大连海事大学 2021-2022（2）学期　　序号：

信息系统分析与设计课程实践报告

项目名称：音乐信息管理系统

项目组成员：林晨、杨灏哲、贾征强、徐达

姓名	学号	任务	成绩
林晨	2220191537	选题，制定开发计划，需求分析，系统设计，UI 设计与实现，歌曲录入、检索部分功能编码，统筹项目进度，视频制作。	
杨灏哲	2220191651	需求分析，负责 Git 的建立和维护，数据库设计，概要设计，音乐播放部分功能编码，报告撰写。	
贾征强	2220192615	需求分析，运行测试，数据流程图制作，运行测试，社交部分功能编码，报告撰写。	
徐达	2220192614	需求分析，数据字典编写，汇总数据，业务流程图，注册登录功能编码，报告撰写。	

项目组成员完成工作量：

姓名	学号	主要完成工作	占总工作量比例
林晨	2220191537	选题，制定开发计划，需求分析，系统设计，UI 设计与实现，歌曲录入、检索部分功能编码，统筹项目进度，视频制作。	30
杨灏哲	2220191651	需求分析，负责 Git 的建立和维护，数据库设计，概要设计，音乐播放部分功能编码，报告撰写。	30
贾征强	2220192615	需求分析，运行测试，数据流程图制作，运行测试，社交部分功能编码，报告撰写。	30
徐达	2220192614	需求分析，数据字典编写，汇总数据，业务流程图，注册登录功能编码，报告撰写。	10

目录

第一章 系统分析 1
1.1 需求描述 1
1.1.1 系统开发背景 1
1.1.2 可行性分析 1
1.1.3 需求描述 1
1.2 业务流程图 2
1.2.1 过程识别 2
1.2.2 过程的归并和分析 3
1.2.3 建立业务过程与组织机构的联系 5
1.3 数据流程图 10
1.4 功能分析图 17
1.5 数据字典 18
1.6 数据加工处理 25
1.6.1 用户信息管理子系统 25
1.6.2 歌曲管理子系统 25
1.6.3 歌单管理子系统 26
1.6.4 歌曲评论管理子系统 27
1.6.5 管理员子系统 27

第二章 系统设计 29
2.1 开发环境设计 29
2.1.1 计算机通信网络环境设计 29
2.1.2 系统开发平台选择 29
2.2 功能结构图设计 30
2.3 输入/输出设计 30
2.4 数据库设计 31
2.4.1 数据库概要设计 32
2.4.2 数据库逻辑设计 35
2.4.3 数据库物理设计 36
2.5 信息分类编码设计 38

第三章 系统实现 40
3.1 功能实现与界面设计 40
3.2 物理数据库建立 45

3.3 程序编码规范 48

第四章 总结 49

第一章 系统分析

1.1 需求描述

1.1.1 系统开发背景

音乐的魅力在生活中是极其大的，不同的国家、不同语言的人，可以从音乐中体会到相同的情感，可以加强人与人之间的联系，我们也可以从音乐中了解他国，因为音乐是人类共同的食粮，它也可以可以让身体放轻松，纾解压力。因此为使人们能够随时听到音乐，我们需要开发一款音乐系统，为用户提供了便利，使得人们可以通过手机等设备在任何时候欣赏到音乐。

1.1.2 可行性分析

- 技术可行性：提供快速的搜索匹配，同时给用户提供交流讨论的平台
- 经济可行性：开发成本相对较低，同时符合当代社会简单快捷的特点
- 运营可行性：在小的音乐圈试点，然后慢慢扩大使用者范围
- 社会因素可行性分析：方便实用的音乐播放器便于被人们接收。

1.1.3 需求描述

- 登录、注册功能：通过此功能对用户身分经行识别和用户基本信息的存储。
- 歌曲录入功能：歌曲管理人员承担的是对平台上所有上传歌曲的管理工作，对库中的音乐进行审核、统计等管理
- 歌曲检索功能：基于文本检索通过输入歌曲名、歌手名或者歌词来检索歌曲，通过对音乐库中的音乐进行特征标记完成，每首音乐都有歌名、歌手和歌词信息。同时还可以根据歌词片段搜索，也可以根据不同地区，风格，歌手来搜索
- 歌曲播放功能：进行音乐的播放、暂停、调整播放顺序等基本的播放器功能
- 评论：用户可以对歌曲发表自己的见解
- 歌单：用户可以创建歌单对歌曲进行管理

1.2 业务流程图

业务过程是管理各类资源的各种相关活动和决策的组合。管理人员，通过管理这些资源，支持管理目标。定义业务过程是 BSP 方法的核心，业务过程应独立于组织机构，要从企业的全部管理工作中，分析归纳出相应的业务过程。

定义业务过程的基本步骤为：过程识别、过程的归并和分析、建立业务过程与组织机构的联系。

1.2.1 过程识别

（1）定义计划和控制过程

根据战略计划和管理控制等来识别业务过程，具体定义如下表所示。

表 1.1 定义计划和控制过程表

战略规划	管理控制
组织计划	歌曲信息管理
系统设计	歌单信息管理
目标开发	用户信息管理
系统维护	评论信息管理
用户分析	歌手信息管理

（2）定义产品和服务过程

任何产品都有其生命周期：要求阶段、获得阶段、服务阶段、退出阶段。按产品和服务的生命周期阶段来识别其相应的过程，在生命周期的每一个阶段都相应地有一些过程。具体定义如下表所示。

表 1.2 定义产品和服务过程表

要求阶段	获得阶段	服务阶段	退出阶段
需求分析	前端实现	系统测试	系统运行
系统分析	后端实现	系统交付	系统维护
系统设计	数据库实现		系统更新
数据库设计			

（3）定义支持资源过程

BSP 把支持资源，描述成企业为实现其目标的消耗品和使用物。其基本资源

有四类：材料、资金、设备、人员。对每一类支持资源，按生命周期的各个阶段进行过程识别，基本过程如下表所示。

表 1.3 定义支持资源过程表

资源	生命周期阶段			
	要求阶段	获得阶段	服务阶段	退出阶段
资金	财务计划	资金获得	财务管理	会计支付
人员	人事计划	分工计划	专业开发	系统运行
材料	需求产生	需求获得	库存控制	产品交付
设备	设备计划	建设管理	设备维护	设备报损

1.2.2 过程的归并和分析

对于前面从战略规划和管理控制、主要产品和服务、支持企业资源这三个来源中识别出的过程，进行汇总分析，以减少过程在层次上的不一致性和重叠，并把同类型的的业务过程归类。并在此基础上，绘制业务流程图。具体业务流程图如下。

