

一种多选项的存储与高效查询的解决方案

文章分类: *JavaDemo*; 标签: *JavaCodeSnippet*; 作者: *Hackyle*;
更新时间: *Wed Jul 26 19:11:45 CST 2023*

1. 背景

2. 设计思想

1. 将多选项转换为数字

1. 例1

2. 例2

2. 查询原理

3. 项目启动

1. 新增数据演示

2. 查询数据演示

4. 查询示例

1. 构造数据

2. 插入到数据库

3. 查询选择了"1-编程"这个选项的记录

4. 查询选择了"2-听音乐唱歌, 5-看电影"这些选项的记录

本文主要内容

• 对于多选项的值, 如何保存? 本文提供了一种非常规的方案。

• 对于记录在数据库中的多选项的值, 如何快速查询那些记录是包含了某个(某些)选项? 本文使用了“与位运算”解决查询问题。

• 源码地址: <https://github.com/HackyleShawe/JavaDemos/tree/master/Examples/multi-options-storage-query-demo>

文章前置知识: SpringBoot、JdbcTemplate、位运算(与运算)、jQuery

内容导览

• 背景

• 设计思想

◦ 将多选项转换为数字

▪ 例1

▪ 例2

◦ 查询原理

• 项目启动

◦ 新增数据演示

◦ 查询数据演示

• 查询示例

◦ 构造数据

◦ 插入到数据库

◦ 查询选择了"1-编程"这个选项的记录

◦ 查询选择了"2-听音乐唱歌, 5-看电影"这些选项的记录

背景

在项目开发中, 如何保存多选项的值呢? 例如下图中的职业发展和兴趣爱好

新增数据

Name:

Gender: ☒ Man ☐ Woman

Address:

CareerOption: ☐ 收入无上限 ☐ 培训与发展 ☐ 职业价值感 ☐ 行业稳定性 ☐ 社交与人脉 ☐ 塑造个人品牌 ☐ 团队综合素质 ☐ 终身学习

InterestOption: ☐ 编程 ☐ 听音乐唱歌 ☐ 篮球 ☐ 玩游戏 ☐ 看电影 ☐ 享美食 ☐ 健身 ☐ 旅游 ☐ 看书 ☐ 写作

- 最容易想到的就是, 选择了哪些选项, 就把该选项值存储起来。
- 在数据库层面设置一个VARCHAR, 例如选择了"收入无上线、培训与发展、职业价值感", 前端就传递"收入无上线、培训与发展、职业价值感", 数据库就保存为"收入无上线、培训与发展、职业价值感"。

可是这样存在一个问题, 我要查询那些人选择了其中的某个、某些选项, 就很难实现。例如, 查询那些人的兴趣爱好是"编程、篮球", 查询兴趣爱好是"看书、写作"的人数有多少。

为了解决这种多选项的高效查询问题, 本人设计了一种方法, 可以实现快速、高效地查询多选项中有某个、某些选项的所有记录。

设计思想

主要思想

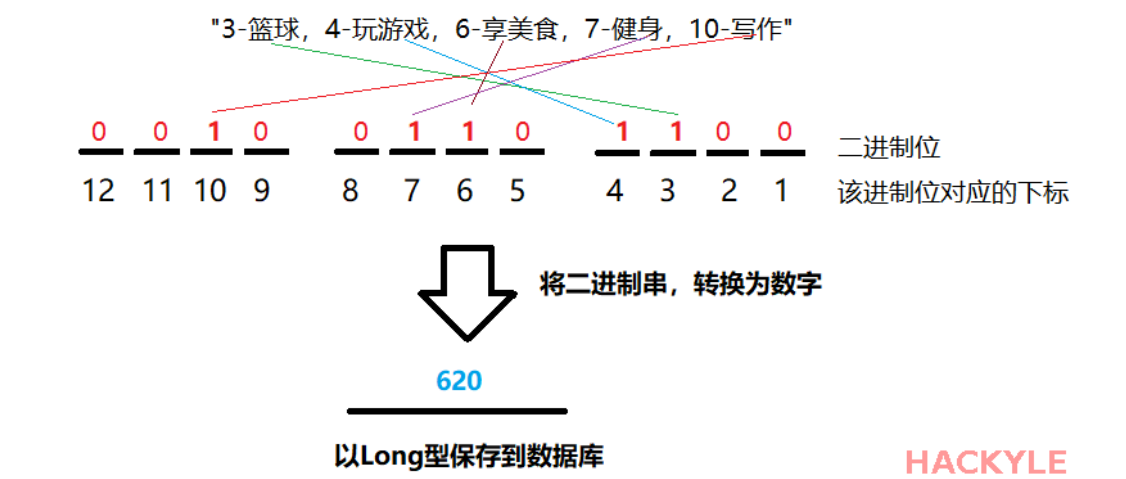
- 将多选项进行编号: 例如: 对职业发展的多选项进行编号为: 1-收入无上限, 2-培训与发展, 3-职业价值感, 4-行业稳定性, 5-社交与人脉, 6-塑造个人品牌, 7-团队综合素质, 8-终身学习; 对兴趣爱好的多选项进行编号为: 1-编程, 2-听音乐唱歌, 3-篮球, 4-玩游戏, 5-看电影, 6-享美食, 7-健身, 8-旅游, 9-看书, 10-写作。

