

uMySQL 博途库文件

安装及使用说明(V2.1)

By: ZhenJiang MingRun Info. Tech Co. Ltd.,

WB.H

2020-10-9

修正记录:

2020-4-7 文档创建(By:华文博)

2020-5-15 西门子论坛网友 SuperTai 反馈问题

字符串拼接问题的解决方法(By:华文博)

2020-10-9 调整连接断开后再次连接会报错的问题

完善安装过程的截图指导

字符串拼接中出现中文字符无法执行的问题(By:华文博)

目 录

1、	基本技术条件	5
1.1	库的基础知识	5
1.2	库的开发环境	6
1.3	库的导入方式	6
2、	uMySQL 库的内容描述	8
2.1	uMySQL 库原理及函数	8
2.2	Calc_Encryption_SHA1 使用方法	10
2.3	uMySQL_Aux_ConnectValue	11
2.4	uMySQL_Connect	12
2.5	uMySQL_Use	13
2.6	uMySQL_Query	14
2.7	Wchar_To_UTF8	15
3、	MySQL 数据库部分	16
3.1	数据库版本及安装	16
3.2	安装过程	16

3.3 几点特别需要注意的地方	26
3.4 MySQL 数据库软件	30
4、使用过程	31
4.1 博图部分	31
4.2 数据库效果	36
5、使用问题及网友反馈	38
5.1 问题反馈	38
5.2 关于字符串拼接	38

1、基本技术条件

1.1 库的基础知识

库文件(Library)是一种程序的集合，也就是将常用的函数、函数块、结构体(数据类型)、数据块整合在一个文件中，

库文件在Portal中分为项目库和全局库：

项目库：每个项目都有自己的库，即项目库。在项目库中，可以存储想要在项目中多次使用的对象。项目库始终随当前项目一起打开、保存和关闭。

也就是说，项目库是针对项目的库文件。

全局库：除了项目库之外，可以使用可供多个项目使用的全局库。全局库共有以下三个版本：

A. 系统库

西门子将自己开发的软件产品包含在全局库中。这些库包括可以在项目中使用的现成函数和函数块。这些自带的库无法更改。自带的库无法根据项目进行自动装载。如果在兼容性模式下处理项目，则将加载各 TIA Portal 产品版本对应的库。对于所有其它项目，则将加载最新 TIA Portal 版本提供的库。

B. 企业库

企业库由用户所在组织集中提供，例如，位于网络驱动器上的某个中央文件夹中。TIA Portal 可对相应的企业库进行自动管理。现有版本的企业库更新后，系统将提示用户将相应的企业库更新为最新版本。

C. 用户库

全局用户库与具体项目无关，因此可以传送给其它用户。如果所有用户都需要以写保护方式打开全局用户库，则可对全局用户库进行共享访问。例如，将该库放置在网络驱动器上。

与此同时，用户仍可以使用自己在较低 TIA Portal 版本中创建的全局用户库。但是，如果要继续使用旧版本 TIA Portal 中的全局用户库，则必须先将该库进行升级。

1.2 库的开发环境

当前项目库开发基于 Potal V15.1.

注：如果低于此版本的 Potal 请升级后使用。

1.3 库的导入方式

您拿到的博途库文件由以下几部分组成：

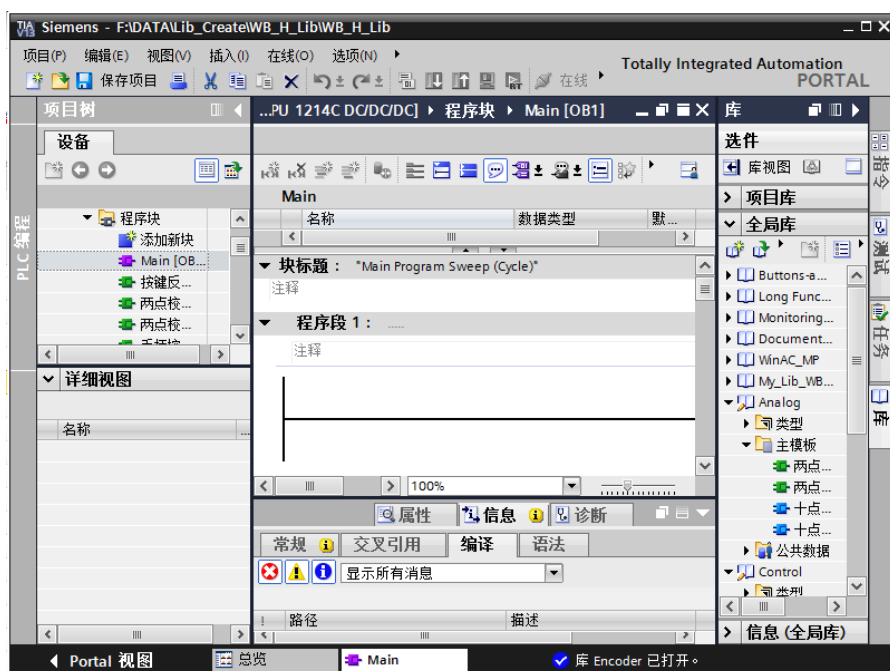
uMySQL_V2.0	2020/4/7 15:32	文件夹	
uMySQL_V0.5.zap15_1	2019/8/15 17:18	Siemens TIA Por...	553 KB
uMySQL库文件使用说明书.docx	2016/2/24 15:54	Microsoft Word ...	2,112 KB

uMySQL_V2.0 uMySQL 库文件 (TIA Portal V15.1)