用户业务图

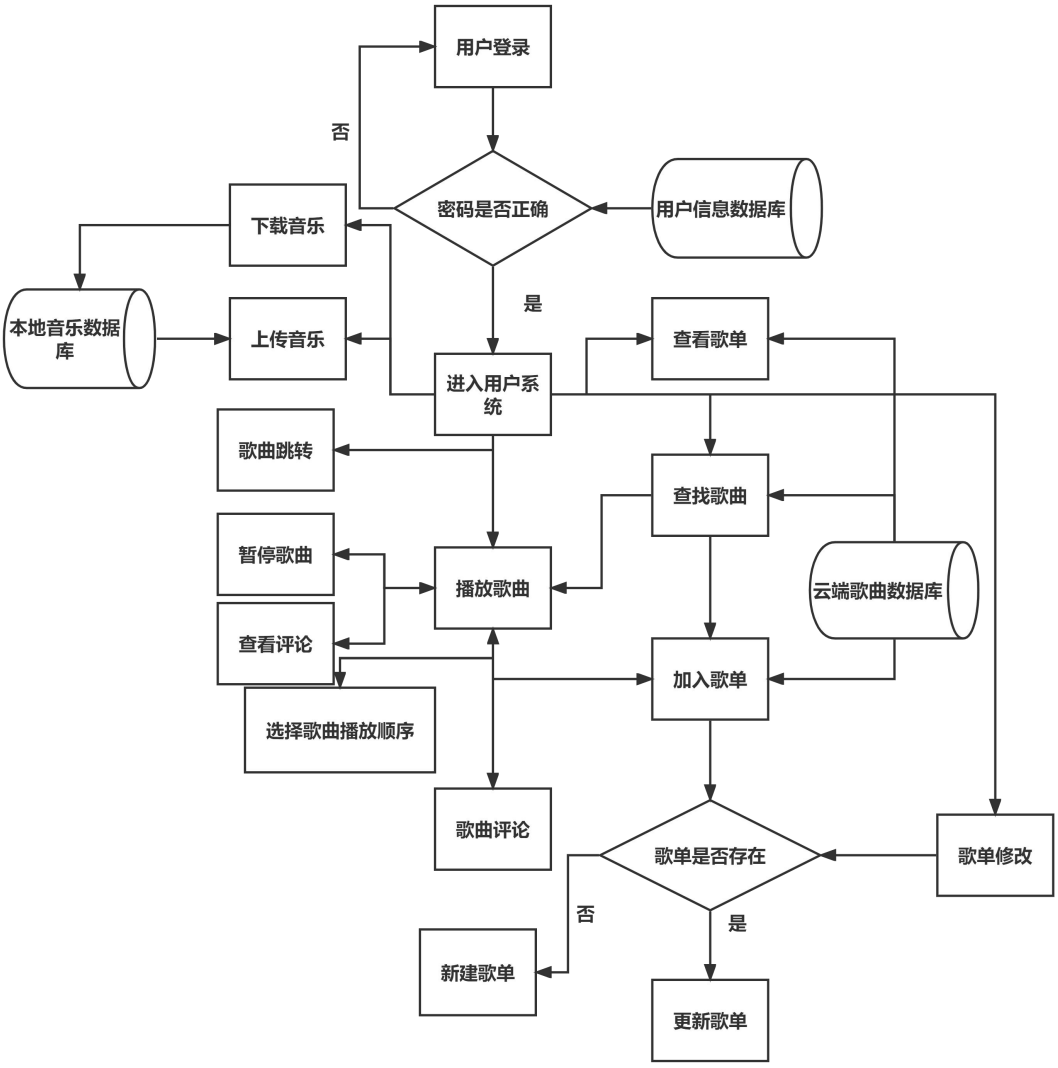

图 1.1 用户业务流程图

管理员业务图

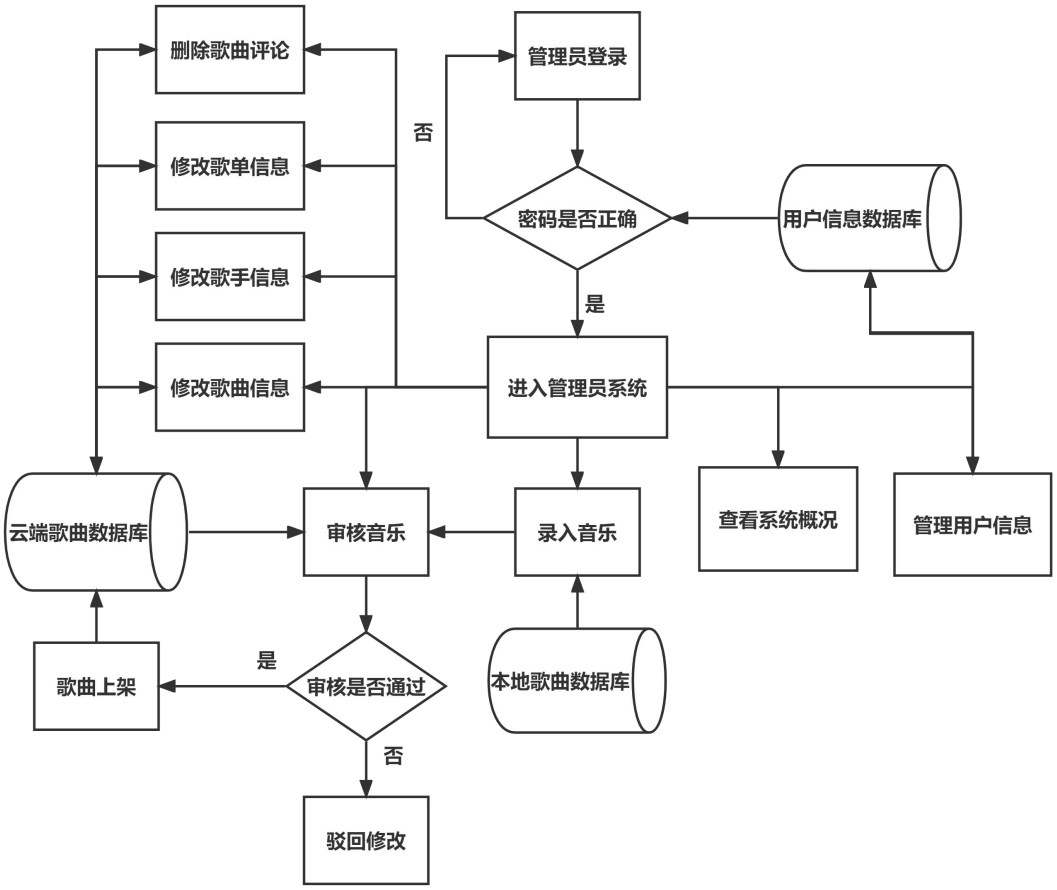

图 1.2 管理员业务流程图

1.2.3 建立业务过程与组织机构的联系

过程（功能）和数据之间存在产生和使用的关系，用 C-U 矩阵来表示他们之间的关联关系。

功能和数据的交叉点上标以 C 或 U；C（Create）表示这个数据类由相应的功能产生，U（Use）表示这个功能使用相应的数据类。具体的 C-U 矩阵如下表所示。

表 1.4　C-U 矩阵表

数据 / 过程	用户个人信息表	本地曲数据库	云端歌曲信息表	歌手表	歌单表	歌单歌曲关系表	用户评论表	歌曲评论关系表	管理员个人信息表	歌曲审核表
用户注册	C									
用户登录	U									
用户信息修改	U									
用户听歌			U	U						
用户查询歌曲			U	U						
用户上传歌曲		U	C	C						C
用户下载歌曲		C	U	U						
用户创建歌单					C	C				
用户删除歌单					U	U				
用户修改歌单					U	U				
用户					U	U				

分享歌单										
用户评论歌曲							C	C		
用户修改评论							U	U		
用户删除评论							U	U		
管理员登录									U	
管理员审核音乐										U
管理员导入音乐			C							U
管理员管理用户	U									

通过 C-U 矩阵，我们可以进行子系统划分和确定子系统的实施顺序。

表 1.5　C-U 矩阵划分子系统表

数据	用户个人信息表	本地歌曲数据库	云端歌曲信息表	歌手表	歌单表	歌单歌曲关系表	用户评论表	歌曲评论关系表	管理员个人信息表	歌曲审核表

过程										
用户注册	用户信息管理子系统									
用户登录										
用户信息修改										
用户听歌		歌曲管理子系统								
用户查询歌曲										
用户上传歌曲										C
用户下载歌曲										
用户创建歌单					歌单管理子系统					
用户删除歌单										
用户修改歌单										
用户分享歌单										
用户评论							歌曲评论管理子系统			

歌曲									
用户修改评论									
用户删除评论									
管理员登录									管理员子系统
管理员审核音乐									
管理员导入音乐			C						
管理员管理用户	U								

根据以上分析，将整个系统划分为用户信息管理子系统、歌曲管理子系统、歌单管理子系统、歌曲评论管理子系统、管理员子系统五个子系统。

1.3 数据流程图

数据流程图（Data Flow Diagram，DFD/Data Flow Chart），简称数据流图，是一种描述系统数据流程的主要工具，它用一组符号来描述整个系统中信息的全貌，综合地反映出信息在系统中的流动、处理和存储情况。

数据流程图有两个特征:

（1）抽象性

数据流程图把具体的组织机构、工作场所、物质流都去掉，只剩下信息和数据存储、流动、使用以及加工情况

（2）概括性

指数据流程图把系统对各种业务的处理过程联系起来考虑，形成一个总体。

我们通过对系统需求的分析和业务流程的分析，将整个系统划分为用户信息管理子系统、歌曲管理子系统、歌单管理子系统、歌曲评论管理子系统、管理员子系统。以下是我们的数据流图：

顶层数据流图：

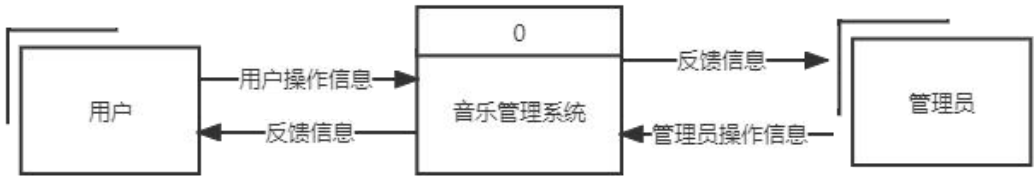

图 1.3 顶层数据流图

用户和管理员通过向音乐管理系统输入操作信息，系统经过处理后，形成相应的反馈信息，在图形界面展示出来。

第一层数据流图：

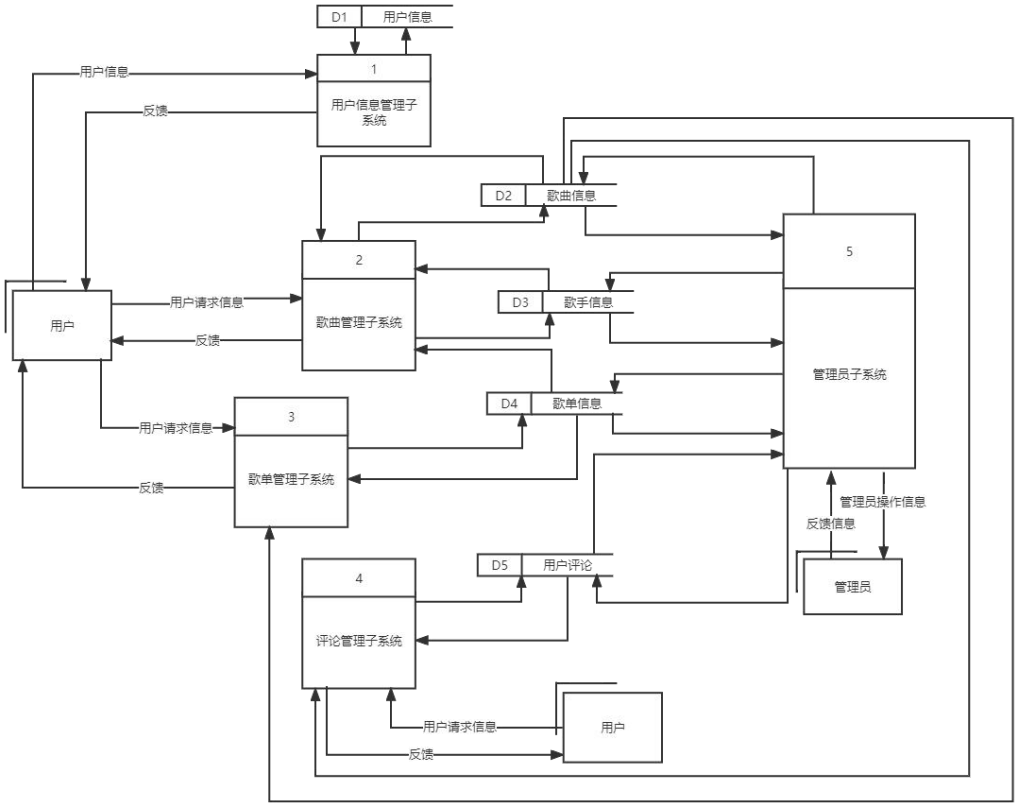

图 1.4 第一层数据流图

用户将用户信息输入到用户信息管理子系统，子系统通过与用户信息表交互，对用户信息实现增删改查，并生成反馈信息，在用户界面显示。

用户将歌曲管理请求信息（包括歌曲查看、查询、上传、下载）输入到歌曲管理子系统中，子系统通过调用相应的模块，进行歌曲信息的管理，并生成反馈信息，在用户界面显示。

用户将歌单管理请求信息（包括歌单创建、删除、修改、分享、查看）输入到歌单管理子系统中，子系统通过调用相应的模块，进行歌曲信息的管理，并生成反馈信息，在用户界面显示。

用户将评论管理请求信息（包括评论创建、查看、修改、删除）输入到歌曲评论管理子系统中，子系统通过调用相应的模块，进行评论信息的管理，并生成反馈信息，在用户界面显示。

管理员将管理操作信息（包括管理歌曲、评论、歌单、用户、歌手信息和查看平台概况请求）输入到管理员子系统中，子系统通过调用相应的模块，进行整个平台的管理，并生成反馈信息，在用户界面显示。

