# 宅单词

# 数据库设计说明书

**组员**：郑斌、陈一聪、张玉麟，王少滨
周宇靖、叶先锻、李定成、王弘毅、杨明伟

**指导教师**：单红

云玩家 编写

2020.04.11

# 目录

# 第一章 引言

## 1.1 编写目的

- 本数据库设计说明书是关于宅单词数据库设计，主要包括数据逻辑结构设计、数据字典以及运行环境、安全设计等。
- 本数据库设计说明书读者：用户、系统设计人员、系统测试人员、系统维护人员。
- 本数据库说明书是根据系统需求分析设计所编写的。
- 本系统说明书为开发软件提供了一定的基础。

## 1.2 背景

中国在线教育 APP 月独立设备总数持续增长，截止到 2019 年 6 月达 4.1 亿台；搜题、批改作业等作业类产品（作业帮、快对作业、小猿搜题等）和背单词、查词翻译等单词类产品（有道词典、百词斩 等）从 2012 年起就一直是在线教育两大流量之王，艾瑞咨询数据显示，2019Q3，在线教育领域流量最大的 APP 依然是作 业帮、有道词典、快对作业等作业/单词类产品。2019 年中国在线教育市场规模预计达 3133.6 亿元，同比增长 24.5%，预计未来 3 年市场规模增速保持 在 18-21%之间，增速持续降低但增长势头保持稳健。用户对在线教育的接受度不断提升、在线付费意识逐渐养成以及线 上学习体验和效果的提升是在线教育市场规模持续增长的主要原因。

## 1.3 参考资料

a) 软件工程导论

b) 数据库系统概论

c) Baidu

# 第二章 外部设计

## 2.1 标识符和状态

数据库软件的名称：mysql community 5.7.21
数据库的名称：neet_word

## 2.1 命名约定

由于 mysql 表名是否区分大小写因数据库配置而异，因此这里规定，同意使用下划线分割单词，而不是使用驼峰法。这样的好处是能够在无法使用驼峰法命名表名时，可以使名称更加直观，数据库表名应该有意义，并且易于理解，最好使用可以表达功能的英文单词或缩写，如果用英文单词表示，建议使用完整的英文单词，表名不可以太长，最好不要超过 3 个英文单词长度（22 个字母）。

数据库表名统一使用 t_开头，视图使用 v_开头，外键约束则用”fk_当前表名_关联表名”来约束外键的命名规范。对于触发器，可以使用驼峰法命名，且以动词+名词的形式说明触发器的作用。

对于表中的列而言，统一使用’表名的前两个字母_’后接列名。一样使用下划线来区分单词，同时注意列名不要过长，要适当的控制名字的长短，在表创建完成前，应该为表添加表的注释。

## 2.3 设计约定

1) 所有字段在设计时，除以下数据类型 timestamp、image、datetime、smalldatetime、uniqueidentifier、binary、sql_variant、binary、varbinary 外，必须有默认值。字符型的默认值为一个空字符值串’’；数值型的默认值为数值 0；逻辑型的默认值为数值 0；

   其中：系统中所有逻辑型中数值 0 表示为“假”；数值 1 表示为“真”。

   datetime、smalldatetime 类型的字段没有默认值，必须为 NULL。

2) 当字段定义为字符串形时建议使用 varchar 而不用 nvarchar。

   注：在 MySQL5.0 以上的版本中，varchar 数据类型的长度支持到了 65535，也就是说可以存放 65532 个字节的数据，起始位和结束位占去了 3 个字节。

   默认值为 NULL

3) **字段的描述**

   a. 字段必须填写描述信息（注释）

   b. 尽量遵守第三范式的标准（3NF）

   表内的每一个值只能被表达一次（列名不重复）

   表内的每一行都应当被唯一的标示（标识唯一性，如自动增长主键）

   表内不应该存储依赖于其他键的非键信息

4) **加索引规则**

a) 表建好后数据库自动为表生成一个索引（为自动增长的列生成唯一索引），如果在对这列添加索引，数据库会给一个警告,内容大概是，已经为这列添加了索引，建议修改索引名称和自动增长列名保持一致，为了方便使用。

