**版本修订批准记录**

佳都新太科技股份有限公司

XXXX年XX月

多维数据整合V0.5

时空碰撞&轨迹查询

设计说明书

2019年3月

【 历次版本修订记录，版本修订描述重要说明】

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **日期** | **版本号** | **版本修订描述** | **作者** | **审核** | **批准** |
| 2020-10-16 | V0.5 | 初建 | 贾志忠 |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**注：版本修订批准记录，如不需要可删除此页.**

**目录**

[1 范围 1](#_Toc2080)

[1.1 标识 1](#_Toc19129)

[1.2 系统概述 2](#_Toc15313)

[1.3 文档概述 2](#_Toc3857)

[2 引用文档 2](#_Toc28199)

[3 功能需求/业务设计 3](#_Toc8466)

[4 设计目标 3](#_Toc18366)

[4.1 功能目标 3](#_Toc17824)

[4.1.1 要素时间维度统计 3](#_Toc9658)

[4.1.2 要素时空分布统计 3](#_Toc10714)

[4.1.3 实体强关系分析 3](#_Toc23328)

[4.1.4 console模块 4](#_Toc19853)

[4.1.5 查询比对 4](#_Toc26107)

[4.1.6 碰撞关联 4](#_Toc1600)

[4.1.7 轨迹查询 4](#_Toc32442)

[4.1.8 关系挖掘 4](#_Toc1951)

[4.1.9 关系拓展 4](#_Toc23400)

[4.1.10 研判数据管理及数据碰撞 5](#_Toc1108)

[4.2 性能目标 5](#_Toc3497)

[4.2.1 数据规模预估 5](#_Toc21134)

[4.2.2 系统硬件配置 5](#_Toc22949)

[4.2.3 非功能性能指标 5](#_Toc6590)

[4.2.4 可靠性需求 6](#_Toc5447)

[4.2.5 离线统计分析性能目标 6](#_Toc31581)

[4.2.6 实时统计分析性能目标 6](#_Toc23164)

[5 设计思路 7](#_Toc30384)

[5.1 技术选型 7](#_Toc27728)

[5.1.1 任务调度 7](#_Toc2025)

[5.1.2 任务并行执行框架 7](#_Toc28194)

[5.1.3 REST服务框架 7](#_Toc17689)

[5.2 重点难点分析 8](#_Toc17731)

[5.2.1 关系查询后，如何提供数据反查性能 8](#_Toc4035)

[5.2.2 如何提高多时空数据碰撞的性能 9](#_Toc187)

[5.2.3 关系数据的存储方式 11](#_Toc26652)

[5.3 风险评估 12](#_Toc16395)

[5.3.1 抓拍人脸的身份确认 12](#_Toc5947)

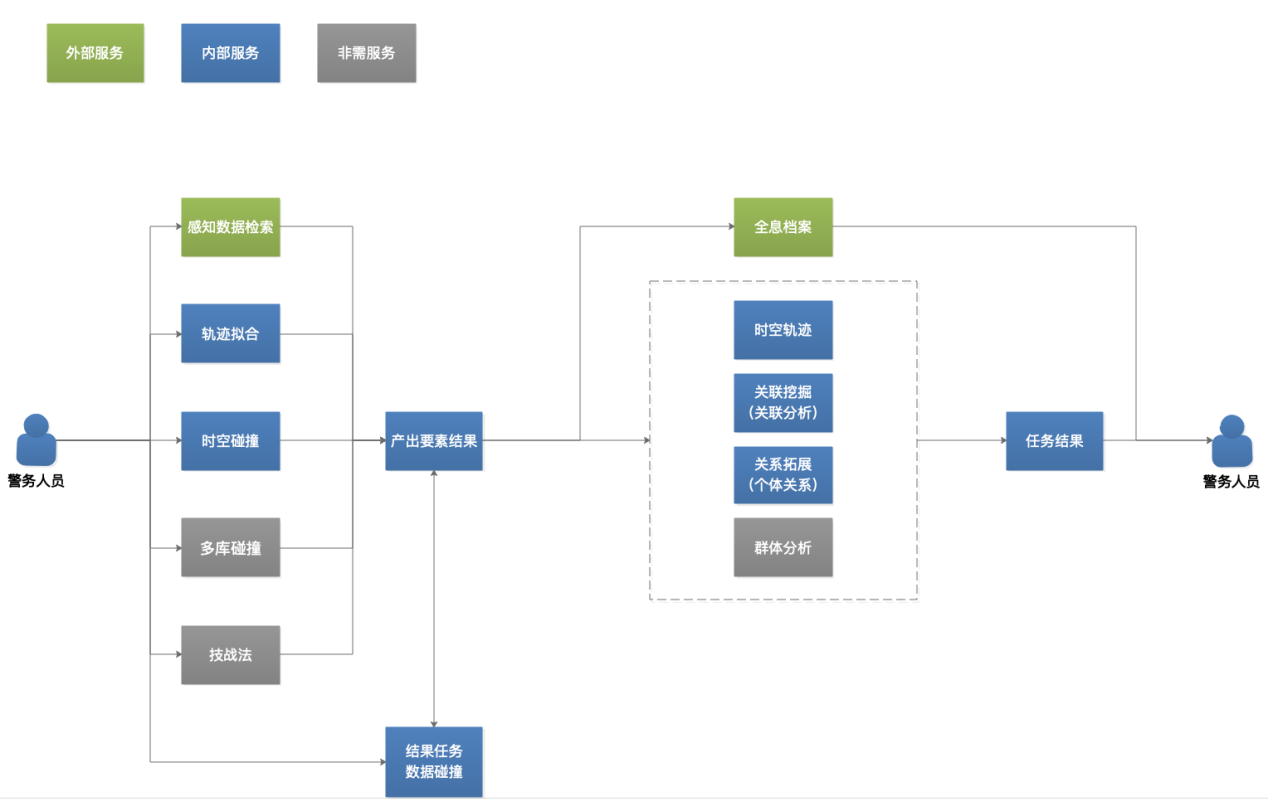
[5.3.2 要素时空分布统计的风险 12](#_Toc6433)

[6 架构设计 17](#_Toc20416)

[6.1 架构设计 17](#_Toc16095)

[6.1.1 逻辑架构图 17](#_Toc8381)

[6.1.2 整体业务流程 18](#_Toc29921)

[ 18](#_Toc13545)

[6.1.3 整体数据流程 18](#_Toc615)

[ 19](#_Toc26721)

[6.1.4 系统架构图 19](#_Toc9607)

[6.1.5 物理部署图 20](#_Toc20871)

[7 子系统/模块划分及设计说明 21](#_Toc20768)

[7.1 规则统计模块 21](#_Toc32189)

[7.1.1 涉及的技术点 22](#_Toc4892)

[7.2 数据挖掘模块 23](#_Toc26893)

[7.2.1 离线数据挖掘分析 23](#_Toc11777)

[7.2.2 实时数据分析 23](#_Toc12286)

[7.3 console模块 23](#_Toc2028)

[7.3.1 关系定义的管理 23](#_Toc13605)

[7.3.2 规则统计配置管理 24](#_Toc25183)

[7.3.3 规则统计执行记录 24](#_Toc55)

[7.4 研判任务管理 24](#_Toc10637)

[7.5 REST服务 24](#_Toc2721)

[7.5.1 直接关系查询服务 25](#_Toc24334)

[7.5.2 关系挖掘或拓展服务 26](#_Toc3576)

[7.5.3 单表时空碰撞服务 27](#_Toc796)

[7.5.4 对象时空分布统计查询服务 28](#_Toc19125)

[7.5.5 研判结果的数据碰撞服务 29](#_Toc14537)

[7.5.6 多实体间关系分析 29](#_Toc12835)

[8 数据结构 30](#_Toc1226)

[8.1 关联相关 30](#_Toc29796)

[8.1.1 关系定义元数据结构 30](#_Toc17363)

[8.1.2 关系存储结构 31](#_Toc15355)

[8.2 规则统计相关 32](#_Toc7319)

[8.2.1 配置表 32](#_Toc27193)

[8.2.2 执行实例表 33](#_Toc4831)

[8.3 数据研判任务 34](#_Toc1150)

[8.3.1 研判任务表 34](#_Toc19550)

[8.3.2 研判任务分析结果表 34](#_Toc16439)

[9 待确认问题 35](#_Toc23854)

范围

标识

* 缩略语

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 缩略语 | 说明/定义 | 英文全称 | 中文译名 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

* 名词术语

|  |  |  |
| --- | --- | --- |
| 序号 | 名称 | 说明/定义 |
| 1 | 实体 | 客观世界存在的，对实践活动或认识活动的承担者，如人员、车辆、手机、SIM卡等。 |
| 2 | 要素 | 能够唯一标识实体的最小单位，如人员的身份证件号码、人员的人像照片。 |
| 3 | 关联 | 同一实体不同要素之间的连接，如人员的身份证件号码与人像照片。 |
| 4 | 关系 | 不同实体的相同或不同要素之间的连接，如人员的身份证件号码与车辆的车牌号，人员甲的身份证件号码与人员乙的身份证件号码。 |
| 5 | 关系挖掘 | 针对不同实体、不同要素之间（但可能为同一归属人或使用人）的关系分析，如甲的身份证件号码与甲登记或驾驶车辆的关系。根据已知要素，挖掘可能与其相关的其他要素，用于全面刻画与已知要素相关的线索。 |
| 6 | 关系拓展 | 针对不同实体、相同要素之间的关系分析，如甲的身份证件号码与乙的身份证件号码的关系、甲的车辆与乙的车辆的关系。根据已知要素，拓展与之有关系的相同要素，用于扩展与已知要素相关的其他线索。 |
| 7 | 强关系 | 指由同一个记录中不同实体要素直接推导出来的关系，如根据车辆登记数据，直接推导出的车牌号码与车辆登记人的关系 |
| 8 | 弱关系 | 指根据数据统计及分析算法，间接推导出来的关系，如根据数据碰撞推导出来的车车同行、人人同行关系 |