第二层数据流图：

用户信息管理子系统

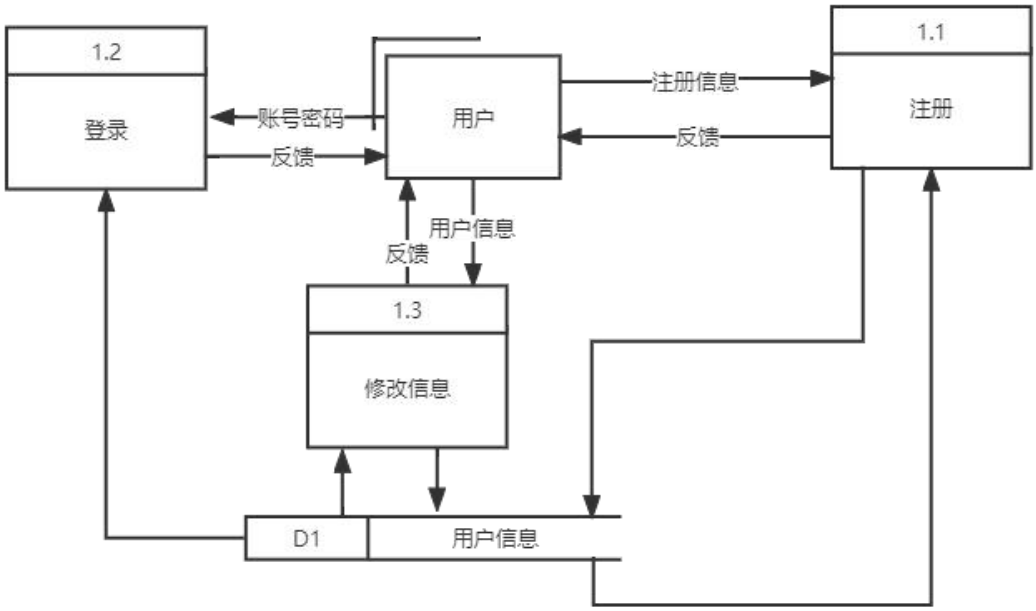

图 1.5 第二层数据流图——用户信息管理子系统

用户将账号密码输入系统，经过登录模块处理，与用户信息表中的信息相比对，最后将反馈信息返回给用户。

用户将注册输入系统，经过注册模块处理，与用户信息表中的信息相比对，最后将反馈信息返回给用户。

用户将修改后的用户信息输入到修改信息模块，该模块将信息与数据库中的信息相比对，修改数据库中的用户信息，并将反馈信息返回给用户。

歌曲管理子系统

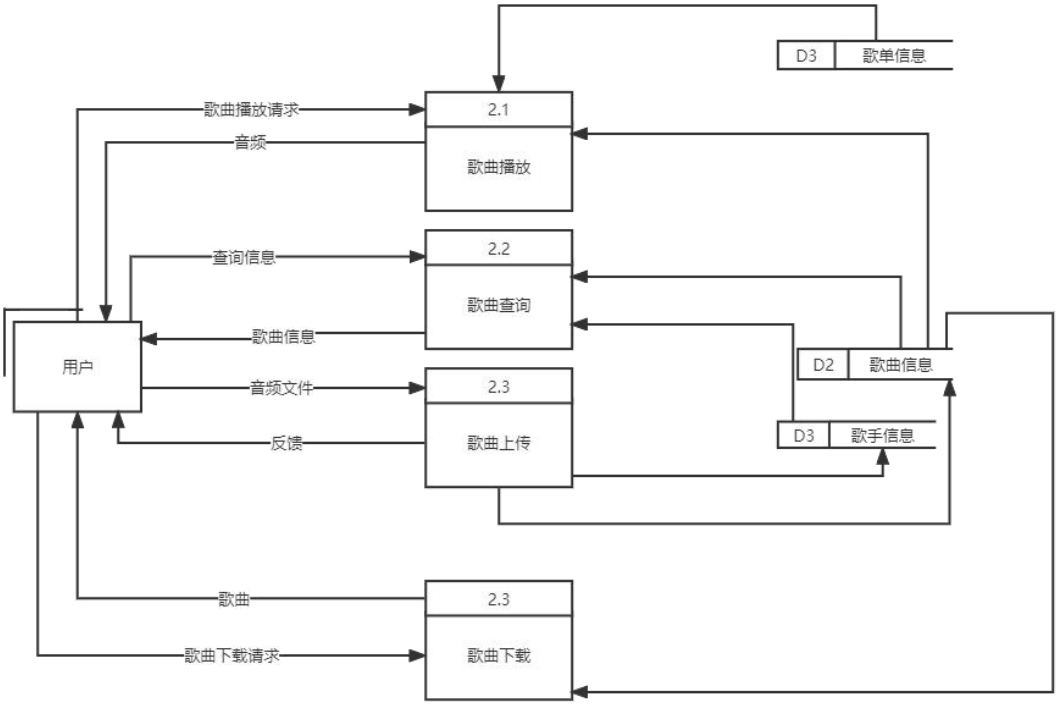

图 1.6 第二层数据流图——歌曲管理子系统

用户将歌曲播放请求输入到歌曲播放模块，该模块根据用户播放请求中的歌曲 ID 或歌单 ID，将信息与数据库中的信息相比对，提取出数据库中的歌曲信息，并将歌曲的音频返回给用户。