b) 如果在添加索引时，建议索引名称和数据库列名保持一致，为了方便使用。

c) 如果字段事实上是与其它表的关键字相关联而未设计为外键引用，需建索引

d) 如果字段与其它表的字段相关联，需建索引。

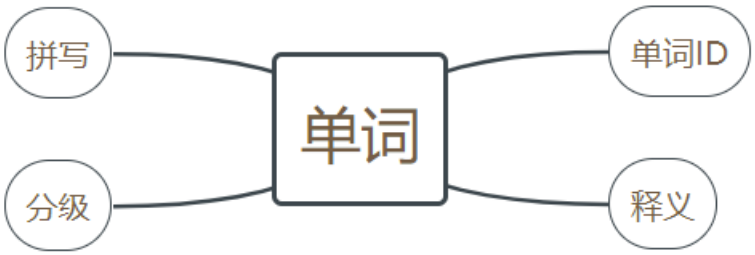
e) 如果字段需做模糊查询之外的条件查询，需建索引。

f) 除了主关键字允许建立簇索引外，其它字段所建索引必须为非簇索引。

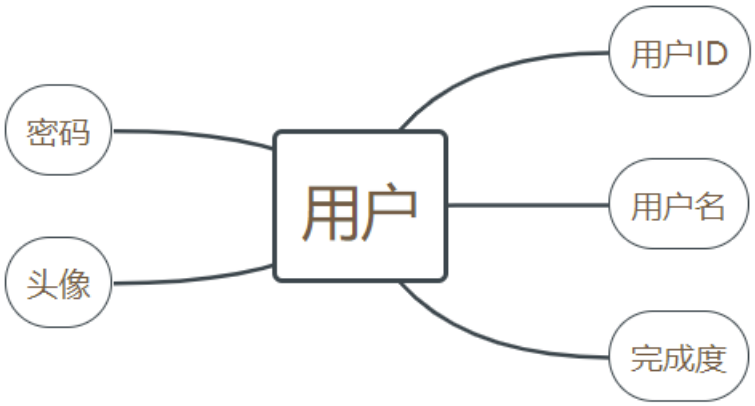
# 第三章 结构设计

## 3.1.1 概念结构设计

**单词：**

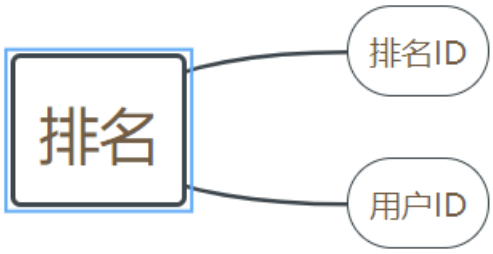



**用户：**

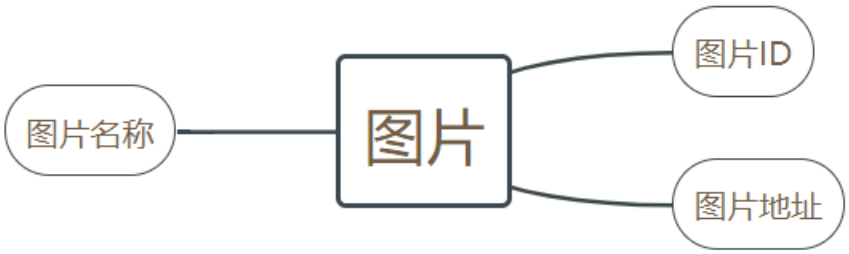



单词本：

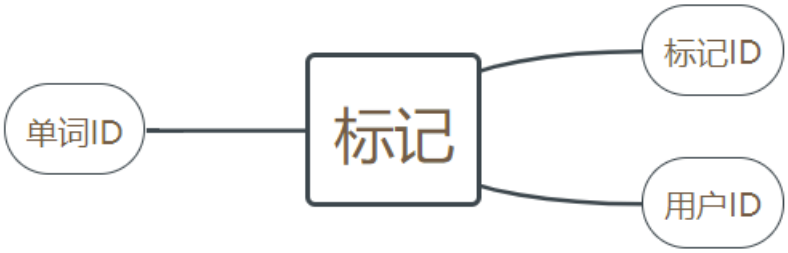

排名：



图片

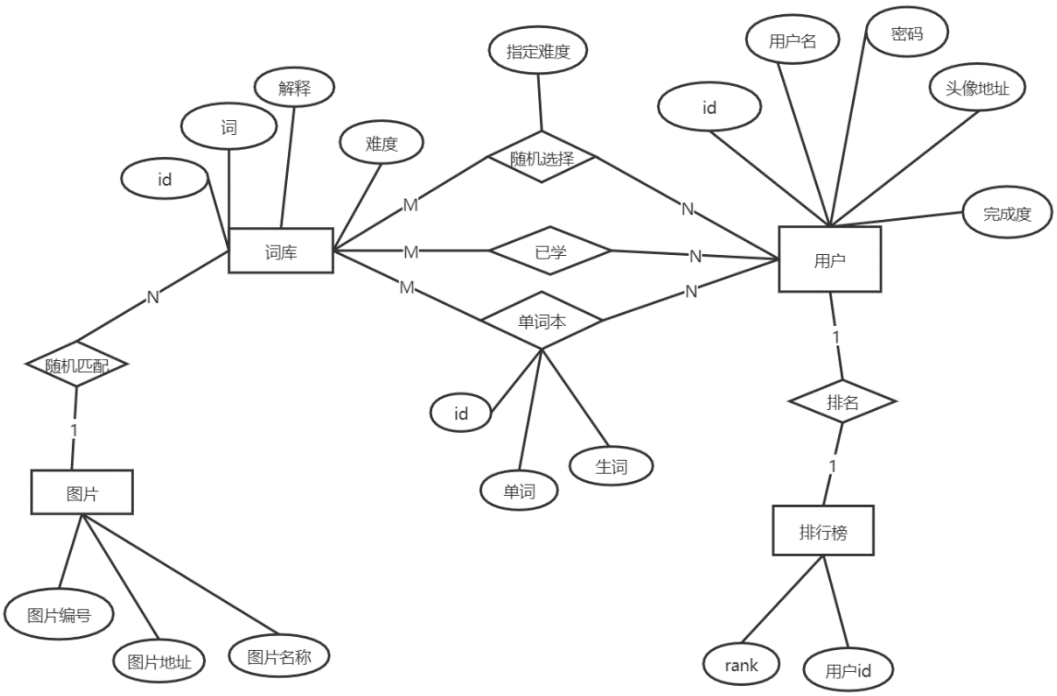



记录：




整体关系：

宅单词1.0 ER图




# 第四章 接口设计

## 4.1 传输方式

HTTPS 传输 端口 443
method: POST
Content-Type: application/json \ application/x-www-form-urlencoded

## 4.2 HTTP Header 说明

| HTTP 头名称 | 字段说明 | 样例 |
|---|---|---|
| Authorization | 签名字段,用来鉴权 | eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.<br>eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.<br>TJVA95OrM7E2cBab30RMHrHDcEfxjoYZgeFONFh7HgQ |
| Content-type | 内容 | application/json |

| 名称 | 字段说明 |
|---|---|
| Authorization | 令牌,使用无状态的 jwt 来生成,通过权限验证来查看用户是否有权限使用当前功能 |

| content-type | 用来标记用户上传的内容,规定此次均为 application/json |
| --- | --- |

## 4.3 返回结构

```
{
    code: 200, //响应代码
    msg: '', //消息,正常时不带此条,出错时附加错误信息
    data: {} //数据
}
```

## 4.4 接口概述

| 类型 | 接口名 | 地址 | 说明 |
| --- | --- | --- | --- |
| 后台接口 | 登录 | /login | 登录接口,使用微信登录 |
| 后台接口 | 获取单词书列表 | /books/list | 获取单词书列表,方便选择 |
| 后台接口 | 下载单词书 | /books/{id} | 下载单词书,重定向至下载页面 |
| 后台接口 | 设置学习计划 | /config | 配置接口统一 |
