|  |
| --- |
| 说明书摘要 |

一种基于定位的疫情接触情况的计算查询方法，该方法基于目标标签的查询需求，根据对实时定位数据的计算结果，从而获得目标标签的接触情况，包括如下步骤：在实时定位的过程中，计算每一条实时定位数据对应的位置状态，并插入到预设的映射表中；对于当前时刻的实时定位数据中的每一个标签，根据其位置状态与映射表，计算与其他标签的最近距离，并将最近距离作为实时定位数据的新增字段的值保存到数据库中；响应于目标标签的查询需求，在数据库中根据新增字段对实时定位数据进行过滤。本发明的有益效果在于：大大减少了数据库的读取数量；加快了对疫情接触情况的计算查询速度；防止因查询量过大导致的数据库崩溃。

|  |
| --- |
| 摘要附图 |



|  |
| --- |
| 权利要求书 |

1.一种基于定位的疫情接触情况的计算查询方法，所述方法基于目标标签的查询需求，根据实时定位数据，从而获得目标标签的接触情况，目标标签的查询需求包括：goal\_tagid，begintime、endtime、safe\_distance与safe\_time，goal\_tagid为待查询的目标标签，begintime与endtime为待查询的时间区间，safe\_distance为亲密接触的临界距离，safe\_time为亲密接触的临界持续时间，其特征在于，包括如下步骤：

步骤1，在实时定位数据所在的数据库表中新增一个字段为CONTACT\_DIS，代表最近接触距离；

步骤2，根据预设的最大安全距离max\_dis\_safe，在实时定位的过程中，对每一条实时定位数据计算对应的位置状态state，并插入到预设的第一映射表中，其中所述实时定位数据包括：tagid、timestamp、sceneid、x与y；

步骤3，对于当前时刻的实时定位数据中的每一个标签，根据其位置状态state与第一映射表，计算与其他标签的最近距离，并将最近距离保存到数据库CONTACT\_DIS字段中；

步骤4，响应于目标标签goal\_tagid的查询需求，在数据库中获取在待查询的时间区间内CONTACT\_DIS小于safe\_distance的目标标签goal\_tagid的实时定位数据；

步骤5，删掉所述目标标签goal\_tagid的实时定位数据中数组长度小于（safe\_time/p）的，p为实时定位数据的定位频率；

步骤6，根据当前目标标签goal\_tagid的实时定位数据的时间戳，查找其他所有标签在所述时间戳下的实时定位数据，并输出目标标签的接触情况。

2.根据权利要求1所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，所述max\_dis\_safe为10m。

3.根据权利要求1或2所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，所述第一映射表为散列表或二叉树，第一映射表的key值为state=[sceneid,xx,yy]，其中，xx=[x/max\_dis\_safe]，yy=[y/max\_dis\_safe]，第一映射表的value值为一个数组或一个链表，数组或链表中的每一个元素为一个tagid。

4.根据权利要求3所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，state=sceneid\*1000\*1000+xx\*1000+yy，且实施定位数据的x与y限定在（0,1000m）之间。

5.根据权利要求1所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，所述步骤3具体包括：

步骤S31，对于当前时刻的实时定位数据中的每一个标签，根据其state与第一映射表找到需要计算距离的其他标签，其中，所述其他标签满足接触判断公式：

（state-1<=state\_other<=state+1）&&

（[state/1000]-1<=[state\_other/1000]<=[state/1000]+1）&&

（[state/1000000]==[ state\_other/1000000]）

其中，state\_other为所述其他标签的state；

步骤S32，对所述每一个标签逐个计算与所述其他标签的距离，选出最小距离作为contact\_dis；

步骤S33，将contact\_dis作为所述实时定位数据的CONTACT\_DIS字段插入到数据库中。

6.根据权利要求5所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，

步骤2还包括：根据预设的第二数值与第三数值，在实时定位的过程中，分别对每一条当前时刻的实时定位数据计算对应的第二位置状态与第三位置状态，并分别插入到预设的第二映射表与第三映射表中，其中第二数值小于max\_dis\_safe，第三数值小于第二数值；