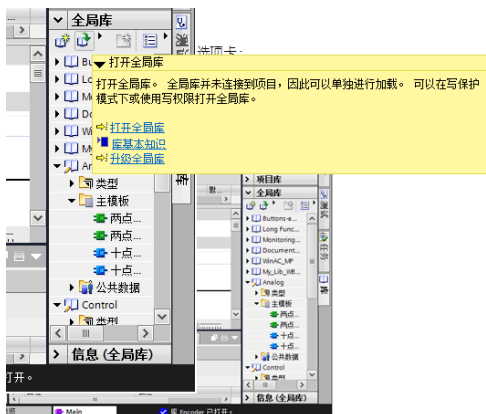
uMySQL_V2.0.zap15_1 源文件部分

uMySQL 库文件使用说明书.docx 库文件简单说明

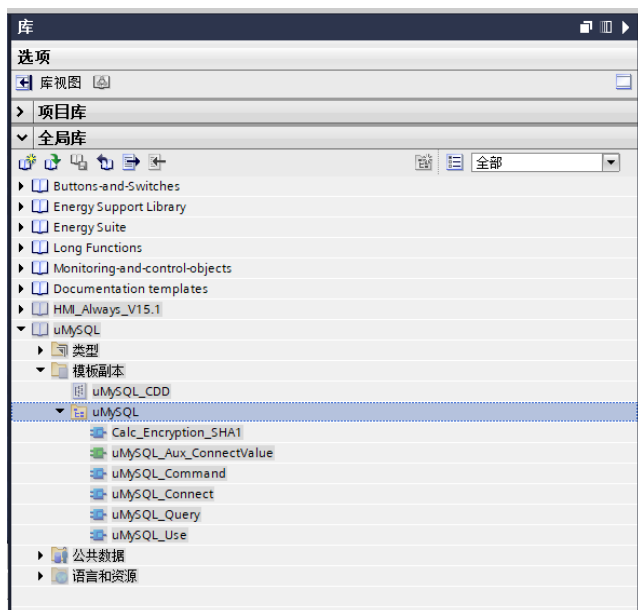
首先打开博途，进入项目视图，在视图的右侧有个库选项卡：



选中库选项卡，选项栏显示如上图所示，点击打开全局库。



选择 uMySQL 库所在文件夹，选中 uMySQL 库文件



此时，uMySQL 库文件会自动导入到博途全局库中。

2、uMySQL 库的内容描述

2.1 uMySQL 库原理及函数

MySQL数据库支持客户端发起TCP/IP连接，只要满足规定的协议，不区分客户端类型，这个是PLC能够连接MySQL数据库的基础。

MySQL数据库连接过程主要分两个阶段：1、握手及连接；2、执行明文SQL语句；这两部分中第二部分很简单，可以理解为把要执行的SQL语句直接发送给数据库，数据库就会执行相应的命令。而握手部分比较复杂，需要有来回的握手，鉴权等，还用到了SHA1校验。

为了方便使用，将用到的函数封装成了以下几个函数/函数块：



具体内容如下：

名称	描述	备注
Calc_Encryption_SHA1	完成SHA1的校验	
uMySQL_Aux_ConnectValue	用于将数据拼接成SQL语句过程中， 将浮点型、整形、布尔型转换成SQL 语句。	
uMySQL_Command	用于拼接SQL语句	
uMySQL_Connect	发起数据库连接	
uMySQL_Query	执行SQL语句，Inser，Update等都 通过这个命令执行。	
uMySQL_Use	因为一个服务器中会有多个数据库， 所有使用过程中可能要切换使用的 数据库，比如打开的时候用的是数据 库DataBase_01,要切换到 DataBase_02，就可以用这个命令	

