Database Module

Detailed design specification

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Reviser | Instructions |
| V1.0 | 2020-9-13 | 林钰淇 | Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. **Introduction**
   1. **Purpose**

The purpose of this manual is to explain the technical scheme of the overall design of the system to users and relevant personnel of software development. From the point of view of system design, this manual explains the overall architecture, processing flow, module division, function allocation, interface design, operating environment, data structure design and error handling design of the system. To provide an overall description of how the logic and data functions of the program system are implemented throughout the design period, thus serving as a basis for detailed program design and coding. This document will be the core document for the design phase. The applicable readers of this summary design specification are: customers, system architects, system developers, testers.

* 1. **References**

[1]https://www.runoob.com/mysql/mysql-install.html

[2]https://www.cnblogs.com/xfxing/p/9322199.html

[3]http://c.biancheng.net/view/7105.html

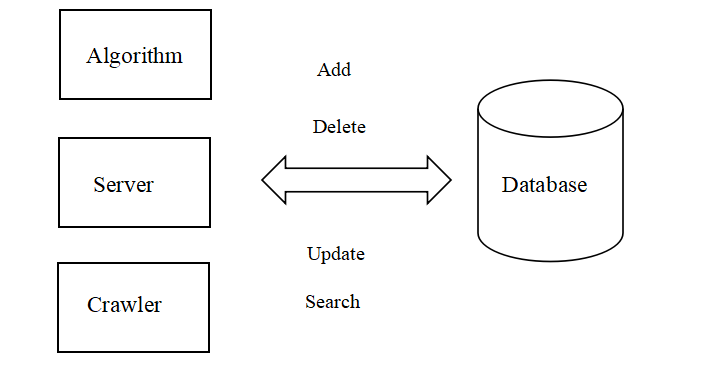
[4]https://www.jb51.net/article/186614.htm

[5]https://www.cnblogs.com/drcoding/p/5329294.html

[6]https://blog.csdn.net/tianxin0928/article/details/80995510?utm\_medium=distribute.pc\_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.channel\_param&depth\_1-utm\_source=distribute.pc\_relevant.none-task-blog-BlogCommendFromMachineLearnPai2-2.channel\_param

1. **The overall design**

The key responsibility of the database module is to establish, organize and maintain the data. For other modules, they can add, update, delete and search for the data if necessary. For instance, we will obtain a series of data from the Crawler Module, and then we will make the transfer job for the future use of the Algorithm Module. The following figure shows the basic relationship between Database Module and other modules.



1. **Detailed design**
   1. **Tables of database module**

UserInfo

|  |  |  |
| --- | --- | --- |
| **user\_name**  **(varchar 20)** | password  (varchar 16) | favorite\_label  (varchar 200) |

SongInfo

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **song\_id**  **(varchar 20)** | song\_name  (varchar 500) | singer  (varchar 20) | song\_sheet\_id\_belonged  (varchar 200) | label  (varchar 200) |

SongSheetInfo

|  |  |  |
| --- | --- | --- |
| **song\_sheet\_id**  **(varchar 20)** | song\_sheet\_name  (varchar 200) | label  (varchar 200) |

UserSongSheet

|  |  |  |
| --- | --- | --- |
| **user\_name**  **(varchar 20)** | **song\_id**  **(varchar 20)** | last\_click\_time  (datetime) |

* 1. **Functions of database module**

UserInfo Table

**1. def addUser(self, user\_name, passwd,label):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**2. def addUserLabel(self, user\_name, label):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**3. def searchUserLabel(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=favorite\_label)

**4. def searchUserInfo(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=res)

**5. def getPassword(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=password)

**6. def judgeUserExist(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

User exist：return dict(successFalse=False, error="this user already exist", data=res)

User not exist：return dict(successTrue=True, error="", data="")

SongInfo Table

**1.def addSongInfo (self, song\_id, song\_name, singer, song\_sheet\_id\_belonged, label):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**2. def searchSongInfo(self, song\_id):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=res)

**3. def updateSongInfo(self, song\_id, song\_sheet\_id\_belonged, label):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**4. def userSearchSong(self, song\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=(song\_name,singer,label))

SongSheetInfo Table

**1. def addSongSheetInfo(self, song\_sheet\_id, song\_sheet\_name, label):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

UserSongSheet Table

**1. def addUserSongSheet(self, user\_id, song\_id, last\_click\_time):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**2. def searchLastClickTime(self, user\_id, song\_id):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=last\_click\_time)

**3. def updateUserSongSheet(self, user\_id, song\_id, last\_click\_time):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**4. def deleteUserSongSheetRecord(self, user\_name, song\_id):**

Error occurred：return {"success": False, "error": str(e)}

Operation succeeded：return {"success": True, "error": ""}

**5. def searchAllSongUserListen(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=song\_id)

Algorithm module

**1. def getMatrix(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=df)

**2. def getDict(self, user\_name):**

Error occurred：return dict(success=False, error=e, data="")

Operation succeeded：return dict(success=True, error="", data=dic)

1. **System operating environment**

Pymysql=0.9.3

numpy=1.15.1

pandas=0.25.1