或诱导用户点播不适宜的歌曲，如含有暴力、色情、政治敏感等内容的歌曲。

在文化方面，本系统适应不同地区和群体的文化特征，提供多种语言和风格的歌曲，满足用户的多元化需求。同时，本系统也需要尊重不同文化的价值观和习俗，不传播或宣扬有损于社会道德和风尚的内容。

综上所述，开发智能点歌系统具备社会可行性。

## 2.4 风险与不确定性

数据库技术发展到今天，其安全性已经大大改进，保障了线上软件的安全性，且本系统不涉及用户的财产，风险较小。本系统还执行了严格的保密措施，对于信息的保护非常严格，仅用于本平台使用，故不具有法律风险；开发此系统时间、资金都较少且可控，不具有投入风险；此系统却有较大市场需求，盈利产出风险较小。

综上所述，开发智能点歌系统的风险和不确定性在接受范围内。

# 需求分析

需求分析是指在软件开发过程中，通过与用户、客户和开发方等进行沟通和协商，分析和定义软件系统所要实现的功能和性能，以及软件系统所要满足的约束条件和质量标准。

需求分析是一个非常重要的环节，因为它直接影响到软件系统的质量、成本和进度。如果需求获取不充分或不准确，可能会导致软件系统无法满足用户的真实需求，或者需要进行频繁的修改和维护，从而增加开发风险和成本。

需求分析主要包括以下几个步骤：

（1）需求获取：采用不同的技术，如访谈、问卷、观察、原型等，收集用户对软件系统的期望和需求。

（2）需求提炼：对收集到的需求进行整理、分类、归纳、抽象、优化等，消除需求之间的冲突和矛盾，形成一致的需求视图。

（3）需求描述：将分析后的需求用规范的语言和格式进行描述，形成需求规格说明文档，作为软件开发的基础和依据。

（4）需求验证：对需求规格说明文档进行检查、评审、测试等，确保需求是正确的、完整的、一致的、可行的、可测试的和可跟踪的。

## 3.1 需求获取

本团队首先对现有的音乐软件及相似软件进行深入调查和分析，并亲自体验点歌这一操作过程，经过大量讨论后最终达成一致，得到智能点歌系统的业务流程如下：

用户通过用户界面访问点歌系统，如果是第一次使用，需要进行注册和登录，如果已经注册过，可以直接登录。用户可以通过手机号、邮箱或第三方账号（如微信、QQ等）进行注册和登录。

用户登录后，可以在**用户界面**中修改自己的个人信息，如昵称、头像、性别、年龄等或自行注销账号。用户也可以在用户界面中查看自己的历史点歌记录和收藏歌单。

用户在**搜索界面**中有两种搜索方式：

①用户可以通过歌名、歌手等歌曲条件进行搜索。

②用户可以输入模糊的查找语句智能查找，如“我想听一首轻松的英文歌”、“给我来一首周杰伦的快歌”等，然后系统会根据用户输入和音乐库中的数据，为用户列举出符合要求的歌曲。

用户可以在**播放界面**中控制音乐的播放、暂停、上一首、下一首等操作。用户也可以在此界面中查看当前播放的音乐的详细信息，如歌名、歌手、专辑、时长、歌词等。

用户可以在**推荐界面**中查看和播放智能推荐的音乐，智能推荐是根据用户的收藏歌单和历史点歌记录，为用户推荐合适的音乐。推荐算法会考虑用户的喜好、场景、心情等因素，动态调整推荐策略。用户可以根据推荐结果来选择歌曲或全部播放。用户也可以对推荐结果进行反馈，如收藏、评价等。