2.2 Calc_Encryption_SHA1 使用方法

```
163 // Strg_IO_Chars(Strg := #password,  
164 // pChars := 0,  
165 // Cnt => #Temp_SHA_Array_Num,  
166 // Chars := #Temp_SHA_Array);  
167 // #Temp_SHA_Array_Num_Int := UINT_TO_INT(#Temp_SHA_Array_Num);  
168  
169 FOR #Temp_Index_i := 0 TO 19 DO  
170     #Temp_SHA_Array[#Temp_Index_i] := #SaltString[#Temp_Index_i];  
171 END_FOR;  
172 FOR #Temp_Index_i := 20 TO 39 DO  
173     #Temp_SHA_Array[#Temp_Index_i] := #Temp_SHA1_2[#Temp_Index_i - 20];  
174 END_FOR;  
175  
176 #Calc_Encryption_SHA1(IN_Str := #Temp_SHA_Array,  
177     Len := 40,  
178     TempOutStr => #Temp_SHA1_3);  
179  
180 FOR #Temp_Index_i := 0 TO 19 DO  
181     #Token[#Temp_Index_i] := #Temp_SHA1[#Temp_Index_i] XOR #Temp_SHA1_3[#Temp_Index_i];  
182     #SND_Buffer[#Temp_NowNum + #Temp_Index_i] := #Token[#Temp_Index_i];  
183 END_FOR;  
184 #Temp_NowNum := #Temp_NowNum + 20;  
185 #SND_Buffer[#Temp_NowNum] := b#16#00;  
186 #Temp_NowNum := #Temp_NowNum + 1;  
187  
188 // DataBase Name      N      Byte      Last Character      16#00  
189 #Temp_StrLen := LEN(#DB_Name);  
190 FOR #Temp_Index_i := 0 TO #Temp_StrLen - 1 DO  
191     #SND_Buffer[#Temp_NowNum + #Temp_Index_i] := #DB_Name[#Temp_Index_i + 1];  
192 END_FOR;  
193 #Temp_NowNum := #Temp_NowNum + #Temp_StrLen;  
194 #SND_Buffer[#Temp_NowNum] := b#16#00;  
195 #Temp_NowNum := #Temp_NowNum + 1;
```

函数在 uMySQL_Connect 中被调用，不需要用户单独使用

2.3 uMySQL_Aux_ConnectValue

```
1 "测试数据块"."wstring" := WSTRING#'INSERT INTO plcdata(T1,T2,T3,T4,T5,T6,T7,T8,T9,T10)VALUES(';
2 曰"uMySQL_Aux_ConnectValue"(Value:="测试数据块".T_Data[0],
3     PREC:=3,
4     Tail:=WSTRING#',',
5     CommandLine:="测试数据块"."wstring");
6
7 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[1],
8     PREC := 3,
9     Tail := WSTRING#',',
10    CommandLine := "测试数据块"."wstring");
11 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[2],
12    PREC := 3,
13    Tail := WSTRING#',',
14    CommandLine := "测试数据块"."wstring");
15 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[3],
16    PREC := 3,
17    Tail := WSTRING#',',
18    CommandLine := "测试数据块"."wstring");
19 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[4],
20    PREC := 3,
21    Tail := WSTRING#',',
22    CommandLine := "测试数据块"."wstring");
23 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_int[0],
24    PREC := 3,
25    Tail := WSTRING#',',
26    CommandLine := "测试数据块"."wstring");
27 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_int[1],
28    PREC := 3,
29    Tail := WSTRING#',',
30    CommandLine := "测试数据块"."wstring");
31 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_int[2],
32    PREC := 3,
33    Tail := WSTRING#',',
34    CommandLine := "测试数据块"."wstring");
35 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Bool[0],
36    PREC := 3,
37    Tail := WSTRING#',',
38    CommandLine := "测试数据块"."wstring");
39 曰"uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Bool[1],
40    PREC := 3,
41    Tail := WSTRING#')',
42    CommandLine := "测试数据块"."wstring");
```

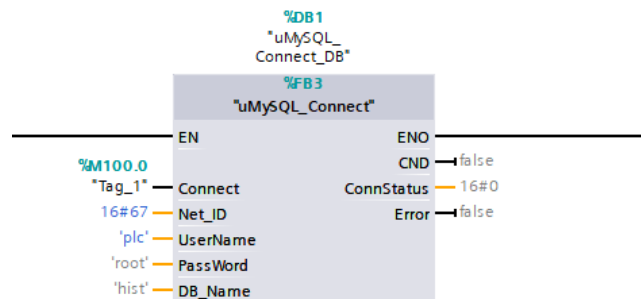
Value: 要转换的变量的值, 由于数据类型是 Variant, 因此此处数据类型可以是 Real、Int、Bool 三种类型, 内部会进行选择转换。

PREC: 当 Value 为浮点型时的小数点后位数。

Tail: 尾巴, 为了拼接 MySQL 语句, 一般是数据后面跟个逗号或者什么的, 特别注意, 最后一个数据, Tail 位置应该是个小括号 ")" (注意红色圈出来的部分, 跟其他都不一样, 其他是 "," 最后一个是 ")")

