# 1、音乐人成员表

## 1.1、关系模式

* 音乐人成员表(R\_MM(N, BD, RL, AY, MID))

|  |  |  |  |
| --- | --- | --- | --- |
| **属性名** | **中文** | **数据类型** | **备注** |
| N | 姓名 | VARCHAR | 主码的属性 |
| BD | 生日 | DATE |  |
| RL | 职务/角色 | VARCHAR |  |
| AY | 活动年份 | VARCHAR |  |
| MID | 音乐人ID | UNSIGNED INT | 主码的属性，外码 |

* 备注：1.主码为（N，MID），默认同一个音乐人内没有姓名相同的成员（接口设计文档目前按照备注1） 2.可以简化为：默认世上没有名字相同的人，以姓名为主码，此时MID仅仅是外码

## 1.2、添加成员接口

[TOC]

##### 简要描述

* 添加成员接口

##### 请求URL

* TODO

##### 请求方式

* POST

##### 传入后端的参数

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **属性名** | **N** | **BD** | **RL** | **AY** | **MID** |
| 中文 | 姓名 | 生日 | 职务、角色 | 活动年份 | 音乐人ID |
| 编号 | 1 | 2 | 3 | 4 | 5 |
| 数据类型 | string | date | string | string | unsigned int |
| 数据宽度 | 32 |  | 16 | 16 | 对应 unsigned int |
| 小数位数 | 0 | 0 | 0 | 0 | 0 |
| 单位 | / | / | / | / | / |
| 值约束 |  | 出生年<=2012 |  |  |  |
| 是否允许为空 | 否 | 是 | 是 | 是 | 否 |

##### 传入后端的为一序列化的JSON对象，结构示例如下

|  |
| --- |
| {  "data": {  "N": "cfm", #姓名，字符串  "BD": 2000-1-1, #生日，是一个date对象，格式暂不清楚，有待查询  "RL": "singer", #职务/角色，字符串  "AY": “2015-2020”，#活动年份，字符串  "MID": 123456789, #音乐人ID，无符号整数  }  }  备注：主码（N,MID）的属性不可以为空，其余可以为空 |

##### 如果添加成功，后端返回一JSON对象，包含该音乐人添加成员后所有成员信息，其中data为数组，结构示例如下

|  |
| --- |
| {  "error": 0，#添加成功  "data":  [  {  "N": "cfm", #姓名，字符串  "BD": 2000-1-1, #生日，是一个date对象，格式暂不清楚，有待查询  "RL": "singer", #职务/角色，字符串  "AY": “2015-2020”，#活动年份，字符串  },  ...  ]  }  备注：返回的成员信息中没有MID（音乐人ID），因为音乐人的成员列表不会显示音乐人ID，且当前必处于该音乐人的信息页面 |

##### 如果添加失败，返回一JSON对象，包含错误码和错误信息，结构示例如下

|  |
| --- |
| {  "error": 1/2/3, #错误类型，整型数，1表示成员已经存在，2表示成员信息缺乏N或者MID，3表示成员信息格式不规范  "msg":"xxx" #具体错误信息描述，字符串  } |

## 1.3、修改成员接口

[TOC]

##### 简要描述

* 修改成员接口（本质为查找并修改）

##### 请求URL

* TODO

##### 请求方式

* POST

##### 传入后端的参数

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **属性名** | **N** | **BD** | **RL** | **AY** | **MID** |
| 中文 | 姓名 | 生日 | 职务、角色 | 活动年份 | 音乐人ID |
| 编号 | 1 | 2 | 3 | 4 | 5 |
| 数据类型 | string | date | string | string | unsigned int |
| 数据宽度 | 32 |  | 16 | 16 | 对应 unsigned int |
| 小数位数 | 0 | 0 | 0 | 0 | 0 |
| 单位 | / | / | / | / | / |
| 值约束 |  | 出生年<=2012 |  |  |  |
| 是否允许为空 | 否 | 是 | 是 | 是 | 否 |

##### 传入后端的为一序列化的JSON对象，结构示例如下

|  |
| --- |
| {  "data": {  "N": "cfm", #姓名，字符串，  "BD": 2000-1-1, #生日，是一个date对象，格式暂不清楚，有待查询  "RL": "singer", #职务/角色，字符串  "AY": “2015-2020”，#活动年份，字符串  "MID": 123456789, #音乐人ID，无符号整数  }  }  备注：修改成员信息时，名字和音乐人ID是不可以更改的（否则就是添加成员），传给后端名字和音乐人ID只是为了作为主码找到对应元组，其余3个属性可以不改也可以改为空，具体什么时候表示某个属性不改或者改为空，取决于前后端约定 |