步骤S31还包括：若其他标签的数量大于N，利用第二位置状态与第二映射表获得第二其他标签；若第二其他标签的数量大于0且小于等于N，用第二其他标签替代其他标签进入步骤S42；若第二其他标签的数量大于N，利用第三位置状态与第三映射表获得第三其他标签；若第三其他标签的数量大于0，用第三其他标签替代其他标签进入步骤S42；若第三其他标签的数量等于0，用第二其他标签替代其他标签进入步骤S42；N取10至100。

7.根据权利要求6所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，第二数值与第三数值分别是3m与1m。

8.根据权利要求6所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，N=10。

9.根据权利要求1所述的一种基于定位的疫情接触情况的计算查询方法，其特征在于，所述步骤4具体包括：

步骤S41，响应于目标标签的查询需求，在数据库中执行语句：goal\_tagid=TAGID && begintime<TIMESTAMP && TIMESTAMP<=endtime && CONTACT\_DIS<safe\_dis；

步骤S42，对时间戳连续的实时定位数据以数组的形式进行保存。

10.根据权利要求1、5或6所述的一种基于定位的疫情接触情况的计算查询方法，根据标签绑定人员的所在区域，对标签ID进行划分，其特征在于，

步骤1还包括：在实时定位数据所在的数据库表中新增一个字段为CONTACT\_ID；

步骤4还包括：获取所述其他标签中的min\_other\_tagid与max\_other\_tagid，分别代表其他标签中的数值最小与数值最大的标签ID，将min\_other\_tagid与max\_other\_tagid作为实时定位数据的CONTACT\_ID保存到数据库中；

步骤7还包括：根据CONTACT\_ID中的min\_other\_tagid与max\_other\_tagid，过滤掉不属于[min\_other\_tagid，max\_other\_tagid]区间的实时定位数据。

|  |
| --- |
| 说明书 |

一种基于定位的疫情接触情况的计算查询方法

**技术领域**

本发明属于定位领域，更具体的，涉及一种基于定位的疫情接触情况的计算查询方法。

**背景技术**

随着定位技术的成熟，该技术越来越多的应用在智慧城市、智慧园区、智慧监狱以及智慧办公的场景中。近两年疫情频繁，因此，如何借助定位信息判断人与人之间的安全距离是亟需解决的问题。

在相关技术中，通过定位信息来判断人与人之间的安全距离，可以直接从数据库中获取所有人员的位置信息，然后分析判断出人与人之间的接触情况，进而锁定安全距离。但是，这种做法非常耗时，且对数据库的压力很大，甚至会导致数据库或后台程序崩溃。

**发明内容**

为解决现有技术中存在的不足，本发明的目的在于，减少数据库读取数据的数量，以及节约时间防止数据库崩溃，进而提出一种基于定位的疫情接触情况的计算查询方法。

本发明采用如下的技术方案。

一种基于定位的疫情接触情况的计算查询方法，所述方法基于目标标签的查询需求，根据实时定位数据，从而获得目标标签的接触情况，目标标签的查询需求包括：goal\_tagid，begintime、endtime、safe\_distance与safe\_time，goal\_tagid为待查询的目标标签，begintime与endtime为待查询的时间区间，safe\_distance为亲密接触的临界距离，safe\_time为亲密接触的临界持续时间，包括如下步骤：

步骤1，在实时定位数据所在的数据库表中新增一个字段为CONTACT\_DIS，代表最近接触距离；

步骤2，根据预设的最大安全距离max\_dis\_safe，在实时定位的过程中，对每一条实时定位数据计算对应的位置状态state，并插入到预设的第一映射表中，其中所述实时定位数据包括：tagid、timestamp、sceneid、x与y；

步骤3，对于当前时刻的实时定位数据中的每一个标签，根据其位置状态state与第一映射表，计算与其他标签的最近距离，并将最近距离保存到数据库CONTACT\_DIS字段中；

步骤4，响应于目标标签goal\_tagid的查询需求，在数据库中获取在待查询的时间区间内CONTACT\_DIS小于safe\_distance的目标标签goal\_tagid的实时定位数据；

