# C++连接 Mysql 数据库 & 使用预处理读写 BLOB

通常 C++链接 mysql 你每次都需要转换数据，传指针，传大小等一系列复杂操作，是不是顺序很混乱，过程很繁杂。mysql 也为你提供了另外一种方法，那就是 MYSQL_BIND。将数据操作统一化，统一麻烦化。mysqlbind 是一个结构体，根据个人不同需求填充各个数据成员可以存储任意类型数据，当然包括 blob。

## 1.预处理语句（推荐）

对于多次执行的语句，预处理执行比直接执行快，主要原因在于，仅对查询执行一次解析操作。在直接执行的情况下，每次执行语句时，均将进行查询。此外，由于每次执行预处理语句时仅需发送参数的数据，从而减少了网络通信量

预处理机制特点：

减少服务器负荷

提高服务器响应的速度

可以提供参数机制，让客户有更多查询方法

预处理机制数据类型

```
MYSQL_STMT 该结构表示预处理语句
MYSQL_BIND 该结构用于语句输入（发送给服务器的数据值）和输出（从服务器返回的结果值）
```

函数:

```
MYSQL_STMT *mysql_stmt_init(MYSQL *mysql)
    //创建 MYSQL_STMT 句柄。对于该句柄，应使用 mysql_stmt_close(MYSQL_STMT *)释放
```

int mysql_stmt_prepare(MYSQL_STMT *stmt, const char *query, unsigned long length)

//给定 mysql_stmt_init()返回的语句句柄，准备字符串查询指向的 SQL 语句，并返回状态值。字符串长度应由“length”参量给出

my_bool mysql_stmt_bind_param(MYSQL_STMT *stmt, MYSQL_BIND *bind)

//用于为 SQL 语句中的参数标记符绑定数据

my_bool mysql_stmt_bind_result(MYSQL_STMT *stmt, MYSQL_BIND *bind)

//mysql_stmt_bind_result()用于将结果集中的列与数据缓冲和长度缓冲关联（绑定）起来

int mysql_stmt_execute(MYSQL_STMT *stmt)

//mysql_stmt_execute()执行与语句句柄相关的预处理查询

int mysql_stmt_store_result(MYSQL_STMT *stmt)

//以便后续的 mysql_stmt_fetch()调用能返回缓冲数据

int mysql_stmt_fetch(MYSQL_STMT *stmt)

//mysql_stmt_fetch()返回结果集中的下一行

my_bool mysql_stmt_close(MYSQL_STMT *)

//关闭预处理语句

## 2.预处理机制步骤

```
MYSQL_STMT * st;
//对处理的数据类型初始化
MYSQL_STMT *mysql_stmt_init(MYSQL*)       st=mysql_stmt_init(MYSQL*);

//将预处理句柄与具体 sql 语句绑定
int mysql_stmt_prepare(MYSQL_STMT * st,char* sql,int length);

mysql_stmt_prepare(st,sql,strlen(str));

//mysql 语句的参数
select * from tablename where id=? and name=?
//给参数赋值
MYSQL_BIND    para[n]
//1.n 根据语句中参数确定(客户-->服务)        2.n 根据语句中的字段数确定(服务-->客户)

memset(para,0,sizeof(para));
//对参数操作
para[0].buffer_type=MYSQL_TYPE_LONG     //设置参数的数据类型
```

```
int id;
para[0].buffer=&id;     //参数传值

para[1].buffer_type=MYSQL_TYPE_STRING
char str[20];
para[1].buffer_length=sizeof(str);
para[1].buffer=str;

//预处理与参数绑定
mysql_stmt_bind_param(st,para);
//执行
mysql_stmt_execute(st);
//释放预处理机制所占的空间
mysql_stmt_close(MYSQL_STMT *)    mysql_stmt_close(st);
```

---

## 示例：

```
#include <stdio.h>
#include <mysql.h>
#include <string.h>

int main(void)
{
    MYSQL *conn = mysql_init(NULL);         //初始化服务器句柄
    /*登陆服务器*/
    if(!mysql_real_connect(conn, "localhost", "root", "", "test", 0, NULL, 0))
    {
        fprintf(stderr, "mysql_real_connect: %s\n", mysql_error(conn));
        return -1;
    }

    MYSQL_STMT *stmt = mysql_stmt_init(conn); //创建 MYSQL_STMT 句柄

    char *query = "insert into stu values(?, ?);";
    if(mysql_stmt_prepare(stmt, query, strlen(query)))
    {
        fprintf(stderr, "mysql_stmt_prepare: %s\n", mysql_error(conn));
        return -1;
    }

    int id; char name[20];
    printf("id name: ");
    scanf("%d %s", &id, name);
```

```
    MYSQL_BIND params[2];
    memset(params, 0, sizeof(params));
    params[0].buffer_type = MYSQL_TYPE_LONG;
    params[0].buffer = &id;
    params[1].buffer_type = MYSQL_TYPE_STRING;
    params[1].buffer = name;
    params[1].buffer_length = strlen(name);

    mysql_stmt_bind_param(stmt, params);
    mysql_stmt_execute(stmt);                //执行与语句句柄相关的预处理

    mysql_stmt_close(stmt);
    mysql_close(conn);

    return 0;
}
```