用户将歌曲查询信息输入到歌曲查询模块，该模块根据用户查询请求中的搜索关键词，在歌曲信息数据表中搜索，搜索出符合条件的歌曲信息，并将歌曲信息返回给用户。

用户将音频文件输入到歌曲上传模块，该模块根据用户上传的音频信息，将歌曲信息存储到数据库中，并将反馈信息返回给用户。

用户将歌曲下载请求信息输入到歌曲下载模块，该模块根据用户的歌曲下载信息，与数据库中的歌曲信息相比对，并将歌曲文件返回给用户。

歌单管理子系统

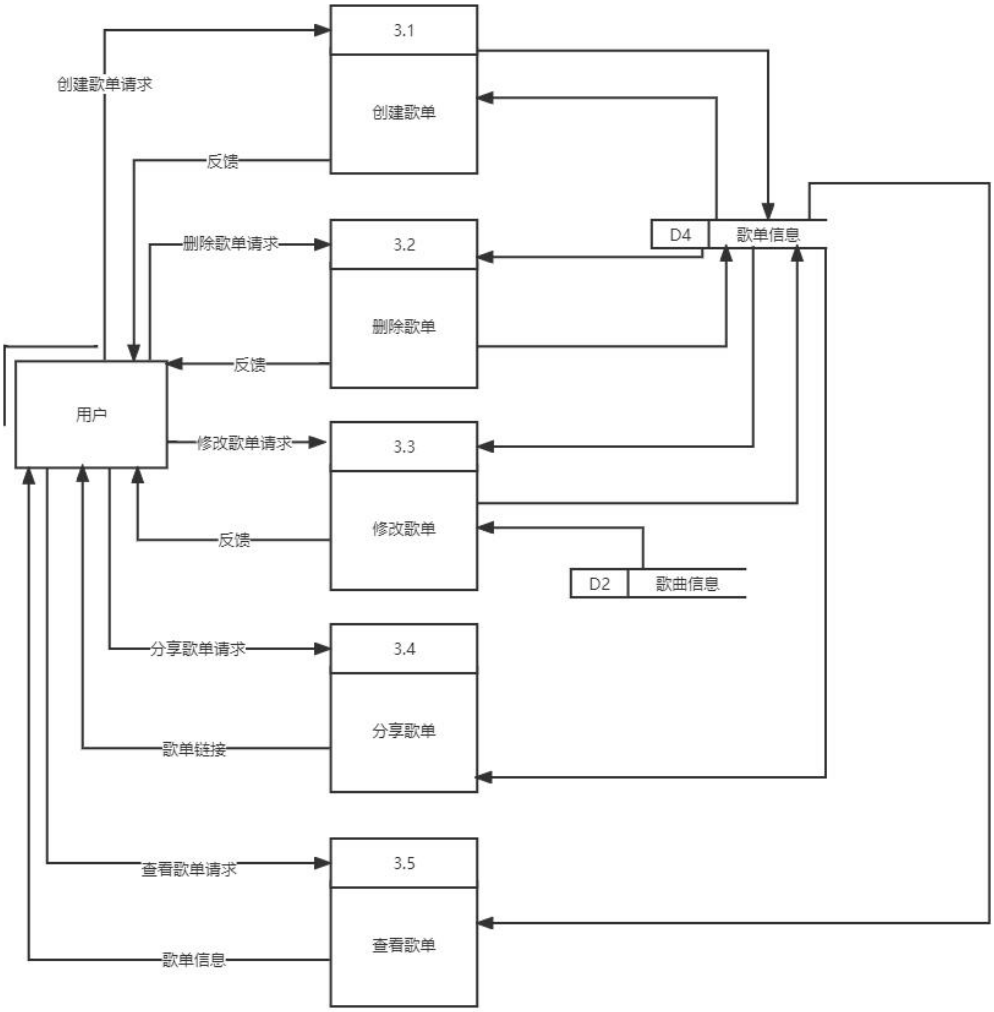

图 1.7 第二层数据流图——歌单管理子系统

用户将创建歌单、删除歌单请求输入到对应模块，对应模块调用代码，在歌单信息的数据库中实现各类操作，并将操作结果反馈返回给用户。

用户将修改歌单的请求输入到修改歌单模块，对应模块调用代码，根据请求中的修改信息，在歌单信息的数据库中执行修改操作，并将操作结果反馈返回给用户。

用户将分享歌单的请求输入到分享歌单模块，对应模块调用代码，根据请求中的歌单 ID，生成歌单链接，并返回给用户。

用户将查看歌单请求输入到查看歌单模块，模块访问并获得歌单及歌单中的歌曲信息，并将歌单信息返回给用户。

歌曲评论管理子系统：

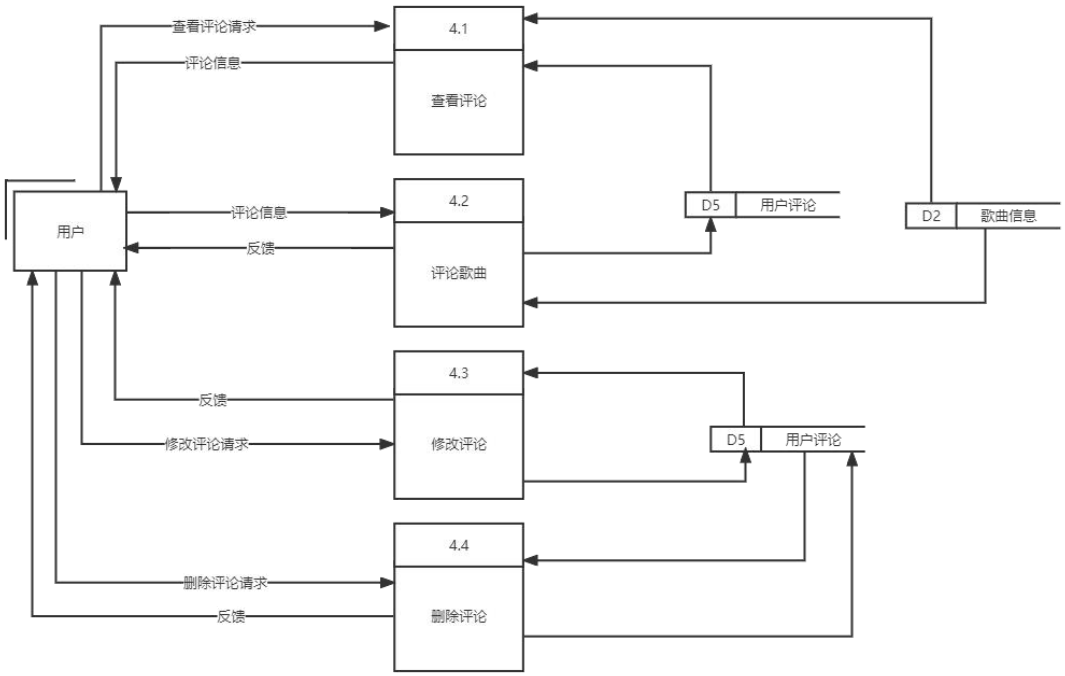

图 1.8 第二层数据流图——歌曲评论管理子系统

用户将查看评论请求输入到查看评论模块，模块访问用户评论数据库并获得该歌曲的用户评论信息，并将评论信息返回给用户。

用户将评论信息输入到评论歌曲模块，该模块根据歌曲信息和评论信息，将评论存入用户评论表中，并生成反馈信息，返回给用户。

用户将修改评论请求输入到修改评论模块，该模块根据修改评论请求中的评论 ID，修改用户评论表中的对应记录，并生成反馈信息，返回给用户。

用户将删除评论请求信息输入到删除评论模块，该模块根据评论 ID，将评论从用户评论表中删除，并生成反馈信息，返回给用户。

管理员子系统：

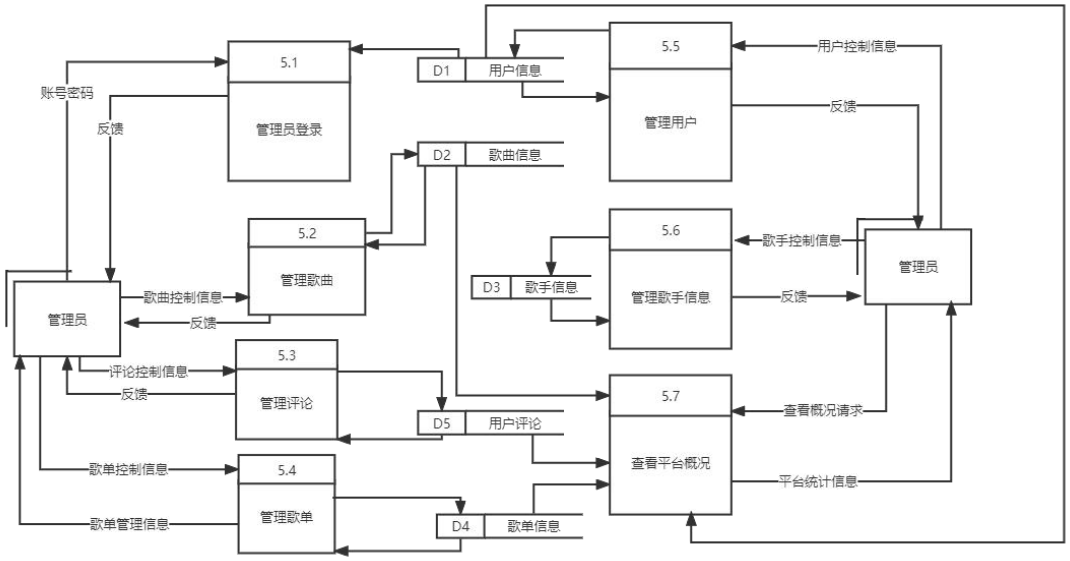

图 1.9 第二层数据流图——管理员子系统

管理员将账号密码输入到管理员登录模块，模块访问用户信息数据库并与表象进行比对，并将反馈信息返回给管理员。

管理员将歌曲控制信息输入到管理歌曲模块，模块根据歌曲控制信息操纵数据库中的歌曲信息并将反馈信息返回给管理员。

管理员将评论控制信息输入到管理评论模块，模块根据评论控制信息操纵数据库中的评论信息并将反馈信息返回给管理员。

管理员将歌单控制信息输入到管理歌单模块，模块根据歌单控制信息操纵数据库中的歌单信息并将反馈信息返回给管理员。

管理员将用户控制信息输入到管理用户模块，模块根据用户控制信息操纵数据库中的用户信息并将反馈信息返回给管理员。

管理员将歌手控制信息输入到管理歌手模块，模块根据歌手控制信息操纵数据库中的歌手信息并将反馈信息返回给管理员。

管理员将查看概况请求输入到查看平台概况模块，模块综合统计用户信息、歌曲信息、用户评论、歌单信息，输出平台统计信息，返回给管理员。

1.4 功能分析图

功能分析图是帮助我们表达设计思路的最有效方式，它是一个连贯的思考过程，是帮助我们放大设计中闪光点的最有利的工具，是系统组织内部各个有机构成要素相互作用的联系方式或形式，以求有效、合理地把组织部分组织起来，为实现共同目标而协同工作。由于组织结构在系统中的基础地位和关键作用，系统所有战略意义上的变革，都必须首先在组织结构上开始。

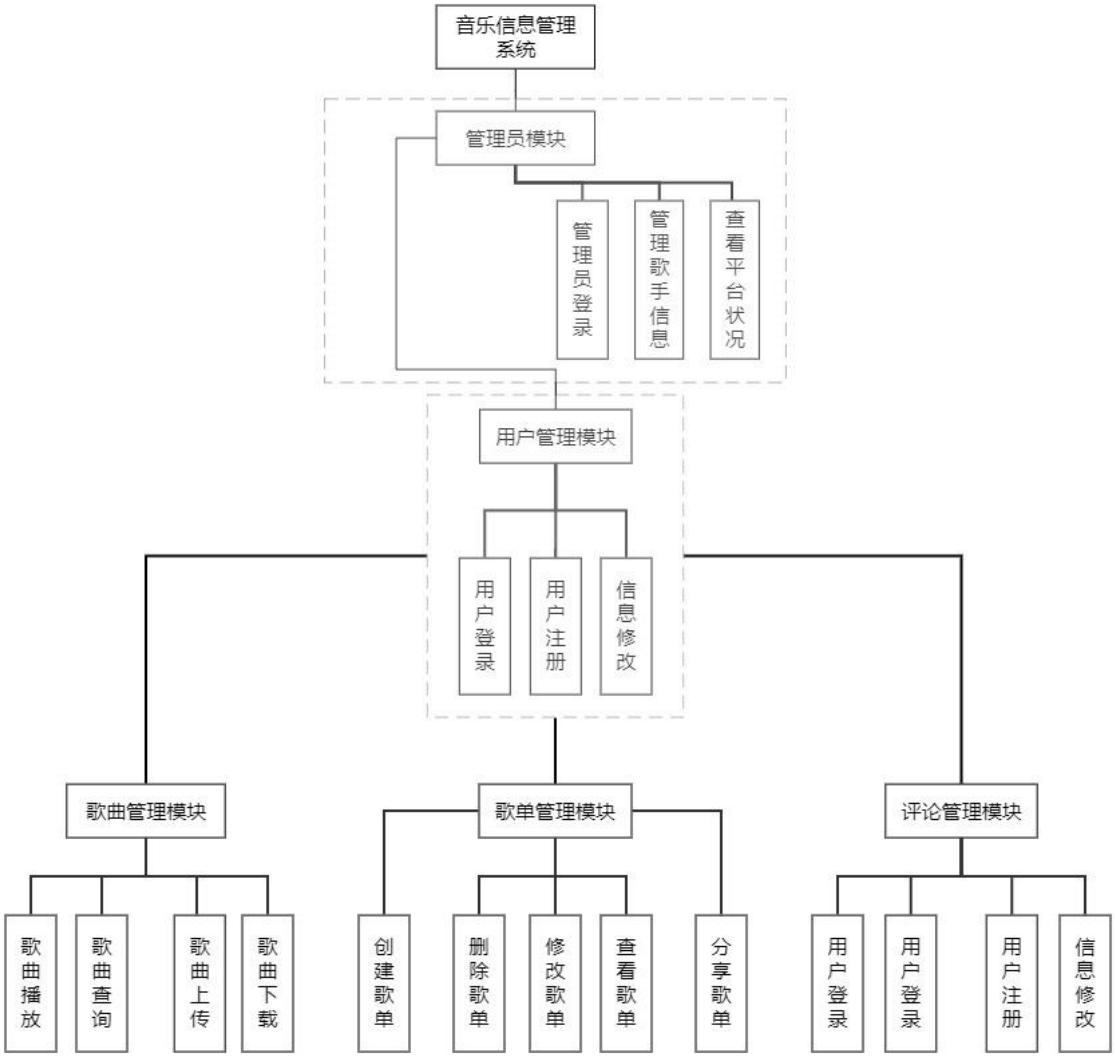

图 1.10 功能分析图

系统功能分析如上，这样的系统组织结构安排，一方面使用户体验感强，使用的功能更加明确，并且让管理员更好的管理用户和歌曲；另一方面精简了系统规模，避免了庞大的系统框架和组织结构基础，为整个系统的实际分析与设计打下基础。

1.5 数据字典

1.5.1 数据项定义

1. 数据项编号：01
数据项名称：用户 id
类型和大小：int
取值范围：
数据含义： 用户编号
2. 数据项编号：02
数据项名称：用户名
类型和大小：varchar(15)
取值范围：
数据含义：用户昵称
3. 数据项编号：03
数据项名称：密码
类型和大小：varchar(20)
取值范围：
数据含义：登入密码
4. 数据项编号：04
数据项名称：性别
类型和大小：varchar(20)
取值范围：
数据含义：用户性别
5. 数据项编号：05
数据项名称：地区
类型和大小：varchar(20)
取值范围：
数据含义：用户地区
6. 数据项编号：06
数据项名称：年龄
类型和大小：int
取值范围：
数据含义：用户年龄
7. 数据项编号：07

数据项名称：电话号码
类型和大小：char(11)
取值范围：
数据含义：用户电话号码
8. 数据项编号：08
数据项名称：类型
类型和大小：int
取值范围：0，1
数据含义：（1 表示管理员，0 表示用户）
9. 数据项编号：09
数据项名称：歌曲 id
类型和大小：int
取值范围：
数据含义：歌曲编号
10. 数据项编号：10
数据项名称：歌曲名
类型和大小：varchar(30)
取值范围：
数据含义：歌曲名称
11. 数据项编号：11
数据项名称：歌曲文件 url
类型和大小：varchar(50)
取值范围：
数据含义：
12. 数据项编号：12
数据项名称：歌曲状态
类型和大小：int
取值范围： 0，1
数据含义：1 为待审核，0 为正常
13. 数据项编号：13
数据项名称：歌手 id
类型和大小：int
取值范围：
数据含义：歌手编号

14. 数据项编号：14
数据项名称：歌单 id
类型和大小：int
取值范围：
数据含义：歌单编号
15. 数据项编号：15
数据项名称：歌单名
类型和大小：varchar(20)
取值范围：
数据含义：歌单名称
16. 数据项编号：16
数据项名称：评论 id
类型和大小：int
取值范围：
数据含义：评论编号
17. 数据项编号：17
数据项名称：评论内容
类型和大小：varchar(200)
取值范围：
数据含义：评论内容
18. 数据项编号：18
数据项名称：歌手名
类型和大小：varchar(30)
取值范围：
数据含义：歌手名字
19. 数据项编号：19
数据项名称：歌手地区
类型和大小：varchar(30)
取值范围：
数据含义：
20. 数据项编号：20
数据项名称：歌手介绍
类型和大小：varchar(200)
取值范围：

数据含义：歌手作品和成就的相关介绍

1.5.2 数据结构定义

1. 数据结构编号：DS01

数据结构名称：用户信息

简述：用户 id，用户名，用户密码，性别，地区，年龄，电话号码

组成：01+0203+04+05+06+07

2. 数据结构编号：DS02

数据结构名称：查询用户信息请求信息

简述：查询用户的相关信息

组成：01+02

3. 数据结构编号：DS03

数据结构名称：权限控制信息

简述：用于更改用户权限，增加管理员等。

组成：01+08

4. 数据结构编号：DS04

数据结构名称：用户更改信息

简述：更改用户的信息如角色等。包括用户手机号，要更改的身份。

组成：01+07+08

5. 数据结构编号：DS05

数据结构名称：注册信息

简述：用户注册时的信息

组成：02+03+04+05+06+07

6. 数据结构编号：DS06

数据结构名称：用户更改信息

简述：更改用户的信息如角色等。包括用户手机号，要更改的身份。

组成：02+07+08

7. 数据结构编号：DS07

数据结构名称：歌曲查询请求信息

简述：用户查询歌曲的信息。包括歌曲名，歌曲编号，歌手编号。

组成：09+10+12

8. 数据结构编号：DS08

数据结构名称：歌曲播放请求信息

简述：用户请求播放的歌曲名，歌曲 id，歌手

组成：09+10+18

9. 数据结构编号：DS09
数据结构名称：歌曲信息
简述：包括歌名，歌曲编号，歌曲状态，歌手
组成：09+10+13+18
10. 数据结构编号：DS10
数据结构名称：歌单控制信息
简述：用于对歌单的查询，增加，修改，删除。包括歌单名，歌单编号。
组成：20+29
11. 数据结构编号：DS11
数据结构名称：歌单信息
简述： 歌单 id，歌单名，歌单的用户
组成：14+15+01
12. 数据结构编号：DS12
数据结构名称：歌单修改信息
简述： 创建歌单，修改歌单，删除歌单的相关信息
组成：14+15
13. 数据结构编号：DS13
数据结构名称：音频文件
简述：歌曲文件，歌曲状态
组成：11+13
14.数据结构编号：DS14
数据结构名称：平台统计信息
简述：平台歌曲，用户，歌单，歌手数量信息的管理
组成：1+9+14+16
15.数据结构编号：DS15
数据结构名称：评论控制信息
简述：评论内容，评论管理
组成：16+17