步骤5，删掉所述目标标签goal\_tagid的实时定位数据中数组长度小于（safe\_time/p）的，p为实时定位数据的定位频率；

步骤6，根据当前目标标签goal\_tagid的实时定位数据的时间戳，查找其他所有标签在所述时间戳下的实时定位数据，并输出目标标签的接触情况。

进一步的，所述max\_dis\_safe为10m。

进一步的，所述第一映射表为散列表或二叉树，第一映射表的key值为state=[sceneid,xx,yy]，其中，xx=[x/max\_dis\_safe]，yy=[y/max\_dis\_safe]，第一映射表的value值为一个数组或一个链表，数组或链表中的每一个元素为一个tagid。

进一步的，state=sceneid\*1000\*1000+xx\*1000+yy，且实施定位数据的x与y限定在（0,1000m）之间。

进一步的，所述步骤3具体包括：

步骤S31，对于当前时刻的实时定位数据中的每一个标签，根据其state与第一映射表找到需要计算距离的其他标签，其中，所述其他标签满足接触判断公式：

（state-1<=state\_other<=state+1）&&

（[state/1000]-1<=[state\_other/1000]<=[state/1000]+1）&&

（[state/1000000]==[ state\_other/1000000]）

其中，state\_other为所述其他标签的state；

步骤S32，对所述每一个标签逐个计算与所述其他标签的距离，选出最小距离作为contact\_dis；

步骤S33，将contact\_dis作为所述实时定位数据的CONTACT\_DIS字段插入到数据库中。

进一步的，步骤2还包括：根据预设的第二数值与第三数值，在实时定位的过程中，分别对每一条当前时刻的实时定位数据计算对应的第二位置状态与第三位置状态，并分别插入到预设的第二映射表与第三映射表中，其中第二数值小于max\_dis\_safe，第三数值小于第二数值；

步骤S31还包括：若其他标签的数量大于N，利用第二位置状态与第二映射表获得第二其他标签；若第二其他标签的数量大于0且小于等于N，用第二其他标签替代其他标签进入步骤S42；若第二其他标签的数量大于N，利用第三位置状态与第三映射表获得第三其他标签；若第三其他标签的数量大于0，用第三其他标签替代其他标签进入步骤S42；若第三其他标签的数量等于0，用第二其他标签替代其他标签进入步骤S42；N取10至100。

进一步的，第二数值与第三数值分别是3m与1m。

进一步的，N=10。

进一步的，所述步骤4具体包括：

步骤S41，响应于目标标签的查询需求，在数据库中执行语句：goal\_tagid=TAGID && begintime<TIMESTAMP && TIMESTAMP<=endtime && CONTACT\_DIS<safe\_dis；

步骤S42，对时间戳连续的实时定位数据以数组的形式进行保存。

进一步的，根据标签绑定人员的所在区域，对标签ID进行划分；

步骤1还包括：在实时定位数据所在的数据库表中新增一个字段为CONTACT\_ID；

步骤4还包括：获取所述其他标签中的min\_other\_tagid与max\_other\_tagid，分别代表其他标签中的数值最小与数值最大的标签ID，将min\_other\_tagid与max\_other\_tagid作为实时定位数据的CONTACT\_ID保存到数据库中；

步骤7还包括：根据CONTACT\_ID中的min\_other\_tagid与max\_other\_tagid，过滤掉不属于[min\_other\_tagid，max\_other\_tagid]区间的实时定位数据。

本发明的有益效果在于，与现有技术相比，本发明具有以下优点：

1. 大大减少了数据库的读取数量。
2. 加快了疫情接触情况的计算查询速度。
3. 防止数据库或后台程序崩溃。

**附图说明**

图1是一种基于定位的疫情接触情况的计算查询方法的流程图。

**具体实施方式**

下面结合附图对本申请作进一步描述。以下实施例仅用于更加清楚地说明本发明的技术方案，而不能以此来限制本申请的保护范围。

在相关场景中，数据库会保存每一个标签某一时刻的位置信息。通常，实时定位数据表（简称为hispos表）至少包含如下字段：TAGID、TIMESTAMP、SCENEID、X与Y。其中，TAGID代表标签ID，TIMESTAMP表示时间戳，SCENEID代表场景ID，X与Y代表TAGID在TIMESTAMP下的二维坐标。也就是说，实时定位数据表中的每一条实时定位数据，均对应一个TAGID、TIMESTAMP、SCENEID、X与Y。表1示例性展示了定位技术下数据库存储实时定位数据的表的基本格式。

