风控指标的构成及指标存储的设计思路

券领取的用户在最近1个月的IP更换频率超过3次

券领取的用户在最近1个月的IP更换的次数总和 > 3

最近1小时内, 用户重新登录的次数大于5

最近1小时内, 用户重新登录的次数总和 > 5

风控规则可以看作由 **关系表达式** 组成

关系表达式 由 左变量（指标），关系运算符, 右变量（阈值）组成

券领取的用户在最近1个月的IP更换频率超过3次

券领取的用户在最近1个月的IP更换的次数总和 > 3

券领取的用户在最近1个月的IP更换的次数总和

最近1小时内，用户重新登录的次数大于5

最近1小时内，用户重新登录的次数总和 > 5

最近1小时, 用户重新登录的次数总和

风控指标由**时间**限制，**空间**限制，**计算方式** 组成

基于滑动窗口思想的风控指标采样思路

风控指标计算的**重点不是如何计算**

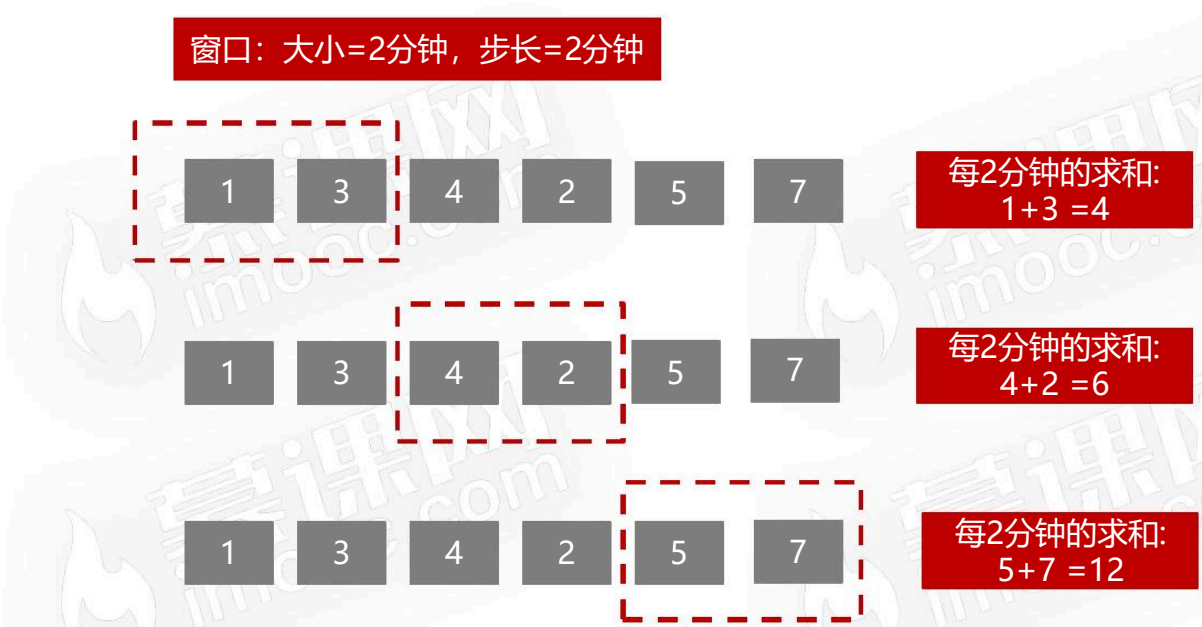
风控指标计算的重点是如何**快速取得指定时间片的数据**

账号在最近3分钟内登陆次数超过5次

相对于现在时间的
动态的时间片

对于动态时间片**如何快速获取数据进行计算？**

对于**指定**动态时间片**如何快速获取结果？**


窗口：大小=2分钟，步长=2分钟
1
3
4
2
5
7
每2分钟的求和:
1+3 =4
1
3
4
2
5
7
每2分钟的求和:
4+2 =6
1
3
4
2
5
7
每2分钟的求和:
5+7 =12