1.5.3 数据流定义

1. 数据流编号：F01
数据流名称：用户信息
简述：系统传递的用户信息。
数据流来源：用户
数据流去向：用户信息管理子系统

2. 数据流编号：F02
数据流名称：反馈信息
简述：系统传递的用户信息
数据流来源：音乐管理系统
数据流去向：用户，管理员
3. 数据流编号：F03
数据流名称：管理员操作信息
简述：管理员需要进行的操作
数据流来源：管理员子系统
数据流去向：管理员
4. 数据流编号：F04
数据流名称：用户需求信息
简述：用户点歌的需求
数据流来源：用户
数据流去向：歌曲管理子系统
5. 数据流编号：F05
数据流名称：用户歌单需求信息
简述：用户对歌单的操作
数据流来源：用户
数据流去向：歌单管理子系统
6. 数据流编号：F06
数据流名称：用户评论需求信息
简述：用户评论信息
数据流来源：用户
数据流去向：评论管理子系统

1.5.4 数据存储

1. 数据存储编号：D1
数据存储名称：用户信息
简述：用户信息
数据存储组成：DS01+DS02+DS04
相关联的处理：管理信息处理，用户信息处理
2. 数据存储编号：D2
数据存储名称：歌曲信息
简述：歌曲编号，音乐名，歌手信息，用户信息，文件 url

数据存储组成：27+06+23+08+DS03+DS04+DS12+07

相关联的处理：用户信息处理，管理信息处理，

3. 数据存储编号：D3

数据存储名称：歌手信息

简述：歌手编号，歌手名，歌手

数据存储组成：

相关联的处理：歌手信息处理，管理信息处理，歌曲统计综合查询，歌曲审核

4. 数据存储编号：D4

数据存储名称：歌单信息

简述：歌单 id，歌单情况

数据存储组成：DS11+DS12

相关联的处理：歌单管理信息处理，歌单查询处理

5. 数据存储编号：D5

数据存储名称：用户评论

简述：用户信息，评论信息

数据存储组成：DS01+17+18

相关联的处理：用户评论处理

1.5.5 外部实体定义

1. 外部实体编号：S01

外部实体名称：用户

简述：使用音乐系统的用户。

有关数据流：来自用户的用户信息，上传歌曲；系统输出给用户的反馈信息

2. 外部实体编号：S02

外部实体名称：管理员

简述：审核音乐，删除音乐，增加音乐，更改音乐状态的管理员

有关数据流：来自管理员的管理请求信息，增删改查音乐；系统输出给音乐管理员的反馈信息，查询歌曲信息

1.6 数据加工处理

本项目主要依托于五大子系统对不同的数据进行加工处理，五大子系统中存在多个数据加工处理模块，对相应的数据进行加工处理。

1.6.1 用户信息管理子系统

注册登录

用户通过选择注册或登录按钮，选择进入注册或登录界面，在界面中输入账号密码和个人信息，提交之后，将信息封装成 JSON 格式，通过 http 发送给远端服务器。服务器判断请求是注册还是登录。

如果是注册，先使用 SELECT 语句在数据库中查找对应的用户名，如果用户名已注册则输出错误信息，返回给客户端；如果用户名未注册，则将用户信息存入数据库中，客户端输出注册成功消息。

如果是登录，先使用 SELECT 语句在数据库中查找对应的用户名，如果用户名未注册则输出未注册错误信息，返回给客户端；如果密码错误，客户端输出密码错误消息。

修改个人信息

用户需要修改个人信息时，在修改页面填写表单，输入修改后的信息，然后封装成 JSON 发送给服务器，服务器将封装后的信息提取出来，修改数据库中的对应项，并返回修改成功消息给客户端，客户端输出修改成功消息和修改后的个人信息。

1.6.2 歌曲管理子系统

歌曲播放

用户选择要听的歌曲，客户端先通过歌曲 ID 判断歌曲是否在本地缓存中，若不在本地缓存中则发送请求给服务器，从服务器下载歌曲及歌曲信息，存储到本地，并播放。如果在本地缓存中则直接播放。

歌曲查询

用户通过搜索框输入要搜索的歌曲，客户端将用户的搜索词发送给服务器，服务器通过 SQL 语句模糊查询，查看是否存在该歌曲，如果不存在则输出错误信息，存在就输出歌曲信息。

上传

用户进入上传界面，输入要上传的歌曲文件以及歌曲信息，点击上传，将信息打包上传至服务器，同时在数据库中更新歌曲信息，并输出操作成功的信息。

下载

用户选择一首歌曲发起下载请求，客户端将歌曲 ID 及下载请求发送给服务器，服务器接收到请求后，根据歌曲 ID 在数据库中搜索歌曲的存储路径，将歌曲发送给客户端，客户端接收到歌曲后将歌曲存储在本地。

1.6.3 歌单管理子系统

创建歌单

用户发送创建歌单请求和创建歌单的信息，客户端将这些信息封装后发送给服务器，服务器接收到请求后根据歌单信息，在数据库中添加，客户端输出“创建成功”提示，并显示歌单。

删除歌单

用户通过点击删除按钮发送删除歌单请求和想要删除歌单的 ID，客户端将这些信息封装后发送给服务器，服务器接收到请求后根据歌单 ID，在数据库中删除，客户端输出“删除成功”提示，并显示删除后的结果。

查看歌单

用户通过点击歌单发送查询歌单请求的和歌单的 ID，客户端将这些信息封装后发送给服务器，服务器接收到请求后根据歌单 ID，在歌曲-歌单表中搜索该歌单的歌曲，并将歌曲的信息，发送到客户端，然后客户端输出歌单。

修改歌单

用户通过点击歌单中的歌曲项目，可以选择删除该歌曲，或点击歌曲，选择加入歌单，然后客户端将请求、歌曲 ID 和歌单 ID 封装成 JSON，发送到服务器，服务器接收到请求后使用对应的 INSERT 或 DELETE 语句，实现歌单歌曲的增加和删除，并在客户端输出修改后的歌单信息。

分享歌单

用户通过点击歌单的分享按钮，可以选择分享歌单，客户端会生成一个带着请求的链接，点击这个链接的用户会将歌单 ID 发送给服务器，服务器接收到请

求后根据歌单 ID，在歌曲-歌单表中搜索该歌单的歌曲，并将歌曲的信息，发送到客户端，然后客户端输出歌单。

1.6.4 歌曲评论管理子系统

查看评论

用户通过点击歌曲，进入歌曲播放界面后，点击查看评论按钮，客户端将查看评论的请求和歌曲 ID 信息封装后发送给服务器，服务器接收到请求后根据歌曲 ID，在评论表中搜索该歌曲的评论，并将歌曲评论的信息，发送到客户端，然后客户端输出评论信息。

评论歌曲

用户通过在评论区输入评论文本后，点击发送。客户端将查看评论的请求和歌曲 ID 信息封装后发送给服务器，服务器接收到请求后根据歌曲 ID，添加该歌曲的评论，并将歌曲评论的信息，发送到客户端，然后客户端输出评论信息。

修改评论

用户通过点击自己的评论文本，点击修改，重新输入评论信息。客户端将修改评论的请求和评论 ID 信息封装后发送给服务器，服务器接收到请求后根据评论 ID，修改该条评论，客户端输出修改后的评论信息。

删除评论

用户通过点击自己的评论文本，点击删除。客户端将删除评论的请求和评论 ID 信息封装后发送给服务器，服务器接收到请求后根据评论 ID，使用 DELETE 语句，删除该条评论，客户端输出删除成功的消息以及删除该条评论后的评论信息。

1.6.5 管理员子系统

管理歌曲

管理员进入管理歌曲页面，客户端发送管理歌曲请求到服务器，服务器将歌曲信息发送到客户端。管理员通过选择歌曲，能对歌曲执行删除，修改等操作，客户端接收到操作请求后，将歌曲 ID 和修改后信息发送给服务器，服务器调用对应 SQL 语句处理之后将结果返回给客户端，客户端输出反馈信息。如果歌曲被删除或修改，对应的歌单表记录也会随之修改。

管理评论

管理员进入管理歌曲界面后，可以通过点击查看歌曲评论按钮进入评论管理页面，通过点击评论文本，选择删除。客户端将删除评论的请求和评论 ID 信息封装后发送给服务器，服务器接收到请求后根据评论 ID，使用 DELETE 语句，删除该条评论，客户端输出删除成功的消息以及删除该条评论后的评论信息。

管理歌单

管理员进入管理歌单页面，客户端发送管理歌单请求到服务器，服务器将歌单信息发送到客户端。管理员通过选择歌单，点击删除按钮，能对歌单执行删除，修改等操作，客户端接收到操作请求后，将歌曲 ID 和修改后信息发送给服务器，服务器调用对应 SQL 语句处理之后将结果返回给客户端，客户端输出反馈信息。

管理用户

管理员进入管理用户页面，客户端发送管理用户请求到服务器，服务器将用户信息发送到客户端。管理员通过选择用户，能对用户执行删除，修改等操作，客户端接收到操作请求后，将用户 ID 和修改后信息发送给服务器，服务器调用对应 SQL 语句处理之后将结果返回给客户端，客户端输出反馈信息。

管理歌手信息

管理员进入管理歌手信息页面，客户端发送管理歌手请求到服务器，服务器将歌手列表发送到客户端。管理员通过选择歌手，能够查看歌手的详细信息，并可以对歌手信息执行删除，修改等操作，客户端接收到操作请求后，将歌手 ID 和修改后信息发送给服务器，服务器调用对应 SQL 语句处理之后将结果返回给客户端，客户端输出反馈信息。

查看平台概况

管理员通过点击查看平台概况按钮，客户端发送查看平台概况请求到服务器，服务器通过 SQL 语句统计出平台的各项数据并通过界面输出。

第二章 系统设计

2.1 开发环境设计

2.1.1 计算机通信网络环境设计

我们使用安卓前端+ Spring Boot 框架后端来构造整个项目。前端负责展示用户界面及处理部分事件，以及向后端发送请求并接受后端传输的数据。后端负责提供接口来处理前端的请求，并将数据加工后返回给前端。

我们使用自己搭建的服务器作为后端，服务器系统为 Ubuntu 20.04.1。通过拨打运营商电话，向运营商申请 IPv4 地址，使我们的服务器能够通过公网访问，但需保证服务器的拨号连接不能中断，否则 IP 地址会发生改变。

我们在服务器上部署 Spring Boot 框架后台，并根据前端的不同请求设计不同接口，前端通过 HTTP 协议，将请求信息发送到服务器的对应接口，服务器提取请求信息，并通过加工，使用回调方法返回给前端，前端根据返回的数据进行界面的控制。

我们前后端之间的数据交换采用 JSON 格式。JSON（JavaScript Object Notation）是一种轻量级的数据交换格式。易于人阅读和编写，可以在多种语言之间进行数据交换。同时也易于机器解析和生成。

2.1.2 系统开发平台选择

我们采用 windows10/11 系统，使用 Android Studio 2020.3.1 API 32 进行安卓前端的开发，使用 IDEA 进行 Spring Boot 框架后台的开发。

服务器采用 Ubuntu 20.04.1 系统，并将开发完毕的 Spring Boot 项目的 Jar 包部署在服务器上。

数据库采用 SQL SERVER 2019，部署在后端服务器上，通过 JDBC 进行访问和操作。使用 SQL Server Management Studio 通过远程连接进行数据库管理。

