

# 《数据库系统实验》

## 实验报告

题目	索引的使用和测试
姓名	刘硕
学号	16340154
班级	软件工程二班

### 一. 实验环境和工具

在 Win10 环境中利用 MySQLworkbench 6.3 数据库可视化工具完成实验。

### 二. 实验内容

- 1) 单记录插入
- 2) 查询记录并排序
- 3) 单记录查询
- 4) 对 test 表的 id 字段建立非聚集索引
- 5) 对 test 表的 mm 字段建立非聚集索引
- 6) 对 test 表的 id 字段建立唯一索引

### 三. 实验设计思路

#### 1. 索引简介

索引是创建在表上，对数据库一列或者多列进行排序的数据结构。通过索引，查询数据时不必读完所有的信息，只是查询索引列，所以索引可以提高查询的速度。

数据在磁盘上以块的形式存储，为了保证对磁盘操作的原子性，访问数据的时候会一并访问所有的数据块。磁盘上的数据块类似于链表，都包含一个数据项和一个指针，指针指向下一个节点的内存地址，并且在物理地址上不需要连续存储。很多记录只能做到按照一个字段排序，所以要查询某个未经排序的字段，就需要使用线性查找，就需要访问  $N/2$  个数据块，其中  $N$  指的是一个所涵盖的所有数据块。如果该字段是非键字段（不包含唯一值），那么就要搜索整个表空间。因此，对于排序的字段，可以使用二分查找，只需访问  $\log_2 n$  个数据块。这样数据库的性能自然就得到了提升。

索引的建立往往遵循以下几个原则：选择唯一性索引；为经常进行排序、分组联合的数据建立索引；为常作为查询条件的数据建立索引；限制索引数目，因为它需要占用内存；尽量使用数据量小的作为索引；尽量使用前缀来索引。

索引优点包括提升检索速度、对于有依赖关系的父表和子表的联合查询可以提升查询速度、使用分组和排序语句进行查询的时候可以查询时分组排序的时间、加速表之间的连接等等.....但是索引也会占用物理空间、增加创建和维护的时间。

## 2. 基本索引

索引的基本种类有普通索引、唯一索引、主键索引、组合索引和全文索引。

普通索引是最简单的索引方式，它没有任何限制。可以直接创建某表的索引，也可以在创建表索引的同时改变表结构、或者创建表的时候直接创建索引。

与普通索引不同，唯一索引的索引列允许有空值但是必须唯一。可以直接创建某表的唯一索引，也可以在创建表唯一索引的同时改变表结构、或者创建表的时候直接创建唯一索引。

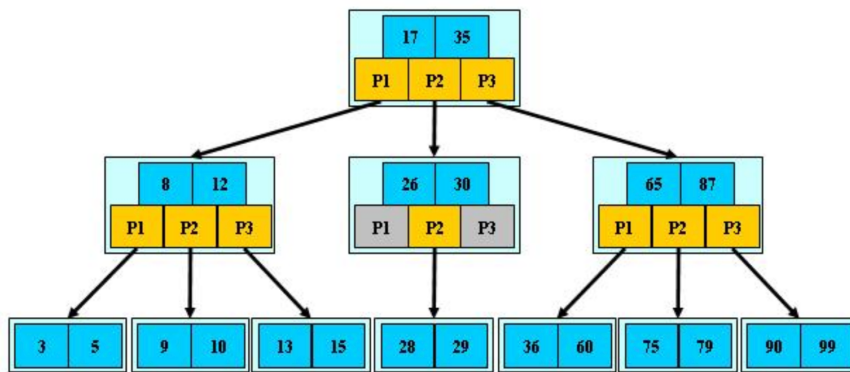
主键索引是一种特殊的唯一索引，一个表只能有一个主键，不能有空值。一般是在创建表的时候创建主键索引。