操作员可以在系统后台界面可以对音乐库进行增删改操作以定期更新音乐库，可以维护用户信息，同时对于一些违禁的用户冻结其使用本系统的资格。

## 3.2 功能建模—DFD

根据以上需求获取的业务描述，画出智能点歌系统的分层数据流图如下：

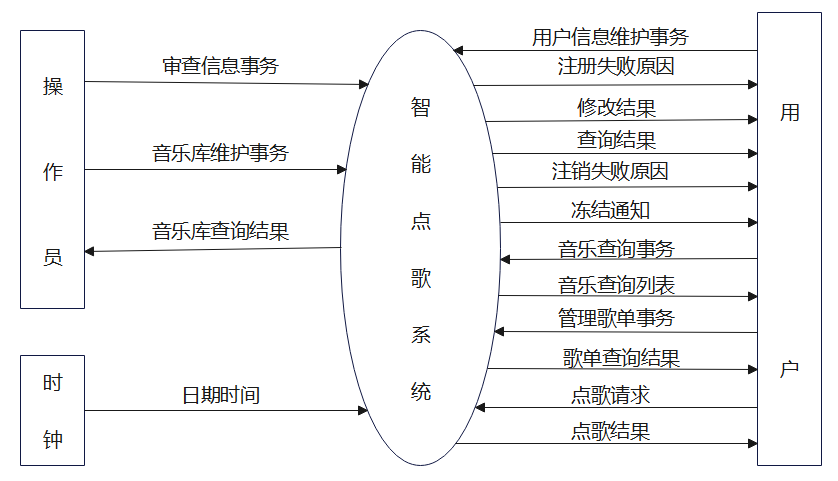


图3-1 顶层数据流图

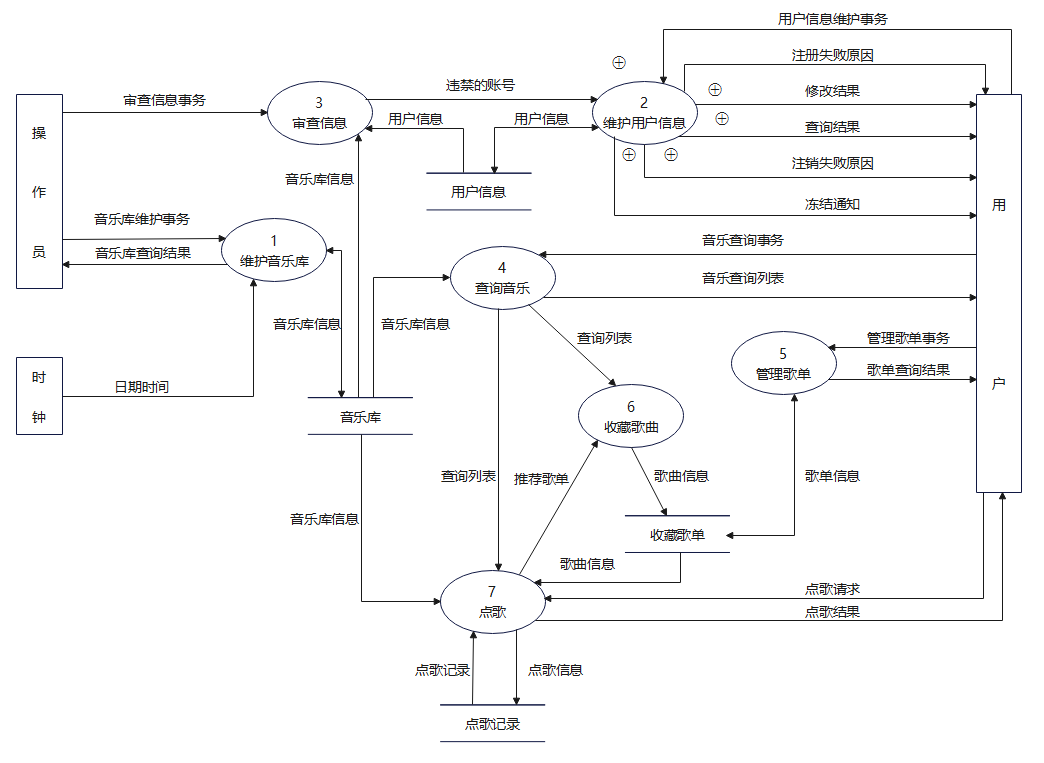


图3-2 一层数据流图

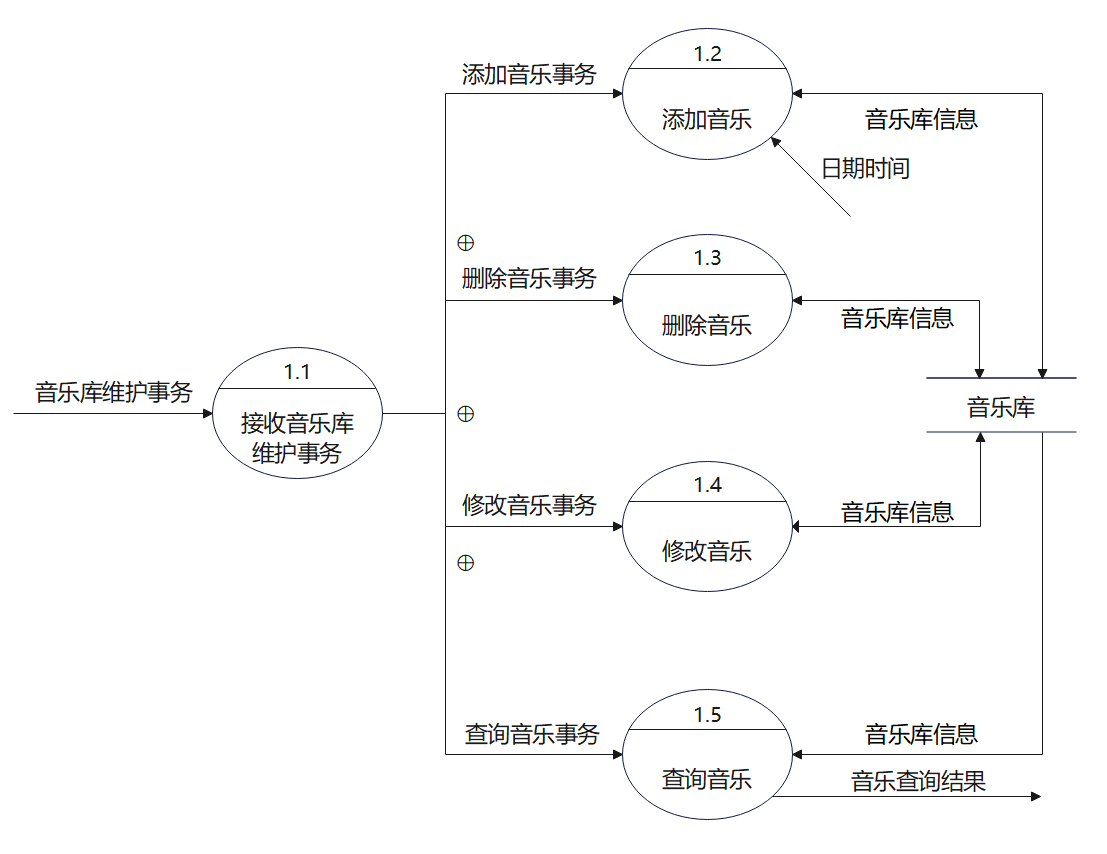


图3-3 维护音乐库数据流图

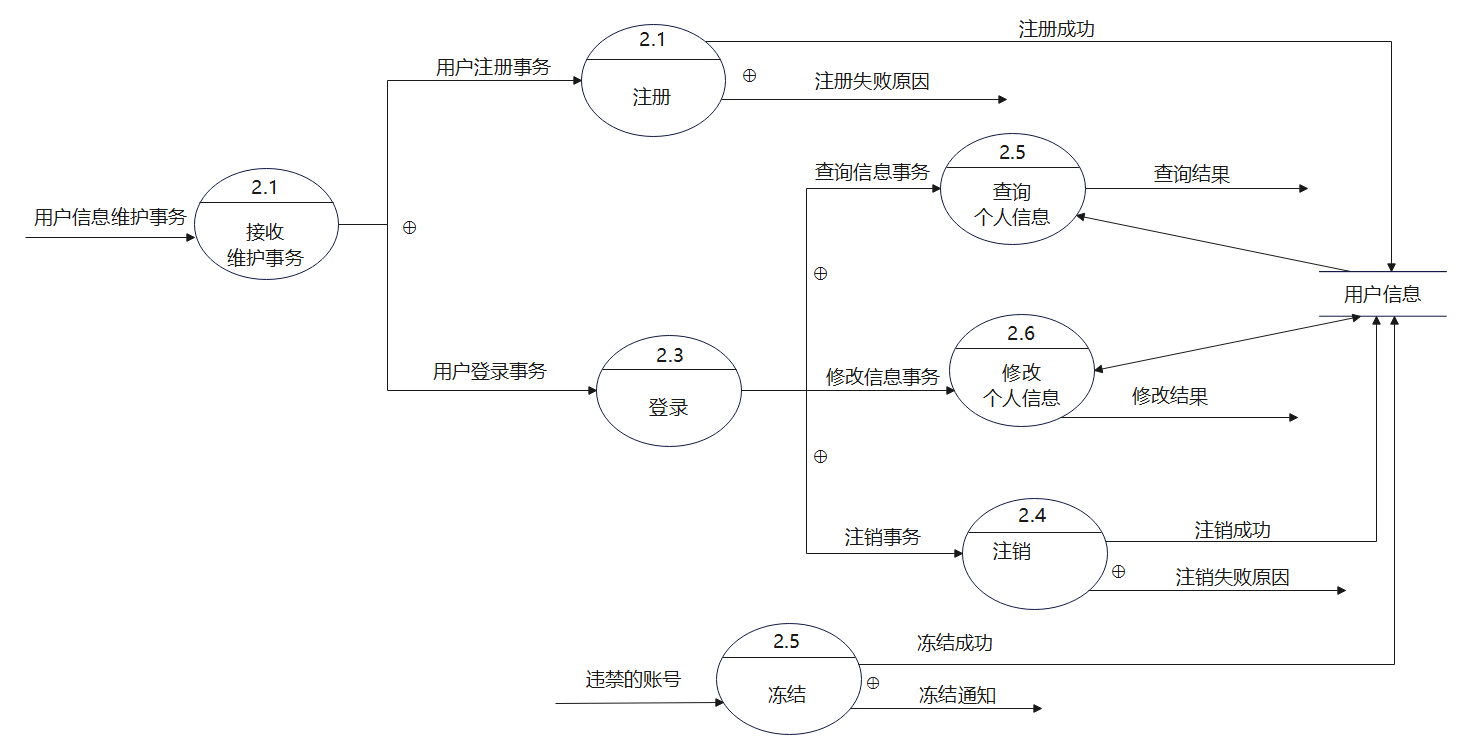


图3-4 维护用户信息数据流图

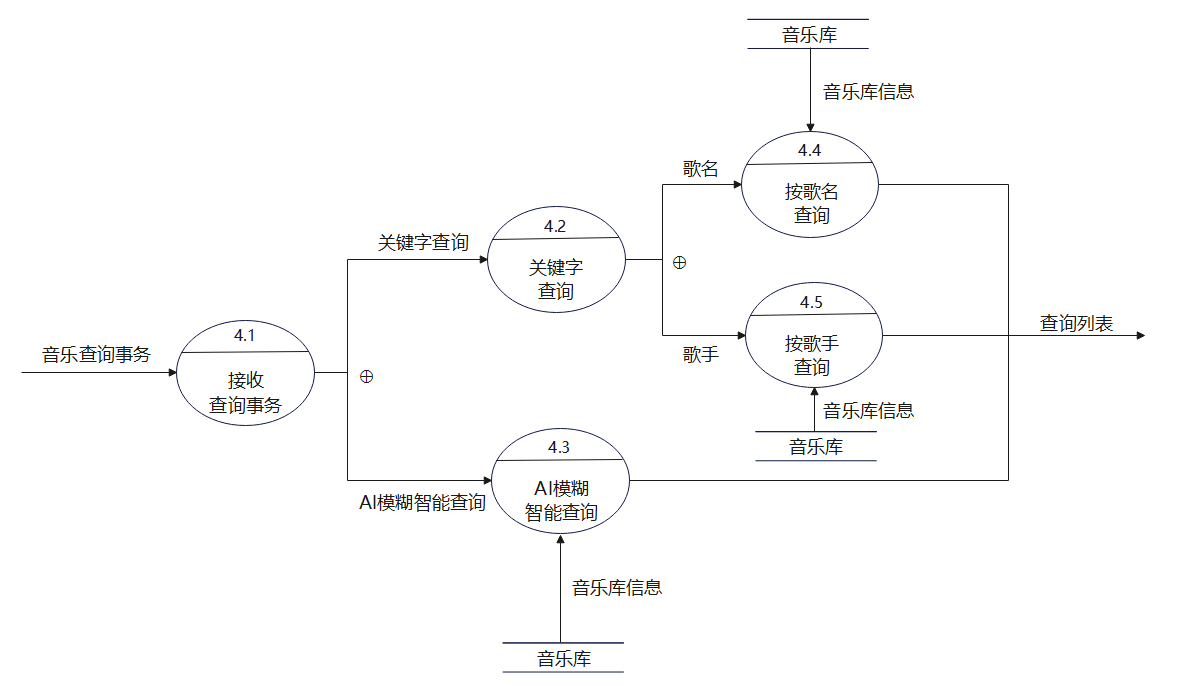


图3-5 查询音乐数据流图

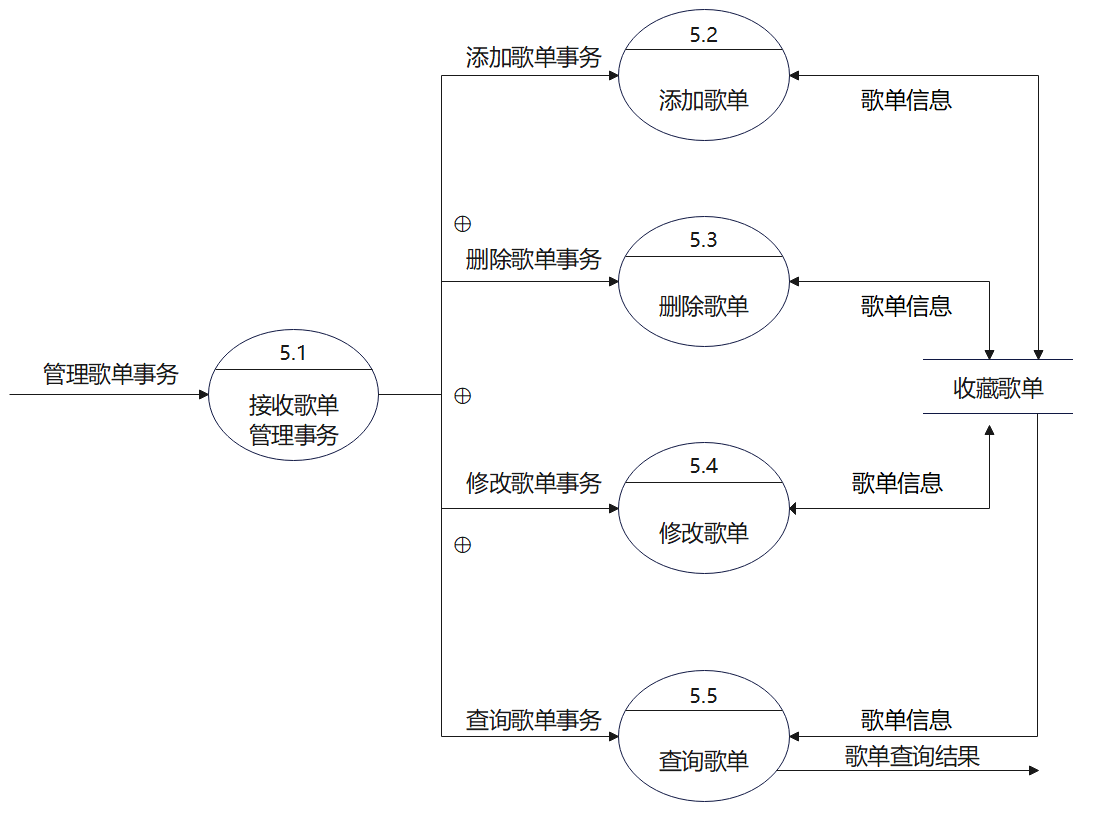


图3-6 管理歌单数据流图

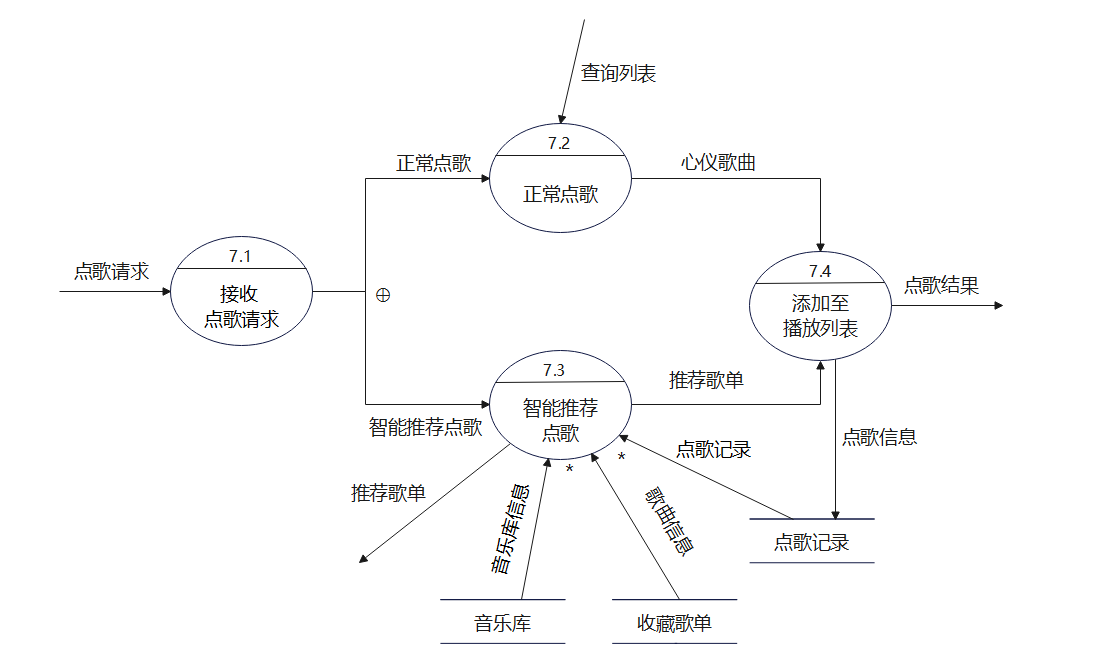


图3-7 点歌数据流图

## 3.3 数据建模—E-R图

通过前面智能点歌系统的业务流程及功能建模，可以分析出系统的数据信息包括：用户信息、音乐库信息、历史点歌记录、收藏歌单信息，其中音乐库包含所有的歌曲，只是歌曲的容器，所以数据存储中只需存储歌曲的信息。通过功能建模的二层数据流图中点歌和收藏歌曲的详细过程可以分析出历史点歌记录是用户点播歌曲是产生的，而收藏歌单是是用户收藏歌曲时产生的。所以，系统E-R图中共包含2个实体：用户和歌曲。智能点歌系统的E-R图如下：