2.2 功能结构图设计

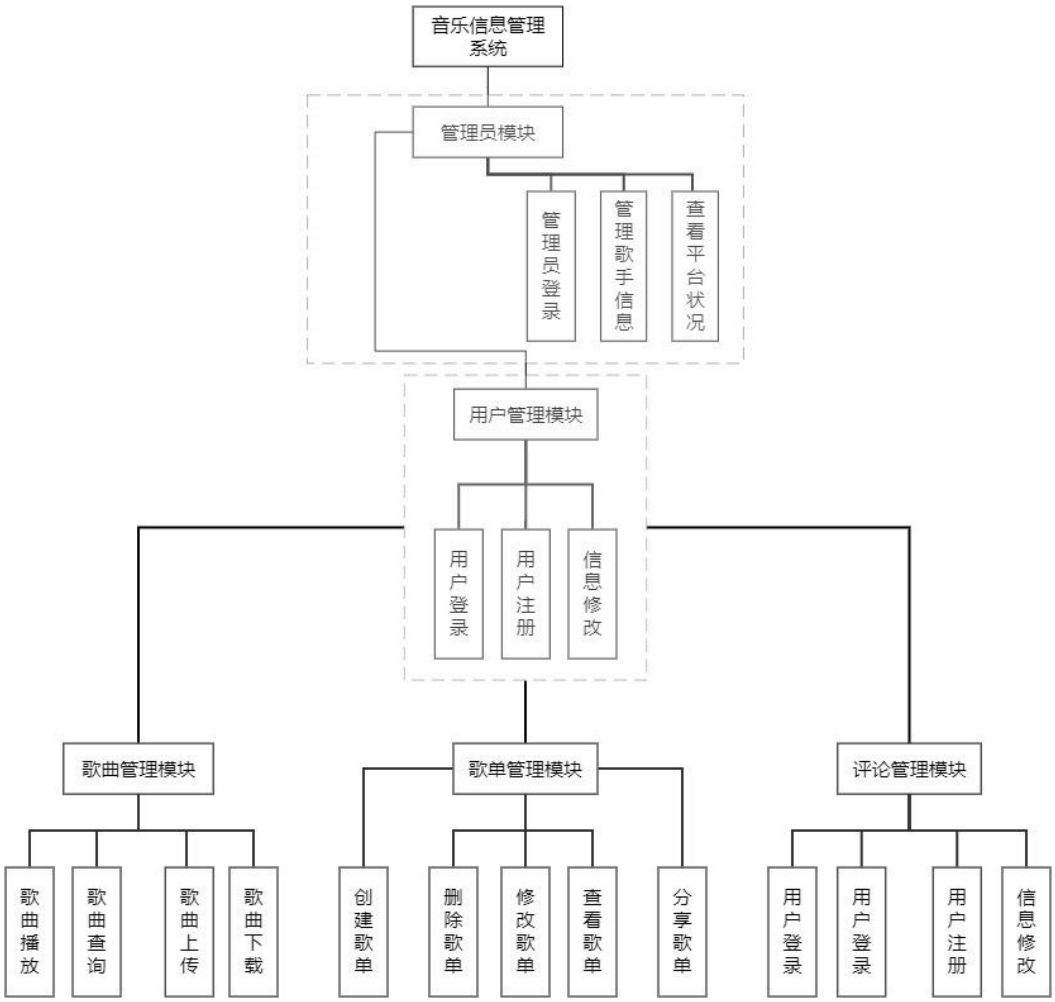

图 2.1 功能结构图

2.3 输入/输出设计

输入输出设计是管理信息系统与用户的界面，一般而言，输入输出设计对于系统开发人员可能并不重要，但对用户来说，却显得尤为重要。

2.3.1 输入/输出设计的意义

(1)它是一个组织系统形象(Cooperation Identify System,CIS)的具体体现；

(2)它能够为用户建立良好的工作环境，激发用户努力学习、主动工作的热情；

(3)符合用户习惯，方便用户操作，使目标系统易于为用户所接受；

(4)为用户提供易读易懂的信息形态。

2.3.2 输入设计

输入设计的目的是提高输入效率，减少输入错误。

(1)输入设计的原则与输入类型

√控制输入量：尽可能利用计算。

√减少输入延迟：批量输入、周转文件输入。

√减少输入错误：采用多种校验方法和验证技术。

√避免额外步骤。

√简化输入过程。

(2)输入设计的任务

(3)输入设备和介质

(4)输入信息的校验

校验对象；数据出错的种类；数据的校验方法；差错的纠正

(5)输入屏幕设计

常用的是人机对话方式，具体有菜单式，填表法，应答式，选择式，提问法。

我们选用菜单式与填表法相结合，从而设计实现用户登录以及注册等相关页面。

2.3.3 输出设计

(1)输出类型与输出内容

输出类型：外部输出；内部输出；中间输出；交互输出；操作输出。

输出内容：

√输出信息使用情况：信息的使用者、使用目的、信息量、输出周期、有效期、保管方法和输出份数。

√输出信息内容：输出项目、精度、信息形式（文字、数字）。

√输出格式：表格、报告、图形等。

√输出设备和介质：设备如：打印机、显示器等；介质如：磁盘、磁带、纸张（普通、专用）等。

(2)输出设计的任务

(3)输出设备和介质

我们选用数据库表的输出格式，在显示器上输出目前所有用户所注册的所有相关信息。

2.4 数据库设计

数据库的设计主要包括数据库概要设计、数据库逻辑设计和数据库物理设计。

2.4.1 数据库概要设计

概念结构设计是整个数据库设计的关键，通过对用户需求进行综合、归并与抽象，形成信息世界的结构，独立于具体的数据模型和 DBMS。E-R 模型是数据库概念设计的主要工具，在数据需求分析的基础上，通过识别系统中的实体和实体的属性，仔细分析实体与实体之间的联系，并用 E-R 图来描述概念结构设计的结果。

