# 第四章 MySQL基础头文件和基础库

在软件开发中，最为典型的思想就是模块化、抽象、分层。实现MySQL这种庞大的软件系统更加需要清晰的模块化设计以及抽象分层。任何软件系统都是建立在操作系统之上的，而且在实现的过程中都会用到一些基础的数据结构和算法，为此MySQL实现了自己的基础类库，包括对底层操作系统调用的封装和基础数据结构和算法的封装。为什么基础的数据结构和算法不使用C++的STL呢？个人思考是：C++的STL是通用化设计，例如它有自己的内存管理系统等等，对于MySQL这种数据库服务器软件来说有时候不能满足需求，而且会完全依赖于STL的实现，自己实现自己的基础库的好处就是更加可控。也可以针对性的做一些性能优化。

MySQL是一个跨平台的数据库管理系统，支持现在市场上流行的几乎所有操作系统。各个操作系统平台的API会有一些差异，为此，MySQL定义了一个my\_global.h文件，该文件主要是一堆的条件编译，用来适配各种不同的操作系统平台和各种编译器，例如：32位平台下，int64被定义为long long，64位平台下则为long，以及最为常用的操作的宏定义，例如求两个数的最大者，最小者、以及各种整数、浮点数的取值范围的定义等。每一个头文件以及C源文件都会包含该文件。该文件的具体内容就不分析了。它位于include/目录下。

前面已经分析了MySQL源码的目录结构，知道其基础库在mysys目录中实现，包括一些基础的数据结构和系统调用的封装。接下来进行剖析。

很多基础库的函数声明都以my开头，对系统调用的封装的声明很多都在my\_sys.h头文件中。具体包括如下几块：

内存管理函数的声明；

一系列的函数指针（例如：信号处理等）；

一系列结构体和枚举的定义；

一系列函数的声明，例如：文件、目录操作函数、格式化输出、cache的读写、字符集、线程等相关的函数。

接下来分析其中的数据结构和对系统调用的封装，为什么要对系统调用进行封装呢？这是为了提供一致的语义，因为Windows下的文件打开和Linux下的不一定一样。

1、基础数据结构和算法

（1）动态数组

动态数组的定义在include/my\_sys.h中，如下：

typedef struct st\_dynamic\_array

{

uchar \*buffer;

uint elements,max\_element;

uint alloc\_increment;

uint size\_of\_element;

} DYNAMIC\_ARRAY;

几个元素的含义如下：

buffer：存放数组中元素的内存区；

elements：数组中元素的个数；

max\_element：数组中可以容纳的最大元素个数；

alloc\_increment：数组在自增长时每次增加的元素个数；

size\_of\_element：元素的大小，例如对于int32型来说该值为4。

实现在mysys/array.c中，具体包括的操作有：初始化数组，在最末尾插入元素（涉及到数组的自增长），删除末端元素，删除某个位置的元素，获取某个位置的元素，设置某个位置的元素，删除整个数组，重置数组大小。这些动作中只有删除整个数组和重置数组大小会调整数组指向的缓冲区。删除整个数组将内存释放掉，重置大小，将将缓冲区设置为可以存放目前元素的大小。

（2）base64编解码

这个不多解释，标准的算法实现。

（3）checksum

使用的crc32，不多解释。

作者：许富博

版权所有，文章以学习和交流为主，切勿用于商业用途。

限于本人水平有限，欢迎大家随时指正，联系方式：

[xufubobo@gmail.com](mailto:xufubobo@gmail.com)

[xufubobo@163.com](mailto:xufubobo@163.com)

[1332841493@qq.com](mailto:1332841493@qq.com)