* 本文文件适用的集成系统和软件系统是：XXX系统

统系概述

在AI赋能、大数据迅猛发展的时代背景下，在各种新形势的变化中，公安智慧新警务强调“警种赋能，警务革新”。公安视频专网逐渐发展成为城市物联网的趋势，亟待解决当前公安视频图像智能化建设应用存在的数据挖掘不深、业务逻辑不清等问题。

建立视频监控、人脸抓拍、车辆卡口、电子围栏、智能门禁等视频图像数据与公安业务数据的融合数据体系，实现视频图像多要素解析比对和多维度关联分析，提升挖掘视频图像价值信息的能力，赋能各警种应用和工作方式创新。

融合视频录像、人脸抓拍、车辆卡口、门禁设备及电围设备等感知数据，横向打破烟囱式数据组织结构，串联多部门、多行业的多源异构数据，纵向基于感知数据的深度挖掘，通过自动分析、综合，实现“关键领域、关键要素、关键过程”的状态估计和身份刻画，进而实现关系挖掘和轨迹分析，最终为态势分析和风险评估等警务决策活动提供循证支持。

文档概述

本文档基于业务需求进行相关的业务设计和技术选型，指导开发人员进行相应业务功能开发。

# 引用文档

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 名称 | 发布单位 | 发布时间 |
| 1 | 公安大数据处理GA/DSJ 200-2019 | 公安部 | 2019-3 |
| 2 | 公安视频图像信息应用系统GA/T 1400 | 公安部 | 2017-5 |
| 3 | 公安视频图像智能化建设应用指南 | 公安部科信局 | 2020-4 |
| 4 | 关于进一步加强视频图像智能化建设服务全警全域应用的指导意见 | 广东省公安厅科信处 | 2020-9 |

# 功能需求/业务设计

参见《多维数据融合v1.0(0925).docx》

# 设计目标

功能目标

**要素时间维度统计**

针对人、车、码的关键要素按时间进行次数统计，如按小时统计每个车牌号码被抓拍的次数，时间的统计颗粒度是小时、天。

**要素时空分布统计**

要素分布库主要包含要素最后分布和要素分布变迁时序。

要素最后分布主要是用于记录各类关键要素最后的分布情况，包括最后出现的时间以及最后出现的 区域，区域可以按照每个地方不同的关注粒度进行抽象，抽象粒度至少要到市行政区划一级。

要素分布变迁时序主要是记录了要素的历史分布明细，是要素最后分布的历史明细表。记录了要素 的历史连续活动区域，以及在该区域内连续活动的最早时间、最晚时间，活动的总天数等。

要素最后分布主要是解决要素最后出现地的问题，要素分布变迁时序主要是解决要素历史曾经出现的区域问题。

注意：

* 因为要素最后分布，没有相关的需要涉及，所以不要此部分设计
* 目前的区域只支持设备，如果要考虑实现支持政务的行政区划(省、市、区、街道、社区、网格),则需要在设备基础表或原始数据中，增加对应的字段。

**实体强关系分析**

统计登记类业务数据、感知轨迹类数据中的强关系。

**console模块**

1. 提供如下3个配置管理功能：

* 要素时间维度统计配置管理
* 要素时空分布统计配置管理
* 强关系分析配置管理

1. 任务执行历史列表：根据选择的日期范围、任务类型【维度统计、强关系分析、时空分布统计】，查看任务的执行情况
2. 任务执行失败列表: 根据选择的日期、任务类型【维度统计、强关系分析、时空分布统计】，查看执行失败或未执行的任务，可以针对失败的任务重新执行：全部或指定的任务列表

**查询比对**

以人员、车辆、手机为重点对象，通过要素标识或要素特征实现单体目标的查询、比对、查询结果信息展示

**碰撞关联**

根据时空条件输入，分析符合在多个时间、区域范围条件内的人员、车辆、侦码集合，返回符合条件的人员信息、车辆信息和侦码信息

注意：至少2个时空条件，每个时空条件的时间跨度不能大于1天

**轨迹查询**

在查询比对或碰撞关联输出的实体要素列表中选择将待分析的实体要素，选择后自动添加至时间操作轴，以时间、空间为过滤条件，展示要素活动区域或要素活动轨迹

**关系挖掘**

针对不同实体、不同要素之间的关系分析。可通过待分析要素，挖掘检索关联的人员、车辆和侦码。系统关联分析建立关系，动态关系通过多点位同时出现的要素建立，赋予置信度；静态关系通过登记数据建立。

**关系拓展**

针对不同实体、相同要素之间的关系分析。可通过待分析要素，拓展检索关联的人员、车辆和侦码。系统关联分析建立关系，动态关系通过同行、同乘、同场所行为、共用等算法建立，赋予置信度；静态关系通过登记数据建立。

**研判数据管理及数据碰撞**

研判任务管理记录分析任务选定的要素，可查看、编辑本用户检索、分析的要素集合，实现多任务的交并差数据碰撞分析。

性能目标

**数据规模预估**

数据规模估测：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | 区县级（150万人） | | 地市级（500万人） | | 发达地市级（2000万人） | |
| 路数 | 单日增量 | 路数 | 单日增量 | 路数 | 单日增量 |
| 人脸抓拍 | 200 | 0.5万/路 | 1000 | 0.5万/路 | 10000 | 0.3万/路 |
| 视频门禁 | 50 | 0.5万/路 | 200 | 0.5万/路 | 15000 | 0.01万/路 |
| 车辆卡口 | 1000 | 0.7万/路 | 3000 | 0.5万/路 | 8000 | 0.7万/路 |
| WIFI探针 | 100 | 5万/路 | 4000 | 5万/路 | 10000 | 3万/路 |
| 电子围栏 | 200 | 10万/路 | 500 | 10万/路 | 2000 | 10万/路 |
| 业务系统 | 10 | 5万/个 | 20 | 10万/个 | 20 | 10万/个 |

**系统硬件配置**

需要提供如下：梁鑫文，冯刚提供

* Snowballdb集群节点数及节点配置
* es集群节点数及节点配置
* hadoop集群节点数及节点配置

**非功能性能指标**

* 查询比对
* 结果响应时间≤3秒
* 时空轨迹
* 时间轴、地图点位统计响应时间≤3秒，聚合时间≤1秒
* 地图显示时间≤3秒
* 时空碰撞
* 支持最大300路不同类型的设备，设备类型包含但不限于人脸抓拍、车辆卡口、WIFI、电子围栏和门禁设备。
* 支持单条件开始、结束时间跨度≤72小时；多条件输入支持最大时间跨度≤6个月。
* 结果响应时间≤5秒
* 数据情况估算：人脸：3000/路/日；车辆：3000/路/日；电围：8万/路/日；
* 支持多条件输入个数≤10；
* 轨迹拟合
* 支持最多10个点位的选择，时间跨度≤72小时
* 轨迹路线生成时间≤1秒，结果响应时间≤5秒
* 数据分析至少实现T+1天的分析，以准实时（延迟1-3小时内）为目标。
* 所有服务的最大并发数为50
* 单个服务并发数为20

**可靠性需求**

* 要求7\*24小时运行
* 因系统自身缺陷造成的故障每1000小时最多发生1次，概率小于1‰

**离线统计分析性能目标**

* 弱关系分析的性能，请关梓文、张燕补充
* SQL规则统计中每个维度指标的统计在3分钟内完成，支持并发或串行进行统计分析。

**实时统计分析性能目标**

实时统计分析的性能，请郑娃龙、张燕补充

# 设计思路

技术选型

**任务调度**

因为没有涉及复杂的DAG调度，基于目前分析系统的xxl-job实现维度统计、强关系分析、要素时空分布统计任务的调度。

目前分析系统是基于xxl-job 2.0.1版本的基础上修改的，可支持单个任务及前后依赖的任务调度，任务重跑操作。详见设计参见xxl-job的官网

https://www.xuxueli.com/xxl-job/

**任务并行执行框架**

数据查询时，需要并行的查询多个表的数据，然后再进行聚合处理。示例如下：

* 时空碰撞查询中的码码碰撞，需要并发的进行不同码【手机号码、MAC、MSI、MEI】多个碰撞分析，然后统一返回
* 关系查询时，要先在要素关系中查询出关系数据，然后再查询要素的其它属性信息如姓名、性别，数据融合后返回。
* 为了提供数据维度统计、要素分布统计、实体强关系分析的效率，需要将这些操作并发的执行。

**选用技术：**

基于京东开源的并行框架[asyncTool](https://gitee.com/jd-platform-opensource/asyncTool" \o "asyncTool" \t "https://gitee.com/jd-platform-opensource/)二次开发，实现不同业务场景的并行处理要求。

[asyncTool](https://gitee.com/jd-platform-opensource/asyncTool" \o "asyncTool" \t "https://gitee.com/jd-platform-opensource/)是目前正在 京东App后台 接受苛刻、高并发、海量用户等复杂场景业务的检验测试，随时会根据实际情况发布更新和bugFix，目前稳定版本是1.3. <https://gitee.com/jd-platform-opensource/asyncTool>

**REST服务框架**

基于如下WEB框架实现：

* springboot 2.1.3
* mybatis-plus 3.2
* springfox-swagger2 2.9.2

重点难点分析

**关系查询后，如何提供数据反查性能**

在关系数据存储中，只存储了要素的类型、标识、关系类型，其它业务属性如姓名、年龄是存储在要素的档案表中。

在查询比对功能中，根据用户输入的证件号码、车牌号码或MAC地址【针对人、车需要先到对应的档案表中找到档案标识】，**找到与要素有直接关联的所有要素，并需要根据要求标识，反查相关的业务属性。**