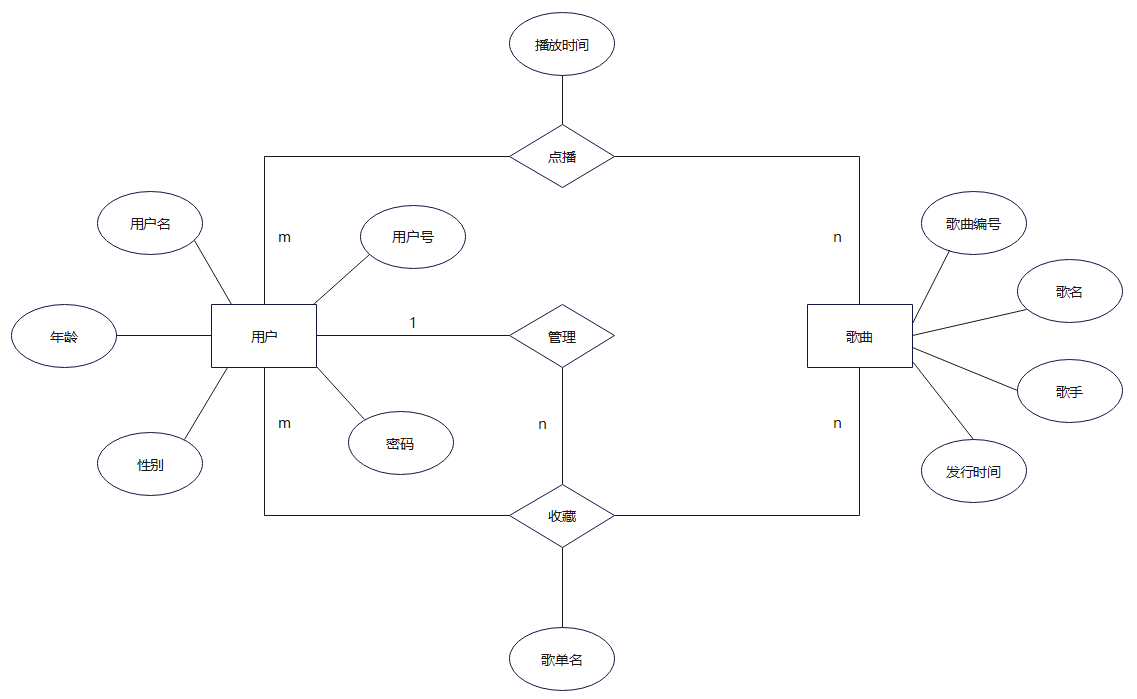


图3-8 E-R图

## 3.4 行为模型—状态转换图

通过前面智能点歌系统的业务流程和功能建模，主要画出系统的点歌状态图：

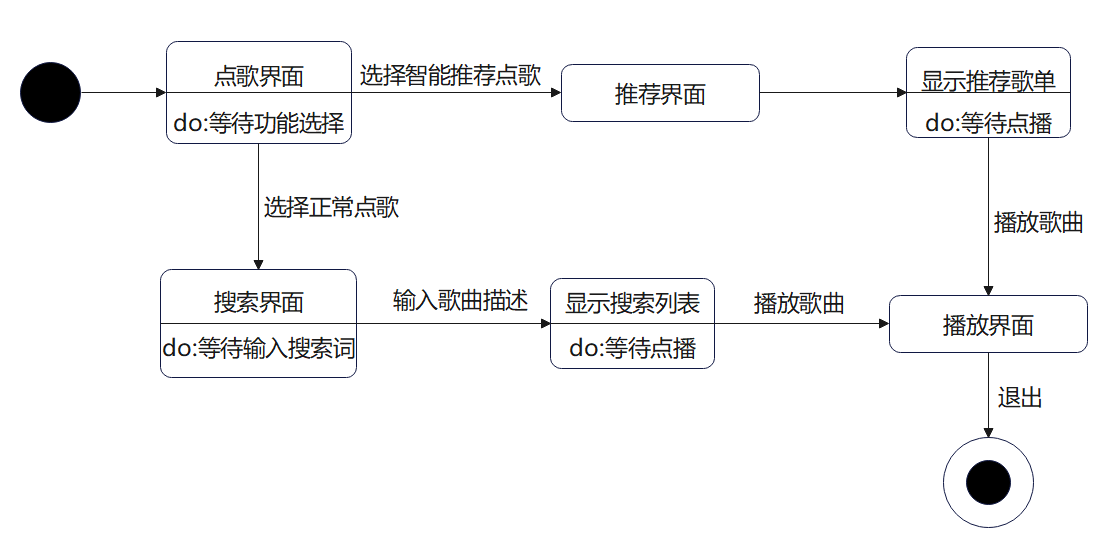


图3-9 点歌状态图

## 3.5 数据字典

智能点歌系统的数据字典可以对其二层数据流图中的所有基本加工、数据流、数据存储、数据源点和数据汇点进行词条方式的描述。为节省篇幅，以下只给出数据源点或数据汇点“用户”的词条、数据源点或数据汇点“操作员”的词条、二层点歌数据流图中的所有基本加工、数据流、数据存储及构成各个数据结构的数据元素的词条描述。

1. **数据源点或数据汇点“用户”的词条**

(1)名称：用户

(2)简述：通过智能点歌系统查询音乐、管理歌单及点歌

(3)相关数据流：

用户输入到系统的数据流：DF2.1（用户信息维护事务）、DF4.1（音乐查询事务）、DF5.1（管理歌单事务）、DF7.1（点歌请求）

系统输出到用户的数据流：DF2.2（注册失败原因）、DF2.3（修改结果）、DF2.4（查询结果）、DF2.5（注销失败原因）、DF2.6（冻结通知）、DF4.2（音乐查询列表）、DF5.2（歌单查询结果）、DF7.2（点歌结果）

1. **数据源点或数据汇点“操作员”的词条**

(1)名称：操作员

(2)简述：通过智能点歌系统对用户和音乐库进行维护

相关数据流：

操作员输入到系统的数据流：DF3.1（审查信息事务）、DF1.1（音乐库维护事务）

系统输出到操作员的数据流：DF1.2（音乐库查询结果）

1. **数据流“点歌请求”的词条**
2. 数据流名：点歌请求；
3. 编号：DF7.1；
4. 简述：需要点歌时发送的请求；
5. 来源：外部实体“用户”；
6. 去向：P7.1（加工“接收点歌请求”）；
7. 数据流“点歌请求”的数据结构描述，即数据元素词条如下。

点歌请求

1. 类型：布尔型，长度为2字节；
2. 取值范围：[“是”，“否”]；
3. 相关的数据元素及数据结构：DF7.2（正常点歌数据流）、DF7.3（智能推荐点歌数据流）、DF7.12（点歌结果数据流）
4. **数据流“正常点歌”的词条**
5. 数据流名：正常点歌；
6. 编号：DF7.2；
7. 简述：点歌方式为正常点歌；
8. 来源：P7.1（加工“接收点歌请求”）；
9. 去向：P7.2（加工“正常点歌”）；
10. 数据流“正常点歌”的数据结构描述，即数据元素词条如下。

正常点歌

1. 相关的数据元素及数据结构： DF7.5（心仪歌曲数据流）、DF7.11（点歌信息数据流）、F3（点歌记录存储文件）、DF7.8（点歌记录数据流）；
2. **数据流“智能推荐点歌”的词条**
3. 数据流名：智能推荐点歌；
4. 编号：DF7.3；
5. 简述：点歌方式为智能推荐点歌；
6. 来源：P7.1（加工“接收点歌请求”）；
7. 去向：P7.3（加工“智能推荐点歌”）；
8. 数据流“智能推荐点歌”的数据结构描述，即数据元素词条如下。

智能推荐点歌

1. 相关的数据元素及数据结构：DF7.9（推荐歌单数据流）、DF7.11（点歌信息数据流）、F3（点歌记录存储文件）、DF7.8（点歌记录数据流）；
2. **数据流“查询列表”的词条**
3. 数据流名：查询列表；
4. 编号：DF7.4；
5. 简述：查询列表的歌曲信息；
6. 来源：P4.4（加工“按歌名查询”）、P4.5（加工“按歌手查询”）、P4.3（加工“AI模糊智能查询”）；
7. 去向：P7.2（加工“正常点歌”）；
8. 数据流“查询列表”的数据结构描述，即数据元素词条如下。

查询列表=歌名+歌手+发行时间

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 发行时间

·类型：DATE，长度为18字节

·取值范围：“2000-01-01”…“2100-12-31”

1. **数据流“推荐歌单”的词条**
2. 数据流名：推荐歌单；
3. 编号： D7.5、DF7.9；
4. 简述：由智能推荐点歌处理产生的推荐歌单信息；
5. 来源：P7.3（加工“智能推荐点歌”）；
6. 去向：P6（加工“收藏点歌”）、P7.4（加工“添加至播哦放列表”）；
7. 数据流“推荐歌单”的数据结构描述，即数据元素词条如下。

推荐歌单=歌名+歌手+发行时间

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 发行时间

·类型：DATE，长度为18字节

·取值范围：“2000-01-01”…“2100-12-31”

1. **数据流“音乐库信息”的词条**
2. 数据流名：音乐库信息；
3. 编号：DF7.6；
4. 简述：音乐库存储的歌曲信息；
5. 来源：F1（存储文件“音乐库”）；
6. 去向：P7.3（加工“智能推荐点歌”）；
7. 数据流“音乐库信息”的数据结构描述，即数据元素词条如下。

音乐库信息=歌名+歌手+发行时间

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 发行时间

·类型：DATE，长度为18字节

·取值范围：“2000-01-01”…“2100-12-31”

1. **数据流“收藏歌单”的词条**
2. 数据流名：收藏歌单；
3. 编号：DF7.7；
4. 简述：收藏歌单的歌曲信息；
5. 来源：F2（存储文件“收藏歌单”）；
6. 去向：P7.3（加工“智能推荐点歌”）；
7. 数据流“收藏歌单”的数据结构描述，即数据元素词条如下。

收藏歌单=歌名+歌手+发行时间

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 发行时间

·类型：DATE，长度为18字节

·取值范围：“2000-01-01”…“2100-12-31”

1. **数据流“点歌记录”的词条**
2. 数据流名：点歌记录；
3. 编号：DF7.8；
4. 简述：点歌记录的歌曲信息
5. 来源：F3（存储文件“点歌记录”）
6. 去向：P7.3（加工“智能推荐点歌”）；
7. 数据流“点歌记录”的数据结构描述，即数据元素词条如下。

点歌记录=歌名+歌手

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. **数据流“心仪歌曲”的词条**
2. 数据流名：心仪歌曲；
3. 编号：DF7.10；
4. 简述：由正常点歌处理产生的心仪歌曲信息；
5. 来源：P7.2（加工“正常点歌”）；
6. 去向：P7.4（加工“添加至播放列表”）；
7. 数据流“心仪歌曲”的数据结构描述，即数据元素词条如下。

心仪歌曲=歌名+歌手+发行时间

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 发行时间

·类型：DATE，长度为18字节

·取值范围：“2000-01-01”…“2100-12-31”

1. **数据流“点歌信息”的词条**
2. 数据流名：“点歌信息”；
3. 编号：DF7.11；
4. 简述：由播放列表播放得到的点歌信息；
5. 来源：P7.4（加工“添加至播放列表”）；
6. 去向：F3（存储文件“点歌记录”）；
7. 数据流“点歌信息”的数据结构描述，即数据元素词条如下。

点歌信息=歌名+歌手

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. **数据流“点歌结果”的词条**
2. 数据流名：点歌结果
3. 编号：DF7.12；
4. 简述：点歌结束后返回给用户的点歌结果信息；
5. 来源：P7.4（加工“添加至播放列表”）；
6. 去向：外部实体“用户”；
7. 数据流“点歌结果”的数据结构描述，即数据元素词条如下。