组合索引是在多个字段上创建的索引，当查询条件中使用了创建索引的第一个字段，索引才会被使用。

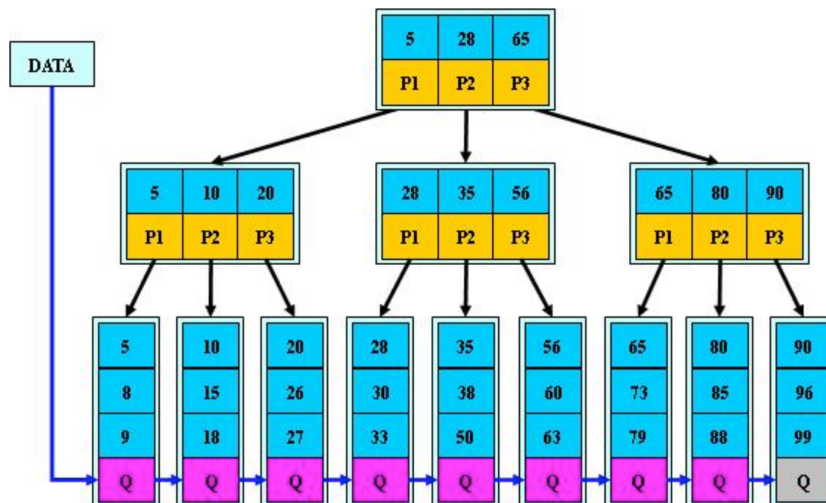
全文索引是用来查找文本中的关键字，而不是直接与索引值进行比较，只用在 `char`、`varchar`、`text` 上才可以建立全文索引。全文索引不是简单的 `where` 数值匹配，应该与 `match against` 混合使用。可以直接创建某表的全文索引，也可以在创建表全文索引的同时改变表结构、或者创建表的时候直接创建全文索引。

## 3. B 树、B+树、B\*树

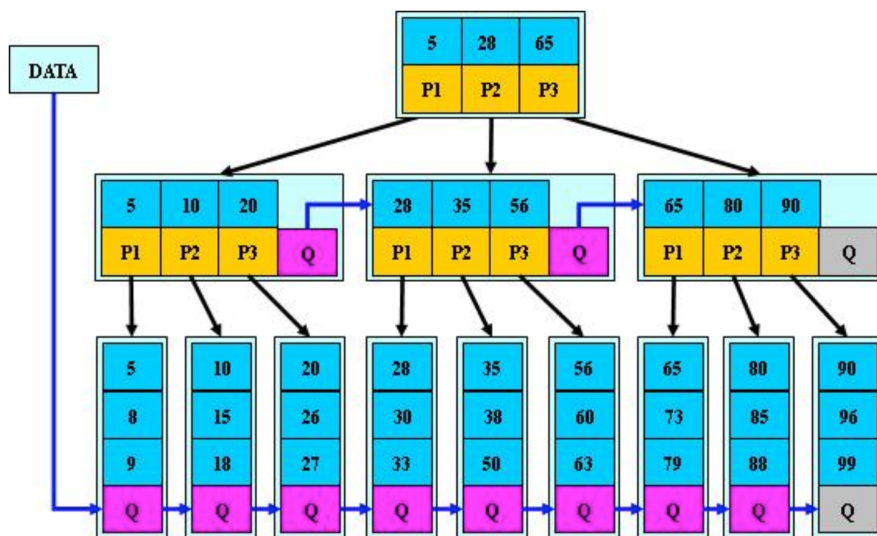
B 树是一种平衡多叉树，一个  $m$  阶 B 树是一棵  $m$  路平衡搜索树。B 树的根节点至少有两个子女；每个非根节点包含的关键字的个数不小于  $\lceil m/2 \rceil$  且不大于  $m-1$ ；除了根节点以外所有节点的度数等于关键字总数加 1；所有的叶子结点在同一层。B 树的关键字集合分布在整颗树中；任何一个关键字出现且只出现在一个结点中；搜索有可能在非叶子结点结束；其搜索性能等价于在关键字全集内做一次二分查找；自动层次控制。



B+树是一种树数据结构，是一个  $n$  叉树，每个节点通常有多个孩子，一棵 B+树包含根节点、内部节点和叶子节点。根节点可能是一个叶子节点，也可能是一个包含两个或两个以上孩子节点的节点。B+树所有关键字都出现在叶子节点的链表中（稠密索引），且链表中的关键字恰好是有序的；不可能在非叶子节点命中；非叶子节点相当于是叶子节点的索引（稀疏索引），叶子节点相当于是存储（关键字）数据的数据层。