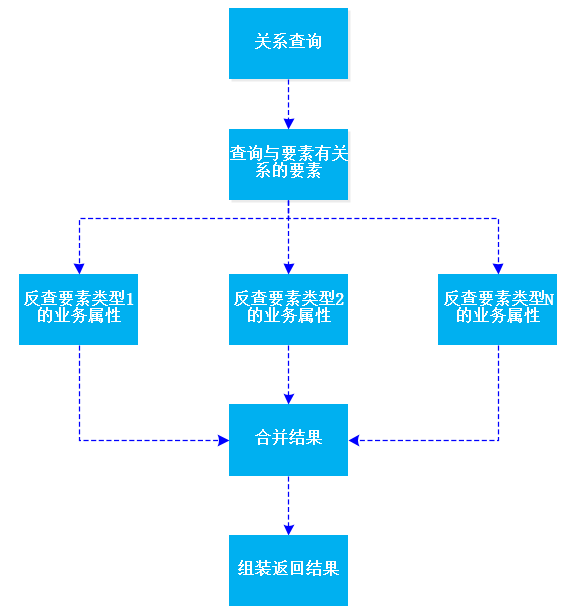
因为要素类型较多，需要并行的反查多张表，并将查询后的结果与关系查询的结果合并返回。

**注意：**

* 关系数据存储在elasticsearch中，要素及标识对应关系如下表格，文档ID是：要素1标识\_关系类型\_要素2的标识。
* 索引是在系统安装时建好的。
* 如果文档ID相同，则后面插入的数据会覆盖以前的数据，即以最新的数据为准
* 为了保存相同的关系不会插入2条数据，引入关系优先级，优先级如下：人档标识，车档标识，手机号，IMEI, IMSI, MAC

|  |  |  |
| --- | --- | --- |
| 要素名称 | 要素标识 | 备注 |
| 人 | 人员档案表的标识 | 兼容人脸聚类产生没有身份信息的人员档案数据的查询 |
| 车 | 车辆档案表的标识 | 根据车辆抓拍数据近实时建档，延迟一个小时内 |
| 手机号 | 手机号 |  |
| IMEI | IMEI号 |  |
| IMSI | IMSI号 |  |
| MAC | MAC地址 | 12位,如02004C4F4F50 |

具体的计算流程如下：



**如何提高多时空数据碰撞的性能**

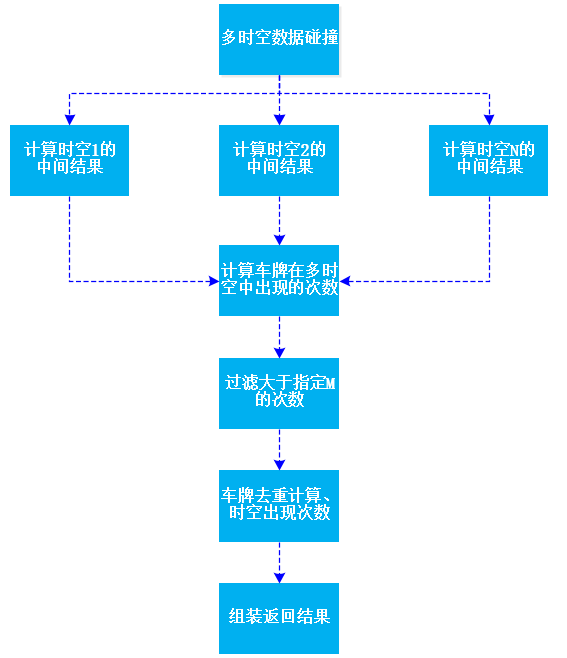
用户根据业务需要，选择多个时空条件，统计多个时空条件中共同出现的要素。如果用户选择的时空范围较大【对应的中间结果集数据量大于50万或更多】、时空条件比较多【大于3个】，则在计算时，则需要先计算多个时空条件对应的中间结果集，然后针对中间结果集求交集，最后计算去重后的车牌及在多个时空中的出现次数，计算量较大，导致一个请求很难在3秒内返回。

**实现方式：基于mppdb的SQL实现数据碰撞【如果把符合条件的中间结果读出到内存然后再基于计算框架如Spark或flink计算，则性能瓶颈在读写数据并性能不高，支持的并发度也低】**

**注意：**

* **基于MPPDB 的SQL实现【MPPDB使用snowballdb】，所以性能取决于数据库，需要根据产品设计的数据量，进行性能测试。**
* **性能测试需要使用真实的数据，否则测试出来的结果不能指导实际的项目。模拟数据只能进行功能性验证。**
* **多时空碰撞基于按要素、按小时统计后的数据进行分析**
* **是基于单表的轨迹数据进行碰撞分析。**
* **针对多表的轨迹数据碰撞，目前缺少业务场景及需要的业务需求。**
* **因为用户在地图划定的空间范围，可能包含多种类型的设备，则需要进行多种感知数据的碰撞分析。因为在WEB展示时,不同的碰撞分析结果是在不同的TAB页展示的，则对应的碰撞是用户在点击TAB页时触发。**

具体的计算流程如下图所示：



**关系数据的存储方式**

#### **存储方案**

**方案一：**

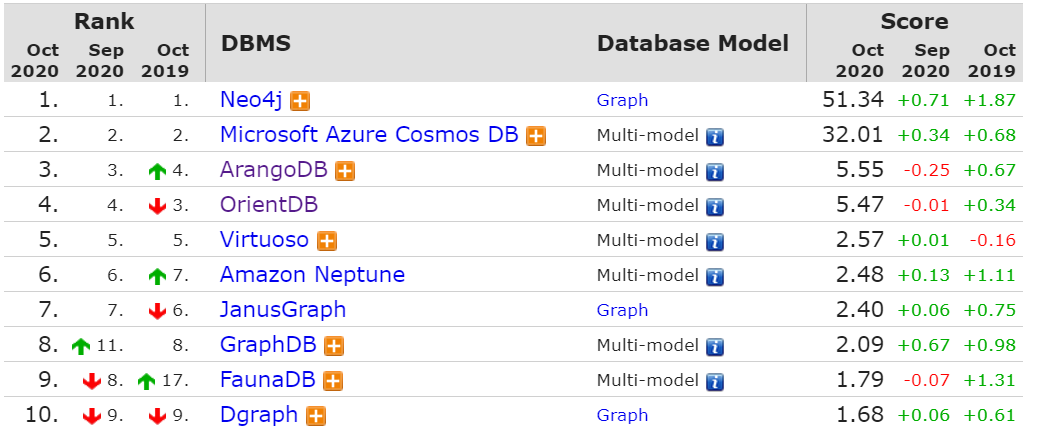
因为一期产品需要只需要计算直接关系，建议基于elasticsearch进行存储，索引结构参见：数据结构中的“关系存储结构”

优缺点如下：

* 优点：数据的更新、删除比较容易
* 缺点：不能进行间接关系、关系挖掘如2个对象的最短距离的查询

**方案二：**

目前陈毅峰在预研图库，预研db-engines网站图数据库排名靠前的几个图库，然后选择一个。目前排名前10的图数据库如下：



优缺点如下：

* 优点：能直接进行间接关系、关系挖掘如2个对象的最短距离的查询
* 缺点：大量关系数据的删除估计比较慢

**推荐使用方案一。**

#### **关系数据的插入和更新**

针对弱关系，目前需要将某一类型的弱关系先删除，然后再批量写入，

针对强关系，则需要如果存在，则更新，否则进行数据插入。

如果基于elasticsearch存储关系数据，则比较方便进行数据的删除和更新

针对于使用开源的图库，则需要测试功能、性能上是否满足。

风险评估

**抓拍人脸的身份确认**

产品崔世鹏确认，针对全量人脸抓拍数据进行聚类分析。风险如下:

实际项目中，可能只对重点区域和重点人员进行聚类，导致按证件号码查询要素关系数据，符合条件的数据很少，会导致根据动态抓拍类数据产生的人员时空分布数据量很少。

**要素时空分布统计的风险**

#### **统计后数量未显著减少**

按要素按小时统计要素时空分布时，绝大部分要素在一个设备一个小时内，只有一条记录，所以**统计产生的记录条数与原始记录条数基于相当，数量比例在95%以上。**

以花都的卡口过车数据分析：

花都每天的卡口过量数量在300万左右，按设备、车牌、小时维度统计后的数据量仍然是300万左右，详见下面的表格。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **过车数据量** | **按设备、车牌、小时统计后，统计数量** | **差值** | **比例** |
| 200901 | 3257742 | 3178408 | 79334 | 97.56% |
| 200902 | 3413713 | 3331626 | 82087 | 97.60% |
| 200903 | 3413218 | 3337452 | 75766 | 97.78% |
| 200904 | 3706877 | 3615602 | 91275 | 97.54% |
| 200905 | 3385510 | 3231379 | 154131 | 95.45% |
| 200906 | 2955187 | 2857488 | 97699 | 96.69% |
| 200907 | 3212024 | 3113662 | 98362 | 96.94% |

**按车牌、小时维度统计后的数据量详见下表，统计后的数量比例在40%左右**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **过车数据量** | **按车牌、小时统计后，统计数量** | **差值** | **比例** |
| 200901 | 3257742 | 1256280 | 2001462 | 38.56% |
| 200902 | 3413713 | 1280489 | 2133224 | 37.51% |
| 200903 | 3413218 | 1305232 | 2107986 | 38.24% |
| 200904 | 3706877 | 1387956 | 2318921 | 37.44% |
| 200905 | 3385510 | 1273180 | 2112330 | 37.61% |
| 200906 | 2955187 | 1189089 | 1766098 | 40.24% |
| 200907 | 3212024 | 1294516 | 1917508 | 40.30% |

