SDS （Simple Dynamic String，简单动态字符串）是 Redis 底层所使用的字符串表示， 几乎所有的 Redis 模块中都用了 SDS。

Sds 在 Redis 中的主要作用有以下两个：

1.实现字符串对象（StringObject）；

2.在 Redis 程序内部用作 char* 类型的替代品。

在 C 语言中，字符串可以用一个 \0 结尾的 char 数组来表示。

这种简单的字符串表示，在大多数情况下都能满足要求，但是，它并不能高效地支持长度计算和追加（append）这两种操作：c

1.每次计算字符串长度（strlen(s)）的复杂度为 O（N） 。

2.对字符串进行 N 次追加，必定需要对字符串进行 N 次内存重分配（realloc）。

在 Redis 内部， 字符串的追加和长度计算很常见， 而 APPEND 和 STRLEN 更是这两种操作，在 Redis 命令中的直接映射， 这两个简单的操作不应该成为性能的瓶颈。

另外， Redis 除了处理 C 字符串之外， 还需要处理单纯的字节数组， 以及服务器协议等内容，所以为了方便起见， Redis 的字符串表示还应该是二进制安全的： 程序不应对字符串里面保存的数据做任何假设， 数据可以是以 \0 结尾的 C 字符串， 也可以是单纯的字节数组， 或者其他格式的数据。

考虑到这两个原因， Redis 使用 sds 类型替换了 C 语言的默认字符串表示： sds 既可高效地实现追加和长度计算， 同时是二进制安全的。

SDS与C字符串的区别：

1.常数复杂度获取字符串长度。

2.杜绝缓冲区溢出。

3.减少修改字符串是带来的内存冲分配次数（1.空间预分配　　2.惰性空间释放）。

4.二进制安全。

以这几个方面来分析C++中string类和SDS的区别。

C++中string类和SDS的区别：

1.string类同样可以以常数复杂度获取字符串长度。因为string类内部具有记录字符串长度的私有成

2.string类杜绝缓冲区溢出。如果在向string类添加元素时，没有足够的空间容纳新容器，则string

3.(1)string类中也有对空间的预分配，并且比SDS更加灵活，我们可以用reserve(n)函数对其预分

(2)string类中并没有明显的惰性空间释放，但是对string来说当它需求大小小于当前容量时也不会退

4.string类同样保证了二进制安全，其内部也并不以空字符为结尾的标志。他同样定义了记录其字符串