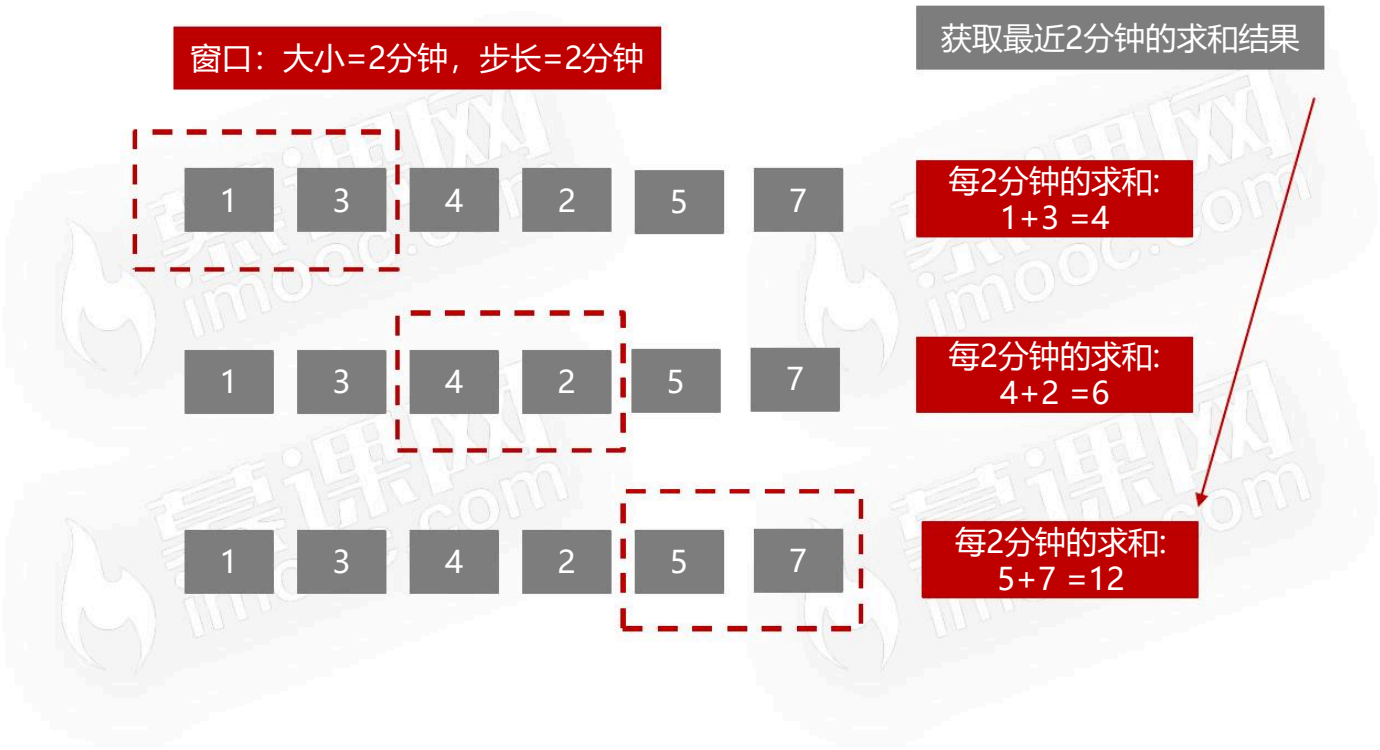
滑动窗口算法：计算连续区间的问题

优点：减少重复计算，降低时间复杂度

滑动窗口算法是一种算法思想

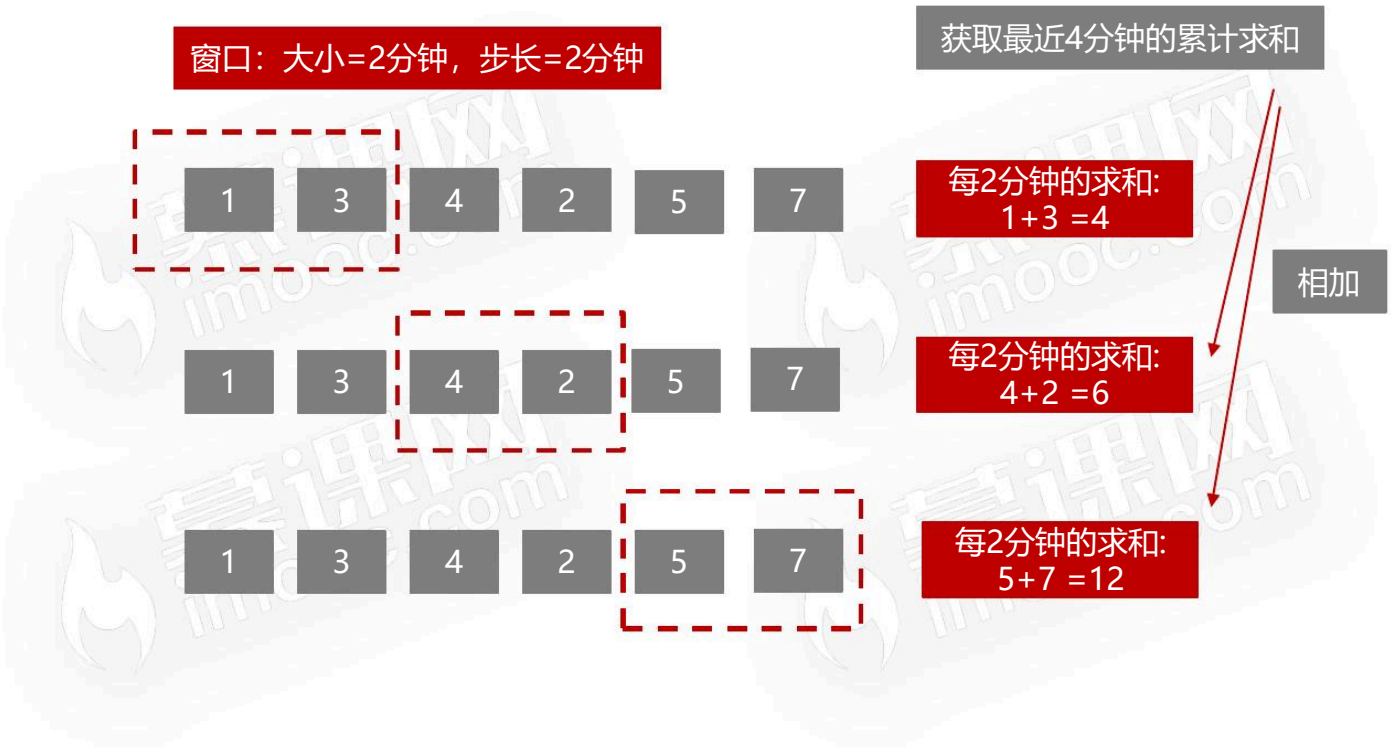
对于动态时间片**如何快速获取数据进行计算？**

对于**指定**动态时间片**如何快速获取结果？**


窗口：大小=2分钟，步长=2分钟
获取最近2分钟的求和结果
1
3
4
2
5
7
每2分钟的求和:
1+3 =4
每2分钟的求和:
4+2 =6
每2分钟的求和:
5+7 =12

对于动态时间片**如何快速获取数据进行计算？**

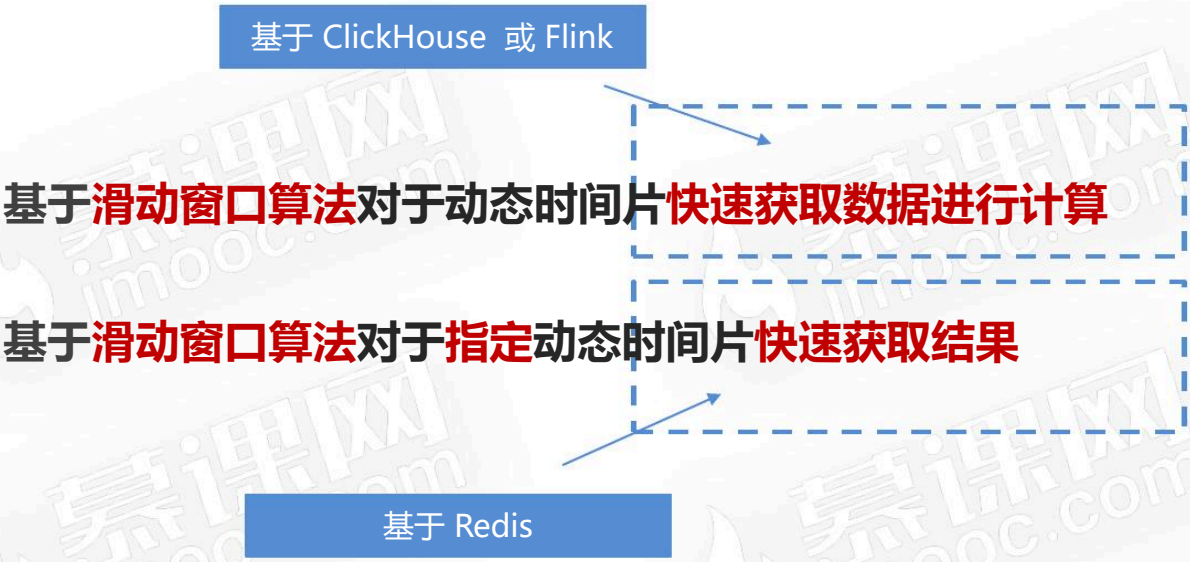
对于**指定**动态时间片**如何快速获取结果？**


窗口：大小=2分钟，步长=2分钟
获取最近4分钟的累计求和
1
3
4
2
5
7
每2分钟的求和:
1+3 =4
相加
每2分钟的求和:
4+2 =6
每2分钟的求和:
5+7 =12

