MySQL 的状态数据大概可以分为两类，一部分是相对动态的数据，例如当前的连接数、内存消耗等，通常来说这些动态数据需要以比较高的频率去收集，最后绘制成图形来观察某段时间内的指标动态变化判断问题

而另外一部分是相对来说比较静态的数据，例如当前数据库中表的数量等，这些相对静态的数据不需要高频率地去收集，一方面来说可以减少数据采集带来的资源消耗，另一方面可以减少数据磁盘空间。

今天我要说的巡检是 MySQL 层的巡检，巡检指标比较少，我这里只是给大家提供一个思路，具体需要采集什么样的指标还是需要大家自己去根据实际需要去定制

比如我这里分几个维度来巡检

1、表

2、索引

3、系统参数

4、账户安全

首先表检查的话我会去做如下表巡检

1.超过 10G 大表(单表 QPS)

2.索引数目超过 8 个的表(索引过多影响增删改效率)

3.碎片超过 1G 的表(影响执行计划)

4.单表超过 2000 万的表(影响 QPS)

5.非默认字符集表(影响执行计划)

6.大字段表(行迁移)

7.字段长度过大的表(浪费)

8.无主键/索引表(影响主从复制)

从索引维度来说可以检查如下

1. 重复索引

2. 字段过多索引

3.未使用的索引

参数检查

1.version

2.innodb_buffer_pool_size

3.innodb_flush_log_at_trx_commit

4.innodb_log_file_size

5.innodb_log_files_in_group

6.innodb_file_per_table

7.innodb_open_files

8.innodb_data_home_dir

9.innodb_flush_method

10.innodb_max_dirty_pages_pct

11.sync_binlog

12.max_connections

13.query_cache_type

14.sort_buffer_size

15.read_buffer_size