用户实体图：属性包括用户 id、用户名、密码、性别、年龄、地区、手机号、用户类型。

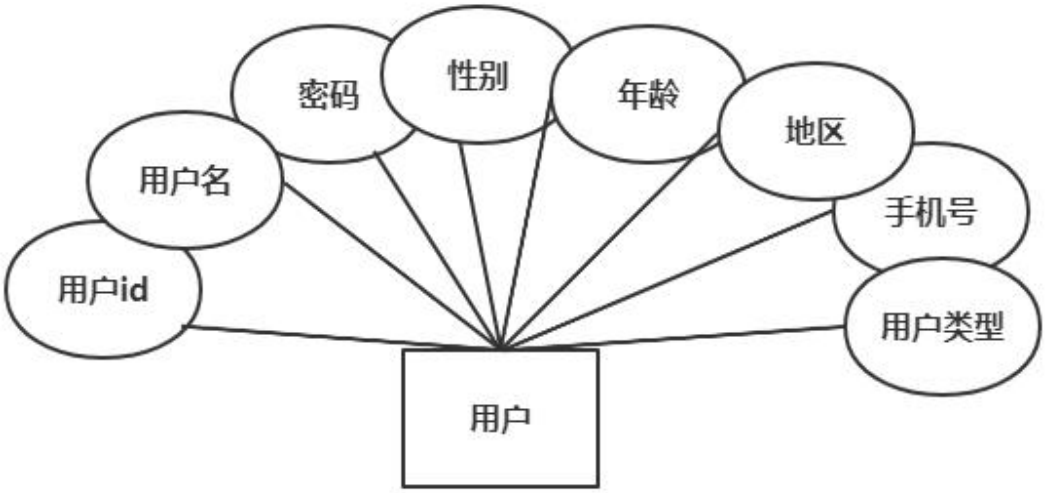

图 2.2 用户实体图

歌曲实体图：属性包括歌曲 id、歌曲名、歌曲 url、歌手、歌曲状态、上传用户。

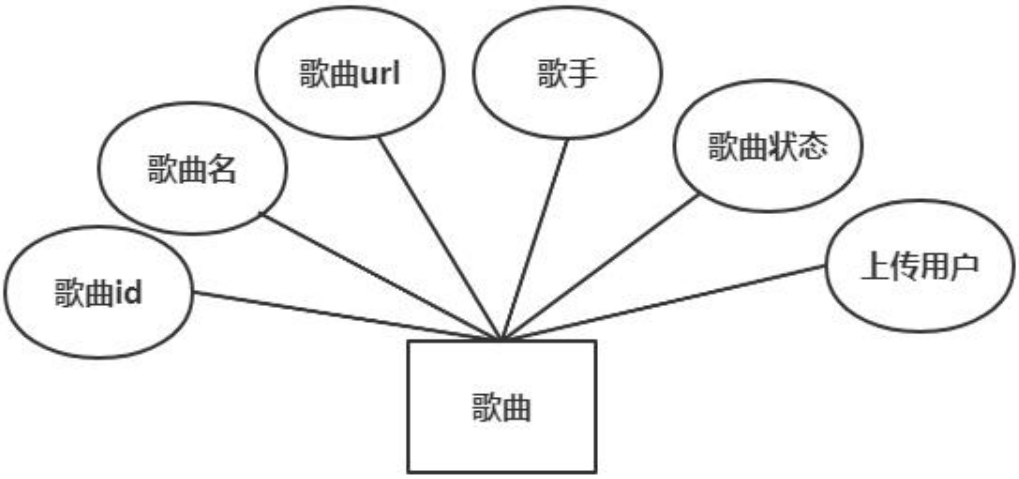

图 2.3 歌曲实体图

歌曲实体图：属性包括歌手 id、歌手名、歌手地区、歌手简介。

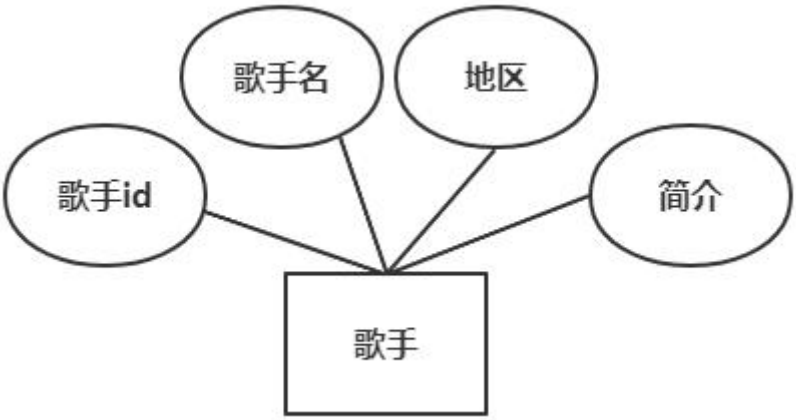

图 2.4 歌手实体图

歌单实体图：属性包括歌单 id、歌单名、创建用户。

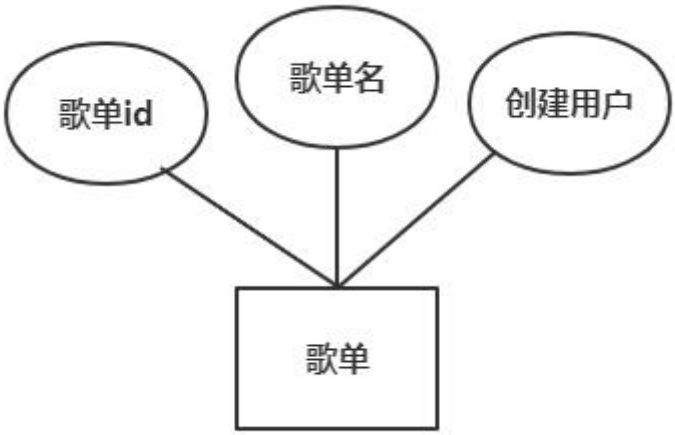

图 2.5 歌单实体图

评论实体图：属性包括评论 id、评论内容、发表用户。

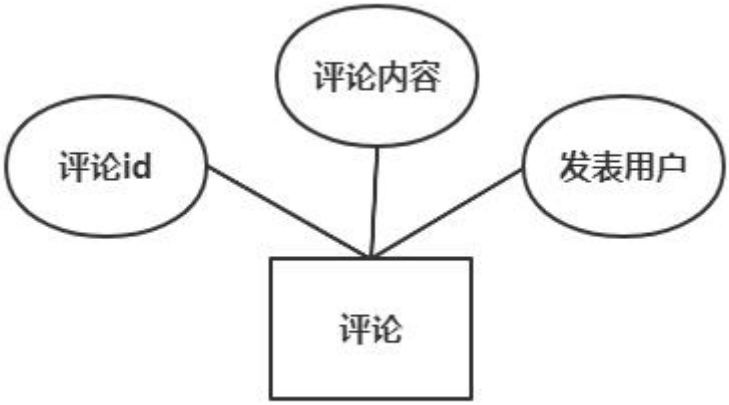

图 2.6 评论实体图

确定以上实体之间的联系，汇总出一个完整的 E-R 图如下：

其中一个用户上传多首歌曲，一个用户创建多个歌单，一个用户发表多条评论，一首歌曲包含多条评论，一个歌手创作多首歌曲，歌曲和歌单是多对多的关系。

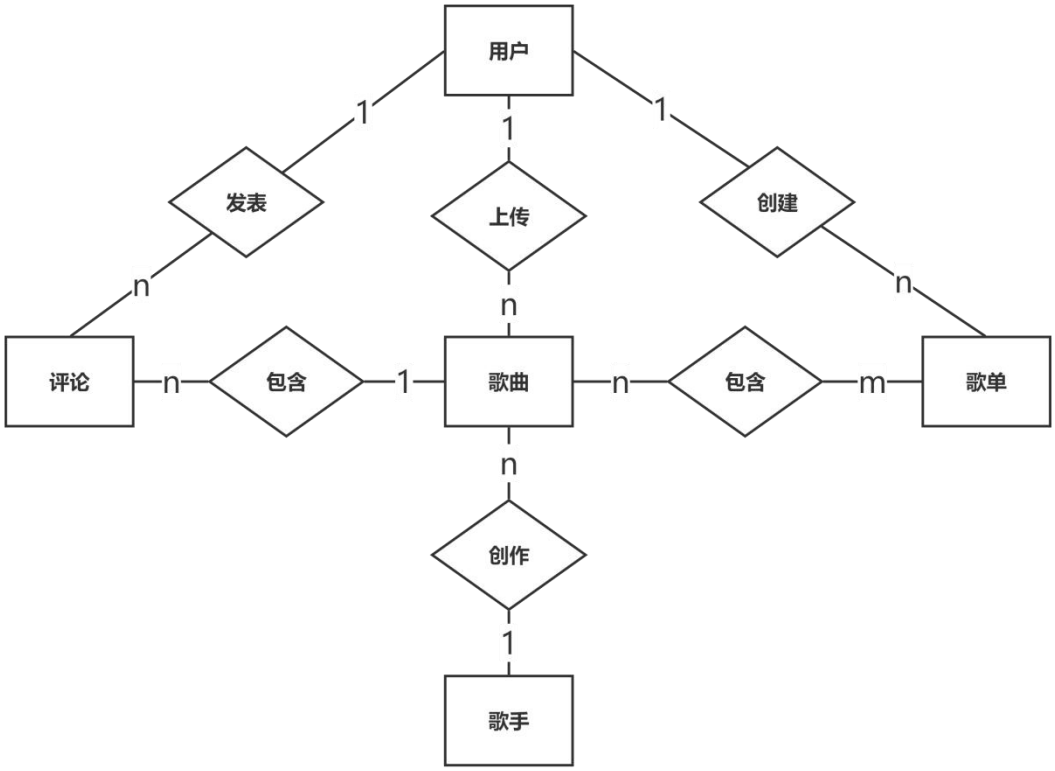

图 2.7 E-R 图

2.4.2 数据库逻辑设计

逻辑结构设计是将概念结构转换为某一数据模型下的逻辑结构，并对其进行优化。在实际的数据库逻辑结构设计时将概念结构的 E-R 模型转换为一系列关系的基本表。关系规范化理论是指导数据库逻辑结构设计的主要理论，是基于关系模型提出，但对其他的数据模型也是适用的。它强调了数据库的逻辑结构要满足一定程度的规范要求，如果达不到一定的规范程度，就会出现不同程度的“异常”情况。提出基于关系模式中数据依赖关系来确定关系模式的规范程度。

数据库逻辑设计需要遵循以下原则：

一个实体型转换为一个关系模式。

对于 1:1 联系：可以转换为一个独立的关系模式，也可以与任意一端对应的关系模式合并。

对于 1:N 联系：可以转换为一个独立的关系模式，也可以与 N 端对应的关系模式合并。

对于 M:N 联系：可以转换为一个独立的关系模式，与该联系相连的各实体的码以及联系本身的属性均转换为关系的属性，关系的码为各实体码的组合。

三个或三个以上实体间的一个多元联系：可以转换为一个独立的关系模式。

具有相同码的关系模式：可合并。

将概要设计阶段形成的 E-R 图转化成逻辑结构

表 2.1 用户表

用户 id	用户名	密码	性别	地区	年龄	手机号	用户类型

表 2.2 歌曲表

歌曲 id	歌曲名	歌曲文件 url	歌手 id	歌曲状态	用户 id

表 2.3 歌单表

歌单 id	歌单名	用户 id	歌曲 id

表 2.4 评论表

评论 id	评论内容	用户 id	歌曲 id

表 2.5 歌手表

歌手 id	歌手名	地区	简介

对上述逻辑表进行分析，发现歌单表和评论表存在非主属性对码的部分依赖，不满足 3NF，继续进行优化，对歌单表和评论表进行分解，效果如下：

表 2.6 用户表

用户 id	用户名	密码	性别	地区	年龄	手机号	用户类型

表 2.7 歌曲表

歌曲 id	歌曲名	歌曲文件 url	歌手 id	歌曲状态	用户 id

表 2.8 歌单表

歌单 id	歌单名	用户 id

表 2.9 歌曲歌单关系表

歌曲 id	歌单 id

表 2.10 评论表

评论 id	评论内容	用户 id

表 2.11 评论歌曲关系表

歌曲 id	评论 id

表 2.12 歌手表

歌手 id	歌手名	地区	简介

以上所有逻辑表均满足 3NF。

2.4.3 数据库物理设计

数据库物理设计是将逻辑设计中的表转化为物理存储表。

数据库中存储表设计如下：

表 2.13 物理存储表

	字段	数据类型	约束条件	复合主键	主键/外键	说明
用户表（user）	u_id	int	auto_increment		主键	用户 id
	u_usernam	varchar(				用户名

	e	15)				
	u_password	varchar(20)				密码
	u_sex	varchar(20)	male or famale			性别
	u_region	varchar(20)				地区
	u_age	int				年龄
	u_phone	char(11)	不能重复,not null			手机号
	u_type	int	1 或 0（1 表示管理员，0 表示用户）			类型
歌曲表（music）	m_id	int	auto_increment		主键	歌曲 id
	m_name	varchar(30)				歌曲名
	m_url	varchar(50)				歌曲文件 url
	m_singer	int			外键	歌手 id
	m_type	int	1 为待审核，0 为正常			歌曲状态
	m_userid	int			外键	用户 id
歌单表（musiclist）	ml_id	int	auto_increment		主键	歌单 id
	ml_name	varchar(20)				歌单名
	ml_userid	int			外键	用户 id
歌曲歌单关系表（music_musiclist）	mml_musicid	int		复合主键	外键	歌曲 id
	mml_listid	int			外键	歌单 id
评论表(comment)	c_id	int	auto_increment		主键	评论 id
	c_content	varchar(				评论内

		200)				容
	c_userid	int			外键	用户 id
歌曲评论关系表（ music_comment)	mc_musicid	int		复 合主键	外键	歌曲 id
	mc_commentid	int			外键	评论 id
歌手表（singer)	s_id	int	auto_increment		主键	歌手 id
	s_name	varchar(30)				歌手名
	s_region	varchar(30)				歌手地区
	s_intro	varchar(200)				歌手介绍

2.5 信息分类编码设计

用户标识

8 位数字

1 位用户初始身分编码：1 用户，2 管理员

7 位特殊标识：自增唯一序列

歌手标识

13 位字符

10 位：姓名英文，中文化为拼音，取前 10 位字符，空值等同于 ’0’

