**现有模式：**

监控中，每间隔N(3)秒，运行一段运算，监控完整运算规则：

第一步， 调用大盘计算存储过程， 大致逻辑为:

先建立一个最终结果临时表，

分别得到 “竞标中”和“未在竞标中”（没有买也没有卖） 的商品，

遍历处理“竞标中”的商品，得到每个需要得到的值， 放入最终结果。

最后处理那些“未在竞标中”中的商品，也并入最终结果。

最终将结果，插入数据表“AAA\_DaPanPrd\_Temp”

第二步， 监控从数据表“AAA\_DaPanPrd\_Temp”中，去除需要的数据进行简单处理，

然后进行压缩保存至服务器上的二进制文件中。

**分析：**

这种模式， 存在的问题大致为：

问题1： 遍历得到需要的值时，单行单个商品运算就比较占资源，随着数据量变大。

问题2： 监控间隔极短的时间就运行一次，即使没有业务发生，也同样会进行计算占用资源，造成无故占用。

问题3： 无法方便进行拓展拆分运算，运算是顺序执行，导致运算时间较长，若服务器资源未满，无法充分利用服务器空闲资源。

问题4： 最终应该将大盘数据准确变化时间，尽可能的短，目前模式无论服务器运算能力多强，都无法很快得结果。

问题5： 最终应该确保随着业务量的增加，只需要简单的拼接服务器即可拓展。代码部署和编写思路基本不用重写，只需要变动业务逻辑部分。

**为了解决这五个问题，新思路大致如下：**

首先尽可能优化单条计算的效率， 大致从sql语法、索引优化角度入手。

建立一个“大盘数据变化时间对照表”。表结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 商品编号 | 最后一次执行统计时间 | 最后一次执行统计时间(开始预存) | 最后一次有效数据变更时间 |
|  | 4:00 |  | 5:00 |

当某商品编号发生了 投标单下达成功、投标单撤销成功、预订单下达成功、预订单撤销成功、有中标数据产生时， 依据商品编号，在“大盘数据变化时间对照表”更新“最后一次有效数据变更时间”(用数据库服务器自带的当前时间更新)，商品编号不存在时，插入新数据,插入时，“最后一次执行统计时间”字段使用 “2000-1-1”。

在监控中， 以100毫秒间隔的进行反复处理。处理逻辑：

从“大盘数据变化时间对照表”中，先处理商品表中有，但时间对照表中没有的，插入。再更新符合条件的预存时间。

最后获取“最后一次有效数据变更时间”晚于或等于“ 最后一次执行统计时间”的数据，没有符合条件的数据就忽略本轮。 获取后，对于更新符合条件的,更新“大盘数据变化时间对照表”中该商品“ 最后一次执行统计时间（开始预存）”为当前时间。

为这些数据进行分组，每组1到10行(视情况验证和调整)，为每组数据，单独开启一个独立并发线程，在线程中计算这组数据内的每一条对应的业务数值。 每个线程内，以一组商品编号作为参数，执行“AAA\_DaPanPrd”存储过程，在存储过程中只计算这一组商品编号的相关数据。

“AAA\_DaPanPrd”存储过程中，每组数据中的每行计算出结果的一刻(不用管其他数据和其他线程)，即在存储过程中直接操作“AAA\_DaPanPrd\_Temp”表(这个表的结构与数据现在用的完全相同),操作时，先删除该商品编号有关的所有数据，再插入根据业务规则计算出来的该商品新数据，由于存在三个合同期限，所以根据不同情况可能会插入1到3条数据，没有合同期限的那种两个横杠的，也插入进去。插入成功后，将“最后一次执行统计时间”更新为“最后一次执行统计时间(开始预存)”时间。

确保当本轮的所有线程都跑完后， 立刻取出“AAA\_DaPanPrd\_Temp”表的数据，压缩二进制保直接保存到数据库服务器硬盘。最后，按照指定间隔执行下一轮。

也可以不用等这一轮跑完就可以开始下一轮，这样实时性更强，但占用资源较高。 目前先不允许重叠轮次。

这样，二进制文件“HomePageByte.txt”和“HomePageByte\_jhxp.txt” 即可在很短的时间内不断更新， 若数据有变化的商品很多很频繁，只需要增加sql集群的节点，即可自动分散运算量。 业务逻辑还是集中在“AAA\_DaPanPrd”存储过程中。