**按天统计车牌数量**

|  |  |
| --- | --- |
| **日期** | **去重后的车牌数量** |
| 200901 | 508203 |
| 200903 | 518298 |
| 200902 | 508853 |
| 200904 | 551761 |
| 200905 | 524847 |
| 200906 | 486018 |
| 200907 | 532777 |

**方案一:**

如果按车牌、按小时统计，并将设备标识聚合到一个字段中，统计后的数据约减少60%，SQL如下：

**select** HPHM,JGRQ,**SUBSTRING**(toString(JGSK) ,7,2) **AS** **HOUR**,groupArray(DEVICE\_ID) **as** DEVICE\_IDS,**count**(\*) **as** total

**from** CAR\_DETECT\_INFO **where** JGRQ **BETWEEN** 200901 **and** 200907 **group** **by** HPHM,JGRQ,**HOUR**

求最后出现时间对应的设备即最后设备:

**select** **SUBSTRING**(CREATE\_TIME,1,10) **as** **day**,IMSI,**COUNT**(),**max**(CREATE\_TIME),**min**(CREATE\_TIME),argMax(EQUID,CREATE\_TIME),argMin(EQUID,CREATE\_TIME),groupArray(EQUID),groupArray(CREATE\_TIME)

**from** TB\_STRENGTHEN\_WIFI\_IMSI **where** **day** = '2020-09-02' **GROUP** **BY** **day**,IMSI **order** **by** **day**

在数据碰撞时，使用如下SQL:

**SELECT** DEVICE\_IDS,arrayIntersect(DEVICE\_IDS,['2','3']) **as** occurNum **from** CAR\_HPHM\_HOUR\_STAT **where** **length**(occurNum)>0

**如果采用这种方式，则需要针对arrayIntersect的过滤效率进行性能测试。注意：这种方式是和snowballdb绑定的，其它MPPDB如华为的gaussDB中是不存在类似函数的。**

**方案二:**

按天、按车牌、按设备统计，将时间按小时或分钟分片（一期按小时分片）然后的bitmap方式【即array方式存储，计算时转换为bitmap】，统计产生的数据量约为原始数据量的87%。

bitmap存储方式：在一个分片中，如果这辆车经过这个设备【不考虑次数】，则标记为1，否则为0，然后将标记为1的小时转换为一个array进行存储，在计算时，将array转换为bitmap进行bitmap交集计算，涉及的函数是：bitmapBuild、bitmapHasAny等，示例SQL如下：

**SELECT** **LENGTH**(bitmapToArray(bitmapAnd(bitmapBuild([2223,2221]),bitmapBuild([2223,2221,2231])))) **AS** res

**SELECT** DEVICE\_IDS,bitmapToArray(bitmapBuild(HOURS )) **FROM** CAR\_HPHM\_HOUR\_STAT **WHERE** bitmapHasAny(bitmapBuild(HOURS),bitmapBuild([5,8]))>0;

**问题如下：**

* 统计后，数据量显著减少，和基于原始数据进行计算，估计性能提升不明显
* 如果一个时空条件跨天，则需要将一条SQL拆分为2条，增加了复杂性，降低了性能

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **过车数据量** | **按设备、车牌、天统计后，统计数量** | **差值** | **比例** |
| 200901 | 3257742 | 2853718 | 404024 | 87.60% |
| 200902 | 3413713 | 2973146 | 440567 | 87.09% |
| 200903 | 3413218 | 2976102 | 437116 | 87.19% |
| 200904 | 3706877 | 3230371 | 476506 | 87.15% |
| 200905 | 3385510 | 2910542 | 474968 | 85.97% |
| 200906 | 2955187 | 2575546 | 379641 | 87.15% |
| 200907 | 3212024 | 2801562 | 410462 | 87.22% |

建表及简单测试示例：

**CREATE** **TABLE** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE **Date**,STAT\_HOUR **Int8**,HPHM **String**,DEVICE\_IDS **Array**(**String**),STAT\_TIME **DateTime**)

Engine=MergeTree() **partition** **by**(STAT\_DATE) **ORDER** **BY** (STAT\_DATE,STAT\_HOUR,HPHM) SETTINGS index\_granularity = 8192;

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-11',12,'粤AN4661',['1','2','3'],'2001-10-12 12:12:44');

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-12',12,'粤AN4661',['1','2','3'],'2001-10-12 12:22:44');

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-13',12,'粤AN4661',['1','2','3'],'2001-10-12 12:22:44');

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-14',12,'粤AN4661',['1','2','3'],'2001-10-12 13:12:44');

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-15',12,'粤AN4661',['1','2','3'],'2001-10-12 12:12:44');

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-16',12,'粤AN4661',['1','2','3'],'2001-10-12 01:12:44');

**insert** **into** CAR\_HPHM\_HOUR\_STAT (STAT\_DATE,STAT\_HOUR,HPHM,DEVICE\_IDS,STAT\_TIME)**values**('2001-10-17',12,'粤AN4661',['1','2','3'],'2001-10-12 23:12:44');

**SELECT** \* **FROM** CAR\_HPHM\_HOUR\_STAT;

**SELECT** DEVICE\_IDS,arrayIntersect(DEVICE\_IDS,['2','3']) **as** occurNum **from** CAR\_HPHM\_HOUR\_STAT **where** **length**(occurNum)>0

**SELECT** STAT\_DATE,STAT\_TIME,toHour(STAT\_TIME),toStartOfHour(STAT\_TIME) **as** **HOUR** **FROM** CAR\_HPHM\_HOUR\_STAT

**方案三：****直接基于过车明细数据进行统计分析**

直接将按按小时、按设备、按要素【指人、车、码】统计后的数据，直接写到snowballdb, 时空分布统计按天统计，统计约有几十个，会不会影响正常的业务数据写入和查询

通过三个方案的比较，确定按如下方案实施：

按方案一【如果按车牌、按小时统计，并将设备标识聚合到一个字段中】进行预统计，满足轨迹查询、时空碰撞需求。

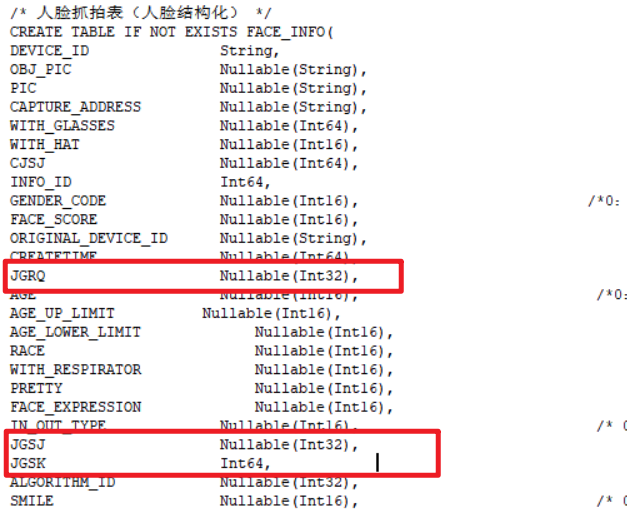
后续如果发现基于统计表的时空碰撞性能比基于方案三的差，则直接基于方案三进行时空碰撞分析。

#### **统计SQL性能低**

目前人、车、码相关的五类感知数据，抓拍时刻、抓拍日期、抓拍时间都是数字类型，如下图所示。在按小时统计时，需要将抓拍时刻转化为字符串然后进行字符串截取获取小时【SUBSTRING(toString(JGSK) ,7,2)】，性能较低。

能否将JGSK转换为datetime类型 或 将JGSJ转换为time类型，提供根据条件筛选的效率。

**聚类后的人脸轨迹表有按小时维度存储，对应的表结构在哪里？郑娃龙**



#### **加强型WIFI-IMSI数据统计分析**

如果按天、IMSI且采集的设备数大于指定值，则统计结果如下：

在至少3个设备出现，数据量减少至2%

在至少2个设备出现，数据量减少至10%

在至少1个设备出现，则数据量减少50%

如果按小时、IMSI且采集的设备数大于指定值，由数据减少量更为明显。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **加强型IMSI数据量** | **按IMSI、小时天统计后，统计数量(>=3个设备出现）** | **>=2个设备出现** | **>=1个设备出现** |
| 2020-9-1 | 640615 | 12600 | 64386 | 353047 |
| 2020-9-2 | 855306 | 19117 | 88722 | 429468 |
| 2020-9-3 | 958493 | 22663 | 99845 | 473904 |
|  |  |  |  |  |
| **日期** | **加强型IMSI数据量** | **按IMSI、小时统计后，统计数量(>=3个设备出现）** | **>=2个设备出现** | **>=1个设备出现** |
| 2020-9-1 | 640615 | 2144 | 35759 | 517863 |
| 2020-9-2 | 855306 | 3424 | 50468 | 689565 |
| 2020-9-3 | 958493 | 4013 | 56790 | 769621 |

统计SQL如下：

**SELECT** **day**,**COUNT**() **from** (

**select** **SUBSTRING**(CREATE\_TIME,1,10) **as** **day**,**SUBSTRING**(CREATE\_TIME,12,2) **as** **hour**,IMSI,uniqExact(EQUID) **AS** NUM,**COUNT**(EQUID),groupArray(EQUID),groupArray(CREATE\_TIME)