CommandLine: 要拼接的 SQL 命令行语句, 一般单独建一个 DB 块, 声明一个 WString 的变量就可以。

2.4 uMySQL_Connect



本函数块的作用是向 MySQL 数据库发起连接，其中几个参数解释如下：

Connect: 启动数据库连接，在使用过程中，这个位要一直保持为 True。

Net_ID: 发起 TCP/IP 连接的连接 ID，系统中不能重复。

UserName: 数据库的用户名。

PassWord: 数据库连接密码。

DB_Name: 要连接的数据库，这个地方意义不大，可以后面通过 uMySQL_Use 切换。

CND: 连接成功。

ConnStatus: 连接状态。

Error: 连接错误。

另外，函数设计成一个 DB，有一些参数是在背景数据块中的，特别注意以下几个地方：

uMySQL_Connect_DB										
名称	数据类型	起始值	保持	可从 HMI...	从 H...	在 HMI...	设定值	监控	注释	
1 Input										
2 Connect	Bool	false							发起连接	
3 Net_ID	CONN_OUC	16#0							连接ID	
4 UserName	String	'root'							数据库用户名	
5 PassWord	String	'root'							数据库密码	
6 DB_Name	String	'hist'							数据库名称	
7 Output										
8 CND	Bool	false							连接成功	
9 ConnStatus	Byte	16#0							连接状态	
10 Error	Bool	false							错误	
11 InOut										
12 Static										
13 IP	TCON_IP_v4									
14 Interfaceld	HW_ANY	0							HWIdentifier of IE-interface submodule	
15 ID	CONN_OUC	16#0							connection reference / identifier	
16 ConnectionType	Byte	16#0B							type of connection: 11=TCP/IP, 19=UDP (17=TCP)	
17 ActiveEstablished	Bool	true							active/passive connection establishment	
18 RemoteAddress	IP_V4								remote IP address (IPv4)	
19 ADDR	Array[1..4] of Byte								IPv4 address	
20 ADDR[1]	Byte	16#C0							IPv4 address	
21 ADDR[2]	Byte	16#A8							IPv4 address	
22 ADDR[3]	Byte	16#1F							IPv4 address	
23 ADDR[4]	Byte	16#A8							IPv4 address	
24 RemotePort	UInt	3306							remote UDP/TCP port number	
25 LocalPort	UInt	0							local UDP/TCP port number	
26 CDD	*uMySQL_CDD*									
27 TempBool	Array[0..19] of Bool									
28 REV_Buffer	Array[0..2048] of B...									
29 SND_Buffer	Array[0..2048] of B...									
30 NetConnect	TCON									
31 NetRcv	TRCV									
32 NetRcv_EN	Bool	false								
33 NetRcv_Len	Int	0								
34 NetRcv_Done	Bool	false								
35 Step	Int	0								
36 MessageLen	Int	0								

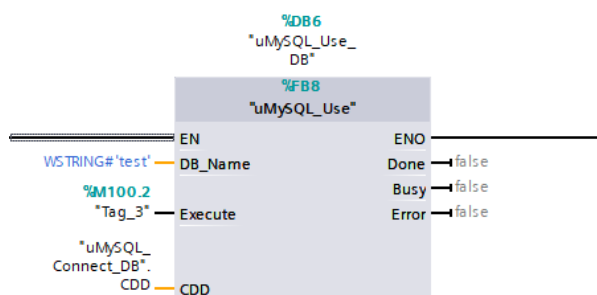
IP->Interfaceld: PLC 网口的 ID, 可以在系统变量中找到, 一般 CPU 本身所带的 X1 默认是 64。

IP->RemoteAddress->ADDR: MySQL 数据库服务器所在的 IP 地址。

IP->RemoteAddress->RemotePort: MySQL 数据库服务器所使用的端口, 默认是 3306。

其他的参数大可不必在意:

2.5 uMySQL_Use



Use 函数用来切换所使用的数据库。

DB_Name: 指定要连接的数据库名称, 数据类型是 WString。

Execute: 执行切换名称, 点动, 指导出现 Done 或者 Error 就可以, 类型为 Bool。

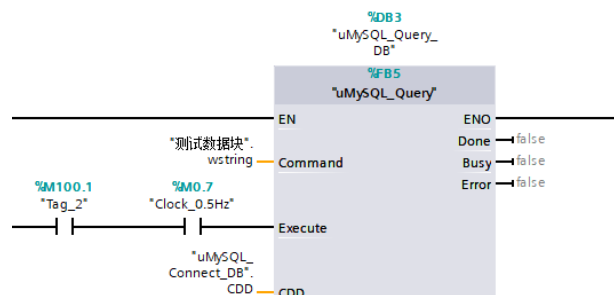
CDD: 这个地方使用 uMySQL_Connect 所生成背景数据块中的.CDD, 类型为 uMySQL_CCD,该数据类型属于库中自定义的数据类型。

Done: 执行成功。

Busy: 正在执行。

Error: 执行错误。

2.6 uMySQL_Query



Query 函数用来执行 SQL 语句。

Command: 要执行的 SQL 语句, 数据类型是 WString。

Execute: 执行动作, 指导出现 Done 或者 Error 就可以, 类型为 Bool。

CDD: 这个地方使用 uMySQL_Connect 所生成背景数据块中的.CDD, 类型为 uMySQL_CCD,该数据类型属于库中自定义的数据类型。

Done: 执行成功。

Busy: 正在执行。

Error: 执行错误。

2.7 Wchar_To_UTF8

```
FOR #Temp_Index_i := 0 TO UINT_TO_INT(#Command_Len) - 1 DO
    #Temp_Dword := "Wchar_To_UTF8"("WChar" := #SendBufferWChar[#Temp_Index_i], Num => #Temp_Num);
    CASE #Temp_Num OF
        1: // 如果是单字符
            #SendBuffer[#Net_Send_Len] := #Temp_Dword.%B0;
            #Net_Send_Len := #Net_Send_Len + 1;
        2: // 如果是两字节
            #SendBuffer[#Net_Send_Len] := #Temp_Dword.%B1;
            #Net_Send_Len := #Net_Send_Len + 1;
            #SendBuffer[#Net_Send_Len] := #Temp_Dword.%B0;
            #Net_Send_Len := #Net_Send_Len + 1;
        ELSE // 如果是其他情况
            #SendBuffer[#Net_Send_Len] := #Temp_Dword.%B2;
            #Net_Send_Len := #Net_Send_Len + 1;
            #SendBuffer[#Net_Send_Len] := #Temp_Dword.%B1;
            #Net_Send_Len := #Net_Send_Len + 1;
            #SendBuffer[#Net_Send_Len] := #Temp_Dword.%B0;
            #Net_Send_Len := #Net_Send_Len + 1;
    END_CASE;
END_FOR;
```