| 后台接口 | 获取情况 | /data | 获取当前学习状况 |
| 后台接口 | 历史记录 | /record | 获取背诵过的单词的历史记录 |
| 后台接口 | 打卡 | /finish | 每日打卡 |
| 后台接口 | 获取图片 | /images | 获取背景图片 |

## 4.5 接口详解

- 登录 POST /login

  使用微信小程序的自带的登录接口,接入微信的账号系统,同时注册

  返回

```
{
    sessionId: '' //微信返回
}
```

- 获取单词书列表 GET /books/list

  返回列表:

```
response:
{
    code: 200,
    data: {
        [{
            bookId: '',
            title: '',
            icon: ''
        }...]
    }
}
```

- 下载单词书(本地缓存) GET /books/{id}

  根据提供的 id 号来下载对应的单词书,调用方式: https://api.xxx.com/books/1, 后台通过重定向将其定向到保存单词表的 cdn 上

- 修改用户配置 **POST** /config

  通过用户提交的键值对来修改对应的用户属性

```
RequestBody:
{
    countPerDay:100, //每日背诵
    bookId: 123, //单词书
    imageType: '' //见下方
}
ResponseBody:
{
    code: 200,
    data: {}
}
```

- 获取用户配置信息 **GET** /config

```
ResponseBody:
{
    countPerDay:100, //每日背诵
    bookId: 123, //单词书
    imageType: '' //见下方
}
```

- 获取用户信息 GET /data

  获取用户信息

```
{
    code: 200,
    data: {
        bookId: 123,
        bookName: '',
        bookImage: '',
        totalWords: 1000 // 单词书总数
        learnedWords: 1 //手动统计,服务端更新
    }
}
```

- 获取历史记录 GET /record

  获取用户的使用记录

  请求实例：https://api.xxx.com/record?count=xxx&page=xxx（后面的参数为分页需求，具体后续再谈）

```
{
    code: 200,
    data: {
        [{
            wordId: 1,
            word: 'abandon',
            translation: '',
            date: '2020-02-02',
            level: 1 //熟练度
        }, {}, ...]
    }
}
```

