数据库设计说明书

|  |  |  |  |
| --- | --- | --- | --- |
| 版本号 | 提交人 | 修改时间 | 修改内容 |
| V0.1 | 马骏 | 2020/4/13 | 设定说明书大体框架 |
| V0.2 | 马骏、蔡泽华、朱紫阳、王力杰 | 2020/4/14 | 完善说明书的大部分内容 |
| V0.3 | 蔡泽华 | 2020/4/18 | 改变了使用的SQL软件，完善对应内容,改变TimeTable与Course表的内容 |
| V0.4 | 林涛、蔡泽华 | 2020/4/19 | 完善数据库安全相关内容 |

# 第一章 引言

**1.1编写目的**

* 此数据库说明书是juice软件的数据库模块设计，通过需求分析后的原型设计里的结构进行编写。
* 此数据库的设计是为了以后编码、测试以及维护阶段的后台数据的存储做准备。应用于系统开发前期，为后期数据库设计指引方向。

**1.2撰写背景**

* 数据库系统：SQLite
* 开发团队：happy\_tree\_friends
* 开发目的：为学生开发的一款个人课表、指纹签到信息统一展现、界面美观、可读性高的APP。

解决学生在上课时间、指纹签到结束时间即将到来，能快捷方便的查看信息。

**1.3预期读者**

* 系统开发人员：开发人员可以根据本文档了解数据库的框架构成。
* 系统测试人员：针对数据库进行功能性测试。
* 系统维护人员：根据数据库框架进行维护。

**1.4术语解释**

|  |  |  |
| --- | --- | --- |
| 序号 | 名称 | 定义 |
| 1 | 数据库 | 用来保存系统数据的软件 |
| 2 | 表 | 数据库中由不同列汇总成的集合 |
| 3 | 列 | 表中一个单独的属性 |
| 4 | 属性 | 实体或者联系所具有的性质 |
| 5 | 实体 | 客观世界中存在的且可互相区别的事物 |
| 6 | 联系 | 客观事物之间的关系 |
| 7 | Java | 一门汇编语言  是面向对象的程序设计语言 |
| 8 | SQLite | 一款轻型的数据库  是遵守ACID的关系型数据库管理系统 |
| 10 | S[QLCipher](user_cancel) | 可运行在安卓平台的数据库加密框架 |
| 11 | ROOM | Android Jetpack架构组件，在SQLite 的基础上提供了一个抽象层 |

**1.5参考资料**

1. 《软件工程》第八版 作者：罗杰 S.普莱斯曼
2. MSD-OA-DES-DataBase 软件开发有限公司《数据库设计说明书》

# 第二章 外部设计

**2.1标识符和状态**

* 数据库软件的名称：SQLite
* 数据库的名称为：juice

|  |  |  |
| --- | --- | --- |
| 表名 | 标识符名称 | 具体功能 |
| Student | stuID | 存入学号等用户信息 |
| CheckIn | stuID | 存放签到系统的信息 |
| stuCourseItem | couID | 存放学生的课程列表 |
| Course | couID | 存放课程的信息 |

**2.2命名约定**

* 数据库里的列名，表名均遵循驼峰命名法则
* 各表都有唯一的主键字段（连接表除外）

**2.3设计约定**

* ER图的设计使用的是ProcessOn工具进行设计
* 数据库模型设计使用了PowerDesigner工具进行设计
* 数据库具体的设计使用了SQLiteBrowser与Navicat Premium
* 数据库的设计遵循了数据库设计第三范式