##### 如果修改成功，返回一JSON对象，包含修改以后的成员信息，结构示例如下

|  |
| --- |
| {  "error": 0，  "data": {  "N": "cfm", #姓名，字符串，  "BD": 2000-1-1, #生日，是一个date对象，格式暂不清楚，有待查询  "RL": "singer", #职务/角色，字符串  "AY": “2015-2020”，#活动年份，字符串  }  }  备注：前端只需将这个修改的成员信息替换修改前该成员的信息显示，其余成员信息保持原来的显示，且不返回MID，其理由和添加成员接口不返回MID的理由相同 |

##### 如果修改失败，返回一JSON对象，包含错误码和错误信息，结构示例如下

|  |
| --- |
| {  "error": 1/2, #错误类型，整型数，1表示要修改信息的成员不存在，2表示修改的成员信息格式不规范  "msg":"xxx" #具体错误信息描述，字符串  } |

## 1.4、删除成员接口

[TOC]

##### 简要描述

* 删除成员接口（本质为查找并删除）

##### 请求URL

* TODO

##### 请求方式

* POST

##### 传入后端的参数

|  |  |  |
| --- | --- | --- |
| **属性名** | **N** | **MID** |
| 中文 | 姓名 | 音乐人ID |
| 编号 | 1 | 2 |
| 数据类型 | string | unsigned int |
| 数据宽度 | 32 | 对应 unsigned int |
| 小数位数 | 0 | 0 |
| 单位 | / | / |
| 值约束 |  |  |
| 是否允许为空 | 否 | 否 |

##### 传入后端的为一序列化的JSON对象，结构示例如下

|  |
| --- |
| {  "data": {  "N": "cfm", #姓名，字符串  "MID": 123456789, #音乐人ID，无符号整数  }  }  备注：主码为（N，MID），凭借主码找到被删除成员 |

##### 如果删除成功，返回一JSON对象，包含删除该成员后的音乐人全部成员信息，其中data为数组，结构示例如下

|  |
| --- |
| {  "error": 0，#删除成功  "data":  [  {  "N": "cfm", #姓名，字符串  "BD": 2000-1-1, #生日，是一个date对象，格式暂不清楚，有待查询  "RL": "singer", #职务/角色，字符串  "AY": “2015-2020”，#活动年份，字符串  },  ...  ]  }  备注：返回的成员信息中没有MID（音乐人ID），因为音乐人的成员列表不会显示音乐人ID，且当前必处于该音乐人的信息页面 |

##### 如果删除失败，返回一JSON对象，包含错误码和错误信息，结构示例如下

|  |
| --- |
| {  "error": 1, #错误类型，整型数，1表示要求删除的成员不存在  "msg":"xxx" #具体错误信息描述，字符串  } |

# 2、歌曲表

## 2.1、关系模式

* 音乐人成员表(R\_AM(SN, SL, ADT, AID))

|  |  |  |  |
| --- | --- | --- | --- |
| **属性名** | **中文** | **数据类型** | **备注** |
| SN | 歌曲名称 | VARCHAR | 主码的属性 |
| SL | 歌曲时长 | INT |  |
| ADT | 试听链接 | VARCHAR |  |
| AID | 唱片ID | UNSIGNED INT | 主码的属性，外码 |

* 备注：1.主码为（SN，AID），默认同一个唱片内没有相同歌名的歌曲（接口设计文档目前按照备注1） 2.可以简化为：默认所有唱片没有歌名相同的歌曲，以歌名为主码，此时AID仅仅是外码

## 2.2、添加歌曲接口

[TOC]

##### 简要描述

* 添加歌曲接口

##### 请求URL

* TODO

##### 请求方式

* POST

##### 传入后端的参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **属性名** | **SN** | **SL** | **ADT** | **AID** |
| 中文 | 歌曲名称 | 歌曲时长 | 试听链接 | 唱片ID |
| 编号 | 1 | 2 | 3 | 4 |
| 数据类型 | string | integer | string | unsigned int |
| 数据宽度 | 32 | 2 | 16 | 对应 unsigned int |
| 小数位数 | 0 | 0 | 0 | 0 |
| 单位 | / | s(秒) | / | / |
| 值约束 |  |  |  |  |
| 是否允许为空 | 否 | 是 | 是 | 否 |

##### 传入后端的为一序列化的JSON对象，结构示例如下

|  |
| --- |
| {  "data": {  "SN": "lively", #歌名，字符串  "SL": 180, #歌曲时长，单位为秒  "ADT": "www.163.com", #试听链接，字符串  "AID": 123456789, #唱片ID，无符号整数  }  } |