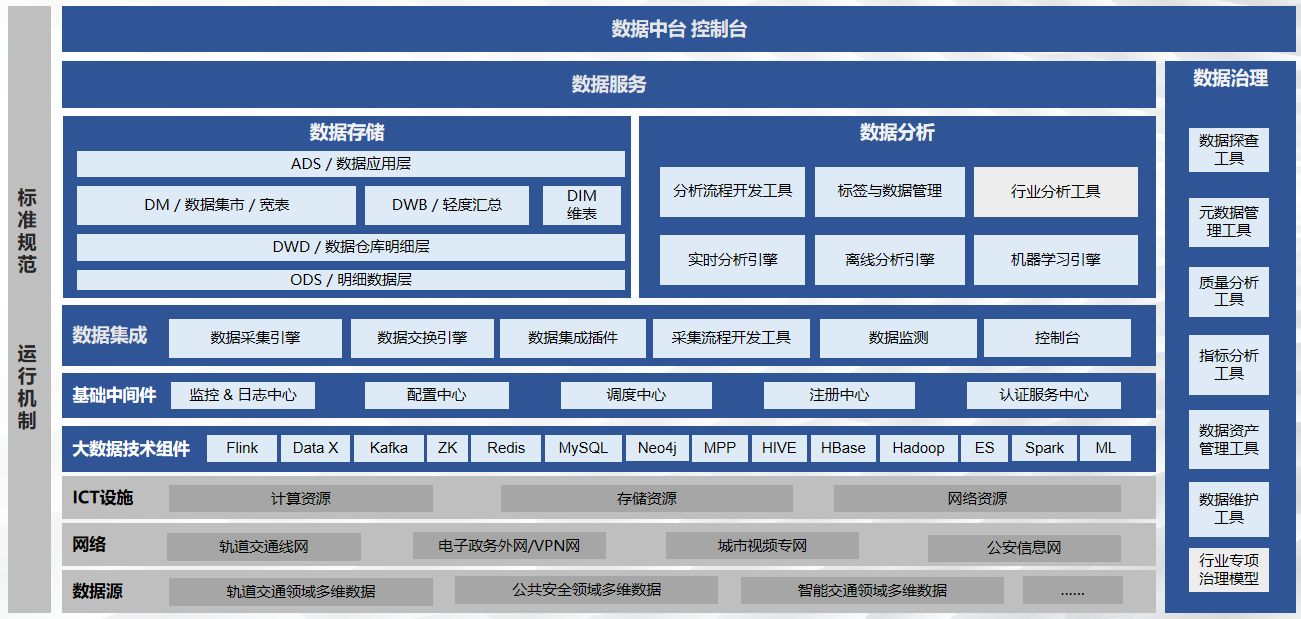
**from** TB\_STRENGTHEN\_WIFI\_IMSI **GROUP** **BY** **day**,**hour**,IMSI **HAVING** NUM>2 **order** **by** **day** ) **group** **by** **day** **order** **by** **day**

即按要素，按小时统计，数据量约减少50%。

# **架构设计**

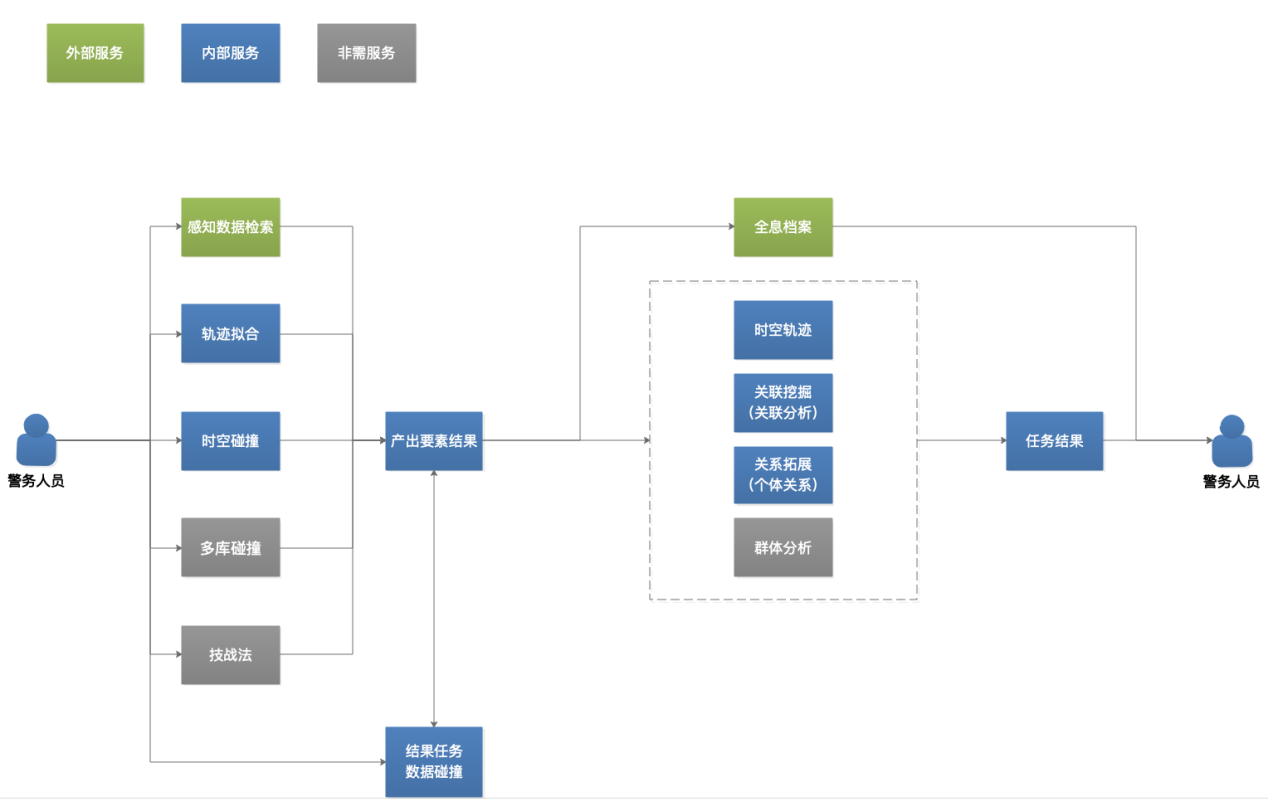
* 1. 架构设计
     1. 逻辑架构图

系统的逻辑分层架构图如下图，本设计重要关注：统计console及相关的统计后台任务。



* + 1. 整体业务流程

从用户使用角度看，整体业务流程如下：



* + 1. 整体数据流程

五类感知数据【人脸抓拍、车辆抓拍、门禁、WIFI、电围】的整体数据接入、标准化、融合的流程如下：



* + 1. 系统架构图

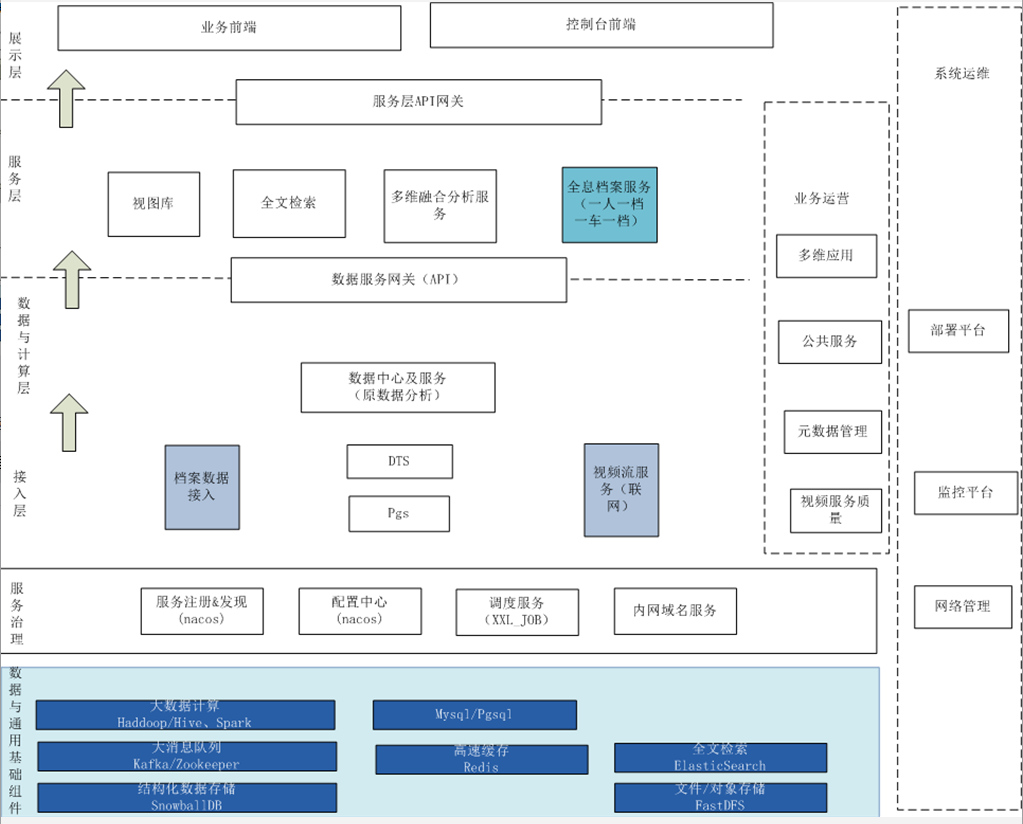
针对数据中心，对应的功能模块如下：

|  |  |  |
| --- | --- | --- |
| **模块名称** | **类型** | **包含的功能列表** |
| console | Springboot | 规则统计配置管理  关系定义的管理  规则统计执行记录及任务重跑  提供的REST服务【7.5章节】 |
| WEB前端 | Vue | console中的WEB |
| 规则统计后台 | java后台 | 与xxl-job的job-engine打包、部署在一起  说明: 基于snowballdb上的数据基于SQL进行统计分析 |
| 实时统计后台 | java后台 | 与xxl-job的job-engine打包、部署在一起  说明: 消费kafka上的数据，进行数据的实时统计分析 |
| ~~实时事件分析后台~~ | ~~java后台~~ | ~~与xxl-job的job-engine打包、部署在一起~~  ~~说明: 消费kafka上的数据，进行数据的实时统计分析~~ |
| 算法分析后台 | java后台 | 与xxl-job的job-engine打包、部署在一起 |

