Group2 miniSQL 测试语句 解释说明版

Part 0 基础说明与功能概览

0.1 基础说明

1. 我们的SQL语句对大小写**不敏感**
2. 每个语句以英文分号作为结束标志
3. 我们**额外支持**的语句主要有: update, help, show, import和output，具体阐述见后

0.2 Bonus一览

1. 类似mysql的错误提示;
2. 更多语句:
    1. update;
    2. show, show table, show index;
    3. import;
    4. output;
    5. help;
3. SQL查询优化（利用index）

Part 1 基础要求语句

> 这部分全部执行正确的语句，异常检测相关语句我们单独置于Part 3进行集中测试
>
> 退出语句放在全部语句的最后进行测试
>
> index优化的测试需要有大量数据支撑才更有效果，因此放在了import语句后

1.1 创建表格

```
create table user (id int, nick_name char(10) unique, gender char(1), score
float, primary key (id));
```

此时可用`show`语句查看已经创建成功，并为主键创建了索引，且已经新建了文件:

```
show table user;
```

```
show;
```

1.2 删除表格

```
drop table user;
```

成功删除表格，可用`show`语句观测到已经没有user表格及为主键创建的索引了:

```
show;
```

为了后续的测试，在这里重建表格:

```
create table user (id int, nick_name char(10) unique, gender char(1), score
float, primary key (id));
```

1.3 创建索引

创建表格的同时已经为主键创建了索引，这里为unique的nick_name创建索引

```
create index user_name_index on user (nick_name);
```

可用`show`语句查看

```
show index user_name_index;
```

1.4 删除索引

```
drop index user_name_index;
```

可用`show`语句查看

```
show;
```

恢复索引

```
create index user_name_index on user (nick_name);
```

1.5 INSERT语句

执行以下插入测试数据的语句：

```
insert into user values (31901001, 'mike', 'M', 94.0);
insert into user values (31901002, 'john', 'M', 75.6);
insert into user values (31902001, 'max', 'F', 33.3);
insert into user values (31902003, 'lucy', 'F', -5.0);
```

1.6 SELECT语句

1.6.1 全部属性搜索

```
select * from user;
```

1.6.2 部分属性搜索

```
select id, nick_name from user;
```

1.6.3 单条件搜索

为节约测试时间起见，我们设置的测试语句有如下特点：

1. **类型不同的数据**的条件语句仅在某些有代表性的地方全部覆盖；
2. **无符合数据**的返回只在某些有代表性的地方写了相应测试语句；
3. **选择全部属性还是某几个属性**已在上述全局搜索部分做了测试，在下面的测试中我们大部分采用了全局搜索，某些地方混用了部分属性；

4. 字符串采用**大小写、单双引号**的测试也仅在相等部分做了不敏感测试。

综上，测试语句对于典型功能做了测试，由于上述的省略的功能测试与我们测试的重点功能**独立不相干**，故实无全局测试的必要，如有需要可在线下验收过程中随时提出，现场测试。

= int 相等，有符合数据&无符合数据

```
select * from user where id = 31901001;
select * from user where id = 88888888;
```

= char 相等

```
select * from user where nick_name = "John";
select * from user where nick_name = 'john';
```

> 大小写、单双引号不敏感，引号只要求配对使用

= float 相等

```
select * from user where score = 33.3;
```

<> int 不等

```
select id, nick_name, score from user where id <> 31901001;
select id, nick_name, score from user where id <> 88888888;
```

> 下面的语句使用的值在表中不存在

<> char 不等

```
select * from user where nick_name <> "john";
select * from user where nick_name <> "bdz";
```

> 下面的语句使用的值在表中不存在

<> float 不等

```
select * from user where score <> -5.0;
select * from user where score <> 100;
```

> 下面的语句使用的值在表中不存在

> >= int

```
select * from user where id > 31901001;
select * from user where id >= 31901001;
select * from user where id > 31900000;
select * from user where id >= 31900000;
select * from user where id > 31902000;
select * from user where id >= 31902000;
select * from user where id > 31902003;
select * from user where id >= 31902003;
select * from user where id > 32000000;
select * from user where id >= 32000000;
```

> - 1, 2: 下界测试;
> - 3, 4: 数值不在表格内，下界;
> - 5, 6: 数值在表格内，中间数据;
> - 7, 8: 上界测试
> - 9, 10: 数值不在表格内，上界

> >= char

```
select * from user where nick_name > 'john';
select * from user where nick_name >= 'john';
select * from user where nick_name > 'bdz';
select * from user where nick_name >= 'bdz';
select * from user where nick_name > 'mike';
select * from user where nick_name >= 'mike';
select * from user where nick_name > 'max';
select * from user where nick_name >= 'max';
select * from user where nick_name > 'zpq';
select * from user where nick_name >= 'zpq';
```