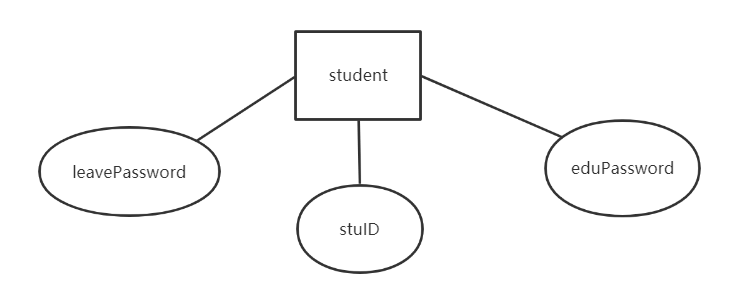
# 第三章 结构设计

**3.1概念结构设计**

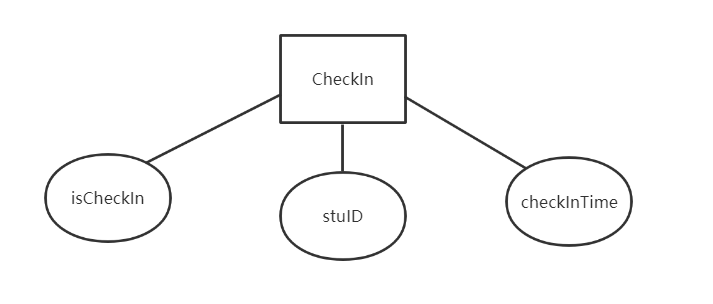
**3.1.1 实体和属性的定义**

* 一个学生拥有一张确认签到的表
* 一个学生拥有一张课程表
* 一个课程表对应数张单个课程的具体描述表
* 每一张表拥有数种列

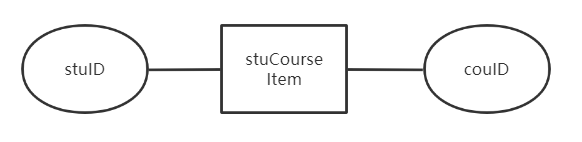
**3.1.2 设计局部ER模式**



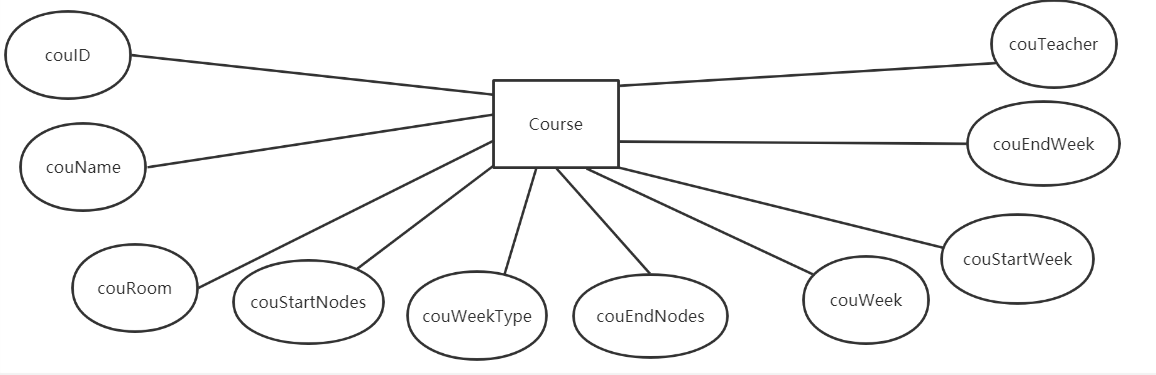
1.1student表ER图



1.2CheckIn表ER图

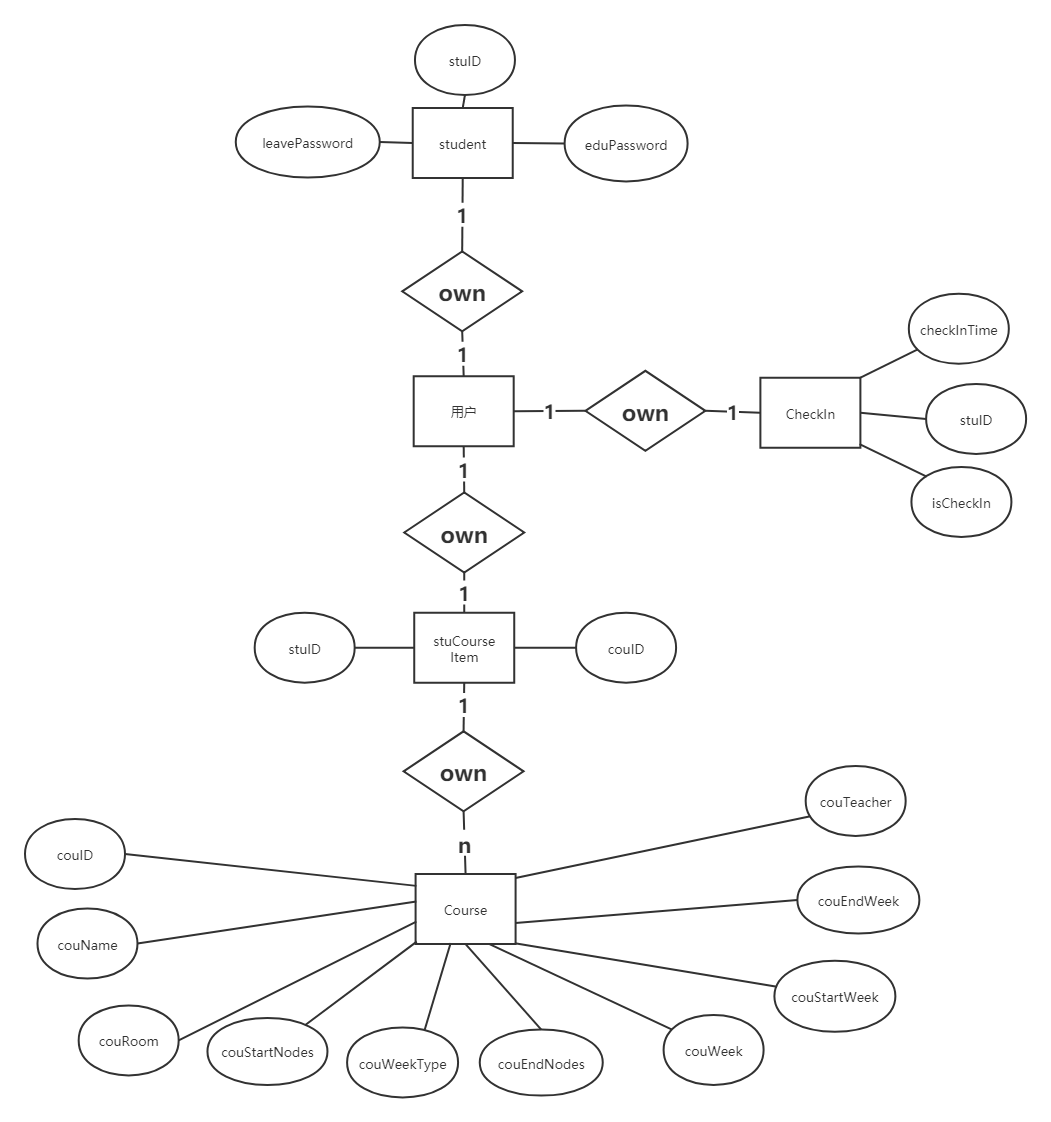


1.3stuCourseItem表ER图



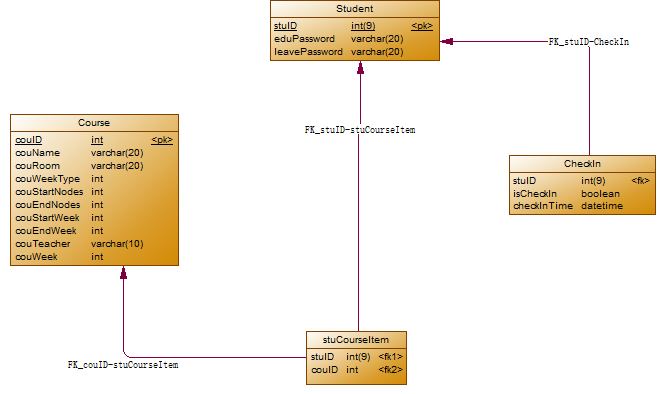
1.4Course表ER图

**3.1.3 设计全局ER模式**



1.5总体ER图

**3.2 逻辑结构设计**



1.6数据库逻辑结构图

**表 Student**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Code | Data Type | Length | Primary | description |
| eduPassword | eduPassword | varchar(20) | 20 |  | 请假系统的密码 |
| leavePassword | leavePassword | varchar(20) | 20 |  | 教育网密码 |
| stuID | stuID | int(9) | 9 | X | 学生的学号 |

# **表 CheckIn**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Code | Data Type | Length | Primary | description |
| checkInTime | checkInTime | datetime |  |  | 签到的具体时间 |
| isCheckIn | isCheckIn | boolean | 9 |  | 是否签到 |
| stuID | stuID | int(9) |  | X | 学生学号 |
| 备注：stuID 外键 -> 连接 student表 | | | | | |