**依赖的功能模块**

|  |  |
| --- | --- |
| **模块名称** | **备注** |
| xxl-job | 任务调度程序 |
| Kafka | kafka集群 |
| mysql | 元数据数据库(pd\_dts、pd\_das两个库) |
| hadoop集群 | 大数据计算集群（包括spark,hive,hdfs,yarn,zk等组件） |
| Snowballdb | MPPDB数据库 |
| Elasticsearch | 用于关系数据存储，两都二选一，待定 |
| 图数据库 |

系统架构图如下:



* + 1. 物理部署图

物理部署图

# **子系统/模块划分及设计说明**

* 1. **规则统计模块**

需要实现的要素时间维度统计、要素时空分布统计、实体强关系分析参见《资源库统计.xlsx》中的对应sheet页。

配置使用的表参见 数据结构 章节

注意：

* 如果按小时统计，统计产生的数据量较大，可能影响业务的正常进行

将配置表中每中业务类型，配置成任务调度系统中的一个任务，按T+1进行调度。如

|  |  |  |
| --- | --- | --- |
| **任务名称** | **调度时间** | **备注** |
| 要素时间维度统计 | 每天凌晨30分 |  |
| 要素时空分布统计 | 每天1:50 |  |
| 实体强关系分析 | 每天3:20 |  |

任务的参数如下：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数描述** | **备注** |
| FREQUENCY\_TYPE | 执行频次类型 |  |
| BUSI\_TYPE | 统计业务类型 |  |
| THREAD\_NUM | 并发的线程数 | 如果扫描到多个统计配置，则根据指定的线程数，并发的执行 |

规则统计后台程序根据调用任务中配置中的参数，从配置表中读取对应的配置，并发执行规则统计。

**一期仅支持以snowballdb、MySQL的数据进行规则统计分析，然后将结果写到snowballdb、MySQL、elasticsearch7.2中。**

**需要确定的是：**

* **要素时空分布统计结果统一放snowballdb,存储一年**
* **要素时间维度统计，结果统一放mysql,存储一年**
* **需要明确各类设备的数量（人脸摄像机、产生WIFI的设备[热点、加强型WIFI、电围、移动终端探测设备]、车辆卡口、微卡口),否则不能预估数据量**
  + 1. **涉及的技术点**

涉及到的技术难点如下：

* 为了缩短规则统计的时间，任务需要要任务并行框架，根据指定的线程数，并发的执行。
* 规则统计结果数据根据不同情况，需要写到mysql、snowballdb、ES(关系数据)
* 因为数据（如感知类数据中的门禁数据）中人员档案标识(PERSON\_ID）不是必填的,如果PERSON\_ID为空则证件号码非空，是则需要在计算前，缓存人员档案标识与证件号码-证件类型的关系数据到JVM.

**疑问：**

**如果门禁表证件号码在档案表中找不到记录，怎么处理？直接忽略掉？**

**门禁表中的PERSON\_ID是否等同于人员档案表中的人员标识**

**涉及门禁的统计，需要将视频门禁【数据在聚类后的人脸抓拍表】和门禁表的数据累加？**

* **SQL定义中，需要根据变量配置统计的时间范围，在执行时，将变量转换为实际的值，支持的变量如下：**

声明方式：${变量名}。可以是引用 "时间基准变量"。

时间基准变量：基准变量为$[...] 格式的，[yyyyMMddHHmmss] 是可以任意分解组

合的，比如：$[yyyyMMdd], $[HHmmss], $[yyyy-MM-dd] 等

|  |  |
| --- | --- |
| ${system.biz.date} | 日常调度实例定时的定时时间前一天，格式为 yyyyMMdd，补数据时，该日期 +1 |
| ${system.biz.curdate} | 日常调度实例定时的定时时间，格式为 yyyyMMdd，补数据时，该日期 +1 |
| ${system.datetime} | 日常调度实例定时的定时时间，格式为 yyyyMMddHHmmss，补数据时，该日期 +1 |
| 后 N 年 | $[add\_months(yyyyMMdd,12\*N)] |
| 前 N 年 | $[add\_months(yyyyMMdd,-12\*N)] |
| 后 N 月 | $[add\_months(yyyyMMdd,N)] |
| 前 N 月 | $[add\_months(yyyyMMdd,-N)] |
| 后 N 周 | $[yyyyMMdd+7\*N] |
| 前 N 周 | $[yyyyMMdd-7\*N] |
| 后 N 天 | $[yyyyMMdd+N] |
| 前 N 天 | $[yyyyMMdd-N] |
| 后 N 小时 | $[HHmmss+N/24] |
| 前 N 小时 | $[HHmmss-N/24] |
| 后 N 分钟 | $[HHmmss+N/24/60] |
| 前 N 分钟 | $[HHmmss-N/24/60] |

* 1. **数据挖掘模块**
     1. **离线数据挖掘分析**

详见张燕、关梓文负责的数据分析设计

关系数据统一存储在es中【后续有技术实力时，考虑存储到图数据中】，图数据中已经包含了关系分布的数据

* + 1. **实时数据分析**

详见张燕、关梓文负责的数据分析设计

档案的抓拍总数的近实时统计（按档案或按档案+设备）属于实时统计分析需求

**注意：此部分包含数据的实时统计分析**

* 1. **console模块**
     1. **关系定义的管理**

需要崔世鹏根据 数据结构中定义的关系中，提供关系定义的基础数据，然后由开发人员转换为此表的初始化SQL语句。

实施方式: 在目前分析子系统中的关系定义表中增加关系类型，并完善其增、删、改、查功能。

* + 1. **规则统计配置管理**

针对规则统计表的配置管理功能，包括增、删、改、查功能，查看某一个规则统计的执行记录，针对失败的记录重跑。

* + 1. **规则统计执行记录**

以列表的方式查看规则统计的执行记录，支持按业务类型、执行频次类型、统计表、结果表、执行状态进行条件查询，并提供针对执行失败的任务，进行重跑。

* 1. **研判任务管理**

用户在进行数据查询、时空碰撞、相似轨迹等分析时，可以将分析结果以研判任务的形式保存，然后再对多个研判任务的分析结果进行交、并、差的结果碰撞。

提供研判任务的保存、任务的列表查询、任务名称的编辑、创建副本、导出、星标、任务的删除、任务的查看、多任务的交并差。

注意：在任务名称编辑、星标时，需要更新记录的更新时间。

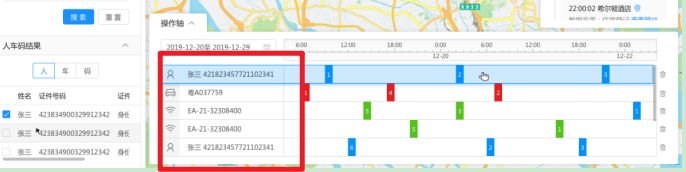
研判任务的分析结果是指轨迹分析时间轴左边的对象列表。

针对人，保存：档案标识、证件类型、证件号码、姓名

针对车，保存：档案标识、车牌颜色、车牌号码

针对码，保存：侦码类型、侦码值

**所有要保存的对象原则上不能超过100个**



* 1. **REST服务**

直接提供REST服务，还是以JAR包提供，需要讨论 @郑娃龙

* + 1. **直接关系查询服务**

根据用户指定的要素条件，查询和要求有直接关系的实体。

说明:

* 针对人,有如下两种情况：
* 如果上传的是图片，则先基于人员档案库作为底库，进行以图搜图，用户选择一个人员档案，再按人员档案标识(PERSON\_ID)进行此人的直接关系查询。
* 如果指定证件号码，则先根据证件号码到人员档案表查找此证件的人员档案，再按人员档案标识(PERSON\_ID)进行此人的直接关系查询。 针对未指定证件类型的，则查询全部证件类型。

**疑问：如果多个人，如何处理，还是只能选择一个**

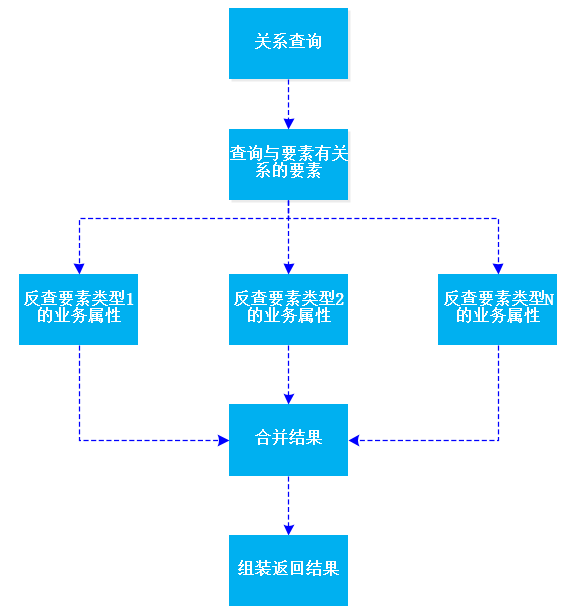
* 针对车: 如果没有选择车牌颜色，则默认为查询所有颜色的车牌

**疑问：如果多个车牌，如何处理，还是只能选择一个**

* **针对码: 侦码类型和侦码值是必填的？**
* 侦码类型具体的字典是：
* 3 手机号
* 4 IMEI
* 5 IMSI
* 6 MAC

|  |  |  |  |
| --- | --- | --- | --- |
| **URI** | /das/rel/search GET请求 | | |
| **功能** | 直接关系查询 --只返回和指定对象有关联的其它对象 | | |
| **参数** | 名称 | 类型 | 描述 |
| actionType | int | 1 人 2 车 3 码 |
|  | topN | int | 返回的记录条数，默认为20条 |
|  | idType | Int | 对象的类型 1 人，2 车，3 手机号，4 IMEI，5 IMSI，6 MAC |
|  | idValue | String | 对象的ID值，针对人车是对应档案表的记录ID，针对码是实际的码值 |
|  | relType | int | 1 要素关联 2 实体关系 -1 全部 默认为1 |
|  | | | |
| **返回结果** | 名称 | 类型 | 描述 |
| data | [object] | 返回符合条件的人车码集合：  [{"ID\_TYPE":,"REL\_ID":,"ID\_VALUE":,"ATTR":{"PERSON\_NAME":"","ID\_TYPE":"111","ID\_NUMBER":"440321199012035621"}]  人的属性：  "ATTR":{"PERSON\_NAME":"","ID\_TYPE":"111","ID\_NUMBER":"440321199012035621"}  车的属性：  "ATTR":{"HPHM":"粤A2DP23","HPYS":2}  码的属性：为空 |