**表1**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| TAGID | TIMESTAMP | SCENEID | X | Y |
| 101 | 1536682067 | 6 | 22.3 | 77.6 |
| 101 | 1536682068 | 6 | 22.7 | 80.3 |
| 101 | 1536682069 | 6 | 23.4 | 81.5 |
| 102 | 1536682067 | 4 | 48.6 | 33.7 |
| 102 | 1536682068 | 4 | 49.2 | 32.2 |
| 102 | 1536682069 | 4 | 50.0 | 32.6 |
| 103 | 1536682067 | 6 | 33.3 | 98.4 |
| 103 | 1536682068 | 6 | 33.1 | 99.9 |
| 103 | 1536682069 | 6 | 33.4 | 100.6 |
| 104 | 1536682067 | 4 | 59.6 | 36.5 |
| 104 | 1536682068 | 4 | 60.2 | 36.2 |
| 104 | 1536682069 | 4 | 61.8 | 36.3 |

需要说明的是，TAGID在实际应用下代表的是挂在脖子上的定位工牌，也可以代表固定在手上的定位手表，也可以是固定在需定位的人员上的电子标签等。实际上，数据库中会有一张标签表，里面记录了标签的类型以及绑定的信息（例如，绑定的是人还是物品，绑定的人的相关信息是什么等等）。为了统一说明，下面将工牌、手表与标签统一用标签替代。SCENEID可以看成是区域号码。例如，一个公司可能有很多层楼，每一层楼都需要不同的基站对该层楼内的标签进行定位。因此，可以对每一层楼用一个不同的SCENEID进行区分。可以理解的是，不同SCENEID下的标签默认为是不可能接触的。

在疫情大数据的项目中，客户有一个最基本的需求就是：查看一个标签（例如，对一个单位进行查询）在某一个时间段内一共与哪些其他标签进行亲密接触。

通常，需求的输入为goal\_tagid，begintime、endtime、safe\_distance与safe\_time这5个参数。需求的输出为多组亲密接触数据，每一组亲密接触数据至少包括：other\_tagid、contact\_begintime这2个查询结果。

其中，goal\_tagid代表待查询的目标标签ID，通常可以是那些被确诊阳性的患者；begintime与endtime为开始时间段与结束时间段，例如endtime可以是goal\_tagid被确诊的时刻，而begintime可以是endtime往前推14天或者21天；safe\_distance为亲密接触的临界距离，例如可以是2m；safe\_time为亲密接触的临界持续时间，例如可以是5秒；other\_tagid为与goal\_tagid亲密接触的一个其他标签ID；contact\_begintime为开始接触时间。也就是说，当goal\_tagid在begintime至endtime的时间段内，与另外一个标签保持了至少safe\_time以上的接触时间，且接触时间内接触距离均小于safe\_distance。那么该另外一个标签与开始接触的时刻就是上述输出结果（即分别为other\_tagid与contact\_begintime）。

在相关技术中，为了实现对上述需求的查询，需要把所有标签在begintime至endtime内的实时定位数据给提取出来。以begintime是endtime往前推14天为例，并假设该单位有1000个人，每天上班时间为10h。又假设实时定位数据的定位频率为1，即每隔1s记录一下位置信息。那么，本次查询至少需要从数据库读取1000\*（14\*10\*3600\*1）=504000000条数据。需要说明的是，这种从硬盘（数据库通常存储在硬盘中）往内存读取数据的操作是非常耗时的。因此，即便是性能再好的数据库，为了一次查询而读取5亿条数据也力所不逮。究其原因，在于没有对历史的实时定位数据进行归纳整理，从而导致只能粗暴的全部遍历式查找。