点歌结果=歌名+歌手

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）

1. **存储文件“音乐库”词条**
2. 文件名：音乐库
3. 编号：F1
4. 简述：操作员维护音乐库时存储的歌曲信息
5. 组成：歌曲编号+歌名+歌手+发行时间
6. 输入：P1.2（加工“添加音乐”）、P1.3（加工“删除音乐”）、P1.4（加工“修改音乐”）
7. 输出：P1.5（加工“查询音乐”）、P3（加工“审查信息”）、P4.3（加工“AI智能模糊查询”）、P4.4（加工“按歌名查询”）、P4.5（加工“按歌手查询”）、P7.3（加工“智能推荐点歌”）
8. 存储方式：关键码方式存取或直接存取

音乐库存储文件的数据结构描述，即数据元素词条如下。

音乐库=歌曲编号+歌名+歌手+发行时间

1. 歌曲编号

·类型：字符型，长度为10字节

·取值范围：“0000000001”…“9999999999”

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）、DF4.9（音乐查询列表数据流）、DF5.3（歌单查询结果数据流）、DF7.12（点歌结果数据流）

1. 歌名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F1（音乐库存储文件）

1. 歌手

·类型：字符型，长度为10字节

·相关的数据元素及数据结构：F1（音乐库存储文件）

1. 发行时间=年+月+日

·类型：日期型

·取值范围：

·年=“0000”…“9999”

·月=“01”…“12”

·日=“01”…“31”

·相关的数据元素及数据结构：F1（音乐库存储文件）

1. **存储文件“收藏歌单”词条**
2. 文件名：收藏歌单
3. 编号：F2
4. 简述：用户收藏歌曲时生成的收藏歌单
5. 组成：用户号+歌曲编号+歌单名
6. 输入：P5（加工“管理歌单”）、P6（加工“收藏歌曲”）
7. 输出：（歌曲信息数据流）
8. 存储方式：关键码方式存取或直接存取

收藏歌单存储文件的数据结构描述，即数据元素词条如下。

收藏歌单=用户号+歌曲编号+歌单名

1. 用户号

·类型：字符型，长度为10字节

·取值范围：“0000000001”…“9999999999”

·相关的数据元素及数据结构：F2（收藏歌单存储文件）、F3（点歌记录存储文件）、F4（用户信息存储文件）

1. 歌曲编号

·类型：字符型，长度为10字节

·取值范围：“0000000001”…“9999999999”

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）、DF4.9（音乐查询列表数据流）、DF5.3（歌单查询结果数据流）、DF7.12（点歌结果数据流）

1. 歌单名

·类型：字符型，长度为20字节

·相关的数据元素及数据结构：F2（收藏歌单存储文件）

1. **存储文件“点歌记录”词条**
2. 文件名：点歌记录
3. 编号：F3
4. 简述：用户点播歌曲时生成的点歌记录
5. 组成：用户号+歌曲编号+播放时间
6. 输入：P7.4（加工“添加至播放列表”）
7. 输出：P7.3（加工“智能推荐点歌”）
8. 存储方式：关键码方式存取或直接存取

点歌记录存储文件的数据结构描述，即数据元素词条如下。

点歌记录=用户号+歌曲编号+播放时间

1. 用户号

·类型：字符型，长度为10字节

·取值范围：“0000000001”…“9999999999”

·相关的数据元素及数据结构：F2（收藏歌单存储文件）、F3（点歌记录存储文件）、F4（用户信息存储文件）

1. 歌曲编号

·类型：字符型，长度为10字节

·取值范围：“0000000001”…“9999999999”

·相关的数据元素及数据结构：F1（音乐库存储文件）、F2（收藏歌单存储文件）、F3（点歌记录存储文件）、DF4.9（音乐查询列表数据流）、DF5.3（歌单查询结果数据流）、DF7.12（点歌结果数据流）

1. 播放时间=年+月+日+时+分

·类型：日期型时间

·取值范围：

年=“0000”…“9999”

月=“01”…“12”

日=“01”…“31”

时=“00”…“23”

分=“00”…“59”

·相关的数据元素及数据结构：F3（点歌记录存储文件）

1. **加工“接收点歌请求”词条**
2. 加工名：接收点歌请求
3. 编号：P7.1
4. 简述：用户输入是否点歌的请求后，系统接收请求并给用户提供两种点歌方式
5. 输入：DF7.1（“点歌请求”数据流）
6. 输出：DF(“正常点歌”数据流）、DF(“智能推荐点歌”数据流）
7. 加工逻辑：用户输入是否点歌的请求后，系统接收请求并给用户提供两种点歌方式
8. 加工“正常点歌”词条
9. 加工名：正常点歌
10. 编号：P7.2
11. 简述：用户通过搜索查询出来的歌曲列表进行歌曲选择点歌
12. 输入：DF（“查询列表”数据流）、DF（“正常点歌”数据流）
13. 输出：DF(“心仪歌曲”数据流）
14. 加工逻辑：用户通过输入的搜索词在音乐库中找到歌曲列表，然后在其中自主选择歌曲点播
15. 加工“智能推荐点歌”词条
16. 加工名：智能推荐点歌
17. 编号：P7.3
18. 简述：系统通过用户的收藏歌单和点歌记录在音乐库中搜寻，把推荐歌单返回给用户进行歌曲选择点歌
19. 输入：DF（“收藏歌单”数据流）、DF（“点歌记录”数据流）、DF（“音乐库”数据流）
20. 输出：DF(“推荐歌单”数据流）
21. 加工逻辑：系统通过对用户的收藏歌单和点歌记录进行情感词提取后，在音乐库中搜寻出和情感词类型相同的歌曲，把推荐歌单返回给用户进行歌曲选择点歌
22. 加工“添加至播放列表”词条
23. 加工名：添加至播放列表
24. 编号：P7.4
25. 简述：用户将歌曲添加到播放列表中等待播放
26. 输入：DF(“心仪歌曲”数据流）、DF(“推荐歌单”数据流）
27. 输出：DF(“点歌信息”数据流）、DF(“点歌结果”数据流）
28. 加工逻辑：用户在查询列表或推荐歌单中根据自己的喜好要求选择歌曲添加至播放列表等待顺序播放

