mysql 索引

索引排序

要排序的字段上加入索引，可以提高速度。

多个单列索引**在**多条件查询**时只会生效**第一个**索引！所以**多条件联合查询时最好建联合索引！

不需要创建索引的场景

1. 对于那些在查询中很少使用或者参考的列不应该创建索引
2. 对于那 些只有很少数据值的列也不应该增加索引。
3. 当修改性能远远大于检索性能时，不应该创建索引。索引本身也占有内存，并且在修改数据时索引也要进行修改

字段适合创建索引

1. 数据量超过300的表应该有索引；经常与其他表进行连接的表，在连接字段上应该建立索引；经常连接查询，需要有索引
2. 经常出现在Where子句中的字段，加快判断速度，特别是大表的字段，
3. 经常用到排序的列上，因为索引已经排序。
4. 经常用在范围内搜索的列上创建索引，因为索引已经排序了，其指定的范围是连续的

索引类型

1. 主键索引

primary key() 要求关键字不能重复，也不能为null,同时增加主键约束
主键索引定义时，不能命名

2. 唯一索引

unique index() 要求关键字不能重复，同时增加唯一约束

3. 普通索引

index() 对关键字没有要求

4 组合索引：

将几个列作为一条索引进行检索，使用最左匹配原则。以**最左边的为起点**任何连续的索引都能匹配上

5. 全文索引

fulltext key() 关键字的来源不是所有字段的数据，而是字段中提取的特别关键字

```
建表：
creat table student(
    stu_id int unsigned not null auto_increment,
    name varchar(32) not null default '',
    phone char(11) not null default '',
    stu_code varchar(32) not null default '',
    stu_desc text,
    primary key ('stu_id'),     //主键索引
    unique index 'stu_code' ('stu_code'), //唯一索引
```

```
    index 'name_phone' ('name','phone'),  //普通索引，复合索引
    fulltext index 'stu_desc' ('stu_desc'), //全文索引
) engine=myisam charset=utf8;

更新：
alert table student
    add primary key ('stu_id'),     //主键索引
    add unique index 'stu_code' ('stu_code'), //唯一索引
    add index 'name_phone' ('name','phone'),  //普通索引，复合索引
    add fulltext index 'stu_desc' ('stu_desc'); //全文索引

删除：
alert table sutdent
    drop primary key,
    drop index 'stu_code',
    drop index 'name_phone',
    drop index 'stu_desc';
 查看索引  show index from 表名;
```

放弃索引的情况

1. 数据类型出现隐式转换的时候也不会使用索引， 当列的类型是字符串，那么一定记得在 where 条件中`把字符常量值用引号引起来`，否则即便这个列上有索引
2. %开头的插叙很自然就没法利用索引了
3. 复合索引的情况下，假如查询条件不包含索引列最左边部分，即不满足最左原则 leftmost，是不会使用复合索引的。
4. 用 or 分割开的条件，如果 or 前的条件中的列有索引，而后面的列中没有索引，那么涉及的索引都不会被用到。
5. order by的字段混合ASC和DESC：SELECT * FROM TAB_NAME ORDER BY KEY_PART1 DESC, KEY_PART2 ASC;
6. 对不同的关键字使用ORDER BY：SELECT * FROM TAB_NAME ORDER BY KEY1, KEY2; ？？？
7. 默认情况下，MySQL对所有的GROUP BY字段进行排序，如果查询包括GROUP BY但是用户想要避免排序结果的消耗，则可以指定ORDER BY NULL禁止排序。
8. 优化嵌套查询， 使用子查询可以一次性的完成很多逻辑上需要多个步骤才能完成的SQL操作，同时也可以避免事务或者表锁死。子查询可以被更有效的连接JOIN替代。
9. - FORCE INDEX

   强制MySQL使用某个索引，使用情况：当where子句取id>1的值，因为数据库中大部分库表都是大于1的，所以会全盘扫描，此时使用use index不可用，所以使用force index。

   比如：select * from tab_name force index(index_name) where id > 1;
10.

查看索引使用情况

如果索引正在工作， Handler_read_key 的值将很高，这个值代表了一个行被索引值读的次数，很低的值表名增加索引得到的性能改善不高，因为索引并不经常使用。
Handler_read_rnd_next 的值高则意味着查询运行低效，并且应该建立索引补救。这个值的含义是在数据文件中读下一行的请求数。如果正在进行大量的表扫描，Handler_read_rnd_next 的值较高，则通常说明表索引不正确或写入的查询没有利用索引，具体如下。

```
MySQL [sakila]> show status like 'Handler_read%';
+-----------------------+-------+
| Variable_name         | Value |
+-----------------------+-------+
| Handler_read_first    | 1     |
| Handler_read_key      | 5     |
| Handler_read_last     | 0     |
| Handler_read_next     | 200   |
| Handler_read_prev     | 0     |
| Handler_read_rnd      | 0     |
| Handler_read_rnd_next | 0     |
+-----------------------+-------+
```