基于滑动窗口的风控指标计算（采样）

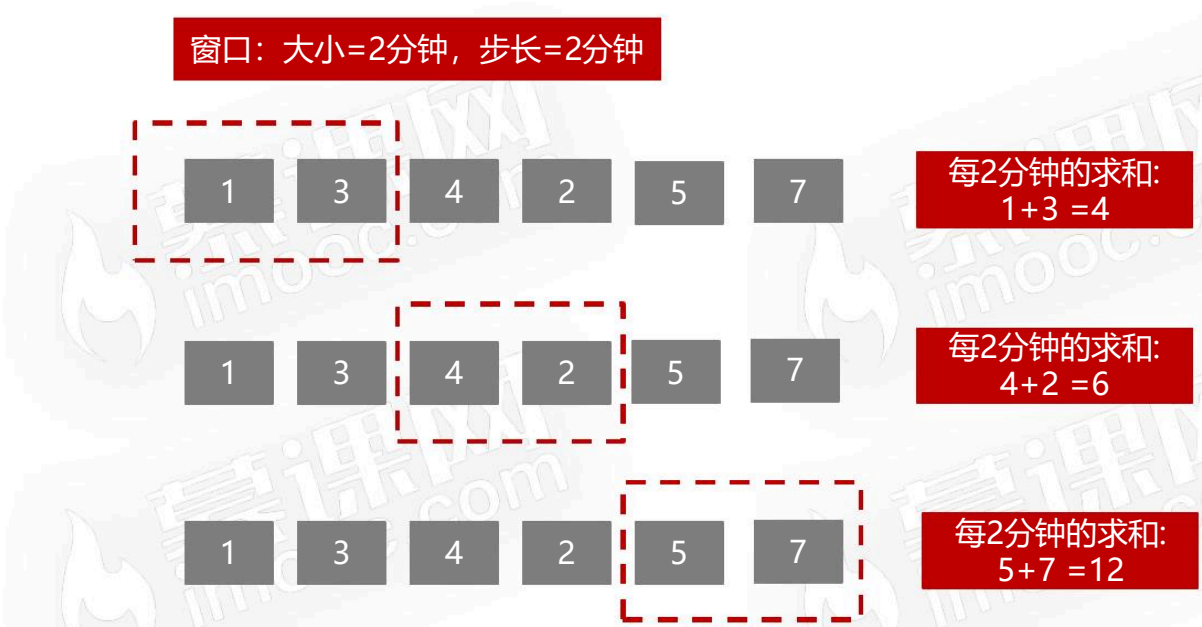
在序列数据中，使用滑动窗口截取序列片段

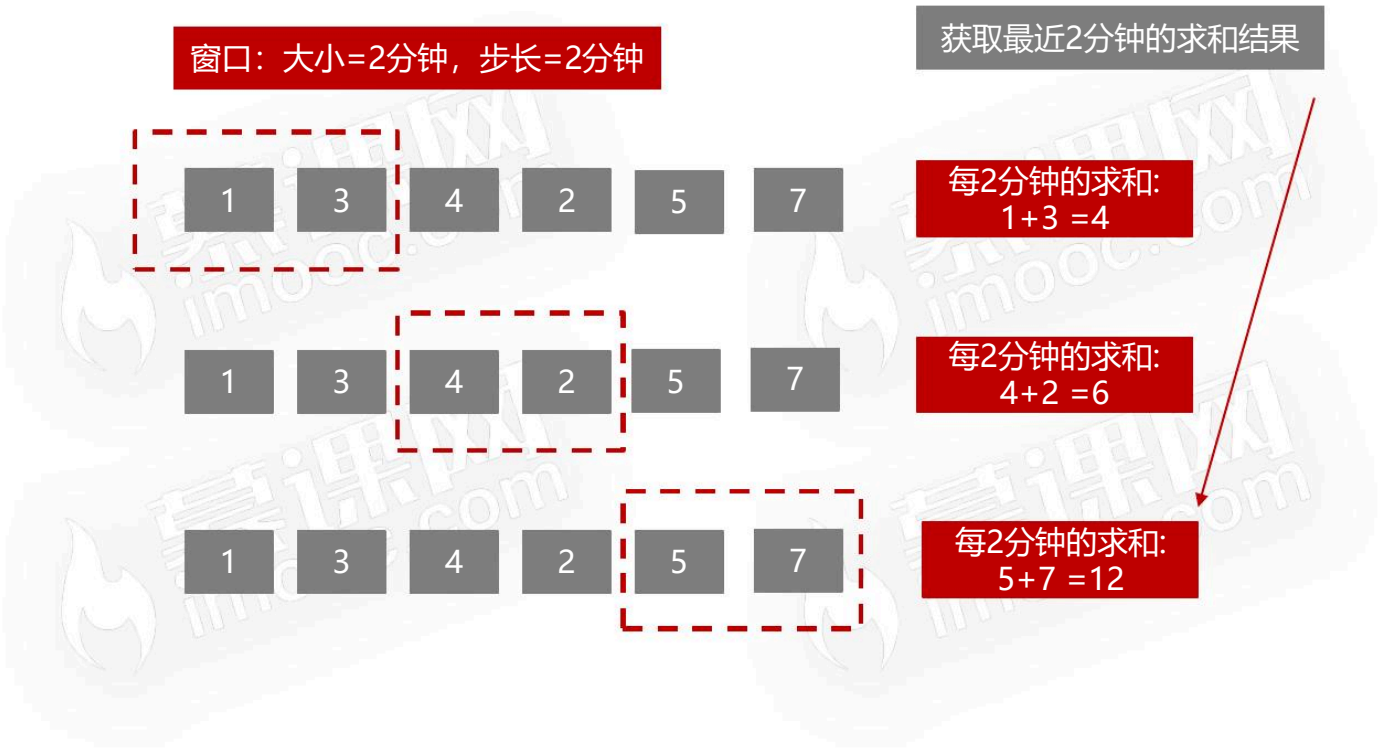
基于 Redis 快速获取风控指标采样的思路

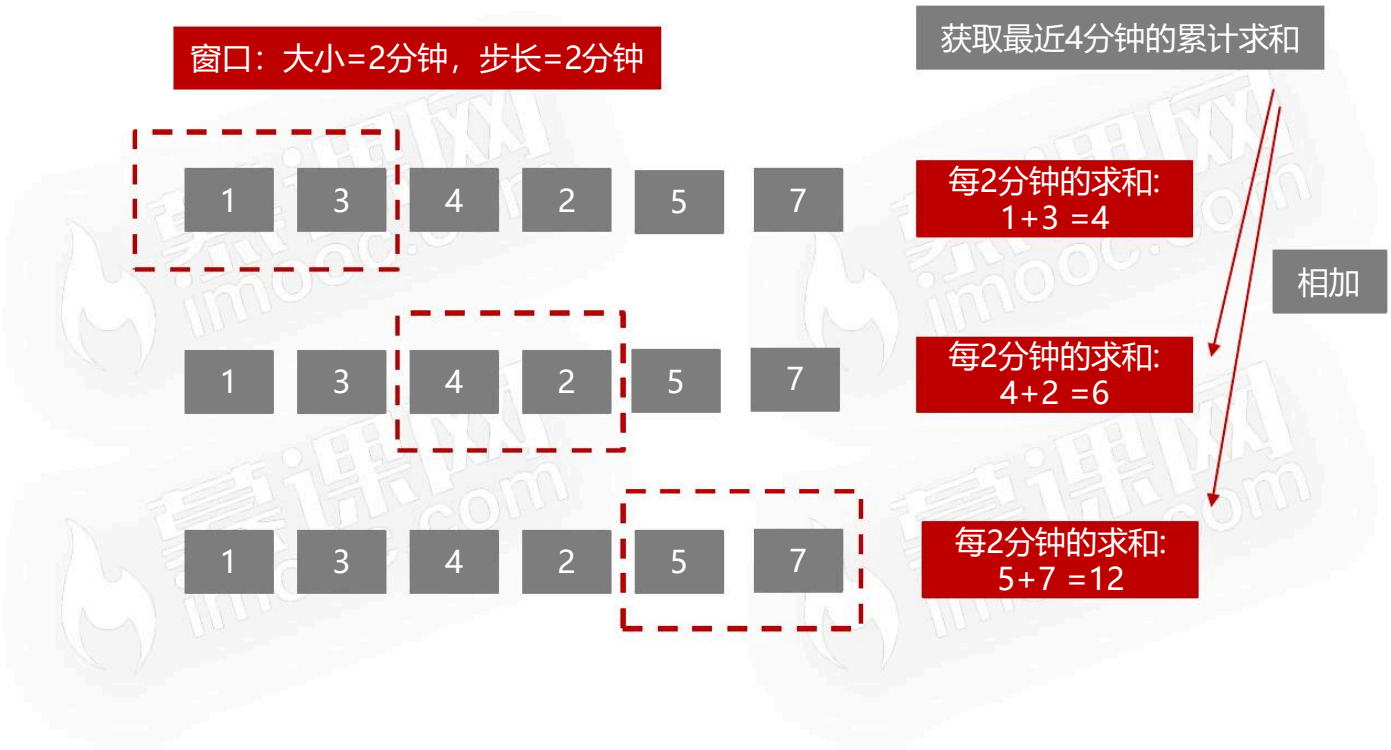

基于 ClickHouse 或 Flink
基于滑动窗口算法对于动态时间片快速获取数据进行计算