B\*树是在 B+树中增加了兄弟节点之间的指针，从而更高效利用节点。B\*树分配新节点的概率更低，空间使用效率更高。



#### 4. 聚集索引和非聚集索引

聚集索引和非聚集索引都是在 B+树的数据结构基础上进行的高级索引。

聚集索引是一种特殊的索引，使数据按照索引的排序顺序放入表中，实际上重组了表的标准。当数据值按照范围查询的时候，聚集索引比较有效。如果需要进行大量数据的修改，不再适合用聚集索引。

非聚集索引建立了表中数据的逻辑顺序，但是记录的物理地址和索引不一定一致。非聚集索引的叶子层并不和实际数据页相重叠，而采用叶子层包含一个指向表中的记录在数据页中的指针方式。非聚集索引层次多，不会造成数据重排。

一个很好的聚集索引和非聚集索引的例子是：汉字字典中的拼音索引是聚集索引，因为索引的顺序和字典中汉字的顺序是一致的；而汉字字典中笔画索引是非聚集索引，虽然笔画可以客观反映汉子的一种逻辑顺序，但是索引的顺序和实际汉字排列的顺序不尽相同。

因此聚集索引和非聚集索引的根本差别在于——索引顺序是否等于实际数据排列顺序。

## 四. 实验完成情况

### 1. 创建示例表

创建示例表 test ，代码部分如下：

```
/**
```

```

* Create tables
*
* Author Nino Lau on 2018.6.15
*/

use jxgl;

CREATE TABLE IF NOT EXISTS test (id int unique AUTO_INCREMENT,
                                rq datetime null,
                                srq varchar(20) null,
                                hh smallint null,
                                mm smallint null,
                                ss smallint null,
                                num numeric(12,3),
                                primary key(id))
                                AUTO_INCREMENT=1 engine=MyISAM;

DELIMITER //
CREATE PROCEDURE `p1`()
begin
    set @i=1;
    WHILE @i<=80000 do
        INSERT INTO TEST(RQ, SRQ, HH, MM, SS, NUM)
            VALUE(NOW(), NOW(), HOUR(NOW()), MINUTE(NOW()),
                SECOND(NOW()), RAND(@i)*100);
        set @i=@i+1;
    END WHILE;
End //

call p1 //
DELIMITER ;

```

### 3. 单记录插入

```

/**
* Insert
*
* Author Nino Lau on 2018.6.15
*/

use jxgl;

DELIMITER //

```

```

Select @i:=max(id) from test;
INSERT INTO TEST(RQ,SRQ,HH,MM,SS,NUM)
VALUES(NOW(),NOW(),HOUR(NOW()),
MINUTE(NOW()),SECOND(NOW()),RAND(@i)*100); //

```

## 4. 查询记录

### 4.1 按 id 排序，查询所有记录

```

/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */

```

```
use jxgl;
```

```
Select * from test order by id;
```

	id	rq	srq	hh	mm	ss	num
	1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
	2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
	3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
	4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
	5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
	6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
	7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
	8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
	9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
	10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
	11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
	12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
	13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
	14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
	15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
	16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
	17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
	18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
	19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870

结果如图，发现记录确实按照 id 由小到大的顺序进行排列。

### 4.2 按 mm 排序，查询所有记录

```

/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */

```

```
use jxgl;
```

Select \* from test order by mm;

id	rq	srq	hh	mm	ss	num
1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
20	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

结果如图，发现记录确实按照 mm 由小到大的顺序进行排列。

### 4.3 单记录查询

```
/**  
 * Query  
 *  
 * Author Nino Lau on 2018.6.15  
 */
```

use jxgl;

Select id from test where id=51;

Result Grid		Filter Rows:
id		
51		
NULL		

结果如图，实现了单记录查询。

## 5. 对 test 表的 id 字段建立非聚集索引

### 5.1 对 test 表的 id 字段建立非聚集索引

```

/**
 * Insert
 *
 * Author Nino Lau on 2018.6.15
 */

```

use jxgl;