Wchar_To_UTF8 函数用来将 WCHAR 转换为 1 字节、2 字节、3 字节 UTF-8 编码。

Wchar: 要进行转换的 Wchar 字符。

Num: 返还执行后的 UTF8 需要几个字符。

返回值: 返回一个 DWORD 类型的数据, .%B0 用来存放第一个字节

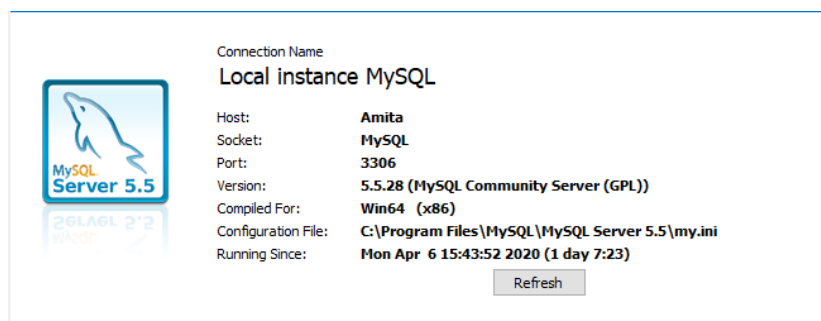
, .%B1 用来存放第二个字节

, .%B2 用来存放第三个字节

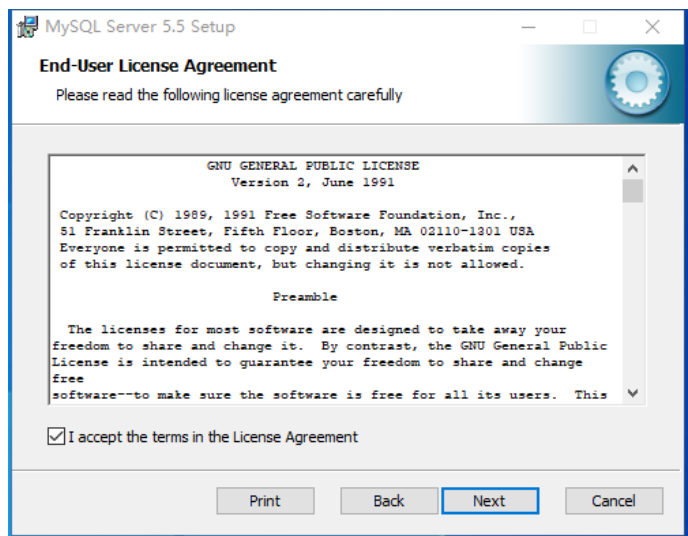
3、MySQL 数据库部分