# 4 总体设计

## 4.1 体系结构设计

由上一章的一层数据流图和二层数据流图综合分析，可以看出此系统包含信息维护、音乐查询、歌曲收藏、歌单管理及点歌的事务处理，所以具有事务型特征。

（1）智能点歌系统的上层结构图设计如下：

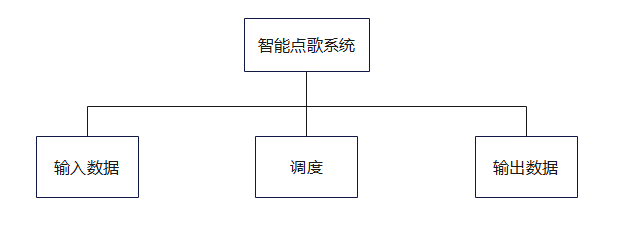


图4-1 智能点歌系统的上层结构图

（2）“输入数据”模块的分解如下：

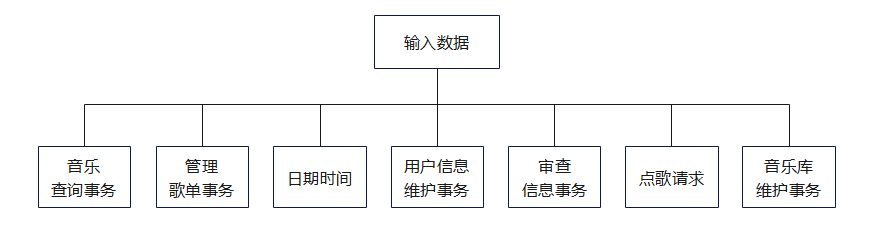


图4-2 输入模块的分解

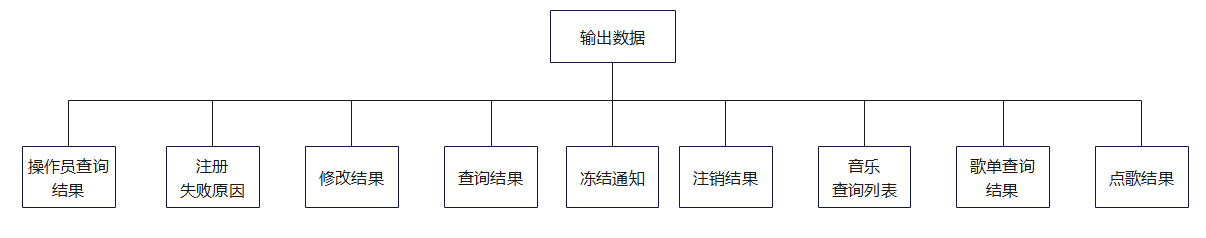
（3）“输入数据”模块的分解如下：

图4-3 输出模块的分解

（4）“调度”模块的的下层事务模块设计如下。其中的审查信息和维护音乐库可以不作为本体系结构的设计内容，直接在软件后台管理员入库实现即可。

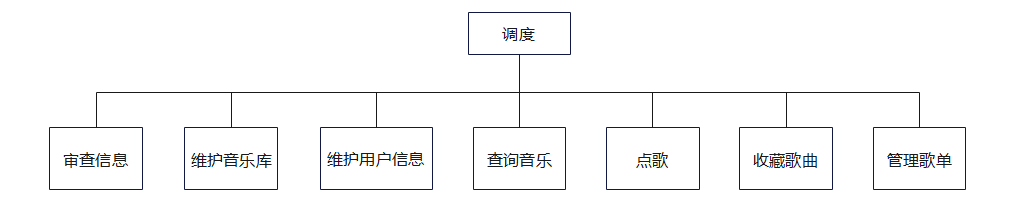


图4-4 调度模块的下层事务模块

（5）设计“事务”层的下层“操作”模块及“细节”模块如下：

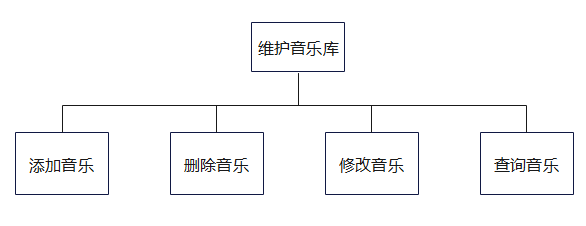


图4-5 维护音乐库模块的分解

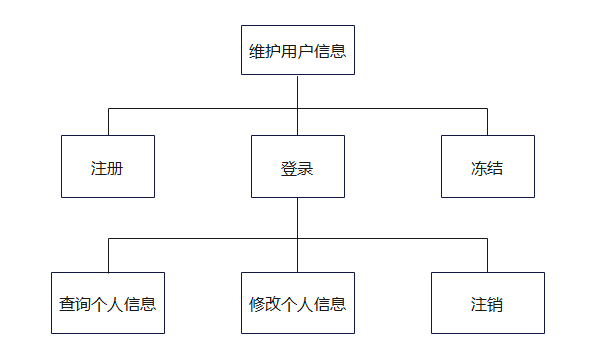


图4-6 维护用户信息模块的分解

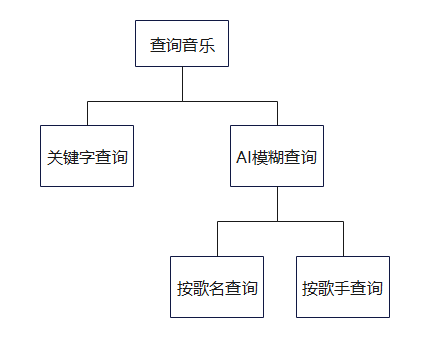


图4-7 查询音乐模块的分解

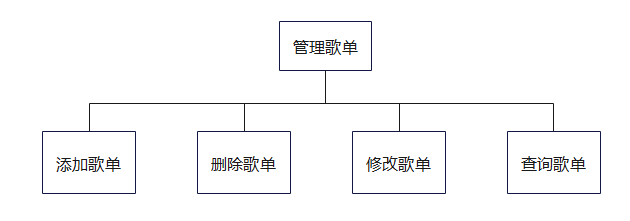


图4-8 管理歌单模块的分解

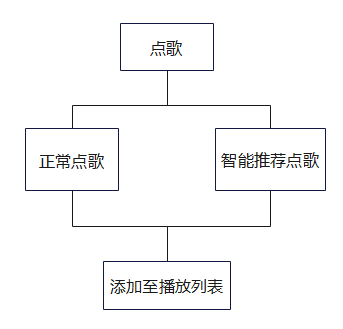


图4-9 点歌模块的分解

（6）按体系结构的启发式设计原则进行结构图整合、精化和调整，得到最终智能点歌系统的体系结构图，如图4-10所示。

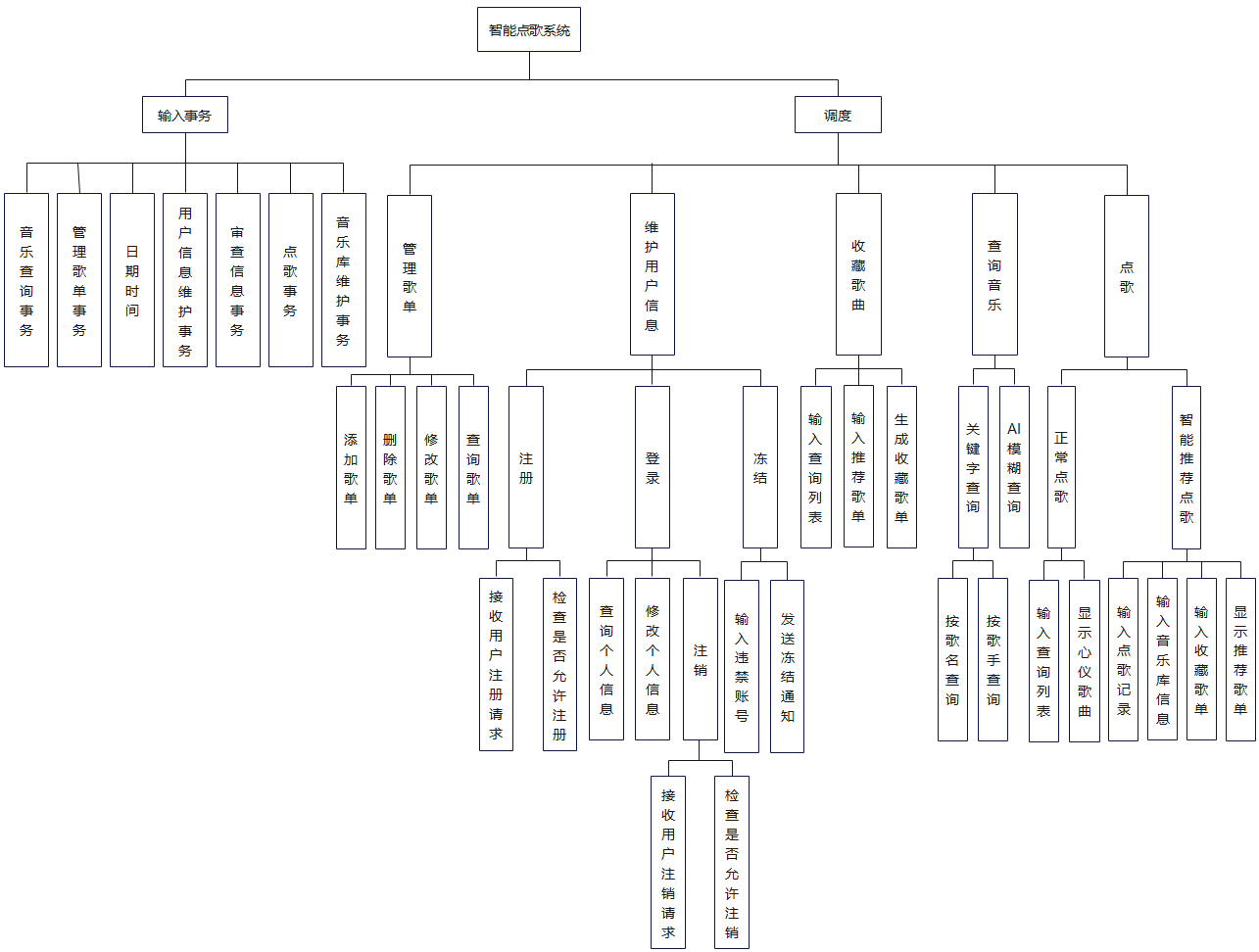


图4-10 智能点歌系统的体系结构

## 4.2 接口设计

接口设计主要包括3方面:软件与人(用户)之间的交互界面设计、软件与硬件及其他软件系统之间的接口设计、模块之间的接口设计。

对智能点歌系统接口设计分析如下。

（1）智能点歌系统是面向大众的,因此界面需布局简洁，操作舒适、简单且步骤清晰；除此还需让软件具有一定的品位, 界面的风格和色彩应该符合用户的审美和喜好。

（2）软件与硬件接口主要是音乐播放接口。播放接口是智能点歌系统的核心接口之一，它需要提供歌曲的播放控制功能，包括播放、暂停、停止、快进、快退等操作。播放接口需要与音频播放器、音频解码器等组件进行交互，以保证音频的正常播放。除此软件应该支持多种音乐数据源，如本地文件、在线资源、云端存储等，并提供统一的接口和格式，以方便软件获取和处理音乐数据。

