MongoDB 性能优化之分页查询

最常见的分页采用的是skip+limit这种组合方式，这种方式对付小数据倒也可以，但是对付上几百上千万的大数据，只能力不从心。通过如下思路改善，可以大大提高查询速度：条件查询+排序+限制返回记录。边查询，边排序，排序之后，抽取第一次分页中的最后一条记录，作为第二次分页的条件，进行条件查询，以此类推....
下面通过一个例子演示一下：

1.准备数据

```
>for(var i = 0 ;i < 2000000;i++)
{db.user.save({"index":i,"name":"zhangsan"+i,"age":(i+10)%100});}
>db.user.count();
```

2.采用skip+limit方式查询

查询以zhangsan2开头的，并按照index倒叙的第100~109条数据

```
>
db.user.find({name:/^zhangsan2/}).sort({"index":-1}).skip(100).limit(10).explain();
{
        "cursor" : "BasicCursor",
        "isMultiKey" : false,
        "n" : 10,
        "nscannedObjects" : 2000000,
        "nscanned" : 2000000,
        "nscannedObjectsAllPlans" : 2000000,
        "nscannedAllPlans" : 2000000,
        "scanAndOrder" : true,
        "indexOnly" : false,
        "nYields" : 3,
        "nChunkSkips" : 0,
        "millis" : 2223,
        "indexBounds" : {

        },
        "server" : "100.205:27017"
}
```

可以看到查询一次需要2223ms。

3.查询下一页优化处理

当时使用上面方法，查询110~119条时，依旧耗时2530ms

```
>
db.user.find({name:/^zhangsan2/}).sort({"index":-1}).skip(110).limit(10).explain();
```

```
{
        "cursor" : "BasicCursor",
        "isMultiKey" : false,
        "n" : 10,
        "nscannedObjects" : 2000000,
        "nscanned" : 2000000,
        "nscannedObjectsAllPlans" : 2000000,
        "nscannedAllPlans" : 2000000,
        "scanAndOrder" : true,
        "indexOnly" : false,
        "nYields" : 3,
        "nChunkSkips" : 0,
        "millis" : 2530,
        "indexBounds" : {

        },
        "server" : "100.205:27017"
}
```


下面我们通过程序改善一下上面下一页的查询，根据第一次100查询，我们知道index的最小值为299890，由于是按照index的倒叙查询，所以下一页的数据为index小于299890的后续10条，查询如下：


```
> db.user.find({name:/^zhangsan2/,index:{"$lt":299890}}).sort({"index":-1}).limit(10).explain();
{
        "cursor" : "BasicCursor",
        "isMultiKey" : false,
        "n" : 10,
        "nscannedObjects" : 2000000,
        "nscanned" : 2000000,
        "nscannedObjectsAllPlans" : 2000000,
        "nscannedAllPlans" : 2000000,
        "scanAndOrder" : true,
        "indexOnly" : false,
        "nYields" : 1,
        "nChunkSkips" : 0,
```

```
        "millis" : 1523,
        "indexBounds" : {

        },
        "server" : "100.205:27017"
}
```

比较一下与上面的查询下一页的效率有显著的提高，当数据越大的情况下，效果会更明显。