Create index indexname1 on test(id);

56	14:51:31	use jxgl	0 row(s) affected	0.000 sec
57	14:51:31	Create index indexname1 on test(id)	80001 row(s) affected Records: 80001 Duplicates: 0 Warnings: 0	0.344 sec

## 5.2 单记录插入

```

/**
 * Insert
 *
 * Author Nino Lau on 2018.6.15
 */

```

use jxgl;

```

DELIMITER //
Select @i:=max(id) from test;
INSERT INTO TEST(RQ,SRQ,HH,MM,SS,NUM)
VALUES(NOW(),NOW(),HOUR(NOW()),
MINUTE(NOW()),SECOND(NOW()),RAND(@i)*100); //

```

## 5.3 查询记录

### 5.3.1 按 id 排序，查询所有记录

```

/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */

```

use jxgl;

Select \* from test order by id;



	id	rq	srq	hh	mm	ss	num
	1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
	2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
	3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
	4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
	5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
	6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
	7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
	8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
	9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
	10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
	11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
	12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
	13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
	14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
	15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
	16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
	17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
	18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
	19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
	20	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

结果如图，发现记录确实按照 id 由小到大的顺序进行排列。

### 5.3.2 按 mm 排序，查询所有记录

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select * from test order by mm;
```

	id	rq	srq	hh	mm	ss	num
	1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
	2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
	3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
	4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
	5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
	6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
	7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
	8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
	9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
	10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
	11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
	12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
	13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
	14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
	15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
	16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
	17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
	18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
	19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
	20	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

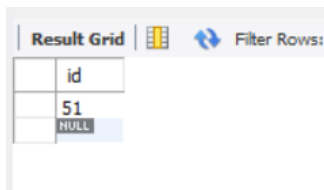
结果如图，发现记录确实按照 **mm** 由小到大的顺序进行排列。

### 5.3.3 单记录查询

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select id from test where id=51;
```



The screenshot shows a 'Result Grid' window with a single row of data. The column header is 'id' and the value is '51'. Below the value '51', the word 'NULL' is visible, likely indicating a null value in another column. The window also has a 'Filter Rows' button.

id
51

结果如图，实现了单记录查询。

## 5.4 删除索引

```
Drop index indexname1 on test;
```

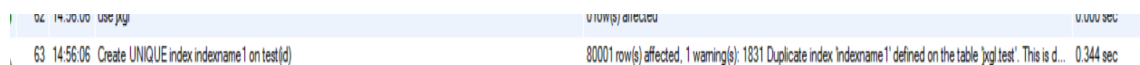
## 6. 对 **test** 表的 **mm** 字段建立非聚集索引

### 6.1 对 **test** 表的 **mm** 字段建立非聚集索引

```
/**
 * Insert
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Create index indexname1 on test(id);
```



The screenshot shows a command log with two entries. The first entry is a command to use the 'jxgl' database. The second entry is a command to create a unique index 'indexname1' on the 'id' column of the 'test' table. The log shows that 80001 rows were affected, with 1 warning about a duplicate index.

Command	Rows Affected	Warnings	Time
use jxgl;	0	0	0.000 sec
Create UNIQUE index indexname1 on test(id)	80001	1 warning(s): Duplicate index 'indexname1' defined on the table 'jxgl.test'. This is d...	0.344 sec

## 6.2 单记录插入

```
/**
 * Insert
 *
 * Author Nino Lau on 2018.6.15
 */

use jxgl;

DELIMITER //
Select @i:=max(id) from test;
INSERT INTO TEST(RQ,SRQ,HH,MM,SS,NUM)
VALUES(NOW(),NOW(),HOUR(NOW()),
MINUTE(NOW()),SECOND(NOW()),RAND(@i)*100); //
```

## 6.3 查询记录

### 6.3.1 按 id 排序，查询所有记录

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */

use jxgl;