鉴于上述分析，本公开实施例提出了一种方法，使得在查询该需求时不再需要从数据库提取海量的实时定位数据。同时，不增加数据库额外的存储负担。

步骤S1，初始化过程中，在hispos表中新增一个字段为CONTACT\_DIS，该字段用于记录一条实时定位数据中tagid在timestamp下与最近标签的距离。需要说明的是，为了精准专业的进行表述，当表示数据库字段的时候（表1的第一行），用大写字母进行说明（例如，TAGID、TIMESTAMP、SCENEID、X、Y与CONTACT\_DIS）；当表示具体的实时定位数据时（表1的第2行及往后），改为小写字母（例如，tagid、timestamp、sceneid、x、y与contact\_dis）。为了适应不同类型病毒的不同的安全临界距离，假设各个病毒的safe\_distance最大不可能超过10m。以下均用最大安全距离值max\_dis\_safe表示10m。

步骤S2，在实时定位的过程中，对每一条当前时刻的实时定位数据计算对应的位置状态state，并插入到预设的映射表中。程序开始之初会初始化一张映射表map（本质为散列表或二叉树），其key值为state，其value值为一个数组或一个链表，数组或链表中的每一个元素为一个tagid。需要说明的是，state可以是一个向量 [sceneid,xx,yy]，其中，xx=[x/max\_dis\_safe]，yy=[y/max\_dis\_safe]，[]表示取整符号。优选的，state最好是一个数值，从而加快key值的计算。由于实际场景下，定位超过1000m必然处于不同的sceneid，所以在一个sceneid下，是不可能出现坐标跨度大于1000m的情况。此外，由于坐标都是相对的，因此，软件完全有能力设定坐标系，使得坐标x与y的范围只能在（0,1000m）之间。基于这种思路，state的值可以是state=sceneid\*1000\*1000+xx\*1000+yy。需要说明的是，映射表中实时定位数据的timestamp是相等的。当timestamp刷新时，映射表需要清空。

步骤S3，对于当前时刻的实时定位数据中的每一个标签，根据其state与映射表计算与其他标签的最近距离，即contact\_dis。并插入到数据库中。具体包括：

步骤S31，对于当前时刻的实时定位数据中的每一个标签，根据其state与映射表找到需要计算距离的其他标签，其中，该其他标签满足接触判断公式：

（state-1<=state\_other<=state+1）&&

（[state/1000]-1<=[state\_other/1000]<=[state/1000]+1）&&

（[state/1000000]==[ state\_other/1000000]）

其中，state\_other为其他标签的state。

步骤S32，对每一个标签逐个计算其与其他标签的距离，选出最小距离，即contact\_dis。

步骤S33，将contact\_dis作为实时定位数据的CONTACT\_DIS字段插入到数据库的hispos表中。

步骤S4，当客户的需求输入时，在数据库中获取goal\_tagid在begintime至endtime之间contact\_dis小于safe\_dis的目标标签的实时定位数据。即数据库的搜索语句为“goal\_tagid=TAGID && begintime<TIMESTAMP && TIMESTAMP<=endtime && CONTACT\_DIS<safe\_dis”。这些实时定位数据按照TIMESTAMP从小到大排列，并在内存中，对连续timestamp的实时定位数据以数组的形式进行保存。因此，可以理解的是，目标标签的实时定位数据是由多个数组组成，每一个数组均由连续的timestamp的实时定位数据构成。

步骤S5，删掉目标标签的实时定位数据中数组长度小于（safe\_time/p）的。其中，p为实时定位数据的定位频率，例如在表1中，定位频率p为1。

步骤S6，针对现有的目标标签的实时定位数据的时间戳，查找其他所有标签在该时间戳下的实时定位数据并输出最终结果。也就是说，相比于相关技术中的全部查找，本公开过滤掉了时间戳不吻合的实时定位数据。只针对性的查找时间戳可能存在密切接触的实时定位数据。

