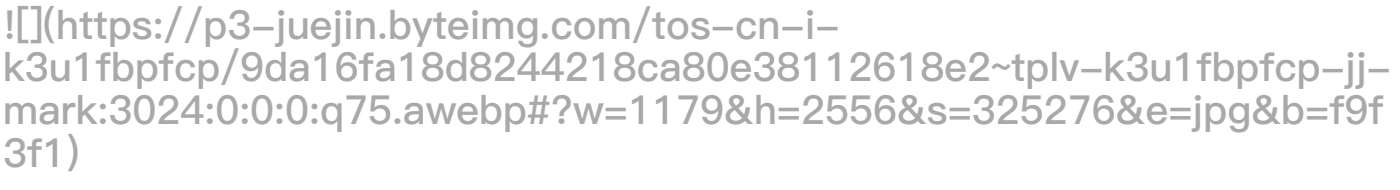
教你写一个电商商品排行榜功能

> 在电商平台上，会有很有榜单的信息，比如新品榜、畅销榜。通过这些榜单，可以让用户直观的对比各个商品之间的销量对比，促使更多的用户下单或者加购。本文介绍如何实现一个简单的榜单功能

榜单定义

在某多 app 上，可以看到下面的小米手表排行榜：



需要实现几个功能：

* 榜单是针对某类商品进行统计，比如手机排行是统计品类为手机的商品。
* 排行统计可以是按照某一个维度，比如订单量、一个月回购量、或者几个指数汇总成一个分数进行排行。上图就是根据热卖指数进行排行。
* 除了排行之外，还需要展示霸榜的天数，比如霸榜榜首 n 天、霸榜前三 n 天。上面排名第一就展示了霸榜榜首的天数。

榜单实现思路

统计榜单至少需要两张表，榜单主表和榜单明细表。

* 主表记录榜单信息，比如`xx月xx手机畅销榜`，统计维度、统计时间范围等等。
* 明细表主要记录，商品有信息、排行、统计销量、分数等信息。

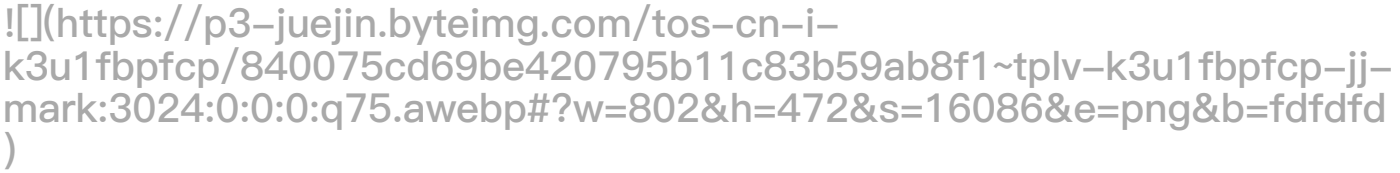
创建好榜单之后，每天定时跑统计任务，将更新榜单明细表中。统计的维度

，需要根据具体需求来查询数据库，然后在计算分数，最后计算排名。

以上实现比较简单，主要是去除相关数据进行汇总计算。最重要的一个功能就是霸榜天数的求解。

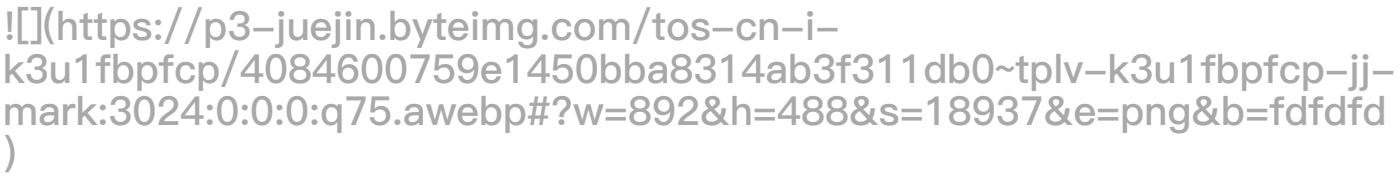
霸榜天数
====

霸榜分成两种：霸榜榜首 n 天、霸榜前三 n 天。这就需要多加一个表，榜单明细日志表，记录每天榜单明细。比如以下记录三天的榜单明细日志记录：



霸榜天数是**从后往前**统计，需要统计连续的天数，首先定位到最后一天，也就是4日，前三的商品 C、A、B 三个商品。统计榜首，就一直排在榜首，直到前面不是榜首的商品。比如上面的图中，4日榜首是商品 C,这就往前推算，3日是不是商品 C，如果不是霸榜1天。如果是的话，就继续往前查，天数累加，直到统计的是非商品。

霸榜前三就相对复杂一点，需要统计两个数据，也就是排名三的往前统计，因为榜首已经统计了，所以只需要统计第2和第3的商品，也就是商品 A 和商品 B。往前找数据：



统计榜首

创建榜单明细日志类：

```
```

```
@Data
@AllArgsConstructor
class RankDetailLog{

    /**
     * 商品
     */
    private String product;

    /**
     * 排行
     */
    private Integer rank;

    /**
     * 时间
     */
    private String createTimeStr;

}
```

