大于 >：

 EXPLAIN select uid, user\_name from recommend\_user  where uid >215     
返回：

id  select\_type   table         type      possible\_keys          key        key\_len       ref      rows        Extra

1   SIMPLE       recommend\_user       ALL       uid                    NULL        NULL         NULL      8          Using where  
没有走索引UID

当我 只查询UID

EXPLAIN select uid from recommend\_user  where uid >215

返回：

id         select\_type         table                 type          possible\_keys          key       key\_len       ref        rows     Extra

1           SIMPLE             recommend\_user        range         uid                    uid       4             NULL       6        Using where; Using index

走了UID索引

总结：

不等于 ！=  和大于>的时候 只有搜索的是 索引列（并只有索引列） 才会用到索引，<, <=, =,不会有这种情况

order by:

explain select id from collect where vtype=1 order by id asc;

+----+-------------+---------+------+---------------+-------+---------+-------+-------+-----------------------------+

| id | select\_type | table   | type | possible\_keys | key   | key\_len | ref   | rows | Extra                       |

+----+-------------+---------+------+---------------+-------+---------+-------+-------+-----------------------------+

| 1 | SIMPLE      | collect | ref | vtype         | vtype | 5       | const | 93237 | Using where; Using filesort |

+----+-------------+---------+------+---------------+-------+---------+-------+-------+-----------------------------+

看上面，最后的Extra 部分：竟然出现 Using filesort ！ 我晕，全表查询。但是 prossible keys 已经使用到了vtype啊？ 看来并不是where 条件的问题。

为了测试，将2条语句分别执行下看：  
mysql> select id from collect where vtype=1 order by id asc limit 0,20;  
+----+  
| id |  
+----+  
| 1 |  
| 2 |  
| 3 |  
| 4 |  
| 5 |  
| 6 |  
| 7 |  
| 8 |  
| 9 |  
| 10 |  
| 11 |  
| 12 |  
| 13 |  
| 14 |  
| 15 |  
| 16 |  
| 17 |  
| 18 |  
| 19 |  
| 20 |  
+----+  
20 rows in set (10.28 sec)  
查找20条数据，花了 10.28 秒！ 非常的晕。  
再看：  
mysql> select id from collect where vtype=1 limit 0,20;  
+----+  
| id |  
+----+  
| 1 |  
| 2 |  
| 3 |  
| 4 |  
| 5 |  
| 6 |  
| 7 |  
| 8 |  
| 9 |  
| 10 |  
| 11 |  
| 12 |  
| 13 |  
| 14 |  
| 15 |  
| 16 |  
| 17 |  
| 18 |  
| 19 |  
| 20 |  
+----+  
20 rows in set (0.01 sec)  
少了order by 基本上秒查！  
为什么会出现以上问题呢？order by id asc, id 字段可是主键啊，按理说应该是非常快的索引，但是mysql 好像并没有用到。再测试  
mysql> explain select \* from collect order by id asc;  
+----+-------------+---------+------+---------------+------+---------+------+--------+----------------+  
| id | select\_type | table   | type | possible\_keys | key | key\_len | ref | rows   | Extra          |  
+----+-------------+---------+------+---------------+------+---------+------+--------+----------------+  
| 1 | SIMPLE      | collect | ALL | NULL          | NULL | NULL    | NULL | 103997 | Using filesort |  
+----+-------------+---------+------+---------------+------+---------+------+--------+----------------+  
看上面的分析，我直接select \* order by id asc ,也是用到了filesort ，用到了这个肯定查询在10秒以上了。  
mysql> explain select id from collect order by id asc;  
+----+-------------+---------+-------+---------------+---------+---------+------+--------+-------------+  
| id | select\_type | table   | type | possible\_keys | key     | key\_len | ref | rows   | Extra       |  
+----+-------------+---------+-------+---------------+---------+---------+------+--------+-------------+  
| 1 | SIMPLE      | collect | index | NULL          | PRIMARY | 4       | NULL | 103997 | Using index |  
+----+-------------+---------+-------+---------------+---------+---------+------+--------+-------------+  
而上面的，将\* 改成id ,Using index 了，看来直接查找id 会使性能提高。  
mysql> explain select id,url,title from collect order by id asc;  
+----+-------------+---------+------+---------------+------+---------+------+--------+----------------+  
| id | select\_type | table   | type | possible\_keys | key | key\_len | ref | rows   | Extra          |  
+----+-------------+---------+------+---------------+------+---------+------+--------+----------------+  
| 1 | SIMPLE      | collect | ALL | NULL          | NULL | NULL    | NULL | 103997 | Using filesort |  
+----+-------------+---------+------+---------------+------+---------+------+--------+----------------+  
1 row in set (0.00 sec)  
而再加上另外的字段，order by 就没任何性能提高了。还是Using fielsort ！  
从上面可以看出order by 不能乱用！不知道网上哪个傻逼说用了limit 最好用order by 这样会让分页的性能加快！简直是放屁。经过实际测试，limit 和order by 没有任何关联！而以前写程序的时候经常会用order by id asc 。 可能是数据从来没有上过10万条的缘故，感觉还行，今天用了10万条采集数据做测试，慢得一塌糊涂！  
好了，现在仔细想想order by 用到索引的场合：  
1） 如果select 只查询索引字段，order by 索引字段会用到索引，要不然就是全表排列；  
2） 如果有where 条件，比如where vtype=1 order by vtype asc . 这样order by 也会用到索引！  
3） 综上，如果order by 真的影响limit的话，那么就请在没有where 查询的时候order by id(主键), 有where 查询的时候，order by (索引) 字段。  
4） 别迷信网上文章，多explain，并且要相信mysql 依然很坚挺，别出现慢的情况就想，免费的性能就这样  
慎用 order by 大于 不等于。。。。。