##### 如果添加成功，后端返回一JSON对象，包含该唱片添加歌曲后所有成员信息，其中data为数组

|  |
| --- |
| {  "error": 0，#添加成功  "data":  [  {  "SN": "lively", #歌名，字符串  "SL": 180, #歌曲时长，单位为秒  "ADT": "www.163.com", #试听链接，字符串  },  ...  ]  }  备注：返回的成员信息中没有AID（唱片ID），因为唱片的歌曲列表不会显示唱片ID，且当前必处于该唱片的信息页面 |

##### 如果添加失败，返回一JSON对象，包含错误码和错误信息

|  |
| --- |
| {  "error": 1/2/3, #错误类型，整型数，1表示唱片已经存在，2表示唱片信息缺乏SN或者AID，3表示唱片信息格式不规范  "msg":"xxx" #具体错误信息描述，字符串  } |

## 2.3、修改歌曲接口

[TOC]

##### 简要描述

* 修改歌曲接口（本质为查找并修改）

##### 请求URL

* TODO

##### 请求方式

* POST

##### 传入后端的参数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **属性名** | **SN** | **SL** | **ADT** | **AID** |
| 中文 | 歌曲名称 | 歌曲时长 | 试听链接 | 唱片ID |
| 编号 | 1 | 2 | 3 | 4 |
| 数据类型 | string | integer | string | unsigned int |
| 数据宽度 | 32 | 2 | 16 | 对应 unsigned int |
| 小数位数 | 0 | 0 | 0 | 0 |
| 单位 | / | s(秒) | / | / |
| 值约束 |  |  |  |  |
| 是否允许为空 | 否 | 是 | 是 | 否 |

##### 传入后端的为一序列化的JSON对象，结构示例如下

|  |
| --- |
| {  "data": {  "SN": "lively", #歌曲名，字符串，  "SL": 180, #歌曲时长，单位为秒  "ADT": "www.163.com", #试听链接，字符串  "AID": 123456789, #唱片ID，无符号整数  }  }  备注：修改歌曲信息时，歌名和唱片ID是不可以更改的（否则就是添加歌曲），传给后端歌名和唱片ID只是为了作为主码找到对应元组，其余2个属性可以不改也可以改为空，具体什么时候表示某个属性不改或者改为空，取决于前后端约定 |

##### 如果修改成功，返回一JSON对象，包含修改以后的歌曲信息，结构示例如下

|  |
| --- |
| {  "error": 0，  "data": {  "SN": "lively", #歌曲名，字符串，  "SL": 180, #歌曲时长，单位为秒  "ADT": "www.163.com", #试听链接，字符串  "AID": 123456789, #唱片ID，无符号整数  }  }  备注：前端只需将这个修改的歌曲信息替换修改前该歌曲的信息显示，其余歌曲信息保持原来的显示。不返回AID，其理由和添加歌曲接口不返回AID的理由相同。 |

##### 如果修改失败，返回一JSON对象，包含错误码和错误信息，结构示例如下

|  |
| --- |
| {  "error": 1/2, #错误类型，整型数，1表示要修改的歌曲不存在，2表示修改的歌曲信息格式不规范  "msg":"xxx" #具体错误信息描述，字符串  } |

## 2.4、删除歌曲接口

[TOC]

##### 简要描述

* 删除歌曲接口（本质为查找并删除）

##### 请求URL

* TODO

##### 请求方式

* POST

##### 传入后端的参数

|  |  |  |
| --- | --- | --- |
| **属性名** | **SN** | **AID** |
| 中文 | 歌名 | 唱片ID |
| 编号 | 1 | 2 |
| 数据类型 | string | unsigned int |
| 数据宽度 | 32 | 对应 unsigned int |
| 小数位数 | 0 | 0 |
| 单位 | / | / |
| 值约束 |  |  |
| 是否允许为空 | 否 | 否 |

##### 传入后端的为一序列化的JSON对象，结构示例如下

|  |
| --- |
| {  "data": {  "SN": "lively", #歌曲名，字符串  "AID": 123456789, #歌曲ID，无符号整数  }  }  备注：主码为（SN，AID），凭借主码找到被删除歌曲 |

##### 如果删除成功，返回一JSON对象，包含删除该歌曲后的唱片全部歌曲信息，其中data为数组，结构示例如下

|  |
| --- |
| {  "error": 0，#删除成功  "data":  [  {  "SN": "lively", #歌名，字符串  "SL": 180, #歌曲时长，单位为秒  "ADT": "www.163.com", #试听链接，字符串  },  ...  ]  }  备注：返回的歌曲信息中没有AID（唱片ID），因为唱片的歌曲列表不会显示唱片ID，且当前必处于该唱片的信息页面 |

##### 如果删除失败，返回一JSON对象，包含错误码和错误信息，结构示例如下

|  |
| --- |
| {  "error": 1, #错误类型，整型数，1表示要求删除的歌曲不存在  "msg":"xxx" #具体错误信息描述，字符串  } |