- 查询：前端还是只传选项编号；后端将其转换为数字；在数据库层面使用位运算中的与运算，匹配包含了参数选项的记录。

将多选项转换为数字

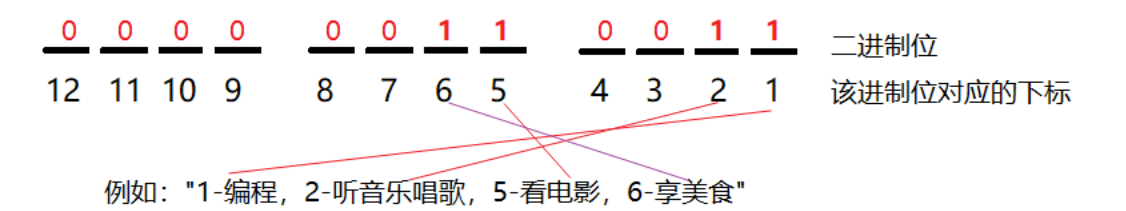
现以兴趣爱好为例，为其多选项定义编号：1-编程，2-听音乐唱歌，3-篮球，4-玩游戏，5-看电影，6-享美食，7-健身，8-旅游，9-看书，10-写作

根据二进制位下标与十进制数的互转：



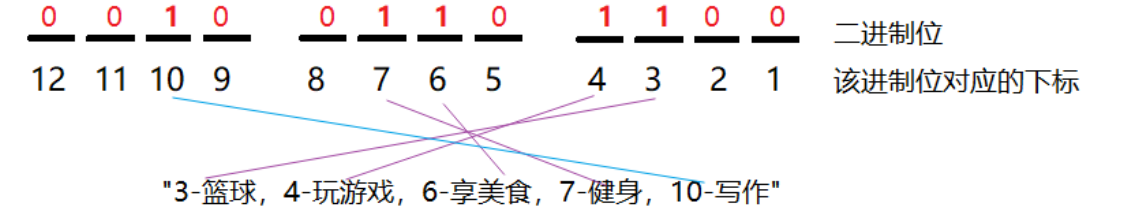
例1

用户A勾选的兴趣爱好为："1-编程，2-听音乐唱歌，5-看电影，6-享美食"
前端传递的串为：1,2,5,6
转换为二进制串：0011 0011
转换规则：在有选项编号出现的位下标的位置上填充1，其他位置填充0
转换为十进制后落库：51



例2

用户B勾选的兴趣爱好为："3-篮球，4-玩游戏，6-享美食，7-健身，10-写作"
前端传递的串为：3,4,6,7,10
转换为二进制串：0010 0110 1100
转换为十进制后落库：620



查询原理

查询的核心思想：与（&）位运算

(包含)

需求：查询所有选有"听音乐唱歌，看电影"的记录



前端传递来的查询条件："2,5"

转换查询条件为二进制（原理同新增保存时）：0010 0010

转换为十进制：34



根据**与运算**原理，编写为SQL: SELECT * FROM person WHERE
interests & 34 = 34 AND deleted = 0;

与运算过程：现假定数据库中interests有两条记录

1. 243(10), 0000 1111 0011(2)

2. 908(10), 0011 1000 1100(2)