Select * from test order by id;
```

	id	rq	srq	hh	mm	ss	num
	1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
	2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
	3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
	4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
	5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
	6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
	7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
	8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
	9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
	10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
	11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
	12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
	13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
	14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
	15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
	16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
	17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
	18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
	19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
	20	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

结果如图，发现记录确实按照 id 由小到大的顺序进行排列。

### 6.3.2 按 mm 排序，查询所有记录

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select * from test order by mm;
```

id	rq	srq	hh	mm	ss	num
1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
20	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

结果如图，发现记录确实按照 mm 由小到大的顺序进行排列。

### 6.3.3 单记录查询

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select id from test where id=51;
```

Result Grid		Filter Rows:
	id	
	51	
	NULL	

结果如图，实现了单记录查询。

## 6.4 删除索引

Drop index indexname1 on test;

删除索引。

## 7. 对 test 表的 id 字段建立唯一索引

### 7.1 对 test 表的 id 字段建立唯一索引

```
/**
 * Insert
 *
 * Author Nino Lau on 2018.6.15
 */
```

use jxgl;

Create Unique index indexname1 on test(id);

66	14:58:29	use jxgl	0 row(s) affected	0.000 sec
67	14:58:29	Create UNIQUE index indexname1 on test(id)	80001 row(s) affected, 1 warning(s): 1831 Duplicate index 'indexname1' defined on the table 'jxgl.test'. This is d...	0.360 sec

### 7.2 单记录插入

```
/**
 * Insert
 *
 * Author Nino Lau on 2018.6.15
 */
```

use jxgl;

DELIMITER //

Select @i:=max(id) from test;

```
INSERT INTO TEST(RQ,SRQ,HH,MM,SS,NUM)
VALUES(NOW(),NOW(),HOUR(NOW()),
MINUTE(NOW()),SECOND(NOW()),RAND(@i)*100); //
```

## 7.3 查询记录

### 7.3.1 按 id 排序，查询所有记录

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select * from test order by id;
```

	id	rq	srq	hh	mm	ss	num
	1	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
	2	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
	3	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
	4	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
	5	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
	6	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
	7	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
	8	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
	9	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
	10	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
	11	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
	12	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
	13	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
	14	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
	15	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
	16	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
	17	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
	18	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
	19	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
	20	2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

结果如图，发现记录确实按照 id 由小到大的顺序进行排列。

### 7.3.2 按 mm 排序，查询所有记录

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select * from test order by mm;
```

	id	rq	srq	hh	mm	ss	num
1		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.540
2		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.559
3		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.577
4		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.595
5		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.614
6		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.632
7		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.650
8		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.669
9		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.687
10		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.705
11		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.723
12		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.742
13		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.760
14		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.778
15		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.797
16		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.815
17		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	40.833
18		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	65.852
19		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	90.870
20		2018-06-15 14:34:32	2018-06-15 14:34:32	14	34	32	15.888

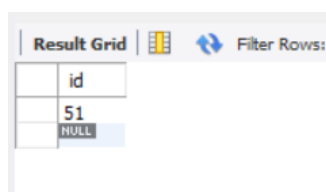
结果如图，发现记录确实按照 mm 由小到大的顺序进行排列。

### 7.3.3 单记录查询

```
/**
 * Query
 *
 * Author Nino Lau on 2018.6.15
 */
```

```
use jxgl;
```

```
Select id from test where id=51;
```



id
51
NULL

结果如图，实现了单记录查询。

## 7.4 删除索引

```
Drop index indexname1 on test;
```

删除索引。

## 五. 实验心得

这次实验主要是让我们熟悉数据库中的索引这个重要概念。因为课程进度较快，我本来想只是实现这个实验就好了，可是实验过程中一些问题让我对索引这个概念产生了兴趣。于是这个实验中期又进行了复习，包括基本的索引以及 B 树、B+树、B\*树等等。通过复习相关概念和数据结构，我不仅仅掌握的代码层的实现，同时对索引有了更深刻的了解。“纸上得来终觉浅，绝知此事要躬行”，我认为这个实验正好又帮助我更好的理解了聚集索引和非聚集索引的性质。另外，上文提到的汉字字典的例子，是我翻阅好多网站找到的。这个例子特别好地阐述了聚集索引和非聚集索引的区别。总之，这次实验让我受益匪浅。