基于滑动窗口算法对于指定动态时间片快速获取结果
基于 Redis

账号在最近3分钟内登陆次数超过5次

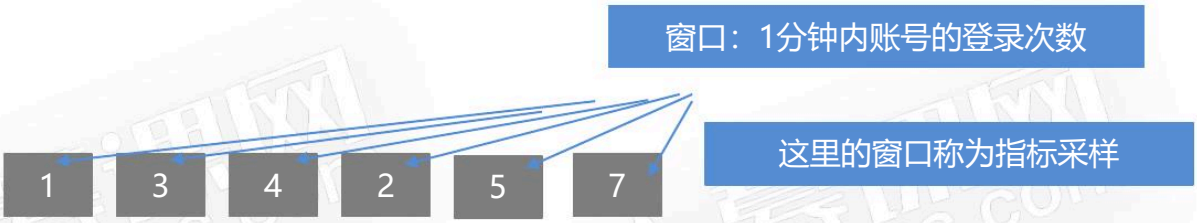
如何在任何时间点，通过 Redis 获取到账号最近3分钟的登录次数


窗口：大小=2分钟，步长=2分钟
1
3
4
2
5
7
每2分钟的求和:
1+3 =4
1
3
4
2
5
7
每2分钟的求和:
4+2 =6
1
3
4
2
5
7
每2分钟的求和:
5+7 =12