- 打卡 POST /finish

  具体调用 https://api.xxx.com/finish?count=xxx

  count 为今日完成数量

- 获取图片类型(列表) GET /images

  具体调用有区分

```
当需要获取类型列表时,使用
https://api.xxx.com/images?type=list
ResponseBody:
{
    code: 200,
    data: {
        [{
            type: '',
            description: ''
        }, ...]
    }
}

当需要获取某种类型时,使用
https://api.xxx.com/images?type=%e4%ba%8c%e6%ac%a1%e5%85%83
ResponseBody:
{
    code: 200,
    data: {
        type: '',
        description: '',
        urls: ['']  //免费图片来源
    }
}
```

# 第五章 数据库设计概述

## 5.1 数据库字典设计

无

## 5.2 安全保密性设计

数据库的安全性在开发时暂时由阿里云的轻量级服务器提供，通过配置网络防火墙以及安全策略组，进出白名单来控制访问。上线后，定制专门的数据库服务器，通过中间服务器来供后端访问。

对数据安全来说，通过定期备份，保证数据不会因服务器的问题而出现损坏。对数据库的读写操作记录 log，确保能够及时发现问题或者攻击。剩下的交给 mysql 原生的数据库故障恢复系统。

使用 binlog2sql，在出现操作需要回滚时，能帮助我们快速回滚数据。数据库备份软件则使用 mysql 原生的备份就行。

对数据完整性，使用第三范式来约束表的建立，并使用外键来建立表与表之间的联系。

## 4.3 数据库实施

### 4.3.1 创建数据库

```
CREATE DATABASE  IF NOT EXISTS `neet_word` /*!40100 DEFAULT CHARACTER S
ET utf8 */;
```

### 4.3.2 创建表

```
USE `neet_word`;
-- MySQL dump 10.13  Distrib 5.7.17, for macos10.12 (x86_64)
--
-- Host: localhost    Database: neet_word
-- ------------------------------------------------------
-- Server version 5.7.21

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_
CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO'
 */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `t_image`
--

DROP TABLE IF EXISTS `t_image`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `t_image` (
  `im_id` int(11) NOT NULL AUTO_INCREMENT,
  `im_name` varchar(255) DEFAULT NULL,
  `im_type` varchar(255) DEFAULT NULL,
  `im_addr` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`im_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;
```

```
--
-- Table structure for table `t_notebook`
--

DROP TABLE IF EXISTS `t_notebook`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `t_notebook` (
  `no_id` int(11) NOT NULL,
  `no_word_id` int(11) DEFAULT NULL,
  `no_user_id` int(11) NOT NULL,
  `no_raw_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`no_id`,`no_user_id`),
  KEY `fk_notebook_user_idx` (`no_user_id`),
  KEY `fk_notebook_word_idx` (`no_word_id`),
  KEY `fk_notebook_word_raw_idx` (`no_raw_id`),
  CONSTRAINT `fk_notebook_user` FOREIGN KEY (`no_user_id`) REFERENCES `
t_user` (`ur_id`),
  CONSTRAINT `fk_notebook_word` FOREIGN KEY (`no_word_id`) REFERENCES `
t_word` (`wo_id`),
  CONSTRAINT `fk_notebook_word_raw` FOREIGN KEY (`no_raw_id`) REFERENCE
S `t_word` (`wo_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `t_rank`
--

DROP TABLE IF EXISTS `t_rank`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `t_rank` (
  `ra_id` int(11) NOT NULL,
  `ra_user_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`ra_id`),
  KEY `fk_rank_user_id_idx` (`ra_user_id`),
  CONSTRAINT `fk_rank_user_id` FOREIGN KEY (`ra_user_id`) REFERENCES `t
_user` (`ur_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `t_record`
```

```
--

DROP TABLE IF EXISTS `t_record`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `t_record` (
  `re_id` int(11) NOT NULL AUTO_INCREMENT,
  `re_word_id` int(11) DEFAULT NULL,
  `re_user_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`re_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `t_user`
--

DROP TABLE IF EXISTS `t_user`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `t_user` (
  `ur_id` int(11) NOT NULL AUTO_INCREMENT,
  `ur_name` varchar(45) DEFAULT NULL,
  `ur_password` varchar(255) DEFAULT NULL,
  `ur_avatar` varchar(255) DEFAULT NULL,
  `ur_complete_rate` float DEFAULT NULL,
  PRIMARY KEY (`ur_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Table structure for table `t_word`
--

DROP TABLE IF EXISTS `t_word`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `t_word` (
  `wo_id` int(11) NOT NULL AUTO_INCREMENT,
  `wo_word` varchar(255) DEFAULT NULL,
  `wo_translate` varchar(255) DEFAULT NULL,
  `wo_level` int(11) DEFAULT NULL,
  PRIMARY KEY (`wo_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

```
/*!40101 SET character_set_client = @saved_cs_client */;
/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;

/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2020-04-13 16:45:52
```