# **表 stuCourseItem**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Code | Data Type | Length | Primary | description |
| stuID | stuID | int(9) | 9 |  | 学生学号 |
| couID | couID | int |  |  | 课程专门的ID |
| 备注：stuID 外键 ->连接Student  couID 外键 ->连接Course | | | | | |

# **表 Course**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Code | Data Type | Length | Primary | description |
| couID | couID | int |  | X | 课程的ID |
| couName | couName | varchar(20) | 20 |  | 课程名 |
| couRoom | couRoom | varchar(20) | 20 |  | 本课程所使用的教室 |
| couStart  Nodes | couStart  Nodes | int |  |  | 这门课是从当天的  第几节课开始 |
| couEndNodes | couEndNodes | int |  |  | 这门课是从当天的  第几节课结束 |
| couStartWeek | couStart  Week | int |  |  | 这门课程开始于第几周 |
| couEnd  Week | couEnd  Week | int |  |  | 这门课程结束于第几周 |
| couTeacher | couTeacher | varchar(10) | 10 |  | 授课老师 |
| couWeek | couWeek | int |  |  | 这门课是在星期几上 |
| couWeek  Type | couWeek  Type | int |  |  | 单双周的判断 |

# 第四章 运用设计

**4.1 数据字典设计**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | | | **Student** | | | |
| 主键 | | | | stuID | | | |
| 序号 | 字段名称 | 数据类型  （精度范围） | 允许为空Y/N | 唯一  Y/N | 长度 | 默认值 | 约束条件/说明 |
| 1 | edu  Password | varchar(20) | N |  | 20 |  | 教育网密码 |
| 2 | leave  Password | varchar(20) | N |  | 20 |  | 请假系统的密码 |
| 3 | stuID | int(9) | N | Y | 9 |  | Primary Key/学生的学号 |
| 备注 | | | | **功能：存储学生信息** | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | | | **CheckIn** | | | |
| 主键 | | | | stuID | | | |
| 序号 | 字段名称 | 数据类型  （精度范围） | 允许为空Y/N | 唯一  Y/N | 长度 | 默认值 | 约束条件/说明 |
| 1 | checkIn  Time | datetime | Y |  |  |  | 签到时间 |
| 2 | isCheck  In | boolean | Y |  |  |  | 签到状态 |
| 3 | stuID | int(9) | N | Y | 9 |  | Foreign Key /  学生学号 |
| 备注 | | | | **功能：查看签到信息** | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | | | **stuCourseItem** | | | |
| 主键 | | | | 无 | | | |
| 序号 | 字段名称 | 数据类型  （精度范围） | 允许为空  Y/N | 唯一  Y/N | 长度 | 默认值 | 约束条件/说明 |
| 1 | stuID | int(9) | N |  | 9 |  | Foreign Key/  学生学号 |
| 2 | couID | int | N | Y |  |  | Foreign Key/  课程专门的ID |
| 备注 | | | | **功能：连接Student表与Course表** | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 表名 | | | | **Course** | | | |
| 主键 | | | | couID | | | |
| 序号 | 字段名称 | 数据类型  （精度范围） | 允许为空  Y/N | 唯一  Y/N | 长度 | 默认值 | 约束条件/说明 |
| 1 | couID | int | N | Y |  |  | Primary Key/  每门课程的ID |
| 2 | couName | varchar(20) | N |  | 20 |  | 课程名 |
| 3 | couRoom | varchar(20) | N |  | 20 |  | 课程使用的  教室 |
| 4 | couStart  Nodes | int | N |  |  |  | 这门课是  从当天的  第几节课开始 |
| 5 | couEnd  Nodes | int | N |  |  |  | 这门课是  从当天的  第几节课结束 |
| 6 | couStart  Week | int | N |  |  |  | 这门课程  开始于第几周 |
| 7 | couEnd  Week | int | N |  |  |  | 这门课程  结束于第几周 |
| 8 | cou  Teacher | varchar(10) | N |  | 10 |  | 授课老师 |
| 9 | couWeek | int | N |  |  |  | 这门课是  在星期几上 |
| 10 | couWeek  Type | int | N |  |  |  | 单双周的判断 |
| 备注 | | | | **功能：存储课程信息** | | | |