窗口：大小=2分钟，步长=2分钟
获取最近2分钟的求和结果
1
3
4
2
5
7
每2分钟的求和:
1+3 =4
每2分钟的求和:
4+2 =6
每2分钟的求和:
5+7 =12


窗口：大小=2分钟，步长=2分钟
获取最近4分钟的累计求和
1
3
4
2
5
7
每2分钟的求和:
1+3 =4
相加
每2分钟的求和:
4+2 =6
每2分钟的求和:
5+7 =12

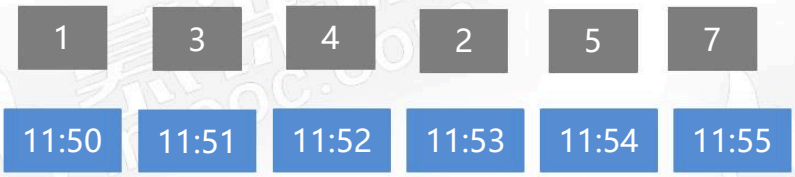
如何在任何时间点，通过 Redis 快速获取到账号最近3分钟的登录次数


窗口：1分钟内账号的登录次数
1
3
4
2
5
7
这里的窗口称为指标采样


1
3
4
2
5
7
11:40
11:45
11:48
11:51
11:54
11:55
（暂定）Redis 数据的写入时间
Redis 存储的账号1分钟内登录次数的数据


假设现在时间是 12:00，需要获取 11:57 ~ 12:00 账号的登录次数
1
3
4
2
5
7
11:40
11:45
11:48
11:51
11:54
11:55
（暂定）Redis 数据的写入时间
Redis 存储的账号1分钟内登录次数的数据

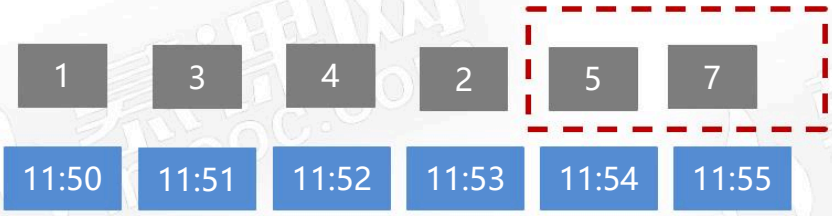
第1种情况: 假设现在时间是 12:00，需要获取 11:57 ~ 12:00 账号的登录次数



现在时间(12:00)和最后更新时间(11:55) 时间差已超过3分钟

结果： 最近 3 分钟内账号登录次数为 0

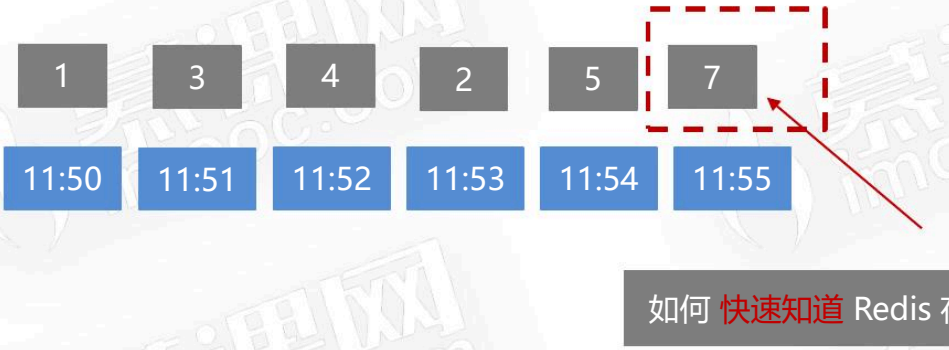
第2种情况: 假设现在时间是 11:57，需要获取 11:54 ~ 11:57 账号的登录次数