（3）模块之间的接口可根据数据流图中各加工之间的联系来确定,根据系统的功能和需求，将系统划分为不同的模块，每个模块负责特定的功能。模块之间的接口应满足以下要求：

①模块之间的接口应该尽量简化和标准化，避免过多或过复杂的参数和返回值，以提高模块之间的通信效率和可读性。

②模块之间的接口应该尽量解耦和抽象化，避免模块之间的直接依赖和耦合，以提高模块之间的独立性和可复用性。

③模块之间的接口应该尽量稳定和兼容，避免模块之间的接口变动导致其他模块出现错误或异常，以提高模块之间的可靠性和可维护性。

## 4.3 数据设计

（1）**实体的映射。**E-R图中有2个实体，映射为2个表，表模式如下：

用户信息（用户号、用户名、性别、年龄、密码）

歌曲信息（歌曲编号、歌名、歌手、发行时间）

（2）**联系的映射。**

用户实体和收藏歌单联系之间为一对多联系，可以在“多”端收藏歌单联系中加入“一”端用户实体的主键作为外键，形成两个表的联系。

用户实体和歌曲实体之间有两层联系且均为多对多联系，必须产生两个新的关系模式形成两个表的联系，该关系模式设计为：

点歌记录（用户号、歌曲编号、播放时间）

收藏歌单（用户号、歌曲编号、歌单名）

（3）**映射汇总。**

①用户信息（用户号、用户名、性别、年龄、密码）

其中，用户号为主键。

②歌曲信息（歌曲编号、歌名、歌手、发行时间）

其中，歌曲编号为主键。

③点歌记录（用户号、歌曲编号、播放时间）

其中，用户号、歌曲编号和播放时间的组合为主键；用户号为外键，与用户信息表的主键相关联；歌曲编号为外键，与歌曲信息表相关联。

根据程序性能的需求，可以按播放时间将点歌记录表横切为历史点歌记录表和近期点歌记录表。

④收藏歌单（用户号、歌曲编号、歌单名）

其中，用户号、歌曲编号、歌单名的组合为主键；用户号为外键，与用户信息表的主键相关联；歌曲编号为外键，与歌曲信息表相关联。

根据程序性能的需求，可以按歌单名将收藏歌单表横切为不同歌单。

（4）**设计视图**。根据系统功能需求，需设计两个视图：

①用户需要查询的点歌记录详细信息：

点歌记录详细信息（用户号、歌名、歌手、发行时间、播放时间）

②用户需要查询的收藏歌单详细信息：

收藏歌单详细信息（用户号、歌单名、歌名、歌手、发行时间）

# 5 详细设计

## 5.1 算法设计

智能点歌系统的所有模块中，其中关键字查询、模糊查询、智能推荐歌曲模块相对复杂，其流程图如下：

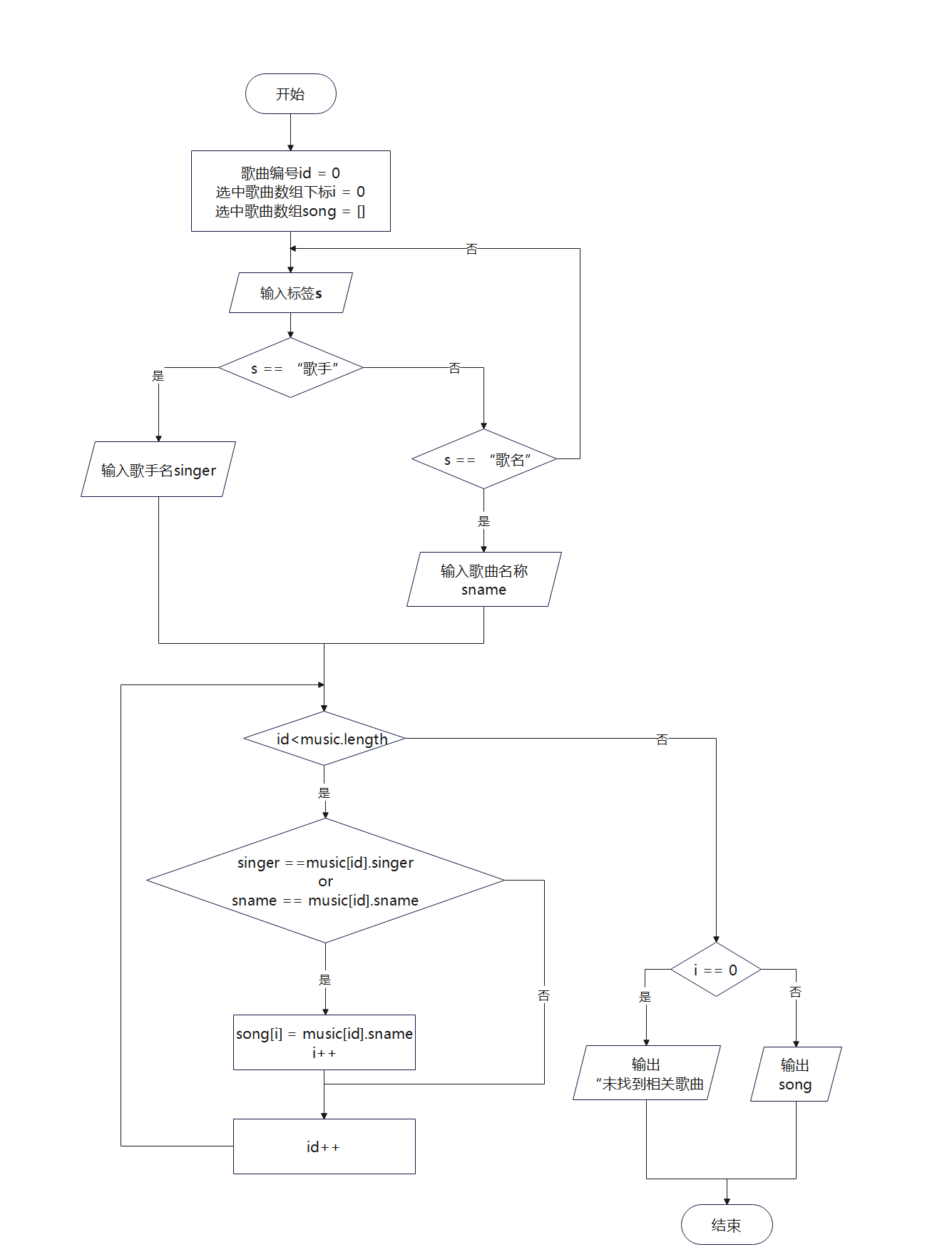


图5-1 关键字查询模块的流程图

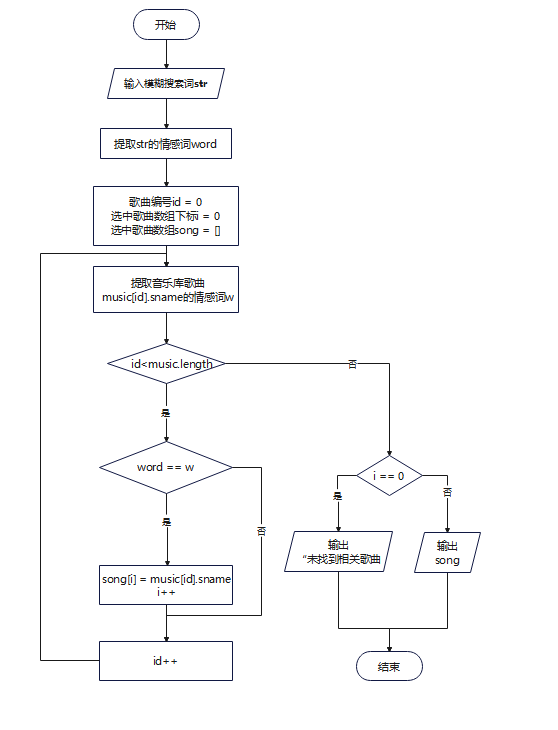


图5-2 模糊查询模块的流程图

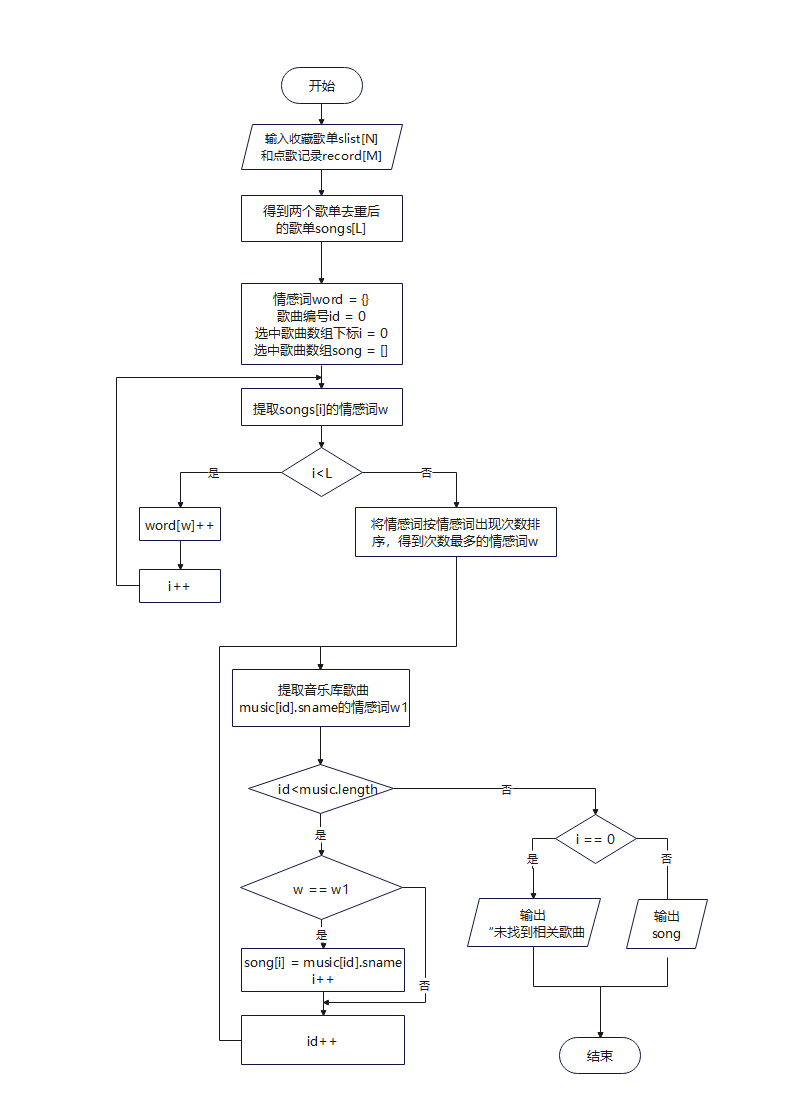


图5-3 智能推荐歌曲模块的流程图

## 5.2 数据结构细节和数据操作的设计

（1）数据库数据文件和日志文件设计

智能点歌系统的数据库根据音乐库中歌曲的数量、注册用户的数量、每个用户的平均点歌数量及点播歌曲次数来确定；根据估算每年平均增长的歌曲、注册用户及点歌次数来确定数据库数据文件的最大初始大小和增长方式；根据数据库的操作频率和平均数据操作数量来确定日志文件的最大初始大小和增长方式。

（2）设计基本表的具体结构

智能点歌系统的数据库中一共设计了5个表，各表的表结构如下，其中包含了主键、外键、索引键、取值范围、默认值和是否允许为空的完整性约束。

表5-1 用户信息表(T\_User)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字 段 | 含 义 | 数据类型 | 键 值 | 取值范围 | 允许空否 |
| UNO | 用户号 | char(10) | 主键 |  | 否 |
| UName | 用户名 | varchar(20) |  |  |  |
| Sex | 性别 | char(1) |  | 男、女 |  |
| Age | 年龄 | int(4) |  |  |  |
| UPWD | 密码 | varchar(20) |  |  |  |

表5-2 歌曲信息表(T\_Song)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 字 段 | 含 义 | 数据类型 | 键 值 | 取值范围 | 允许空否 |
| SNO | 歌曲编号 | char(10) | 主键 |  | 否 |
| SName | 歌名 | varchar(20) |  |  |  |
| Singer | 歌手 | varchar(20) |  |  |  |
| IssueDate | 发行时间 | date |  |  |  |

表5-3 点歌记录表(T\_OrderingRecord)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字 段 | 含 义 | 数据类型 | 键 值 | | 取值范围 | 允许空否 |
| UNO | 用户号 | char(10) | 主键 | 外键 |  | 否 |
| SNO | 歌曲编号 | varchar(10) | 外键 |  | 否 |
| OTime | 播放时间 | datetime |  |  |  |  |

表5-4 收藏歌单表（V\_CollectionList)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 字 段 | 含 义 | 数据类型 | 键 值 | | 取值范围 | 允许空否 |
| UNO | 用户号 | char(10) | 主键 | 外键 |  | 否 |
| SNO | 歌曲编号 | char(10) | 外键 |  | 否 |
| LName | 歌单名 | varchar(20) |  |  |  |  |

（3）设计视图的具体结构

①用户需要查询的点歌记录详细信息视图如下表所示。

表5-5 点歌记录详细信息视图（V\_OrderingRecord)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字 段 | 含 义 | 数据类型 | 数 据 源 | 索 引 |
| UNO | 用户号 | char(10) | T\_OrderingRecord.UNO |  |
| UName | 用户名 | varchar(20) | T\_User.UName |  |
| SName | 歌名 | varchar(20) | T\_Song.SName | 是 |
| Singer | 歌手 | varchar(20) | T\_Song.Singer | 是 |
| IssueDate | 发行时间 | date | T\_Song.IssueDate | 是 |
| OTime | 播放时间 | datetime | T\_OrderingRecord.OTime | 是 |

1. 用户需要查询的收藏歌单详细信息视图如下表所示。

表5-6 收藏歌单详细信息视图（V\_CollectionList)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字 段 | 含 义 | 数据类型 | 数 据 源 | 索 引 |
| UNO | 用户号 | char(10) | T\_CollectionList.UNO |  |
| LName | 歌单名 | varchar(20) | T\_CollectionList.LName | 是 |
| UName | 用户名 | varchar(20) | T\_User.Uname |  |
| SName | 歌名 | varchar(20) | T\_Song.SName |  |
| Singer | 歌手 | varchar(20) | T\_Song.Singer |  |

（4）设计存储过程和触发器

为了提高软件的工作效率及保护数据库的安全性和完整性，需要设计存储过程和触发器。

用户点歌的过程可以通过执行一系列的存储过程或触发器来完成。用户可以通过精确查询和模糊查询两种方式来查询歌曲，精确查询又分为按歌名查询和按歌手查询。因此建立三个存储过程P\_SelectSong、P\_SeclectSinger和P\_SeclectFuzzy，分别以歌名、歌手、自然语言为参数查询T\_Song信息表。用户可以收藏查询到的歌曲，系统生成新的歌单或向原有的歌单中插入一行信息，生成新的歌单用存储过程P\_InsertCList来完成，插入收藏歌曲用存储过程P\_InsertCollection来实现。用户也可以播放查询到的歌曲，系统向点歌记录中插入一行信息，此插入操作用一个存储过程P\_InsertRecord来实现，也可以将歌曲添加至播放列表，系统向播放列表中插入一行信息，此插入操作用一个存储过程P\_InsertOrder来实现，若无正在播放的播放列表，则系统生成一个新的临时的播放列表，此操作利用一个触发器来实现。播放列表播放了一首歌曲时系统也向点歌记录中插入一行信息，此插入操作一样用一个存储过程P\_InsertRecord来实现。 用户点歌时可以选择正常点歌和智能推荐点歌，正常点歌时系统调用查询歌曲模块的一系列存储过程，因此不需建立新的存储过程；智能推荐点歌时系统根据用户的点歌记录和收藏歌单从音乐库中随机选择30首用户可能喜欢的歌曲，此操作用一个存储过程P\_RecommendSong来完成。用户可以将推荐结果添加至播放列表或播放整个推荐列表，实现同上文中的查询。

另外，用户可以管理收藏歌单、维护个人信息，分别通过存储过程P\_ManageCList和P\_MaintainUser来完成。

（5）数据库的安全性设计

①设置用户权限。系统操作员具有最高权限，可以冻结违禁用户和维护音乐库信息。程序员要实现操作员、用户的数据操作功能需具有的权限如下。

表5-7 完成系统操作员的功能数据操作需具有的权限

|  |  |  |
| --- | --- | --- |
| 对象类型 | 对象名称 | 权限 |
| 基本表 | T\_User | Select、Delete |
| T\_Song | Select、Insert、Update、Delete |

表5-8 完成用户的功能数据操作需具有的权限

|  |  |  |
| --- | --- | --- |
| 对象类型 | 对象名称 | 权限 |
| 基本表 | T\_User | Select、Insert、Update、Delete |
| 视图 | V\_OrderingRecord | Select |
| V\_CollectionList | Select |
| 存储过程 | P\_SelectSong | Execute |
| P\_SeclectSinger |
| P\_SeclectFuzzy |
| P\_InsertCList |
| P\_InsertCollection |
| P\_InsertRecord |
| P\_InsertOrder |
| P\_RecommendSong |

