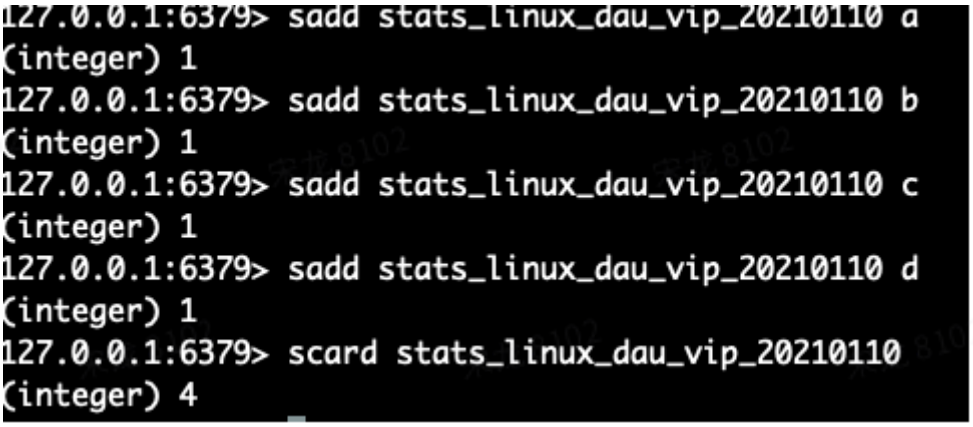
Redis 统计独立用户访问量方案

方案一：使用Set

当一个用户访问的时候，如果用户登陆过，那么我们就使用用户的id，SADD命令，key可以选择URI与对应的日期进行拼凑，当统计某一天的访问量是，直接用`scard`



方案二：bitmap

对于一个32位的int，如果只用来记录id，那么只能够记录一个用户，但如果转成2进制，每位用来表示一个用户，就能够表示32个用户，空间节省了32倍！对于有大量数据的场景，如果使用bitset，可以节省非常多的内存。对于macid，也可以使用哈希算法，把对应的用户标识哈希成一个数字id。bitset非常的节省内存，假设有1亿个用户，也只需要100000000/8/1024/1024约等于12兆内存。

uid	0	0	0	0	0	0	4	3	2	1
二进制位	0	0	0	0	0	0	1	1	1	1

Redis提供了`SETBIT`的方法，可以不停地使用`SETBIT`命令，设置用户已经访问了该页面，也可以使用`GETBIT`的方法查询某个用户是否访问。最后通过`BITCOUNT`可以统计该每天的访问数量。

```
127.0.0.1:6379> SETBIT stats_linux_dau_vip_20211101 100 1
(integer) 0
127.0.0.1:6379> SETBIT stats_linux_dau_vip_20211101 101 1
(integer) 0
127.0.0.1:6379> SETBIT stats_linux_dau_vip_20211101 102 1
(integer) 0
127.0.0.1:6379> getbit stats_linux_dau_vip_20211101 100

(integer) 1
127.0.0.1:6379> bitcount stats_linux_dau_vip_20211101
(integer) 3
```

方案三：概率算法

如果访问量非常大，所需的数量不需要特别精准，可以使用概率算法。Redis中，封装了`HyperLogLog`算法，他是一种基数评估算法，这种算法的特征是一般不存数据具体的值

```
127.0.0.1:6379> pfadd stats_linux_dau_vip_2021 111 222 333 444
(integer) 1
127.0.0.1:6379> pfcount stats_linux_dau_vip_2021
(integer) 4
```

`PFCOUNT`只是一个概率算法，所以可能存在0.81%的误差。 优点占用内存极小，每个 HyperLogLog 键只需要花费 12 KB 内存，就可以计算接近 2^64 个不同元素的基数 Redis HyperLogLog 是用来做基数统计的算法，在输入元素的数量或者体积非常非常大时，计算基数所需的空间总是固定的、并且是很小的。