说明，由于本发明在实时定位的过程中对每一个标签的位置进行了简单标注，因此，在查找客户需求时，没必要对整个时间段内的所有标签的实时定位数据进行查询，只需要查询目标标签存在接触的时间段内的实时定位数据。极大的减少了数据库的读取压力。当客户的需求中safe\_dis的值越小，safe\_time的值越大，本发明的技术方案所能节省的时间越多！效果越好！

在实际项目工程中，标签ID往往体现了其绑定的人员的基本信息。例如智慧办公等场景下，员工长期只会待在同一个地方。例如，可以对在18楼办公的所有员工的标签ID均以18开头，而对在19楼办公的所有员工的标签ID均以19开头。假设18楼办公区与19楼办公区的工位分别是150个与250个，那么给18楼员工与19楼员工分发的标签ID号可以分别是18001~18150与19001~19250。很显然，这种通过区域来划分标签ID的做法是非常通用的。基于此，本公开实施例的方案可以进一步的优化：

在步骤S1，初始化过程中，在hispos表中还要新增一个字段CONTACT\_ID。

相应的，在步骤S3，对于当前时刻的实时定位数据中的每一个标签，根据其state计算与其他标签的最近距离之后，还需要获取其他标签中的min\_other\_tagid与max\_other\_tagid。其中，min\_other\_tagid与max\_other\_tagid为其他标签中的数值最小与最大的标签ID。将min\_other\_tagid与max\_other\_tagid保存在字段CONTACT\_ID中。加入其他标签的数量只为一个，那么min\_other\_tagid与max\_other\_tagid为同一个数字。

相应的，在步骤S6中，可以根据CONTACT\_ID的区间，过滤掉不属于CONTACT\_ID的区间的实时定位数据。例如，标签18007与标签18011、18067、18080这三个标签满足接触判断公式，那么标签18007对应的contact\_id为18011:18080。那么在步骤S7中进行查找的时候，只需要查找从18011至18080所有标签的实时定位数据即可。

当很多标签聚集在一块的时候，属于散列表的最坏情况，这将导致步骤S4中“根据其state计算与其他标签的最近距离”是极其耗时的。鉴于此，本算法对此给出了一种优化策略。

在步骤S2中，对每一条当前时刻的实时定位数据（timestamp相等）计算对应的第二位置状态与第三位置状态，并分别插入到第二映射表与第三映射表。与（第一）映射表的唯一区别在于，（第一）映射表中的max\_dis\_safe为10m，而第二映射表与第三映射表的max\_dis\_safe为3m与1m。需要说明的是，此处的3m与1m的设计遵循自然常数e的衰减规律，从而实现用最少的映射表满足最高的计算效率。

在步骤S31中，若其他标签的数量大于N，利用第二位置状态与第二映射表获得第二其他标签，所述第二其他标签满足接触判断公式（此时公式中的state均为第二位置状态）。若第二其他标签的数量大于0且小于等于N，用第二其他标签替代其他标签进入步骤S32。若第二其他标签的数量大于N，利用第三位置状态与第三映射表获得第三其他标签，所述第三其他标签满足接触判断公式（此时公式中的state均为第三位置状态）。若第三其他标签的数量大于0，用第三其他标签替代其他标签进入步骤S32；若第三其他标签的数量等于0，用第二其他标签替代其他标签进入步骤S32。通常，N的值可以取10~100。一般N优选为10。

通过这种以空间（多了2张映射表，空间消耗翻2倍）换时间（从期望上讲，由于第三映射表的max\_dis\_safe只有第一映射表的十分之一，这意味着第三映射表的其他标签比第一映射表的其他标签减少了99%）的方法，使标签之间最近距离的计算速度大大提高。

本发明申请人结合说明书附图对本发明的实施示例做了详细的说明与描述，但是本领域技术人员应该理解，以上实施示例仅为本发明的优选实施方案，详尽的说明只是为了帮助读者更好地理解本发明精神，而并非对本发明保护范围的限制，相反，任何基于本发明的发明精神所作的任何改进或修饰都应当落在本发明的保护范围之内。

|  |
| --- |
| 说明书附图 |



**图1**