查询流程如下：



* + 1. **关系挖掘或拓展服务**

根据用户指定的要素及关系类型列表，查询和要素有直接关系的实体。

|  |  |  |  |
| --- | --- | --- | --- |
| **URI** | /das/rel/mining GET请求 | | |
| **功能** | 直接关系查询 | | |
| **参数** | 名称 | 类型 | 描述 |
| actionType | int | 1 人 2 车 3 码 |
|  | topN | int | 返回的记录条数，默认为20条 |
|  | idType | Int | 对象的类型 1 人，2 车，3 手机号，4 IMEI，5 IMSI，6 MAC |
|  | idValue | String | 对象的ID值，针对人车是对应档案表的记录ID，针对码是实际的码值 |
|  | relIds | [] | 指定的要查询的关系标识列表,如果为空，则查询全部关系 |
|  | relType | int | 1 要素关联 2 实体关系 -1 全部 默认为1 |
|  | score | int | 支持度，置信度 0-100内的整数 |
|  | | | |
| **返回结果** | 名称 | 类型 | 描述 |
| data | [object] | 返回要素的所有符合条件的关系，如果有多种节点类型，则统一在一个列表中返回。  返回符合条件的人车码集合：  [{"ID\_TYPE":,"REL\_ID":,"ID\_VALUE":,"ATTR":{"PERSON\_NAME":"","ID\_TYPE":"111","ID\_NUMBER":"440321199012035621"}]  人的属性：  "ATTR":{"PERSON\_NAME":"","ID\_TYPE":"111","ID\_NUMBER":"440321199012035621"}  车的属性：  "ATTR":{"HPHM":"粤A2DP23","HPYS":2}  码的属性：为空 |

* + 1. **单表时空碰撞服务**

根据用户的多维时空条件，进行多时间的数据碰撞。

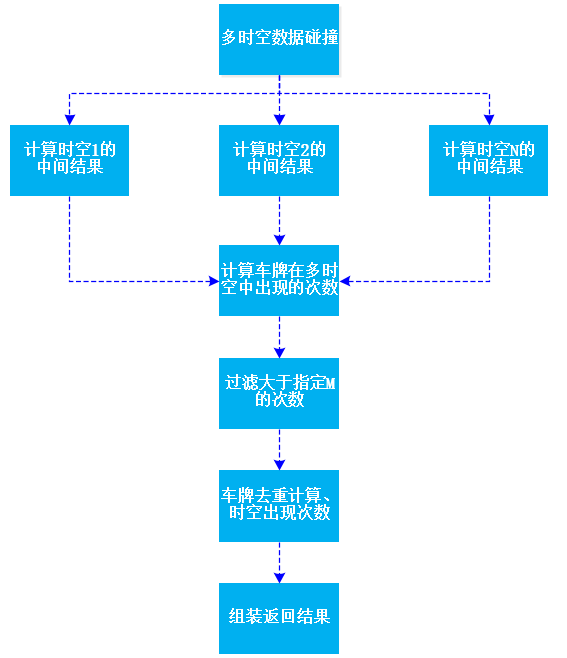
注意：

因为多时空碰撞逻辑复杂，为了保证请示能在3秒内响应，则需要限制中间结果集的大小，限制如下：

* 每个时空条件中的时间跨度3天内，即时间范围只能在72小时内
* 时间选择的颗粒度是到小时
* 每个时空条件中选择的一个种类的设备数不能超过300个
* 因为多维时空碰撞查询的是统计表，统计是T+1进行的，所以时间不能选择今天

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **URI** | /das/collide/single POST请求 | | | |
| **功能** | 直接关系查询 | | | |
| **参数** | 名称 | | 类型 | 描述 |
| actionType | | int | 1 人 2 车 3 码 |
|  | topN | | int | 返回的记录条数，默认为20条 |
|  | list | startTime | string | 2020-10-21 00:00:00 |
|  | endTime | string | 2020-10-22 00:00:00 |
|  | deviceIds | [] | 设备标识的列表, ["","",""] |
|  | | | | |
| **返回结果** | 名称 | | 类型 | 描述 |
| data | | [object] | 对象类型：1 人，2 车，3 手机号，4 IMEI，5 IMSI，6 MAC  [{"ID\_TYPE":1，"ID\_VALUE":"111","REGION\_NUM":3}]  注意：  针对人是人员档案表的档案标识，  针对车是车牌号码  针对其它类型：是码的实际值 |

具体的计算流程如下图所示：



* + 1. 对象时空分布统计查询服务

根据用户指定的对象列表及时间范围，返回对象的时空分析轨迹。

因为要出统计结果及明细数据，由应用开发直接查询标准表【由郑娃龙确认】

注意：

* 查询的数据是要素时空分布统计表，非要素明细数据如车辆抓拍明细记录。

|  |  |  |  |
| --- | --- | --- | --- |
| **URI** | /service/das/trajectory/stat | | |
| **功能** | 直接关系查询 | | |
| **参数** | 名称 | 类型 | 描述 |
| objType | int | 对象类型 1 人 2 车 3码 |
|  | objIds | [] | 人车指档案标识，码指侦码类型-侦码值 |
|  | startTime | string | 2020-10-21 00 |
|  | endTime | string | 2020-10-22 00 |
|  | | | | endTime | string | 2020-10-22 00 |
| **返回结果** | 名称 | 类型 | 描述 |
|  | data | [object] | STAT\_TIME的格式:yyyyMMddHH  [{"OBJ\_ID":"111","STAT\_TIME":2020102312,"NUM":111,[{"DEVICE\_ID":"440321199012035621","NUM":2}]}] |

* + 1. **研判结果的数据碰撞服务**

根据用户指定的数据研究结果，将多个研究结果进行数据交、并、差分析

注意：

* 针对研判结果的交集计算，是计算出在所有研判结果有出现的人、车、码对象
* 针对研判结果的并集计算，是汇总所有研判结果的人、车、码对象
* 针对研判结果的差集计算，是将并集的结果减去交集的结果后产生的人、车、码对象

|  |  |  |  |
| --- | --- | --- | --- |
| **URI** | /service/das/data/anlyseJudge | | |
| **功能** | 直接关系查询 | | |
| **参数** | 名称 | 类型 | 描述 |
| analyseType | int | 研判类型 1 交 2 并 3差 |
|  | taskIds | [] | 数据研判任务标识的列表 |
|  | | | |
| **返回结果** | 名称 | 类型 | 描述 |
|  | data | [object] | 针对人：返回符合条件的关联人的列表  [{"OBJ\_ID":"111","OBJ\_TYPE":1,"PERSON\_NAME":"","ID\_TYPE":"111","ID\_NUMBER":"440321199012035621"}]  针对车：返回符合条件的关联车的列表  [{"OBJ\_ID":"111","OBJ\_TYPE":1,"HPHM":"粤A2DP23","HPYS":2}]  针对码: 返回符合条件的关联码的列表  [{"OBJ\_ID":"111","OBJ\_TYPE":1,"CODE\_TYPE":3,"CODE":"02004C4F4F50"}] |

* + 1. **多实体间关系分析**

根据用户指定的实体对象列表，分析这些实体对象之间的关系。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **URI** | /das/rel/analyse POST请求 | | | |
| **功能** | 直接关系查询 | | | |
| **参数** | 名称 | | 类型 | 描述 |
|  | List | id | string | 查询的实体对象列表 |
| idType | int |
|  | relType | | int | 1 要素关联 2 实体关系 -1 全部 默认为-1 |
|  | score | | int | 支持度，置信度 0-100内的整数，默认为-1 |
|  | | | | |
| **返回结果** | 名称 | | 类型 | 描述 |
| data | | [object] | 返回存在的这些实体对象之间的关系   [{       "ID1": "18900008881",       "ID1\_ATTR": null,        "ID1\_TYPE": 3,       "ID2": "02-00-4C-4F-4F-50",        "ID2\_ATTR": null,        "ID2\_TYPE": 6,         "REL\_ID": "04-0002-0005",         "REL\_TYPE": 1,          "SCORE": 100         }     ] |

# 数据结构

* 1. **关联相关**
     1. **关系定义元数据结构**

关系定义元数据复用目前分析系统的关系定义表

表名：d\_relation, 存储在mysql的pd\_das库中

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段中文名称 | 字段类型 | 备注 |
| rel\_id | 关系标识 | varchar(20) |  |
| parent\_id | 父关系标识 | varchar(20) |  |
| rel\_name | 关系名称 | varchar(50) |  |
| rel\_attr | 关系产生方式 | tinyint | 1:事实；2:预测 |
| is\_valid | 是否有效 | tinyint | 9 删除 1 有效 |
| is\_dir | 是否目录节点 | tinyint | 1目录节点 2具体关系节点 |
| remark | 备注 | varchar(128) |  |
| create\_time | 创建时间 | datetime |  |
| update\_time | 更新时间 | datetime |  |
| is\_used | 是否被使用 | tinyint | 0：是，1：否 |
| rel\_type | 关系类型 | tinyint | 1 要素关联 2 实体关系 |