**4.2 系统安全和权限设计**

**4.2.1系统安全**

不同于常见的客户-服务器范例，SQLite引擎不是个程序与之通信的独立进程，而是连接到程序中成为它的一个主要部分。整个数据库(定义、表、索引和数据本身)都在宿主主机上存储在一个单一的文件中。使用SQLite来存储数据，数据库文件将存储在应用私有目录，其他应用无权访问，可以保证数据安全性。但是一些Android手机获取了Root权限，这可能导致数据库文件被非法获取，可以读取到数据库中存储的数据。如果数据库中存储着用户的敏感数据，我们的程序就会面临严重的安全漏洞隐患。为了解决这个问题，我们会将用户的敏感数据加密后再写入数据库，以保证用户敏感数据不会泄露。

**4.2.2数据库权限设计**

Android系统内置了SQLite数据库，并且提供了一整套的API用于对数据库进行增删改查操作。

做为一个嵌入式的数据库，安卓SQLite只能由对应的应用访问，每个应用都只能访问自身的数据库。

**4.3 数据库实现**

|  |
| --- |
| /\*  Navicat Premium Data Transfer  Source Server : sqlite  Source Server Type : SQLite  Source Server Version : 3021000  Source Schema : main  Target Server Type : SQLite  Target Server Version : 3021000  File Encoding : 65001  Date: 18/04/2020 15:28:02  \*/  PRAGMA foreign\_keys = false;  -- ----------------------------  -- Table structure for CheckIn  -- ----------------------------  DROP TABLE IF EXISTS "CheckIn";  CREATE TABLE "CheckIn" (  "stuID" int(9) NOT NULL,  "isCheckIn" boolean,  "checkInTime" datetime,  PRIMARY KEY ("stuID"),  CONSTRAINT "FK\_stuID-CheckIn" FOREIGN KEY ("stuID") REFERENCES "Student" ("stuID") ON DELETE RESTRICT ON UPDATE RESTRICT  );  -- ----------------------------  -- Table structure for Course  -- ----------------------------  DROP TABLE IF EXISTS "Course";  CREATE TABLE "Course" (  "couID" int NOT NULL,  "couName" varchar(20) NOT NULL,  "couRoom" varchar(20) NOT NULL,  "couTeacher" varchar(10) NOT NULL,  "couWeek" integer NOT NULL,  "couWeekType" int NOT NULL,  "couStartWeek" integer NOT NULL,  "couEndWeek" integer NOT NULL,  "couStartNodes" integer NOT NULL,  "couEndNodes" integer NOT NULL,  PRIMARY KEY ("couID")  );  -- ----------------------------  -- Table structure for Student  -- ----------------------------  DROP TABLE IF EXISTS "Student";  CREATE TABLE "Student" (  "stuID" int(9) NOT NULL,  "eduPassword" varchar(20) NOT NULL,  "leavePassword" varchar(20) NOT NULL,  PRIMARY KEY ("stuID")  );  -- ----------------------------  -- Table structure for sqlite\_user  -- ----------------------------  DROP TABLE IF EXISTS "sqlite\_user";  CREATE TABLE "sqlite\_user" (  "uname" TEXT,  "isAdmin" BOOLEAN,  "pw" BLOB,  PRIMARY KEY ("uname")  )  WITHOUT ROWID;  -- ----------------------------  -- Table structure for stuCourseItem  -- ----------------------------  DROP TABLE IF EXISTS "stuCourseItem";  CREATE TABLE "stuCourseItem" (  "stuID" integer(9) NOT NULL,  "couID" int NOT NULL,  CONSTRAINT "FK\_stuID-stuCourseItem" FOREIGN KEY ("stuID") REFERENCES "Student" ("stuID") ON DELETE CASCADE ON UPDATE CASCADE,  CONSTRAINT "FK\_stuID-stuCourseItem  FK\_couID-stuCourseItem" FOREIGN KEY ("couID") REFERENCES "Course" ("couID") ON DELETE CASCADE ON UPDATE CASCADE  );  -- ----------------------------  -- View structure for stu\_view  -- ----------------------------  DROP VIEW IF EXISTS "stu\_view";  CREATE VIEW "stu\_view" AS SELECT  stuCourseItem.stuID,  stuCourseItem.couID,  Course.couName  FROM  stuCourseItem  INNER JOIN Course ON stuCourseItem.couID = Course.couID;  PRAGMA foreign\_keys = true; |