16.max_allowed_packet

17.table_open_cache

18.thread_cache_size

19.innodb_thread_concurrency

20.key_buffer_size

21.字符集

22.time_zone

23.默认存储引擎

状态检查

1.opened_files

2.opened_tables

3.max_used_connections

安全检查

1.匿名账户

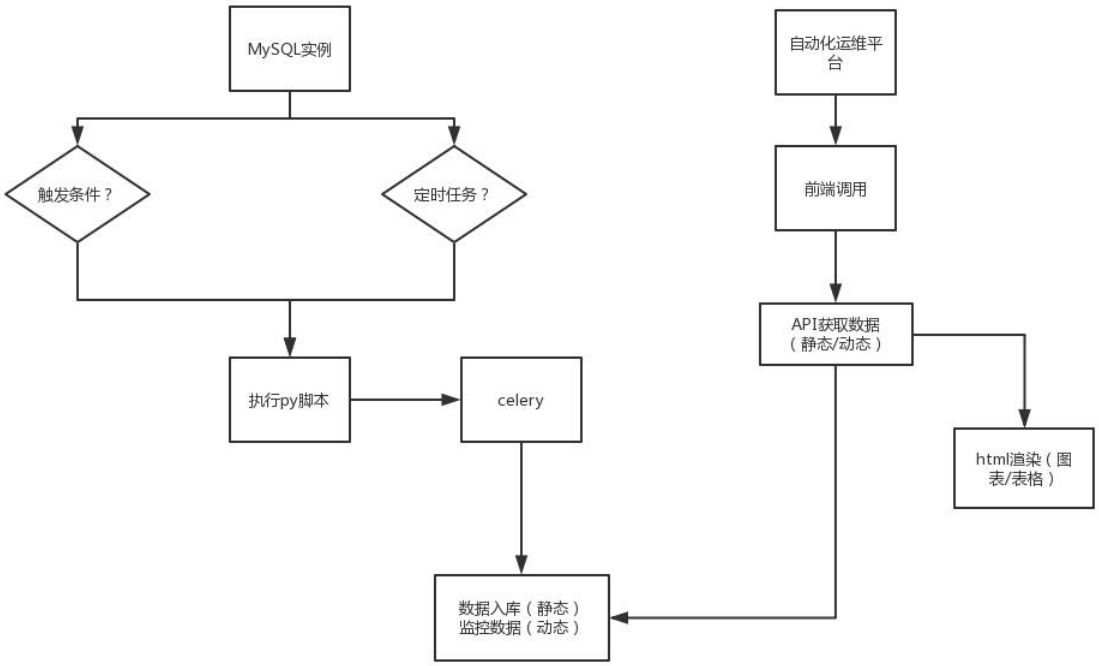
2.无密码账户

3.未限制来源账户

4.高权限账户

之前写过一个简单的 python 脚本，思路其实就是连接到 MySQL，运行一系列的命令获取到相应的指标项，输出到文本，也可以输出到 CSV 的方式

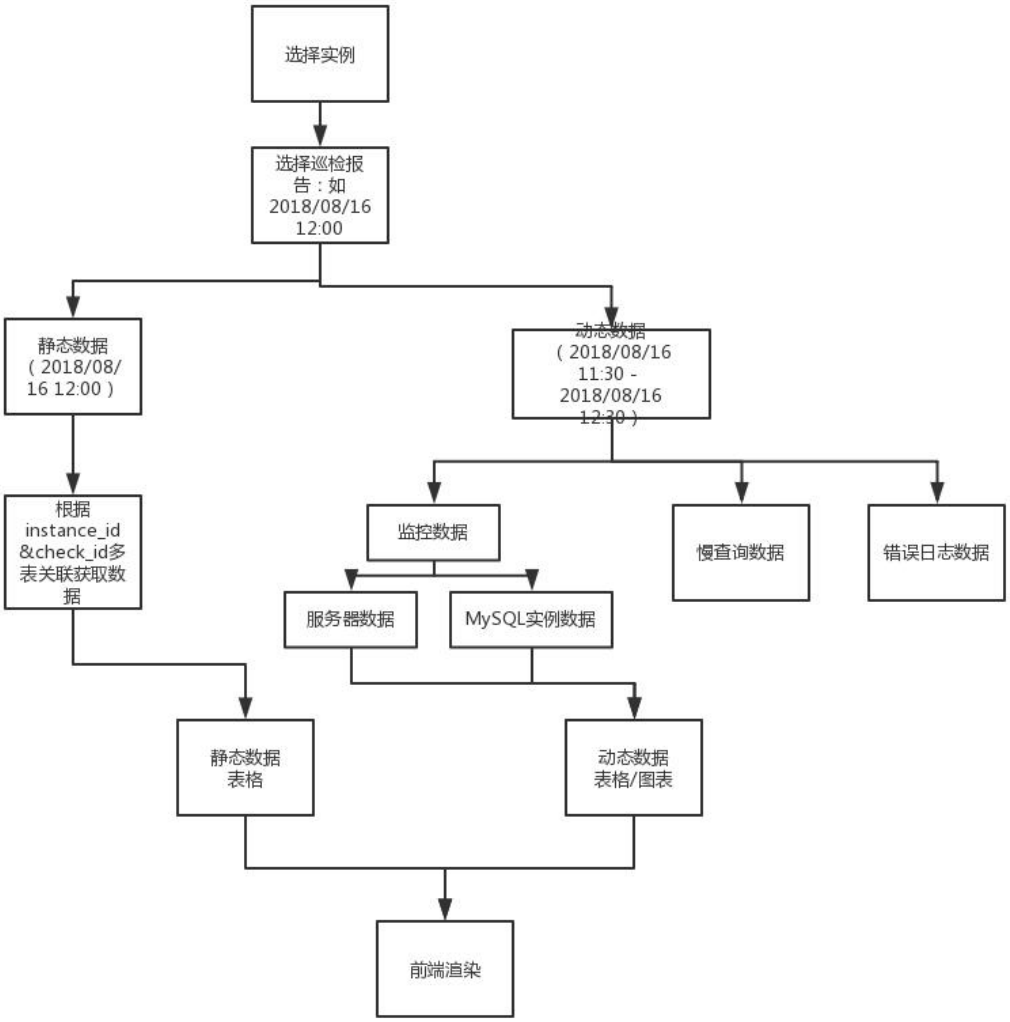
接下来讲一下接入如何接入到自动化运维平台，实际上很多公司都会有 CMDB 系统，也会通过自动化运维平台来管理 MySQL，那么有了这些基础就很好做了。我们的做法是在实例初始化的时候就会默认新建一个管理账号，通过这个管理账号我们可以定期或者手动去做巡检，或者就是定义触发动作去做调用巡检脚本。



在前端展示的时候，我们可以通过 API 获取当前巡检时间的巡检数据+当前巡检时间前后半小时的监控数据（服务器状态+MySQL 监控信息

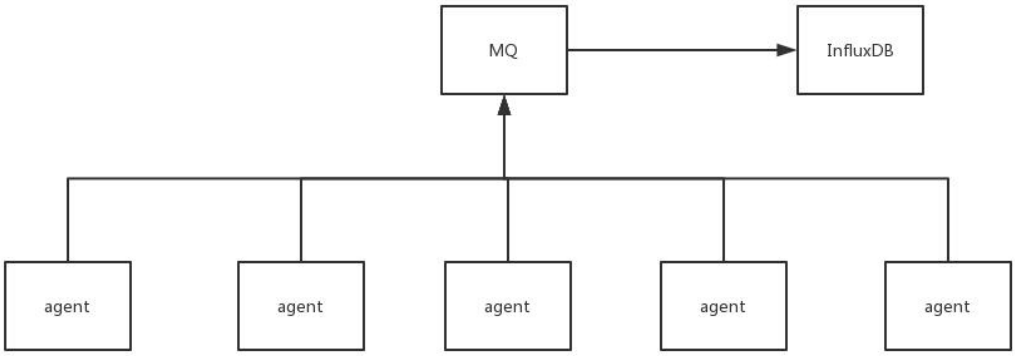
+慢查询+错误日志）

那么我们在展示的时候可以选择用动态的图表来展示，用表格来展示静态数据



后期展望

现在这种架构的不足之处是通过 server 端去通过任务异步去获取数据，如果实例达到一定规模后效率会比较差，另外现在用的是 MySQL 存储状态数据，当量比较多的时候，查询速度也会降低，因此，可以考虑以客户端部署 agent 的方式采集上报数据入库 到 DB 的方式


MQ
InfluxDB
agent
agent
agent
agent
agent