创建模拟数据:

```
RankDetailLog detailLog7 = new RankDetailLog("C",1,"7月4日");
RankDetailLog detailLog8 = new RankDetailLog("A",2,"7月4日");
RankDetailLog detailLog9 = new RankDetailLog("B",3,"7月4日");
RankDetailLog detailLog1 = new RankDetailLog("D",1,"7月2日");
RankDetailLog detailLog2 = new RankDetailLog("B",2,"7月2日");
RankDetailLog detailLog3 = new RankDetailLog("C",3,"7月2日");
RankDetailLog detailLog4 = new RankDetailLog("B",1,"7月3日");
RankDetailLog detailLog5 = new RankDetailLog("C",2,"7月3日");
RankDetailLog detailLog6 = new RankDetailLog("A",3,"7月3日");

List<RankDetailLog> detailLogList = new ArrayList<>();
detailLogList.add(detailLog1);
detailLogList.add(detailLog2);
detailLogList.add(detailLog3);
detailLogList.add(detailLog4);
detailLogList.add(detailLog5);
detailLogList.add(detailLog6);
detailLogList.add(detailLog7);
detailLogList.add(detailLog8);
```

```
detailLogList.add(detailLog9);
```

分组排序：

```
// 按日期分组并排序
Map<String, List<RankDetailLog>> sortedCreateTimeMap =
detailLogList.stream()
        .collect(Collectors.groupingBy(RankDetailLog::getCreateTimeStr, ()
-> new TreeMap<>(Comparator.reverseOrder()), Collectors.toList()));
```

统计榜首：

```
String topProduct = null;
Integer topProductNum = 0;
for (Map.Entry<String, List<RankDetailLog>> entry :
sortedCreateTimeMap.entrySet()) {
    List<RankDetailLog> rankDetailLogList = entry.getValue();
    if (!rankDetailLogList.isEmpty()) {
        RankDetailLog topLog = rankDetailLogList.get(0);
        String currentTopProduct = topLog.getProduct();
        if (topProduct == null) {
            topProduct = currentTopProduct;
            topProductNum = 1;
        } else {
            if (topProduct.equals(currentTopProduct)) {
                topProductNum++;
            } else {
                break;
            }
        }
    }
}
System.out.println("榜首商品：" + topProduct + "，天数：" +
topProductNum);
```

输出：C:1 表示商品霸榜榜首1天。

步骤详解：

* topProduct 标记榜首商品，topProductNum 榜首天数记录。
* 遍历集合，记录榜首商品，然后天数设置1。遍历数据，如果是连续的数据，数量 +1。找不到商品就结束遍历。

统计前三

前三的统计其实是排除了榜首，也就是只统计最新数据的第2和第3的商品。往前汇总统计。

```
Set<String> threeProductSet = new HashSet<>();
Map<String,Integer> threeProductMap = new HashMap<>();
boolean first = true;
for (Map.Entry<String, List<RankDetailLog>> entry :
sortedCreateTimeMap.entrySet()) {
    List<RankDetailLog> rankDetailLogList = entry.getValue();
    if (!rankDetailLogList.isEmpty()) {
        // 只取前三数据
        rankDetailLogList =
rankDetailLogList.subList(0,Math.min(3,rankDetailLogList.size()));
        if (first) {
            for (int i = 0; i < rankDetailLogList.size(); i++) {
                RankDetailLog detailLog = rankDetailLogList.get(i);
                String topThreeProduct = detailLog.getProduct();
                if (i >= 1) {
                    threeProductSet.add(topThreeProduct);
                    threeProductMap.put(topThreeProduct,0);
                }
            }
            first = false;
        } else {
            Set<String> currentThreeProductSet = new HashSet<>();
            for (RankDetailLog detailLog : rankDetailLogList) {
                String topThreeProduct = detailLog.getProduct();
                if (threeProductMap.containsKey(topThreeProduct)) {
threeProductMap.put(topThreeProduct,threeProductMap.get(topThreePro
duct) + 1);
```

```
                    currentThreeProductSet.add(topThreeProduct);
                }
            }
            threeProductSet.removeAll(currentThreeProductSet);
        }
    }
}
System.out.println(threeProductMap);

```

输出：{A=2, B=3}

* 获取到最新商品详情，统计第2和第3的商品，存 threeProductSet 以及计算器集合 threeProductMap 中。
* 为了统计连续性，使用 currentThreeProductSet 存当天的商品信息，再和前面的商品交集，交集的数据就是有连续性商品数据。

总结
==

* 商品榜单需要根据不同的统计维度、参数、统计时间来设计商品榜单表结构。再根据配置的信息，生成榜单详情信息。
* 一般都使用定时方式生成榜单详情信息。
* 需要统计连续霸榜天数，一般有两种方式，一种是统计榜首的连续天数，另一种是前三的连续天数。
+ 统计榜首首先获取最新的榜首商品，商品往前找，连续性的找到前面的数据，如果非连续性就查找结束。
+ 统计前三排除了榜首之外，就是统计第2和第3的商品。获取最新的商品并记录，往前遍历，使用 set 的存储每次遍历的商品，在使用集合的交集来保证统计的连续性。

 原文链接: https://juejin.cn/post/7388444606458282003