---

除了上面的方式可以写入二进制数据之外，还可以用如下常规方法：

## 写入

二进制数据最为常见的就是图片等一些文件信息。虽然我这里不是这类型信息，但确实是二进制数据。具体步骤：

1、定义一个 buffer（如数组）来存储 sql 语句

2、把涉及到二进制数据之前的 sql 语句添加到 buffer 中，可用 sprintf 或 strcpy 等。

3、用 mysql_real_escape_string()函数添加二进制数据到 buffer 中。

4、加上剩余的 sql 语句，形成完整的 sql 语句。

5、利用 mysql_real_query()函数来执行 sql 语句。

具体代码如下：

---

```
#include <stdio.h>
```

```c
#include <stdlib.h>
#include <mysql/mysql.h>
#include <stdint.h>
#include <string.h>

int main(int argc, char *argv[])
{
    MYSQL mysql;
    char sql[256], *end;
    int index, i;
    uint32_t *destIDs;

    if (argc != 2)
    {
        printf("enter error!\n");
        exit(1);
    }
    index = atoi(argv[1]);
    printf("index: %d\n", index);
    destIDs = (uint32_t *)malloc(index * sizeof(uint32_t));
    if (destIDs == NULL)
        printf("malloc error\n");
    for (i = 0; i < index; i++)
        destIDs[i] = i + 1;
    mysql_init(&mysql);
    if (!(mysql_real_connect(&mysql, "localhost", "root", "654321", "dbname", 0,
NULL, 0)))
    {
        fprintf(stderr, "Couldn't connect to engine!\n%s\n",
mysql_error(&mysql));
        perror("");
        exit(1);
    }

    sprintf(sql, "INSERT INTO Task(NumDest, DestIDs) VALUE (%u, ", index);
    end = sql + strlen(sql);
    *end++ = '\'';
    end += mysql_real_escape_string(&mysql, end, (char *)destIDs, index *
sizeof(uint32_t));
    *end++ = '\'';
    *end++ = ')';

    printf("end - sql: %d\n", (unsigned int)(end - sql));
```

```
    if (mysql_real_query(&mysql, sql, (unsigned int)(end - sql)))
    {
        fprintf(stderr, "Query failed (%s)\n", mysql_error(&mysql));
        exit(1);
    }
    mysql_close(&mysql);
    exit(0);
#endif
    return 0;
}
```

## 读取二进制文件

对于二进制文件的读取，也类似。

具体步骤：

1、构造查询字串.

2、执行 mysql _query 查询.（网上有说用 mysql_real_query，未实验）

3、用 mysql_store_result 存储结果.

4、用 mysql_fetch_row 取出一条记录处理.

具体代码如下：

```
void* CFinger_PrintDlg::Read_Mysql()
{
    if (!g_connect) {
        SendMessage(WM_MyMessage, (WPARAM)L"数据库未连接...", NULL);
        return 0;
    }

    MYSQL_RES *res = NULL;
    MYSQL_ROW row;
    unsigned long *row_len;
    char *object = NULL;
    //const char *sql = "select info from final where id=";
    char sql[100] = { NULL };
    unsigned long objsize;
    int ret;
    for (int i = 1; i <= 10; i++) {

        sprintf_s(sql, "select info from final where id='%d'", i);
```

```
        ret = mysql_real_query(&myCont, sql, strlen(sql));
        if (ret) {
            PostMessage(WM_MyMessage, (WPARAM)L"读取失败1", NULL);
        }

        res = mysql_store_result(&myCont);
        if (res == NULL) {
            PostMessage(WM_MyMessage, (WPARAM)L"读取失败2", NULL);
        }

        /* important */
        row = mysql_fetch_row(res);
        row_len = mysql_fetch_lengths(res);  /* get the object's length */
        if (row_len == NULL) {
            PostMessage(WM_MyMessage, (WPARAM)L"读取失败3", NULL);
        }


        objsize = row_len[0];
        object = (char*)malloc(objsize);
        if (object == NULL) {
            PostMessage(WM_MyMessage, (WPARAM)L"读取失败4", NULL);
        }

        memcpy(object, row[0], objsize);


        if (BIOKEY_DB_ADD(ZKFingerHandle, i, objsize, (byte*)object)) {

            PostMessage(WM_MyMessage, (WPARAM)L"读取成功", NULL);
        }

    }
    g_connect = false;
    ;
    //mysql_close(&myCont);
    mysql_free_result(res);
    return NULL;

}
```