查询条件34(10), 0010 0010(2)与**第一条记录匹配原理：**

0000 0001 0010

& 0000 1111 0011

0000 0001 0010 与运算结果仍为查询条件，说明这条记录包含了"2-听音乐唱歌，5-看电影"这些选项

查询条件34(10), 0010 0010(2)与**第二条记录匹配原理：**

0000 0001 0010

& 0011 1000 1100

#-----

0000 0000 0000 与运算结果不为查询条件，说明这条记录不包含"2-听音乐唱歌，5-看电影"这些选项

HACKYLE

项目启动

Step 1: 在application.yml中修改数据库连接参数

Step 2: 执行resources/sql.sql下的SQL文件，初始化数据

Step 3: 从启动类App.java启动

Step 4: 启动成功后进入前端页面: <http://localhost:9898/person.html>

新增数据演示

必要数据，多选项进行勾选：

HACKYLE

首页

文章分类

文章标签

关于

留言

新增数据

Name: AAA

Gender: ☐ Man ☒ Woman

Address: SH

CareerOption: ☒ 收入无上限 ☐ 培训与发展 ☒ 职业价值感 ☐ 行业稳定性 ☒ 社交与人脉 ☐ 塑造个人品牌 ☒ 团队综合素质 ☐ 终身学习

InterestOption: ☒ 编程 ☐ 听音乐唱歌 ☐ 篮球 ☒ 玩游戏 ☐ 看电影 ☒ 享美食 ☐ 健身 ☒ 旅游 ☒ 看书 ☐ 写作

Submit

更新成功

确定

HACKYLE

去数据库查看刚刚新增的记录：

Database > kdb@localhost > schemas > kdb > tables > person

person

1 row

Tx: Auto

DDL

DML

Comma...d (CSV)

pe

Filter Criteria

	id	name	gender	address	careers	interests	create_time	update_time
1	1690356630962	AAA	0	SH	85	425	2023-07-26 15:30:31	2023-07-26 15:30:31

查询数据演示

选择查询条件，点击 “Query” 进行查询：

Title

localhost:9898/person.html

新增数据

Name: AAA

Gender: ☐ Man ☒ Woman

Address: SH

CareerOption: ☒ 收入无上限 ☐ 培训与发展 ☒ 职业价值感 ☐ 行业稳定性 ☒ 社交与人脉 ☐ 塑造个人品牌 ☒ 团队综合素质 ☐ 终身学习

InterestOption: ☒ 编程 ☐ 听音乐唱歌 ☐ 篮球 ☒ 玩游戏 ☐ 看电影 ☒ 享美食 ☐ 健身 ☐ 旅游 ☒ 看书 ☐ 写作

Submit

查询数据

Name:

CareerOption: ☒ 收入无上限 ☐ 培训与发展 ☐ 职业价值感 ☐ 行业稳定性 ☐ 社交与人脉 ☐ 塑造个人品牌 ☐ 团队综合素质 ☐ 终身学习

InterestOption: ☐ 编程 ☐ 听音乐唱歌 ☐ 篮球 ☒ 玩游戏 ☐ 看电影 ☐ 享美食 ☐ 健身 ☐ 旅游 ☐ 看书 ☐ 写作

Query

{ "id": "1690356630962", "name": "AAA", "gender": "0", "address": "SH", "careerOption": "收入无上限, 职业价值感, 社交与人脉, 团队综合素质, ", "interests": "玩游戏, 享美食, 旅游, 看书, ", "createTime": "2023-07-26T07:30:31.000+00:00", "updateTime": "2023-07-26T07:30:31.000+00:00" }

查看运行日志，显示执行的SQL：

```
2023-07-26 15:22:58.766 INFO 7008 --- [main] com.ks.demo.mosq.App : Started App in 8.05 seconds (JVM running for 16.409)
2023-07-26 15:23:23.641 INFO 7008 --- [nio-9898-exec-2] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet 'dispatcherServlet'
2023-07-26 15:23:23.641 INFO 7008 --- [nio-9898-exec-2] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2023-07-26 15:23:23.657 INFO 7008 --- [nio-9898-exec-2] o.s.web.servlet.DispatcherServlet : Completed initialization in 15 ms
2023-07-26 15:30:30.973 INFO 7008 --- [nio-9898-exec-3] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2023-07-26 15:30:31.255 INFO 7008 --- [nio-9898-exec-3] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
query sql :select * from person where deleted = 0 AND careers & 1 = 1 AND interests & 8 = 8
```

