**[C语言访问MYSQL数据库的完整的代码例子](http://www.cnblogs.com/rupeng/archive/2012/10/06/2712841.html)**

1、手写安装带mysql sdk 的mysql  
2、新建控制台项目，项目属性中把  
C:\Program Files\MySQL\MySQL Server 5.5\include 加入“包含目录”  
把C:\Program Files\MySQL\MySQL Server 5.5\lib 加入“库目录”

3、stdafx.h中加入  
#include <winsock.h> //注意顺序，要放在mysql.h前  
#include <mysql.h>//控制台项目中要在mysql.h之前include <winsock.h>

//注意lib添加到“库目录”中，而不是“引用目录”中  
#pragma comment(lib, "libmysql.lib")

4、把libmysql.dll放到生成的exe目录下

5、主要的几个例子：  
//执行基本查询  
void test1()  
{  
 MYSQL \*pConn;  
 pConn = mysql\_init(NULL);  
 //第2、3、4、5参数的意思分别是：服务器地址、用户名、密码、数据库名，第6个为mysql端口号（0为默认值3306）  
 if(!mysql\_real\_connect(pConn,"localhost","root","root","test",0,NULL,0))  
 {    
  printf("无法连接数据库:%s",mysql\_error(pConn));  
  return;  
 }  
 mysql\_query(pConn,"set names gbk");//防止乱码。设置和数据库的编码一致就不会乱码  
 //SET NAMES x 相当于 SET character\_set\_client = x;SET character\_set\_results = x;SET character\_set\_connection = x;  
 //写set character set gbk;查询不会乱码，但是参数化插入会报错。而set names gbk则都不会乱码

 //mysql\_real\_query比mysql\_query多了个参数: 字符串query的长度, 所以适合有二进制数据的query, 而mysql\_query的字符串query不能包含二进制,因为它以\0为结尾  
 //mysql\_query() 不能传二进制BLOB字段，因为二进制信息中的\0会被误判为语句结束。 mysql\_real\_query() 则可以。  
 if(mysql\_query(pConn,"select \* from persons"))  
 {  
  printf("查询失败:%s",mysql\_error(pConn));  
  return;  
 }

 //mysql\_store\_result是把查询结果一次性取到客户端的离线数据集，当结果比较大时耗内存。  
 //mysql\_use\_result则是查询结果放在服务器上，客户端通过指针逐行读取，节省客户端内存。但是一个MYSQL\*连接同时只能有一个未关闭的mysql\_use\_result查询  
 MYSQL\_RES \*result = mysql\_store\_result(pConn);  
 MYSQL\_ROW row;  
 while(row = mysql\_fetch\_row(result))  
 {  
  printf("%s %s\n",row[1],row[2]);  
 }

 mysql\_free\_result(result);  
 mysql\_close(pConn);  
}

//获得更新行数  
void test2()  
{  
 MYSQL \*pConn;  
 pConn = mysql\_init(NULL);  
 if(!mysql\_real\_connect(pConn,"127.0.0.1","root","root","test",0,NULL,0))  
 {  
  printf("无法连接数据库：%s",mysql\_error(pConn));  
  return;  
 }  
 if(mysql\_query(pConn,"update persons set Age=Age+1"))  
 {  
  printf("执行失败：%s",mysql\_error(pConn));  
  return;  
 }  
 printf("更新成功，共更新完成%d条",mysql\_affected\_rows(pConn));  
 mysql\_close(pConn);  
}

//获得自增id  
void test3()  
{  
 MYSQL \*pConn;  
 pConn = mysql\_init(NULL);  
 if(!mysql\_real\_connect(pConn,"127.0.0.1","root","root","test",0,NULL,0))  
 {  
  printf("无法连接数据库:%s",mysql\_error(pConn));  
  return;  
 }  
 mysql\_query(pConn,"set names gbk");  
 if(mysql\_query(pConn,"insert into persons(Name,Age) values('如鹏网',100)"))  
 {  
  printf("执行insert失败%s",mysql\_error(pConn));  
  return;  
 }  
 printf("执行insert成功，新id=%d",mysql\_insert\_id(pConn));  
 mysql\_close(pConn);

}

//参数化查询  
void test4()  
{  
 MYSQL\* pConn;  
 pConn = mysql\_init(NULL);  
 if(!mysql\_real\_connect(pConn,"127.0.0.1","root","root","test",0,NULL,0))  
 {  
  printf("数据库连接失败：%s",mysql\_error(pConn));  
  return;  
 }

 mysql\_query(pConn,"set names gbk");  
 MYSQL\_STMT    \*stmt;      
    MYSQL\_BIND    bind[2];     
 memset(bind,0,sizeof(bind));//把is\_null、length等字段默认值设置为NULL等默认值，否则执行会报错

 stmt = mysql\_stmt\_init(pConn);    
 char\* insertSQL="insert into persons(Name,Age) values(?,?)";  
    if (mysql\_stmt\_prepare(stmt, insertSQL, strlen(insertSQL)))      
    {      
        fprintf(stderr, " mysql\_stmt\_prepare(), INSERT failed,%s\n",mysql\_error(pConn));   
        return;      
    }      
 bind[0].buffer\_type= MYSQL\_TYPE\_STRING;      
    bind[0].buffer= "黑马";      
    bind[0].buffer\_length= strlen("黑马"); //如果设定了buffer\_length，则可以不试用length  
      
    int age=3;

 bind[1].buffer\_type= MYSQL\_TYPE\_LONG;      
    bind[1].buffer= &age;    
 bind[1].buffer\_length = sizeof(age);  
       
    if (mysql\_stmt\_bind\_param(stmt, bind))      
    {      
        fprintf(stderr, " mysql\_stmt\_bind\_param() failed %s\n", mysql\_stmt\_error(stmt));      
        return;  
    }      
      
    if (mysql\_stmt\_execute(stmt))      
    {      
        fprintf(stderr, " mysql\_stmt\_execute(), failed %s\n", mysql\_stmt\_error(stmt));      
        return;     
    }      
 mysql\_stmt\_close(stmt);  
 mysql\_close(pConn);    
 printf("参数化执行SQL结束");  
}