②设置数据加密。用户信息表（T\_User）中的密码是敏感信息，所以应该设置为加密保存。

③设置数据库的备份和还原策略。因为智能点歌系统的数据库每天的数据都有一些变化，所以可以在隔天系统应用时间最少的时间段（一般为3点—5点）进行一次差异数据库备份，每7天进行一次完整数据库备份，只保留近3次完整备份和相应的差异备份。

# 6 编码与测试

## 6.1 系统编码

本团队欲设计一个智能点歌系统Web网站，根据用户的需要和团队程序员的技能等决定使用Java和Python语言。

## 6.2系统测试

6.2.1测试过程

在智能点歌系统的所有功能模块中，按照上一节的关键词查询、模糊查询、智能推荐点歌的控制流图和环路复杂度的计算结果进行基本路径测试，其他模块就不一一进行测试了；另外，还需要进行模块接口、局部数据结构和错误处理的测试等，这里也不一一列举。

智能点歌系统的集成测试可以采用混合增量集成方式。

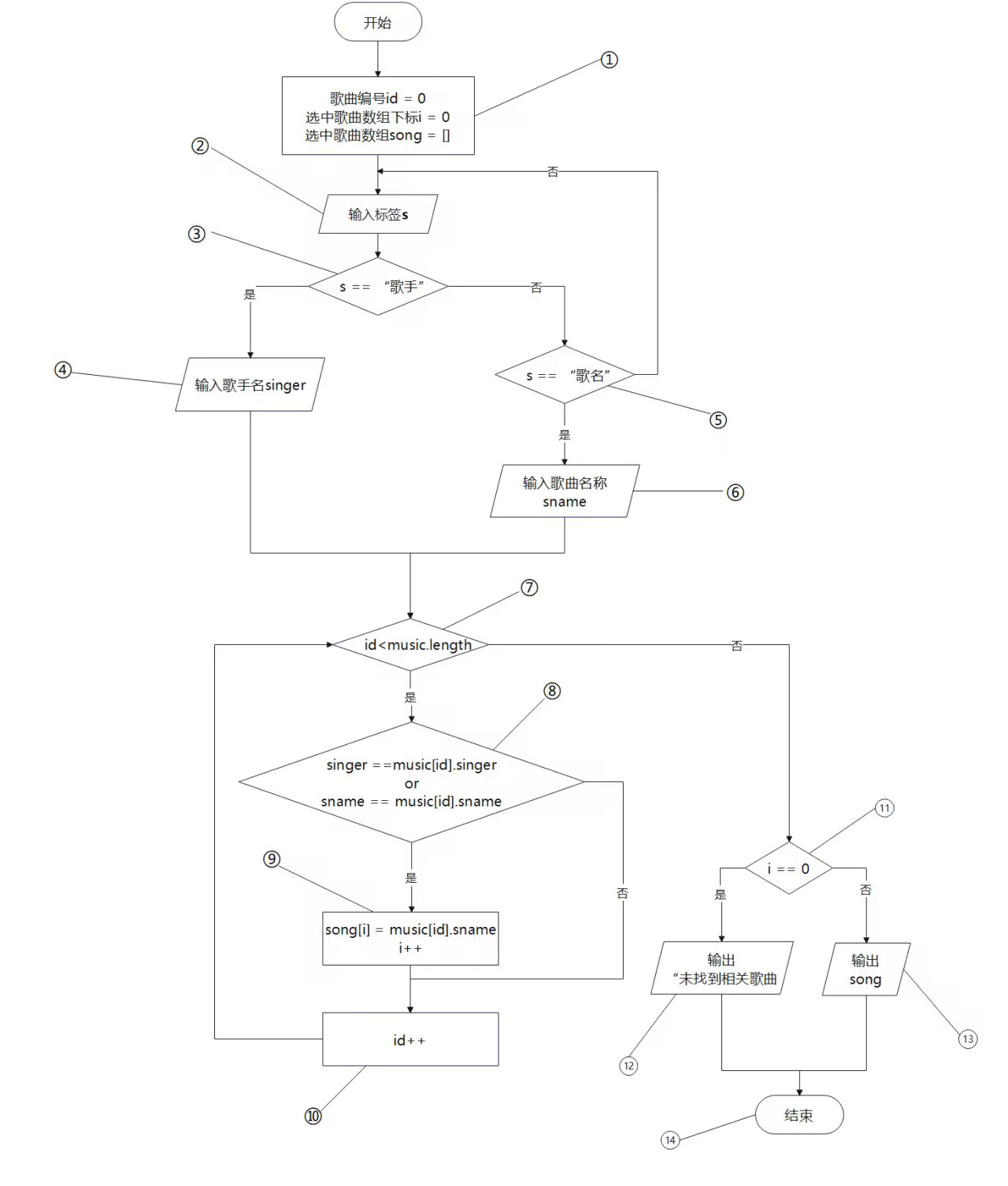
系统测试非常重要，如可靠性测试、压力测试、性能测试、安全性测试、可使用性测试等，对于智能点歌系统都是非常关键的。

在智能点歌系统中，几乎任何人员都是此用户的潜在用户，所以不管是正式验收测试、α测试、还是β测试，最终用户的参与都很方便，很容易测试到用户实际使用程序的多种可能性，从而二发现软件存在的缺陷。

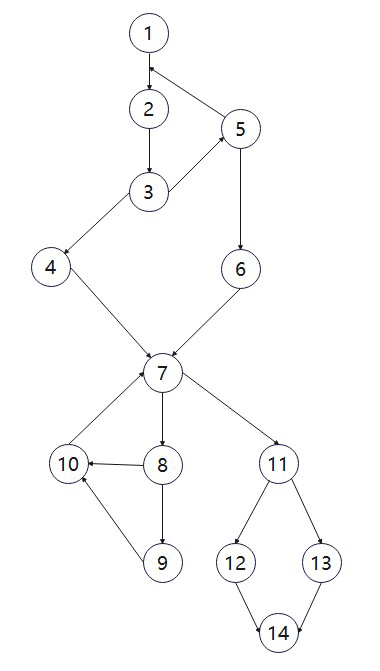
6.2.2单元测试

1. 关键词查询模块
2. 根据关键词查询模块的流程图进行程序效率的分析，计算环路复杂度

①标出关键词查询模块流程图结点，如图所示。



②画出关键词查询模块流图，如图所示。



采用三种计算环路复杂度的方法中的任意一种来计算，其环路复杂度为 6。五条基本路径如下：

第一条：1，2，3，5，2，...

第二条：1，2，3，5，6，7，11，12，14

第三条：1，2，3，5，6，7，11，13，14

第四条：1，2，3，5，6，7，8，10，7...

第五条：1，2，3，4，7，8，10，7，...

第六条：1，2，3，4，7，8，9，10，7，...

1. 按照环路复杂度设计测试用例

假设music.length = 100;

第一条：s = “hhhhkkkkk”；

第二条：此路径不是独立路径，可作为后续路径的子路径测试；

第三条：此路径不是独立路径，可作为后续路径的子路径测试；

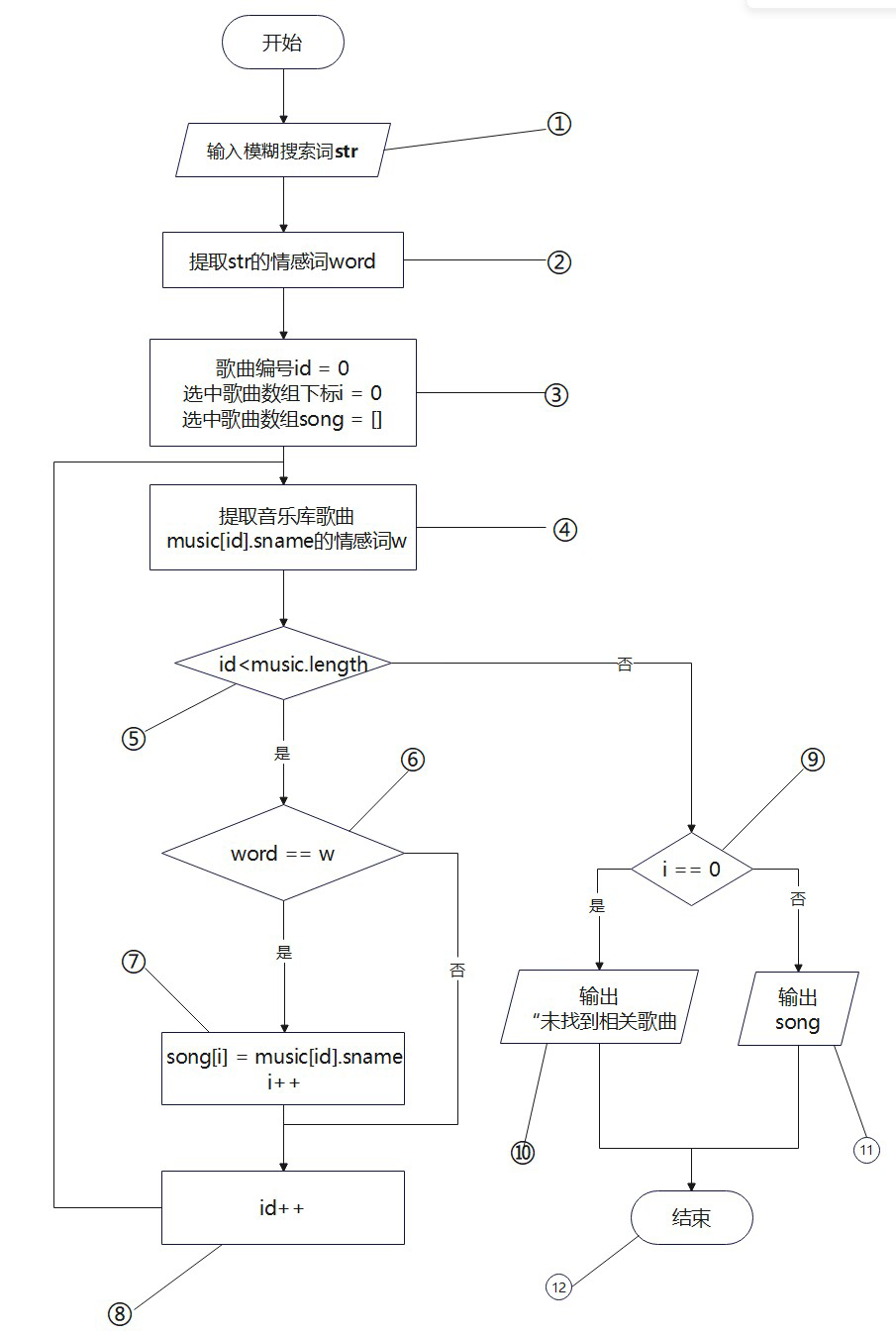
第四条：s = “歌名”,id = 0,sname = “七里香123”,由于music表中没有“七里香123”的歌曲名称，最终输出“未找到相关歌曲”，同时测试了第二条路径；

第五条：s = “歌手”,id = 0,singer = “查琪乐”；由于music表中没有“查琪乐”的歌手，最终输出“未找到相关歌曲”

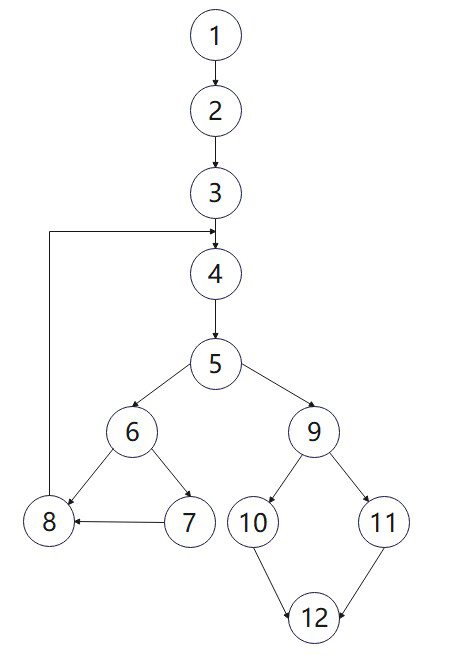
第六条：s = “歌手”,id = 0,singer = “周杰伦”；song存储music[0]到music[99]中singer = “周杰伦”的所有歌曲，最终输出song,同时测试了第三条路径。

1. 模糊查询模块
2. 根据模糊查询模块的流程图进行程序效率的分析，计算环路复杂度

①标出模糊查询模块流程图结点，如图所示。



②画出模糊查询模块流图，如图所示。



采用三种计算环路复杂度的方法中的任意一种来计算，其环路复杂度为 3。

三条基本路径如下：

第一条：1，2，3，4，5，9，10，12

第二条：1，2，3，4，5，9，11，12

第三条：1，2，3，4，5，6，8，4，...