现在时间(11:57)和最后更新时间(11:55) 时间差小于 3 分钟

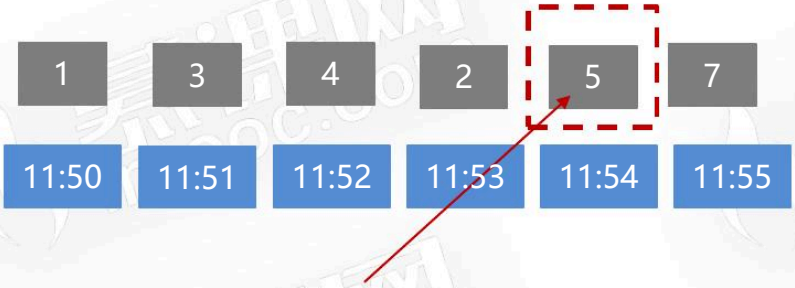
结果： 按照观察, 最近 3 分钟内账号登录次数应该是 5+7 =12

第1种情况: 假设现在时间是 12:00，需要获取 11:57 ~ 12:00 账号的登录次数

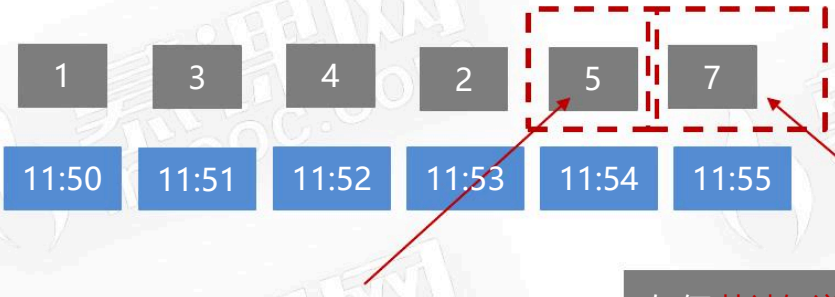


如何 快速知道 Redis 存在 7 这个指标采样值？

第2种情况: 假设现在时间是 11:57，需要获取 11:54 ~ 11:57 账号的登录次数



如何 快速知道 需要包括 5 这个指标采样值？



如何 快速知道 Redis 存在 7 这个指标采样值？

如何 快速知道 需要包括 5 这个指标采样值？

思路：通过 Redis 的 Key 的名称

指标采样 key 的名称: 风控指标唯一id + **指标采样编号**

账号在最近3分钟内登陆次数超过5次

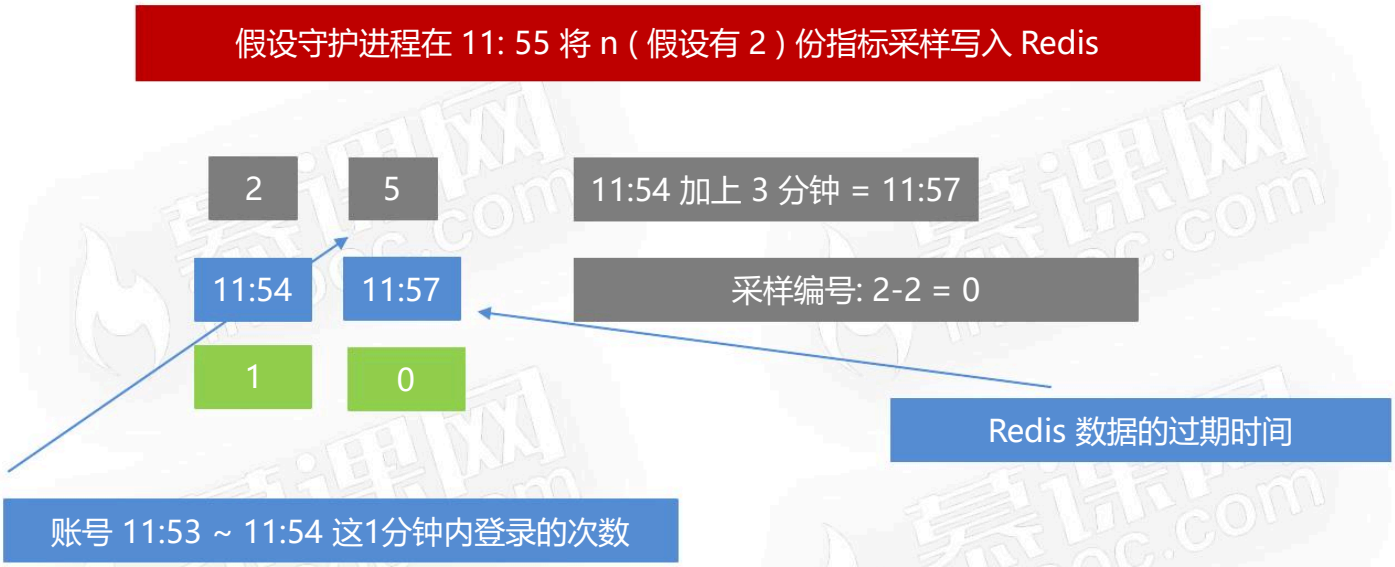
守护进程将3分钟内的指标采样写入 Redis

每个指标采样是 1 分钟内的登录次数

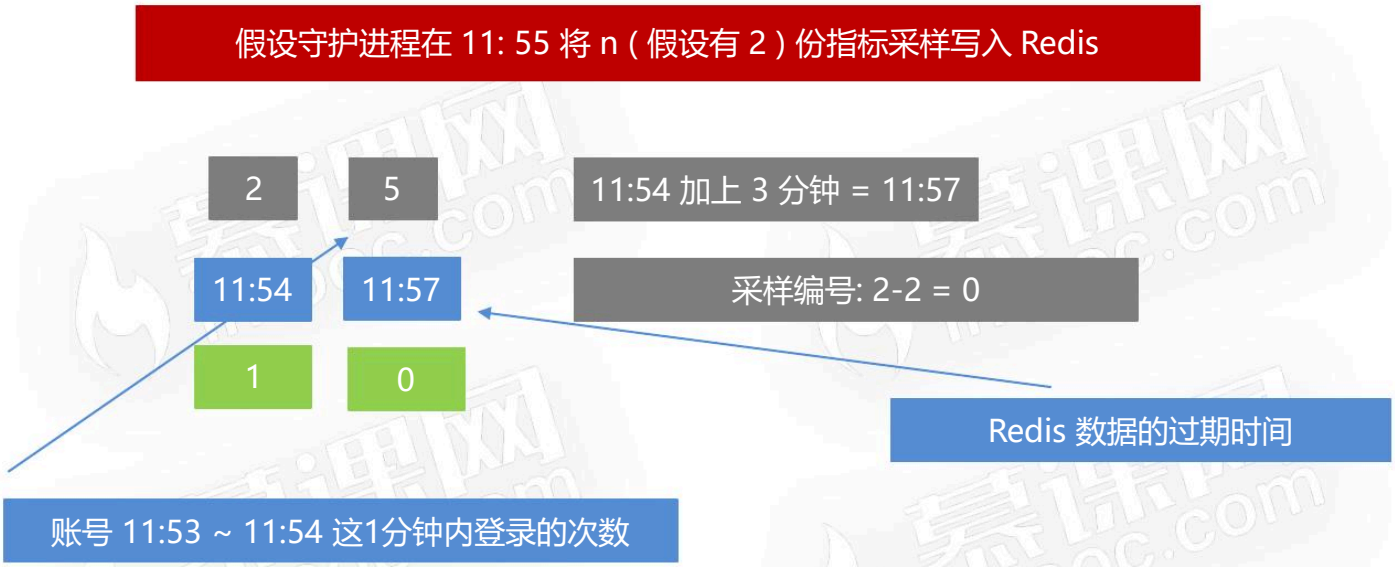
3 分钟内的指标采样最多可以划分为 3 份

3 分钟内的指标采样最少可以划分为 0 份


假设守护进程在 11: 55 将 n (假设有 2) 份指标采样写入 Redis
2
过期时间: 11:51 加上 3 分钟 = 11:54
11:54
采样编号: 2-1 = 1
1
Redis 数据的过期时间
账号 11:50 ~ 11:51 这1分钟内登录的次数


假设守护进程在 11: 55 将 n (假设有 2) 份指标采样写入 Redis
2
5
11:54 加上 3 分钟 = 11:57
11:54
11:57
采样编号: 2-2 = 0
1
0
Redis 数据的过期时间
账号 11:53 ~ 11:54 这1分钟内登录的次数


假设守护进程在 11: 55 将 n (假设有 2) 份指标采样写入 Redis
2
过期时间: 11:51 加上 3 分钟 = 11:54
11:54
采样编号: 2-1 = 1
1
Redis 数据的过期时间
账号 11:50 ~ 11:51 这1分钟内登录的次数