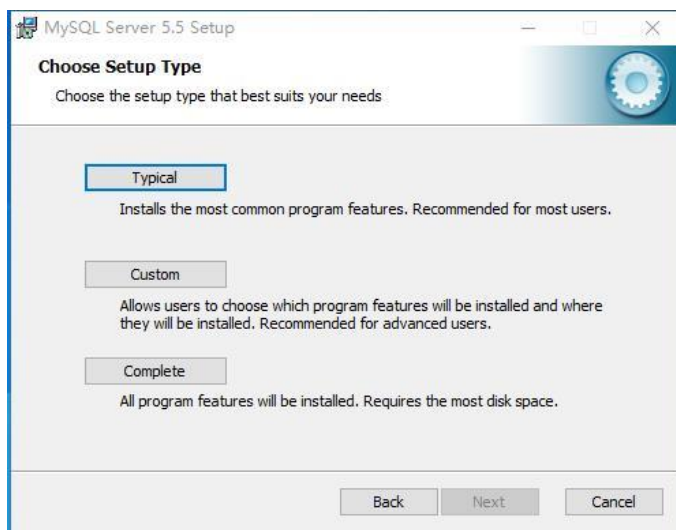
3.1 数据库版本及安装

本人测试使用数据库版本为 5.5.28。

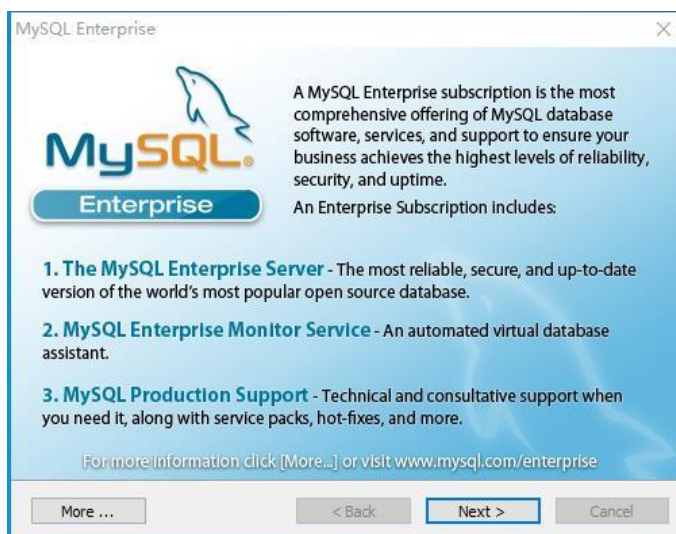
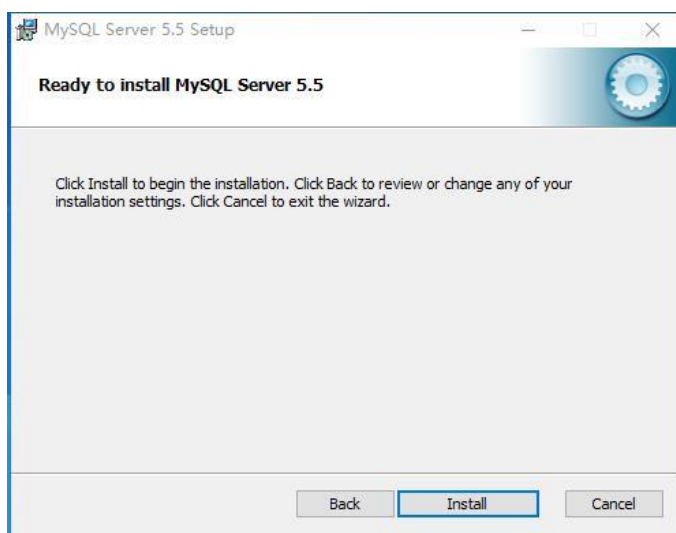


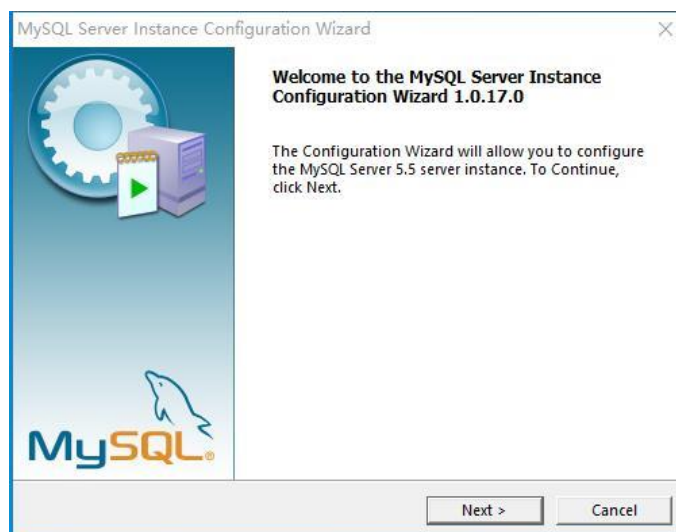
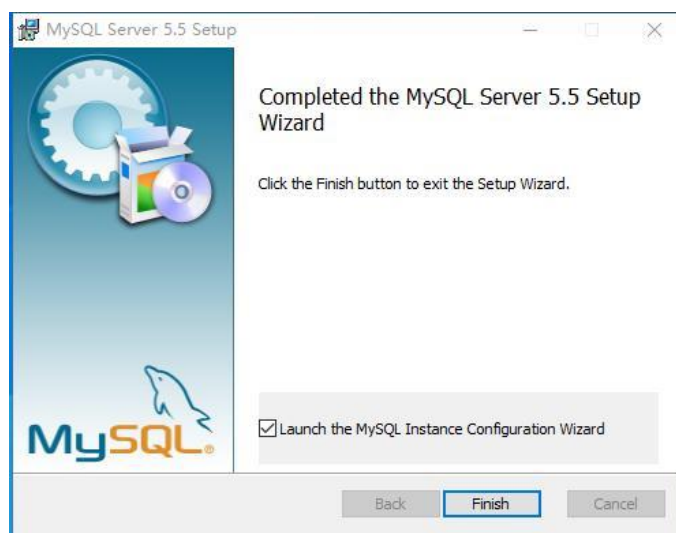
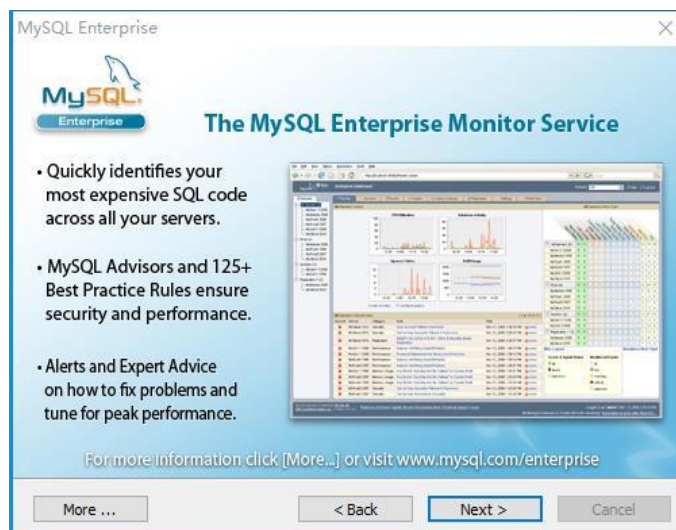
3.2 安装过程