HACKYLE

查询示例

以多选项”兴趣爱好“为例，展示查询的工作原理

构造数据

AA选择了"1-编程, 2-听音乐唱歌, 5-看电影, 6-享美食, 7-健身, 8-旅游"
前端传递的编号串：1,2,5,6,7,8
转换为二进制串：0000 1111 0011
转换为数字：243

https://blog.hackyle.com/article/java-demo/multi-options-storage-query-demo

4/8

转换为数字：908

CC选择了"2-听音乐唱歌，3-篮球，5-看电影，6-享美食，7-健身，9-看书，10-写作"
前端传递的编号串：2,3,5,6,7,9,10
转换为二进制串：0011 0111 0110
转换为数字：886

DD选择了"1-编程，3-篮球，4-玩游戏，6-享美食，7-健身，8-旅游，10-写作"
前端传递的编号串：1,3,4,6,7,8,10
转换为二进制串：0010 0110 1101
转换为数字：749

EE选择了"2-听音乐唱歌，4-玩游戏，5-看电影，7-健身，8-旅游，10-写作"
前端传递的编号串：2,4,5,7,8,10
转换为二进制串：0010 1101 1010
转换为数字：730

FF选择了"1-编程，2-听音乐唱歌，3-篮球，4-玩游戏，5-看电影，7-健身"
前端传递的编号串：1,2,3,4,5,7
转换为二进制串：0000 0101 1111
转换为数字：95

GG选择了"1-编程，3-篮球，4-玩游戏，6-享美食，8-旅游，10-写作"
前端传递的编号串：1,3,4,6,8,10
转换为二进制串：0010 1010 1101
转换为数字：685

插入到数据库

```
1 DROP TABLE IF EXISTS person;
2 CREATE TABLE person (
3     id BIGINT AUTO_INCREMENT,
4     name VARCHAR(50) DEFAULT NULL COMMENT '姓名',
5     gender INT DEFAULT NULL COMMENT '性别，0-女，1-男',
6     address VARCHAR(128) DEFAULT NULL COMMENT '地址',
7     careers BIGINT DEFAULT NULL COMMENT '职业发展多选题',
8     -- 兴趣爱好多选题。可选项：1-编程，2-听音乐唱歌，3-篮球，4-玩游戏，5-看电影，6-享美食
9     -- 例如，全选："11 1111 1111"，保存为十进制=1023，全不选："00 0000 0000"，保存为十
10    -- LONG最大支持64位，最多支持64个多选题的任意选择
11    interests BIGINT DEFAULT NULL COMMENT '兴趣爱好多选题',
12    create_time DATETIME DEFAULT NULL,
13    update_time DATETIME DEFAULT NULL,
14    deleted INT DEFAULT 0 COMMENT '是否删除：0-否，1-是',
15    PRIMARY KEY(id)
16 );
```

```
1 -- 将上文中构造的数据，以SQL的形式插入到数据库中，只以多选题"兴趣爱好"为例
2 INSERT INTO person(name, gender, address, careers, interests, create_time, update_
3 VALUES ('AA', 1, 'SH CN', 1, 243, '2022-12-12','2023-12-12', 0),
4         ('BB', 1, 'SH CN', 1, 908, '2022-12-12','2023-12-12', 0),
5         ('CC', 1, 'SH CN', 1, 886, '2022-12-12','2023-12-12', 0),
6         ('DD', 1, 'SH CN', 1, 749, '2022-12-12','2023-12-12', 0),
7         ('EE', 1, 'SH CN', 1, 730, '2022-12-12','2023-12-12', 0),
8         ('FF', 1, 'SH CN', 1, 95, '2022-12-12','2023-12-12', 0),
9         ('GG', 1, 'SH CN', 1, 685, '2022-12-12','2023-12-12', 0);
```

查询选择了"1-编程"这个选项的记录

AA: 0000 1111 0011
BB: 0011 1000 1100
CC: 0011 0111 0110
DD: 0010 0110 1101
EE: 0010 1101 1010
FF: 0000 0101 1111
GG: 0010 1010 1101