假设守护进程在 11: 55 将 n (假设有 2) 份指标采样写入 Redis
2
5
11:54 加上 3 分钟 = 11:57
11:54
11:57
采样编号: 2-2 = 0
1
0
Redis 数据的过期时间
账号 11:53 ~ 11:54 这1分钟内登录的次数

假设守护进程在 11: 55 将 n (假设有 2) 份指标采样写入 Redis

假设在 11: 56 读取 Redis 的指标采样进行最近3分钟内的登录次数计算

任何时间点，通过Redis 快速获取到最近 n 分钟的指标采样:

解决方案：通过 key（指标唯一id+采样编号），以及设置 key 的过期时间

采样编号: 第一份采样编号为 n-1,最后的采样编号肯定为 0,
做循环, 直到找不到为止

过期时间: 每一份的指标采样超过了 n 分钟就会自动删除

注意 1： 过期时间 = 指标采样的真实计算时间 + n 分钟

任何时间点，通过Redis获取到最近 n 分钟的指标采样:

注意 1： 过期时间 = 指标采样的真实计算时间 + n 分钟

注意 2： 设置好指标采样的单位时间，一般设置为 1 分钟

否则过期时间要减去采样单位时间

风控指标在 Redis 唯一 id 的设计思路

采样编号: 第一份采样编号为 n-1,最后的采样编号肯定为 0,
做循环, 直到找不到为止

不一定设置为n-1， 设置为 n 也可以

或者 g (实际只有 g 份采样) 也可以
(前提：g != n && g < n)

不一定设置为0， 设置为 1也可以

任意时间点，通过 Redis 快速获取到最近 n 分钟的指标采样:

解决方案：通过 key（指标唯一id+采样编号），以及设置 key 的过期时间

指标采样 key 的名称: 风控指标唯一id + 指标采样编号

Redis 存放的数据:

- 风控指标
- 风控指标的采样

从 key 的名称, 能**快速的获取**到风控指标的**关键信息**

Key = 风控指标 Mysql 自增id

Key = 风控指标 Mysql 自增id + 指标主维度 + 计算方式

Key = 风控指标 Mysql 自增id + 指标主维度 + 计算方式 + 版本号

指标Key = 指标 Mysql 自增id + 指标主维度 + 计算方式 + 版本号

采样Key = 指标 Mysql 自增id + 指标主维度 + 计算方式 + 版本号 + 编号