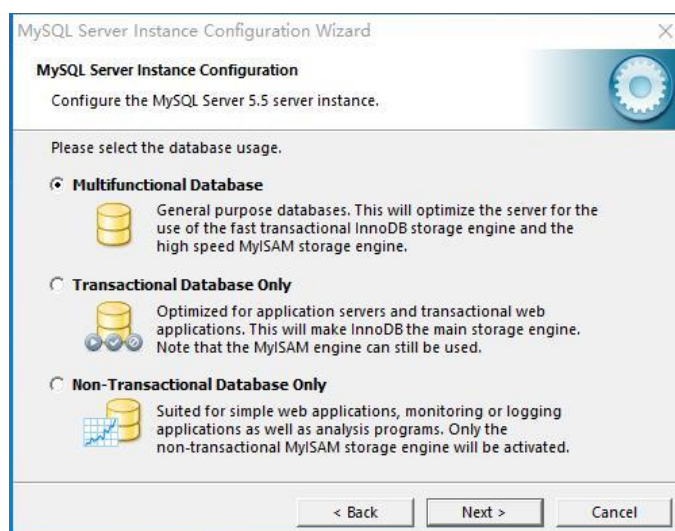
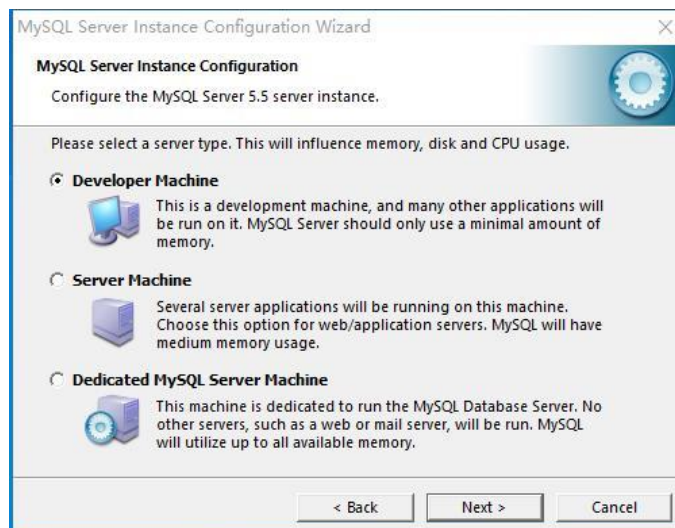
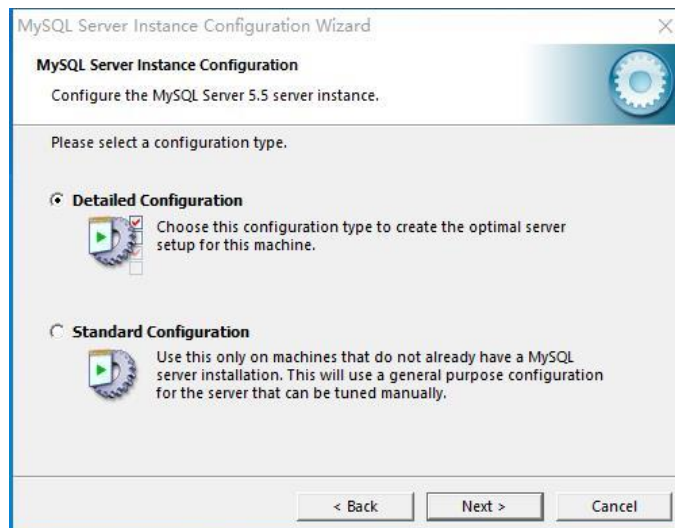