> - 1, 2: 下界测试;
> - 3, 4: 数值不在表格内，下界;
> - 5, 6: 数值在表格内，中间数据;
> - 7, 8: 上界测试
> - 9, 10: 数值不在表格内，上界

> >= float

```
select * from user where score > -5.0;
select * from user where score >= -5.0;
select * from user where score > -10.0;
select * from user where score >= -10;
select * from user where score > 75.6;
select * from user where score >= 75.6;
select * from user where score > 94.0;
select * from user where score >= 94.0;
select * from user where score > 100.5;
select * from user where score >= 100.5;
```

> - 1, 2: 下界测试;
> - 3, 4: 数值不在表格内，下界，4并没有显示表明小数点;
> - 5, 6: 数值在表格内，中间数据;
> - 7, 8: 上界测试
> - 9, 10: 数值不在表格内，上界

< <= int

```sql
select * from user where id < 31901001;
select * from user where id <= 31901001;
select * from user where id < 31900000;
select * from user where id <= 31900000;
select * from user where id < 31902000;
select * from user where id <= 31902000;
select * from user where id < 31902003;
select * from user where id <= 31902003;
select * from user where id < 32000000;
select * from user where id <= 32000000;
```

> - 1, 2: 下界测试;
> - 3, 4: 数值不在表格内，下界;
> - 5, 6: 数值在表格内，中间数据;
> - 7, 8: 上界测试
> - 9, 10: 数值不在表格内，上界

< <= char

```sql
select * from user where nick_name < 'john';
select * from user where nick_name <= 'john';
select * from user where nick_name < 'bdz';
select * from user where nick_name <= 'bdz';
select * from user where nick_name < 'mike';
select * from user where nick_name <= 'mike';
select * from user where nick_name < 'max';
select * from user where nick_name <= 'max';
select * from user where nick_name < 'zpq';
select * from user where nick_name <= 'zpq';
```

> - 1, 2: 下界测试;
> - 3, 4: 数值不在表格内，下界;
> - 5, 6: 数值在表格内，中间数据;
> - 7, 8: 上界测试
> - 9, 10: 数值不在表格内，上界

< <= float

```sql
select * from user where score < -5.0;
select * from user where score <= -5.0;
select * from user where score < -10.0;
select * from user where score <= -10;
select * from user where score < 75.6;
select * from user where score <= 75.6;
select * from user where score < 94.0;
select * from user where score <= 94.0;
select * from user where score < 100.5;
select * from user where score <= 100.5;
```

> - 1, 2: 下界测试;
> - 3, 4: 数值不在表格内，下界，4并没有显示表明小数点;
> - 5, 6: 数值在表格内，中间数据;
> - 7, 8: 上界测试
> - 9, 10: 数值不在表格内，上界

1.6.4 多条件搜索

鉴于1.6.3做了较为全面的搜索，我们这里只进行抽样测试

同类型，双条件

```
select * from user where id < 31902002 and id >= 31901001;
```

不同类型，多条件

```
select * from user where score >= 60.0 and id <= 31902000 and gender = 'M';
```

1.7 DELETE语句

单条件， 单条删除

```
delete from user where id = 31901001;
select * from user;
```

恢复数据

```
insert into user values (31901001, 'mike', 'M', 94.0);
```

单条件，多条删除

```
delete from user where id >= 31901002;
select * from user;
```

恢复数据

```
insert into user values (31901002, 'john', 'M', 75.6);
insert into user values (31902001, 'max', 'F', 33.3);
insert into user values (31902003, 'lucy', 'F', -5.0);
```

多条件，单条删除

```
delete from user where id <= 31901002 and score > 80.0;
select * from user;
```

恢复数据

```
insert into user values (31901001, 'mike', 'M', 94.0);
```

多条件，多条删除

```
delete from user where id > 31902000 and gender <> 'M';
select * from user;
```

恢复数据

```
insert into user values (31902001, 'max', 'F', 33.3);
insert into user values (31902003, 'lucy', 'F', -5.0);
```

全部删除

```
delete from user;
select * from user;
```

恢复数据

```
insert into user values (31901001, 'mike', 'M', 94.0);
insert into user values (31901002, 'john', 'M', 75.6);
insert into user values (31902001, 'max', 'F', 33.3);
insert into user values (31902003, 'lucy', 'F', -5.0);
```

Part 2 扩展功能测试

2.1 update

语句格式:

```
update 表名 set 列名1 = 值1, 列名2 = 值2, ... where 条件子句;
```

单条件，单条单列更新

```
update user set nick_name = 'bdz' where nick_name = 'john';
select * from user;
```

恢复数据

```
update user set nick_name = 'john' where nick_name = 'bdz';
```

单条件，多条单列更新

```
update user set gender = 'F' where gender = 'M';
select * from user;
```