3 位：特殊编码，取字符串形数字编码，为自增序列。

歌曲标识

13 位标识

1 位：上传者区分：1 管理人员上传 2 用户上传

1 位：文件大小区分：1 小歌曲（<1M）2 普通歌曲（1M<且<10M） 3 大歌曲（10M<且<100M）4 常规大小之外歌曲（>100M）

1 位：上传时是否有歌手 1 有歌手信息 2 无歌手信息

10 位：特殊标识符，为自增序列

歌单标识

9 位数字

1 位：创建人员区分 1 用户 2 管理员 3 其他

8 位：自增特殊标识

第三章 系统实现

3.1 功能实现与界面设计

注册登录

图 3.1 注册登录界面

图 3.2　用户界面

图 3.3　用户界面

管理员界面，分为音乐管理，歌单管理，用户管理，歌手管理。点击右侧功能键，打开修改界面。

图 3.4　修改歌手信息界面

播放界面

图 3.5　歌曲播放界面

用户主界面

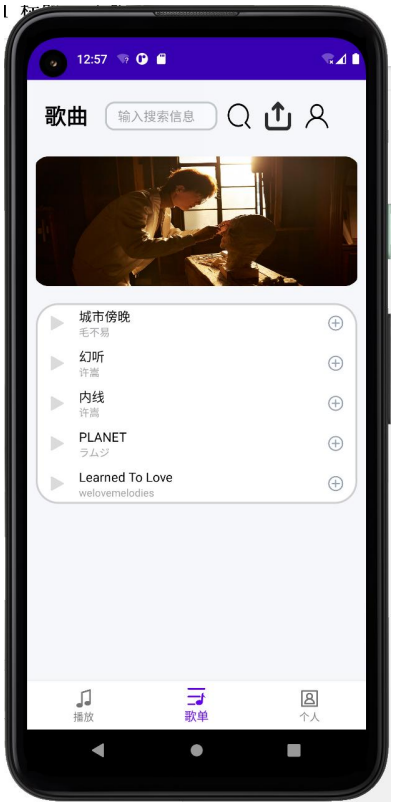

图 3.6　用户主界面

上传界面

图 3.7　歌曲上传界面

搜索页面

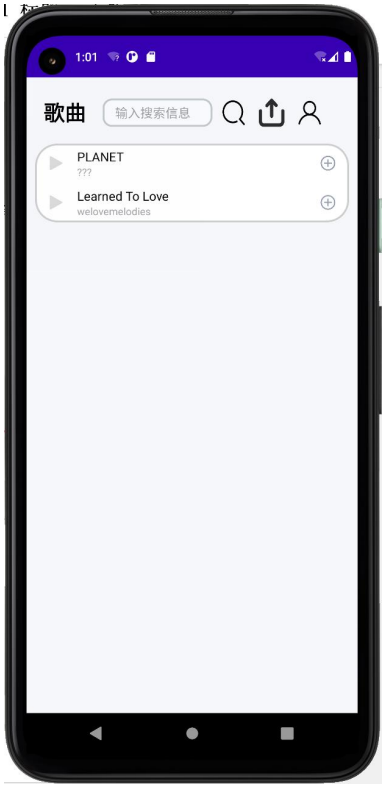

图 3.8　歌曲搜索界面

3.2 物理数据库建立

在 ms sql server 中运行以下脚本，完成数据库和各个表的创建。

```
-- 创建名为 MusicApp 的数据库
CREATE DATABASE MusicApp
GO
USE MusicApp
GO
-- 创建 User 表
CREATE TABLE [User]
(
    u_id INTEGER    IDENTITY(1,1),
    u_username varchar(15),
    u_password varchar(20),
    u_sex CHAR(2)    CHECK(u_sex='男' OR u_sex='女'),
    u_age INTEGER,
    u_phone char(11) unique,
    u_region varchar(100),
    u_like varchar(100),
    u_type INTEGER CHECK(u_type=1 OR u_type=0),
    primary key(u_id),
)
-- 创建 Singer 表
CREATE TABLE Singer
(
    s_idINTEGER    IDENTITY(1,1),
    s_name  varchar(30),
    s_region varchar(30),
    s_intro   varchar(200),
    primary key(s_id),
)
-- 创建 Music 表
CREATE TABLE Music
```

```
(
    m_id    INTEGER   IDENTITY(1,1),
    m_namevarchar(30),
    m_url   varchar(50),
    m_singer    INTEGER,
    m_type  INTEGER CHECK(m_type=1 OR m_type=0),
    m_userid INTEGER,
    primary key(m_id),
    FOREIGN KEY(m_userid) REFERENCES [User](u_id),
    FOREIGN KEY(m_singer) REFERENCES Singer(s_id),
)
-- 创建 Musiclist 表
CREATE TABLE Musiclist
(
    ml_id INTEGER   IDENTITY(1,1),
    ml_name varchar(20),
    ml_userid INTEGER,
    ml_imgurl varchar(200),
    primary key(ml_id),
    FOREIGN KEY(ml_userid) REFERENCES [User](u_id),
)
-- 创建 Music_musiclist 表
CREATE TABLE Music_musiclist
(
    mml_musicid    INTEGER ,
    mml_listid   INTEGER ,
    primary key(mml_musicid,mml_listid),
    FOREIGN KEY(mml_musicid) REFERENCES Music(m_id),
    FOREIGN KEY(mml_listid) REFERENCES Musiclist(ml_id),
)
-- 创建 Comment 表
CREATE TABLE Comment
(
    c_idINTEGER   IDENTITY(1,1),
```

```
    c_content     varchar(200),
    c_userid INTEGER ,
    primary key(c_id),
    FOREIGN KEY(c_userid) REFERENCES [User](u_id),
)
-- 创建 Music_commment 表
CREATE TABLE Music_commment
(
    mc_musicid INTEGER,
    mc_commentid INTEGER,
    primary key(mc_musicid,mc_commentid),
    FOREIGN KEY(mc_musicid) REFERENCES Music(m_id),
    FOREIGN KEY(mc_commentid) REFERENCES Comment(c_id),
)
```

运行以上脚本之后的效果截图如下：

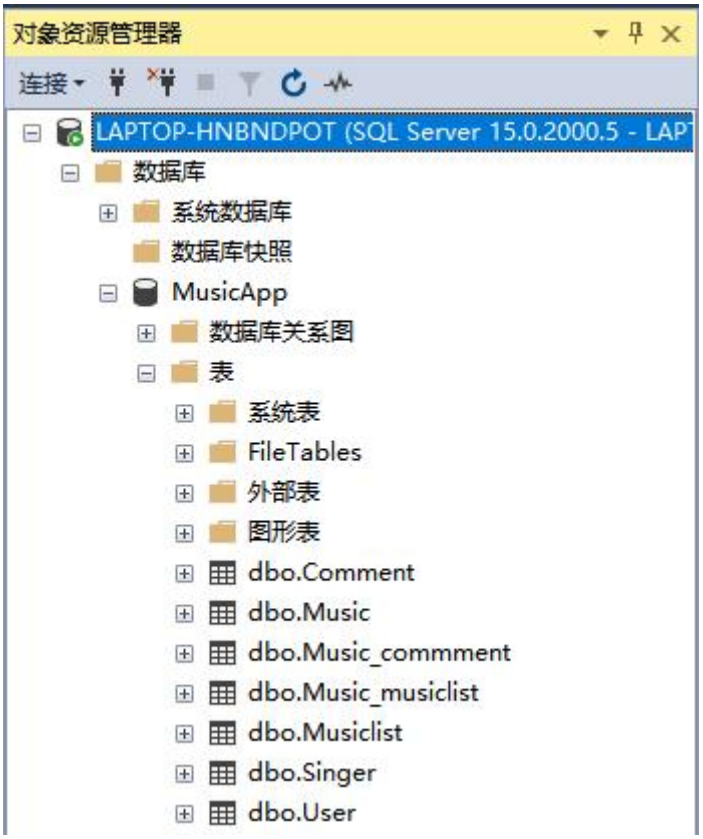

图 3.9 数据库建立图

生成的数据库关系图如下：

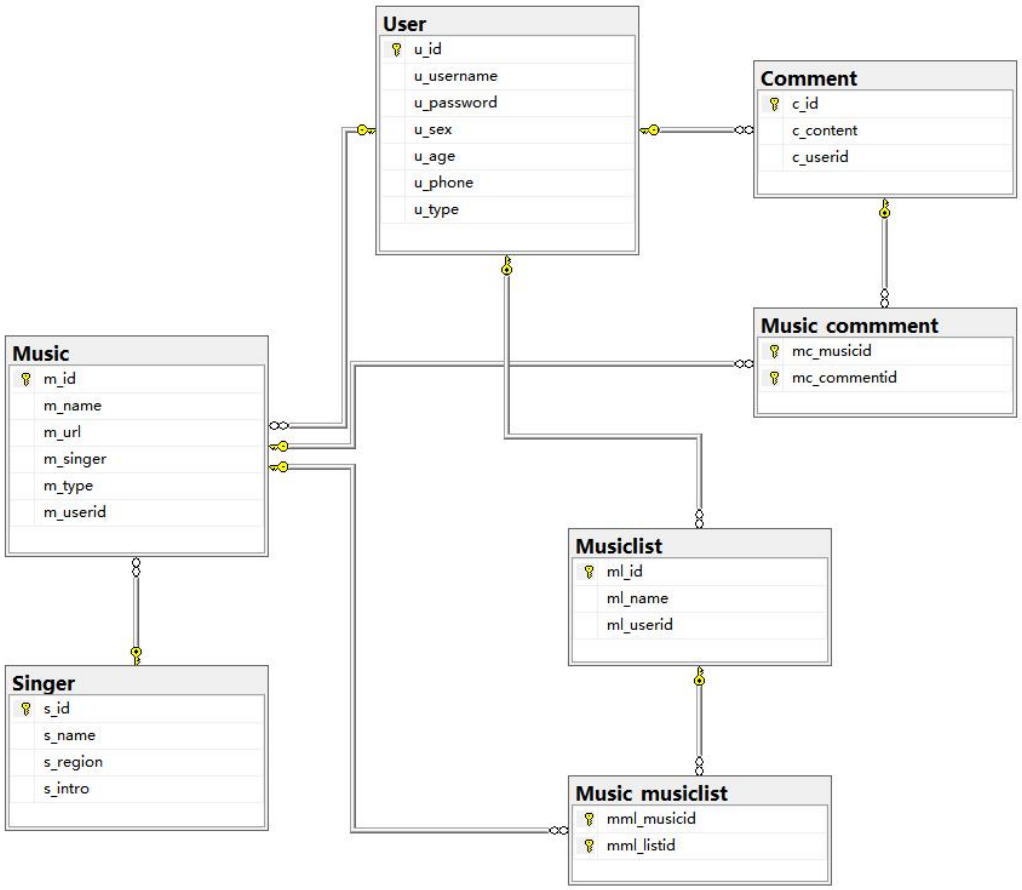

图 3.10 数据库关系图

3.3 程序编码规范

1. 关键词和操作符之间加适当的空格。
2. 相对独立的程序块与块之间加空行
3. 较长的语句、表达式等要分成多行书写。
4. 划分出的新行要进行适应的缩进，使排版整齐，语句可读。
5. 长表达式要在低优先级操作符处划分新行，操作符放在新行之首。
6. 注释要简单明了。
7. 边写代码边注释，修改代码同时修改相应的注释，以保证注释与代码的一致性。
8. 在必要的地方注释，注释量要适中。注释的内容要清楚、明了，含义准确。

第四章 总结

我们小组用了两周时间完成了这次的课程设计，工作量很大，但是很有收获。我们完成了信息系统设计的整体流程，并制作出了一个原型系统。这次的课程设计综合了之前所学的《数据库原理》、《SQL Server》、《信息系统分析与设计》等知识，不仅检验了我们所学习的知识，也培养了我们如何去把握一件事情，如何去做一件事情，又如何完成一件事情。

在团队合作方面，在一开始，我们就进行了分工，我们制作了分工表，每个人负责一部分任务，最后进行整合，在这个过程中，我们锻炼了团队合作的能力，由于第一次合作，一开始有些地方配合的还不是很好，比如安卓开发的同步问题，后来我们决定使用 git 进行版本的控制和同步，达到了很好的效果，大大提升了开发效率，在团队合作中组员之间的沟通是必不可少的，当遇到问题时，沟通是解决问题最有效的方法，我们小组每隔一两天就进行一次汇报工作，汇报自己的进展和遇到的问题，平时我们使用微信群和 qq 群进行实时沟通，遇到棘手的问题，大家一起想办法解决。通过这次实践，我们理解了团队合作和团队管理的重要性，合理的分工，有效的配合，能够达到 1 加 1 大于 2 的效果。我们制定了合理的分工计划和沟通计划，四位组员各司其职，紧密合作，使得效率最大化。

在知识方面，我们进行了一遍信息系统整体的设计，包括系统分析（需求分析、业务分析、数据流图分析、功能分析图分析、数据字典设计、数据加工处理分析）、系统设计（开发环境设计、功能结构图设计、输入/输出设计、数据库设计、信息分类编码设计）、系统实现（功能实现、界面设计、物理数据库建立、程序编码规范设计）等多个模块。通过这次实践，我们复习了书本上学习过的知识，也学到了很多书本上学不到的知识，将知识和实践相结合，对信息系统设计的过程有了更深刻的理解。

在实现的过程中，我们也遇到了很多问题，经过讨论和查找资料，我们最终顺利完成了本次课程设计。比如在数据库的开发过程中，我们设置了一个自增的主键，但是一直无法实现这个效果，后来我们查阅资料发现 SQL Server 和 MySql 的自增语法不一样，SQL Server 是 identity(1,1)，而 MySql 是 auto_increment，更正错误后，问题就迎刃而解了。我们团队多次对于业务图、数据流图、ER 图进行修改，也一步步发现了新的问题，不断进行改进和完善。

课程设计是我们专业课程知识综合应用的实践训练，这也是我们迈向社会，从事职业工作前一个必不少的过程。“千里之行始于足下”，通过这次课程设计，我们深深体会到这句千古名言的真正含义。这次小组分工配合，让我们深刻认识到，我们现在能够认真地进行课程设计，学会脚踏实地迈开这一步，就是在为明

天能够稳健地在社会大潮中奔跑打下坚实的基础。在这次设计过程中，体现出我们整体设计并不断完善一个系统的能力以及综合运用知识的能力，能够学以致用，感受收获劳动成果的喜悦，同时，能够从中发现我们平时学习的不足和薄弱环节，从而加以弥补。

在此也郑重感谢我们的谢老师，正是谢老师无论在以前，还是在现在，严谨细致、一丝不苟的作风一直是我们工作、学习中的榜样，老师循循善诱的教导和不拘一格的思路同样给予我们无尽的启迪。这次课程设计的每个实验细节和每个数据，都离不开老师您的细心教诲与指导，使我们能够很顺利的完成了这次课程设计。