将"1-编程"进行转换

- 转换为二进制: 0000 0000 0001
- 转换为十进制: 1

查询原理: 将查询条件"0000 0000 0001"与AA~GG的二进制位进行与运算后, 仍然为查询条件的记录, 则是选择了"1-编程"这个选项的记录

查询过程

- 将查询条件与AA的二进制位进行与运算:

```
1 | 0000 0000 0001
2 | & 0000 1111 0011
3 | #-----
4 | 0000 0000 0001 与运算结果仍为查询条件, 说明这条记录包含了"1-编程"这个选项
```

- 将查询条件与BB的二进制位进行与运算:

```
1 | 0000 0000 0001
2 | & 0011 1000 1100
3 | #-----
4 | 0000 0000 0000 与运算结果不为查询条件, 说明这条记录不包含"1-编程"这个选项
```

- 其他记录运算同理
- 最终发现只有AA、DD、FF、GG的运算结果符合条件, 这四个记录就是满足"选择了1-编程这个选项的所有记录"

SQL实现

```
1 | select * FROM person WHERE interests & 1 = 1;
```

✓ select * FROM person WHERE interests & 1 = 1;

id	name	gender	address	careers	interests	create_time	update_time	deleted
1	AA	1	SH CN	1	243	2022-12-12 00:00:00	2023-12-12 00:00:00	0
2	DD	1	SH CN	1	749	2022-12-12 00:00:00	2023-12-12 00:00:00	0
3	FF	1	SH CN	1	95	2022-12-12 00:00:00	2023-12-12 00:00:00	0
4	GG	1	SH CN	1	685	2022-12-12 00:00:00	2023-12-12 00:00:00	0

查询选择了"2-听音乐唱歌, 5-看电影"这些选项的记录

目标: 在多项中查询选择了"2-听音乐唱歌, 5-看电影"这些选项的记录

用户的"兴趣爱好"多选项 (二进制形式)

AA: 0000 1111 0011
BB: 0011 1000 1100
CC: 0011 0111 0110
DD: 0010 0110 1101

将"2-听音乐唱歌，5-看电影"进行转换

- 转换为二进制：0000 0001 0010
- 转换为十进制：18

查询原理：将查询条件"0000 0001 0010"与AA~GG的二进制位进行与运算后，仍然为查询条件的记录，则是选择了"2-听音乐唱歌，5-看电影"这个选项的记录

查询过程

- 将查询条件与AA的二进制位进行与运算：

1 | 0000 0001 0010

2 | & 0000 1111 0011

3 | #-----

4 | 0000 0001 0010 与运算结果仍为查询条件，说明这条记录包含了"2-听音乐唱歌，5-看电影"这!

- 将查询条件与BB的二进制位进行与运算：

1 | 0000 0001 0010

2 | & 0011 1000 1100

3 | #-----

4 | 0000 0000 0000 与运算结果不为查询条件，说明这条记录不包含"2-听音乐唱歌，5-看电影"!

- 其他记录运算同理
- 最终发现只有AA、CC、EE、FF的运算结果符合条件，这四个记录就是满足"选择了2-听音乐唱歌，5-看电影"这些选项的所有记录"

SQL实现

1 | select * FROM person WHERE interests & 18 = 18;

✓ select * FROM person WHERE interests & 18 = 18;

Output kdb.person x

Tx: Auto DB ✓ ↺

DDL DML

Comma-separated (CSV)

	id	name	gender	address	careers	interests	create_time	update_time
1	1	AA		1 SH CN	1	243	2022-12-12 00:00:00	2023-12-12 00:00:00
2	3	CC		1 SH CN	1	886	2022-12-12 00:00:00	2023-12-12 00:00:00
3	5	EE		1 SH CN	1	730	2022-12-12 00:00:00	2023-12-12 00:00:00
4	6	FF		1 SH CN	1	95	2022-12-12 00:00:00	2023-12-12 00:00:00

版权声明：非明确标注皆为原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上本文链接及此声明。
原文链接：<https://blog.hackyle.com/article/java-demo/multi-options-storage-query-demo>

留下你的评论

Name:

Email:

© Copy Right: 2022 HACKYLE. All Rights Reserved
Designed and Created by HACKYLE SHAWE
备案号: 浙ICP备20001706号-2