恢复数据

```
update user set gender = 'M' where score > 50;
```

单条件，单条多列更新

```
update user set name = 'bdz', score = '91.5' where name = 'john';
select * from user;
```

恢复数据

```
update user set name = 'john', score = '94.0' where name = 'bdz';
```

多条件，多条多列更新

```
update user set score = 66.6, gender = 'X' where id >= 31901001 and nick_name <
'nick';
select * from user;
```

重建数据

```
delete from user;
insert into user values (31901001, 'mike', 'M', 94.0);
insert into user values (31901002, 'john', 'M', 75.6);
insert into user values (31902001, 'max', 'F', 33.3);
insert into user values (31902003, 'lucy', 'F', -5.0);
```

2.2 show相关语句

语句格式

```
show;
show table 表名;
show index 索引名;
```

说明：

1. 展示表、索引的简略信息；
2. 展示指定表格的信息，信息包含属性名、属性值类型，是否unique/是否为主键；
3. 展示指定索引信息，信息包含索引名，表名，属性名；

测试：

```
show;
show table user;
show index user_name_index;
```

2.3 help

语句格式：

```
help;
```

说明：

展示支持的语句功能提示。

测试：

```
help;
```

2.4 import

语句格式：

```
import 表名 from 文件路径 (属性名1 值属性1 可选unique指定,..., primary key(主键属性名));
```

执行导入文件的语句：

```
import import_test from ./test/test_data/test.csv (id int,name char(10)
unique,gender char(1),primary key(id));
show table import_test;
```

2.5 SQL查询优化

我们在执行select和delete的时候都做了查询优化，如果条件语句中涉及已建立索引的属性，将首先利用索引做第一轮搜索，进而判断以优化查询。

比较创立索引前后的搜索时间。

```
select * from import_test where name = 'qvzmy';
create index import_test_name_index on import_test (name);
select * from import_test where name = 'qvzmy';
```

2.6 output

语句格式：

```
export 路径 from 表名;
```

说明：

将指定的表格以csv的形式输出到指定路径下。

测试：

```
export ./test/test_data/user.csv from user;
```

Part 3 异常提示测试

在miniSQL使用过程中，有诸多异常检测提醒，包括但不限于：SYNTAX Error, INVAILD IDENTIFIER, INVALID VALUE等等，我们也对上述语句实际操作过程中可能出现的错误设置了错误排查，在这里做一个测试

3.1 SQL语法错误 SYNTAX ERROR

Type 1: 不符合SQL语法规范

```
select from user;
select from user
insert into user (31901003, 'ErrorTest', 'F', 88.88);
```

> 第一句没有包含指定的列；
>
> 第二句没有标志结尾的分号
>
> 第三句没有包含values关键字

Type 2: SQL语义解析不合理

```
create table errortest (id int, name char(10), primary key (stu_id));
import import_error_test from ./test/test_data/test.csv (id int,stu_name
char(10) unique,gender char(1),primary key(id));
insert into user (31901008);
```

> 第一句选择的primary key并不在表设定的属性中
>
> 第二句指定的第二个属性stu_name并不在读取的文件所含的属性中
>
> 第三句输入的值数量与对应表的属性数量不匹配

3.2 SQL参量错误 INVALID IDENTIFIER

Type 1-1: 表不存在

```
select * from fake_table;
```

> fake_table 不存在

Type 1-2: 表已经存在

```
create table user (id int, primary key (id));
import user from ./test/test_data/test.csv (id int,stu_name char(10)
unique,gender char(1),primary key(id));
```

> user表已经存在

Type 2: 属性名设定不正确

```
delete from user where stu_name = 'bdz';
```

> stu_name不存在

Type 3: 索引名字设定不正确

```
create index user_name_index on user (id);
```

> 同样的索引已经创建好了

3.3 SQL参数错误 INVALID VALUE

```
insert into user values (31901008, 'blablablablablablablabla', 'M', 66.66);
insert into user values (31901001, 'john', 'M', 66.66);
```

> nick_name过长
>
> unique属性重复

3.4 其他错误

除却以上正确使用过程中可能遇到的因用户不正确操作产生的警示，我们还考虑了因物理储存导致的数据储存的错误提示，大致包括：

- 表数据文件损坏或缺失
- 索引文件损坏或缺失
- 元数据文件损坏或缺失

会产生如下报错（仅展示部分），因较难通过命令行触发错误，因此这里暂不进行测试：

```
ERROR: The catalog folder is missing! The program exits and please rebuild the
folder!
ERROR: ALL the indices may be lost! Please check your files!
```

Part 4 退出

实际可用的有两种退出指令

```
quit;
exit;
```

这里我们采用实验指定的quit

```
quit;
```