第四条：1，2，3，4，5，6，7，8，4，...

1. 按照环路复杂度设计测试用例

假设music.length = 100;

第一条：此路径不是独立路径，可作为后续路径的子路径测试；

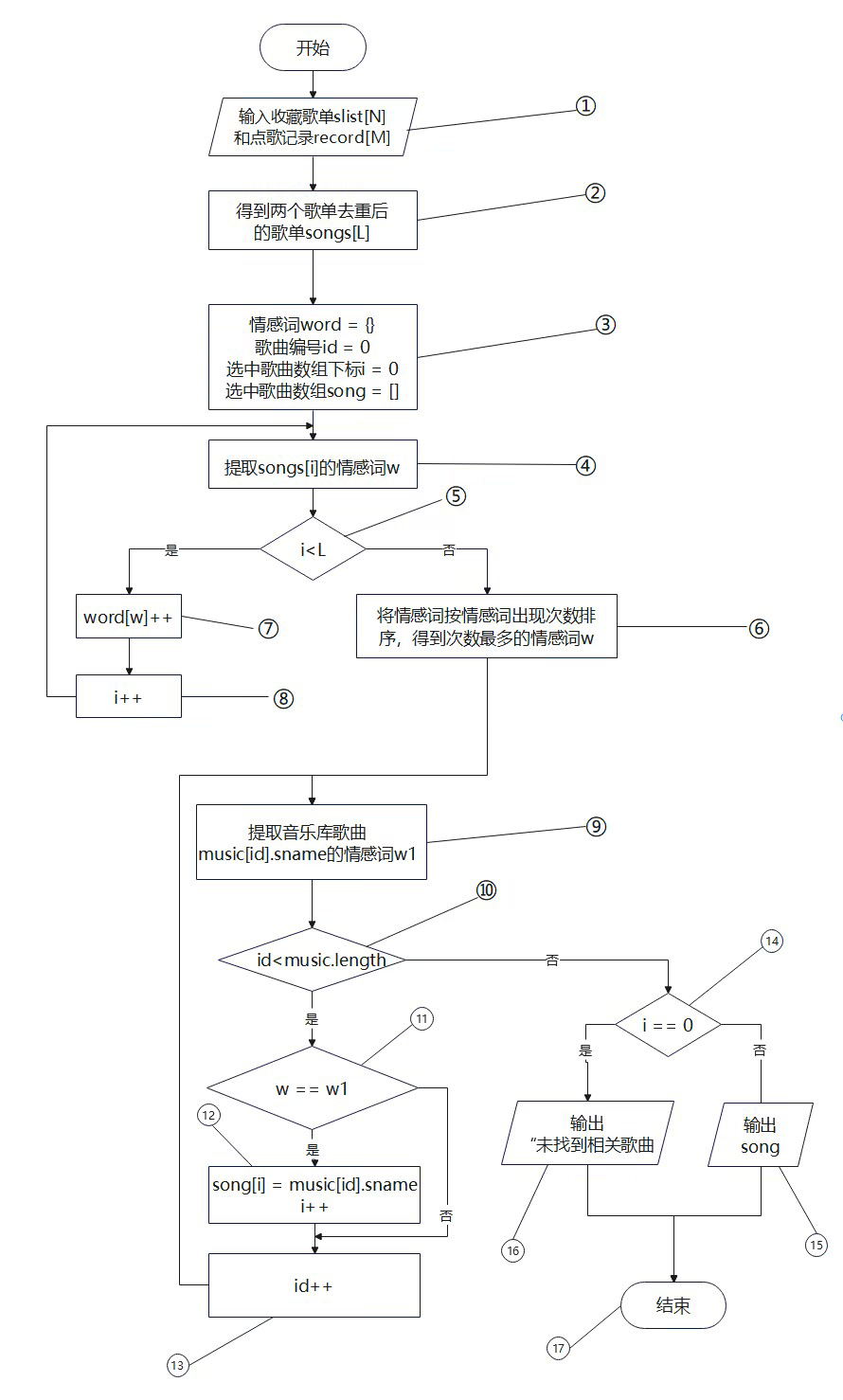
第二条：此路径不是独立路径，可作为后续路径的子路径测试；

第三条：str = “我想听一首jjjjj的歌”，word = “jjjjj”，id = 0,由于music[0]到music[99]中没有w = “jjj”的歌曲标签，最终输出“未找到相关歌曲”，同时测试了第一条路径；

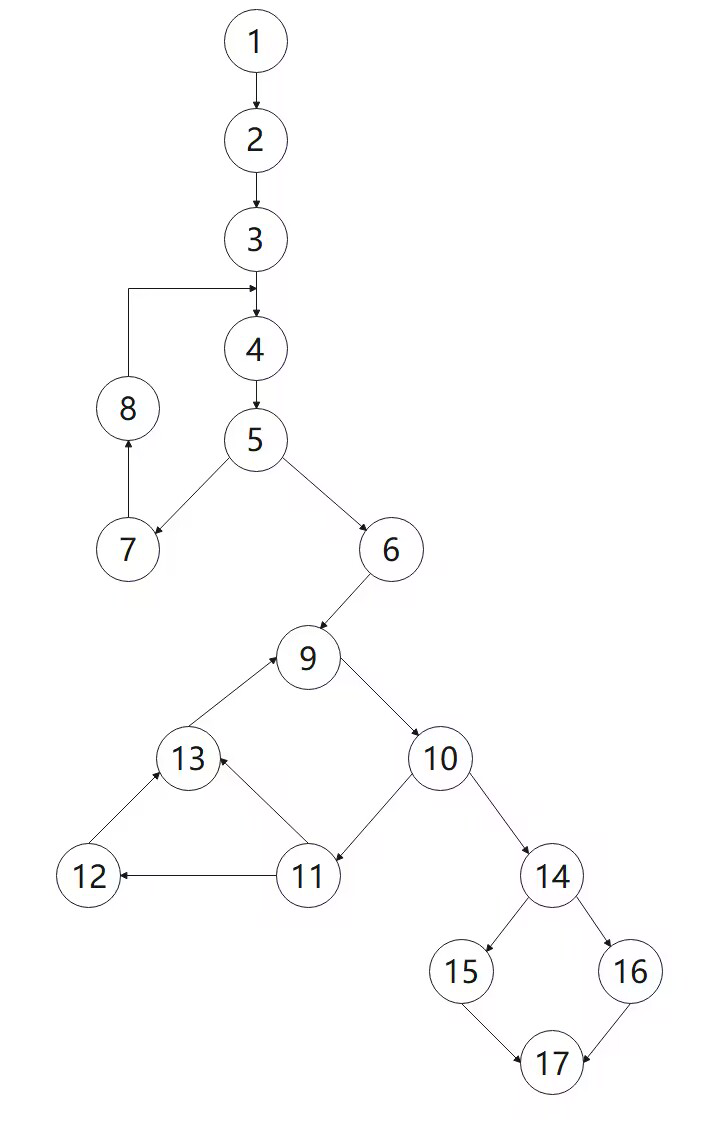
第四条：str = “我想听一首轻快的歌”，word = “轻快”，id = 0,song中存储music[0]到music[99]中w = “轻快”的歌曲，最终输出song，同时测试了第二条路径；

1. 智能推荐点歌模块
2. 根据智能推荐点歌模块的流程图进行程序效率的分析，计算环路复杂度

①标出智能推荐点歌模块流程图结点，如图所示。



②画出智能推荐点歌模块流图，如图所示。



采用三种计算环路复杂度的方法中的任意一种来计算，其环路复杂度为5。五条基本测试路径如下：

第一条：1，2，3，4，5，7，8，4，...

第二条：1，2，3，4，5，6，9，10，14，15，17

第三条：1，2，3，4，5，6，9，10，14，16，17

第四条：1，2，3，4，5，6，9，10，11，13，9，...

第五条：1，2，3，4，5，6，9，10，11，12，13，9，...

1. 按照环路复杂度设计测试用例

本算法与模糊查询算法类似，这里就不再赘述。

6.2.3集成测试

集成测试是一种正规测试过程，必须精心计划，并于单元测试的完成时间协调起来。在指定测试计划时，本系统决定采用增量式集成测试方法来进行组装测试，并按照关键词查询模块、模糊查询模块、智能推荐点歌模块的顺序连接；模块代码编制和测试进度与组抓鬼测试的顺序一致。

6.2.4系统测试

压力测试：在系统运行环境恶劣的情况下，期望系统能够按正常速度运行，并正确输出结果。

性能测试：软件应该能满足要求的功能。

安全性测试：设计一些模拟测试用例对系统进行测试，力图破环系统的保护机构以进入系统。从而检验的系统中已经存在的系统安全性和保密性措施是否发挥作用及有无漏洞。

互连测试：验证两个或多个不同的系统之间的互连性。

可使用性测试：从使用的合理性、方便性等角度对软件系统进行检查，以发现人为因素或使用上的问题。

6.2.5验收测试

发布软件时发布一个beta测试版，由软件的多个用户在一个或多个用户的实际使用环境下进行的测试，收集错误并修改软件。

# 7 展望与致谢

## 7.1 展望

本次《软件工程》课程设计实践，笔者收获良多。在此期间，笔者第一次真正涉及到了整个项目的实践开发，打破了本人对于计算机课程的固有印象。它证明了计算机科学与技术是一门极其注重动手实践的专业，如果缺乏实践，再多的理论学习也无法解决实际开发过程中遇到的各种困难，数据库这门课教会了我们数据如何进行组织和利用。得益于参加过省级大创项目的经验，笔者接触过软件工程课程的相关内容，本次实践得以从更多方面进行完善，但进行本次实践依然对于笔者充满调整，因为有太多的知识盲区让人感到挫败，想要开发一个完整的系统，既要求前端能力，又要求后端能力，仅仅凭借HTML5、CSS、SQL等知识开发起来无疑是非常吃力的，我们需要工具，我们需要框架和设计原则来简化和规范项目的开发，在未来软件工程实践的开发中，笔者将尝试使用它们来进行开发。

本系统只实现了预期目标的部分功能，限于目前的个人水平难以在规定时间内完成其他目标功能的实现，未来笔者将继续投入时间和精力继续开发和完善本系统。

## 7.2 感谢

历时七周的课程设计实践终于结束了！在本论文的最后，我要感谢这段时间以来向我提供帮助的各位老师、同学们和队员的配合。

特别感谢林卫中老师耐心的教育和指导！学生惭愧，没能很好地领悟课程的精髓，未来笔者将更加刻苦学习，不负老师们的期望！

同时也要感谢学习过程中各种资料的作者、免费开源软件的开发者和提供者，感谢你们为笔者完成系统开发所提供的帮助，谢谢你们！