* + 1. **关系存储结构**

存储说明:

* 关系采用顶点-边-顶点三元组的方式进行存储。
* 考虑到数据查询时，需要根据一个要素，查找此要素的所有关系，所以所有关系存储在一个索引中。
* **这个是关系数据存储，不涉及全文检索，只提供根据要素进行关系查询。如果需要支持知识图谱类的搜索，需要使用图数据库**
* **标蓝的6行，针对基于要素轨迹类且使用分析算法产生的关系数据，才有值。**
* 文档ID的产生规则:${**ID1}\_${ID1\_TYPE}\_${REL\_ID}\_${ID2}\_${ID2\_TYPE}**,即使用下划线将多个字段串起来。针对两个对象相同的关系，只存储最新的一条。
* 为了保存相同的关系不会插入2条数据，引入关系优先级，优先级如下：**人档标识，车档标识，手机号，IMEI, IMSI, MAC，如果两个对象类型一致，则比较两个对象的ID，值小的排前面。**

存储在elasticsearch中，使用7.2以上版本，索引名称是: relation\_index

结构如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段中文名称 | 字段类型 | 备注 |
| ID1 | 顶点1的标识 | string | 人：人员档案标识  车：车辆档案标识  码：具体的码值 |
| ID1\_ATTR | 顶点1的关键属性 | object | 人:  {"PERSON\_NAME":""，"ID\_TYPE":"","ID\_NUMBER":}  车：{"HPHM":"","HPYS":}  码: 空 |
| ID1\_TYPE | 顶点1的类型 | smallint | 1 人档标识，2 车档标识，3 手机号，4 IMEI, 5 IMSI, 6 MAC |
| ID2 | 顶点2的标识 | string | 人：人员档案标识  车：车牌号码  码：具体的码值 |
| ID2\_ATTR | 顶点2的关键属性 | object | 人:  {"PERSON\_NAME":""，"ID\_TYPE":"","ID\_NUMBER":}  车：{"HPHM":"","HPYS":}  码: 空 |
| ID2\_TYPE | 顶点2的类型 | smallint | 1 人档标识，2 车档标识，3 手机号，4 IMEI, 5 IMSI, 6 MAC |
| REL\_ID | 关系标识 | varchar(20) |  |
| REL\_TYPE | 关系类型 | tinyint | 1 要素关联 2 实体关系 |
| REL\_ATTR | 关系属性 | object | 关系相关的一些重要属性信息 |
| REL\_GEN\_TYPE | 关系产生方式 | tinyint | 1:事实；2:预测 |
| FIRST\_TIME | 首次关联时间 | datetime |  |
| LAST\_TIME | 最近关联时间 | datetime |  |
| OCCUR\_NUM | 关联发生次数 | int |  |
| SCORE | 支持度，置信度 | Byte | 0-100， 值越大表示置信度越高 |
| FIRST\_DEVICE\_ID | 最早关联设备 | varchar(64) | ID1的设备标识\_ID2的设备标识 |
| LAST\_DEVICE\_ID | 最近关联设备 | varchar(64) | ID1的设备标识\_ID2的设备标识 |

* 1. 规则统计相关
     1. **配置表**

每一个基于规则统计的分析，对应一条记录。按照统计业务类型、执行频次类型，配置到任务调度系统中定期进行调度执行。

STAT\_CONFIG

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段中文名称 | 字段类型 | 备注 |
| CONFIG\_ID | 统计配置标识 | bigint |  |
| STAT\_NAME | 规则统计名称 | varchar(64) |  |
| EXEC\_PARAMS | 运行参数 | varchar(256) | {"startTime":"$[yyyyMMdd-1]",  "endTime":"$[yyyyMMdd]"} |
| STAT\_DATASOURCE\_ID | 统计使用的数据源标识 | bigint |  |
| STAT\_REL\_TBLS | 统计涉及的表名称 | varchar(128) | 如果多个表，则使用,分隔 |
| STAT\_SQL | 统计SQL | text |  |
| STAT\_SQL\_TYPE | 统计SQL语句类型 | tinyint | 1 insert select 2 select 默认为2 |
| RES\_DATASOURCE\_ID | 结果入库的数据源标识 | bigint | 针对统计SQL类型为1的，这两个字段值为可不填 |
| RES\_SQL | 结果入库使用的SQL | text |
| RES\_TBL | 结果表 | varchar(64) | 结果表 |
| CACHE\_CONFIG | 缓存数据的配置 | text | [{"cacheSql":[""]  ,"cacheName":"fusion\_dwunit",  "dataSourceId":2,  "keyColNames":"colName1,colName2",  "fetchSize":10000}] |
| OTHER\_CONFIG | 其它相关的配置 | text | 是一个json对象，保存字段转换配置及针对不同数据源的个性化配置  转换函数  {"transformer":  [{"srcColumnName":"",  "name":"cacheReplace",  "destColumnName":"",  "paras":{"cacheName":"GA\_XZDZ",  "complementFieldName":"DZDM"}}]}  转换函数有：**concat,cacheReplace,genJson,min,sum**  concat(string expr)  expr示例:${A}-${B}  cacheReplace(string cacheName,fieldName)  genJson(string expr)  expr示例: {"ZJHM":"${ZJHM}","ZJDM":"${ZJDM}"}  sum函数主要用于统计数据的累加  min函数主要用于比较两个时间值，取较小的那个 |
| PRE\_SQL | 结果入库前执行的SQL | varchar(128) | 选填，如删除上次运行的统计结果 |
| POST\_SQL | 结果入库后执行的SQL | varchar(128) | 选填，如进行索引优化 |
| FREQUENCY\_TYPE | 执行频次类型 | varchar(16) | hour,day,month |
| STAT\_CATE | 统计分类 | tinyint | 1 要素时空维度统计  2 要素时空分布统计  3 实体关系分析 |
| BUSI\_TYPE | 统计业务类型 | tinyint | 1 人 2 车 3 码 |
| IS\_VALID | 是否有效 | tinyint | 1 有效 2 无效，默认为1 |
| CREATE\_TIME | 创建时间 | datetime |  |
| UPDATE\_TIME | 更新时间 | datetime |  |

* + 1. **执行实例表**

统计任务在执行前，在此表中执行一条记录，状态为准备执行，执行失败，则更新状态为失败，执行成功则更新状态为成功。

STAT\_EXEC\_INSTANCE

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段中文名称 | 字段类型 | 备注 |
| INSTANCE\_ID | 统计执行实例标识 | bigint |  |
| CONFIG\_ID | 统计配置标识 | bigint |  |
| EXEC\_STATUS | tinyint | 1 准备执行  2 执行成功  3 执行失败 |  |
| FREQUENCY\_TYPE | 执行频次类型 | varchar(16) | hour,day,month |
| EXEC\_TIME | 执行使用的时间 | varchar(16) | 执行频次类型不同，则格式不同  hour: yyyyMMddHH  day: yyyyMMdd  month:yyyyMM |
| STAT\_CATE | 统计分类 | tinyint | 1 要素时空维度统计  2 要素时空分布统计  3 实体关系分析 |
| BUSI\_TYPE | 统计业务类型 | tinyint | 1 人2 车3 码 |
| TRIGGER\_TYPE | 触发方式 | Tinyint | 1 调度系统  2 手工调用服务触发 |
| CREATE\_TIME | 创建时间 | datetime | 首次执行时间 |
| UPDATE\_TIME | 更新时间 | datetime | 重跑的执行时间 |

* 1. 数据研判任务
     1. **研判任务表**

用户在进行数据查询、时空碰撞、相似轨迹等分析时，可以将分析结果以研判任务的形式保存，其存储结构如下：

ANAYSE\_JUDGE\_TASK

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段中文名称 | 字段类型 | 备注 |
| TASK\_ID | 研判任务标识 | bigint |  |
| TASK\_NAME | 研判任务名称 | Varchar(128) |  |
| RES\_STAT | 研判分析结果统计 | String | Json字符串  {"person":3,"car":4,"code":10} |
| USER\_ID | int | 用户标识 |  |
| USER\_NAME | varchar(100) | 用户名称 |  |
| STAR\_TAG | int | 星标值 | 默认为-1,值越大，在任务列表展示时，越排在前面 |
| CREATE\_TIME | 创建时间 | datetime | 首次执行时间 |
| UPDATE\_TIME | 更新时间 | datetime | 重跑的执行时间 |

* + 1. **研判任务分析结果表**

用户在进行数据查询、时空碰撞、相似轨迹等分析时，可以将分析结果以研判任务的形式保存，其存储结构如下：

ANAYSE\_JUDGE\_RES

|  |  |  |  |
| --- | --- | --- | --- |
| 字段名称 | 字段中文名称 | 字段类型 | 备注 |
| TASK\_ID | 研判任务标识 | bigint |  |
| OBJ\_ID | Varchar(32) | 对象标识 | 对象标识 人车指档案标识，码指侦码类型-侦码值 |
| OBJ\_TYPE | tinyint | 对象类型 | 对象类型 1 人 2 车 3码 |
| OBJ\_ATTR | Varchar(128) | 对象的重要属性 | 人: {"PERSON\_NAME":""，"ID\_TYPE":"","ID\_NUMBER":}  车：{"HPHM":"","HPYS":}  码: 空 |

# 待确认问题

console涉及的模块【规则统计配置管理、关系定义的管理、规则统计执行记录、提供的REST服务】是在分析子系统的console中添加还是单独新增一个springboot应用？

确认结果：在新版数据分析的console的前后端程序中增加新做的功能模块。