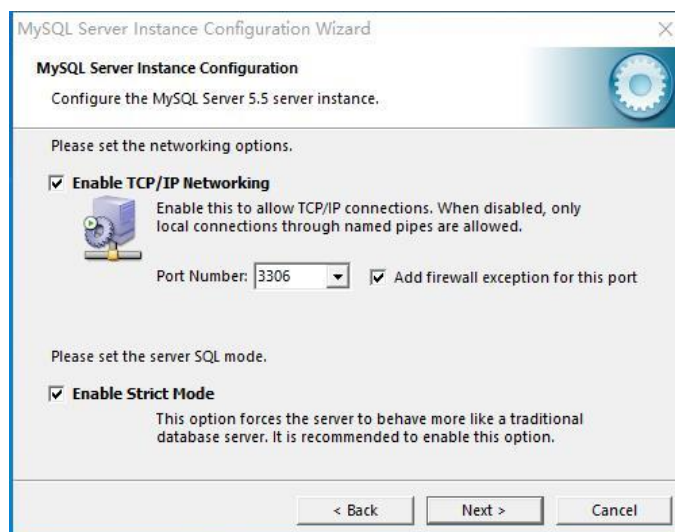
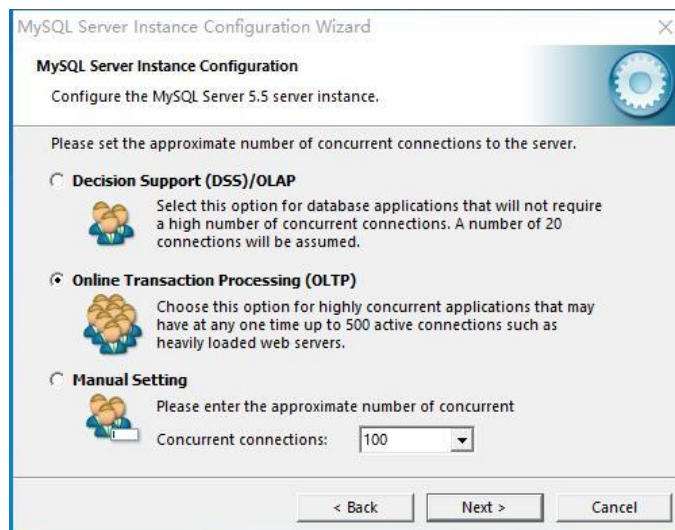
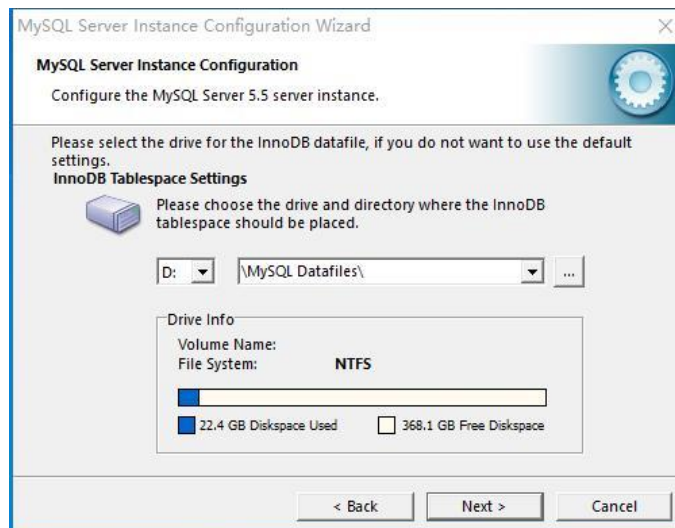


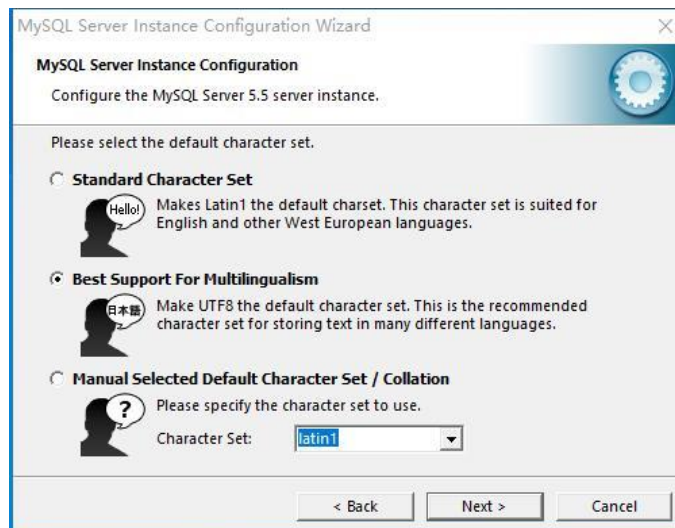
注意选择 Complete 比较靠谱

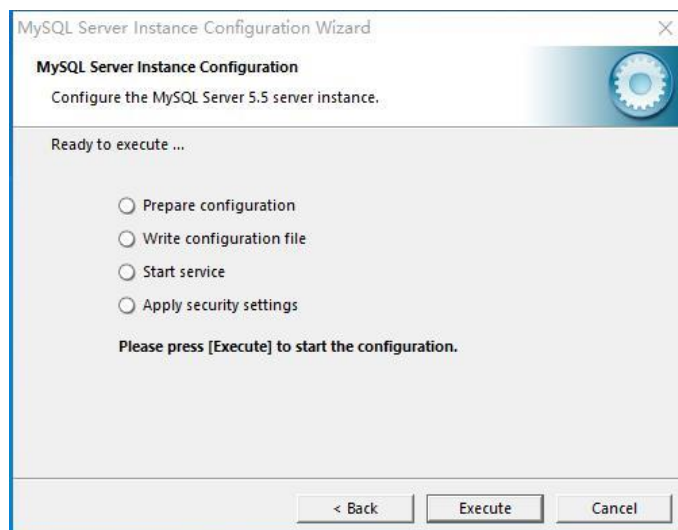




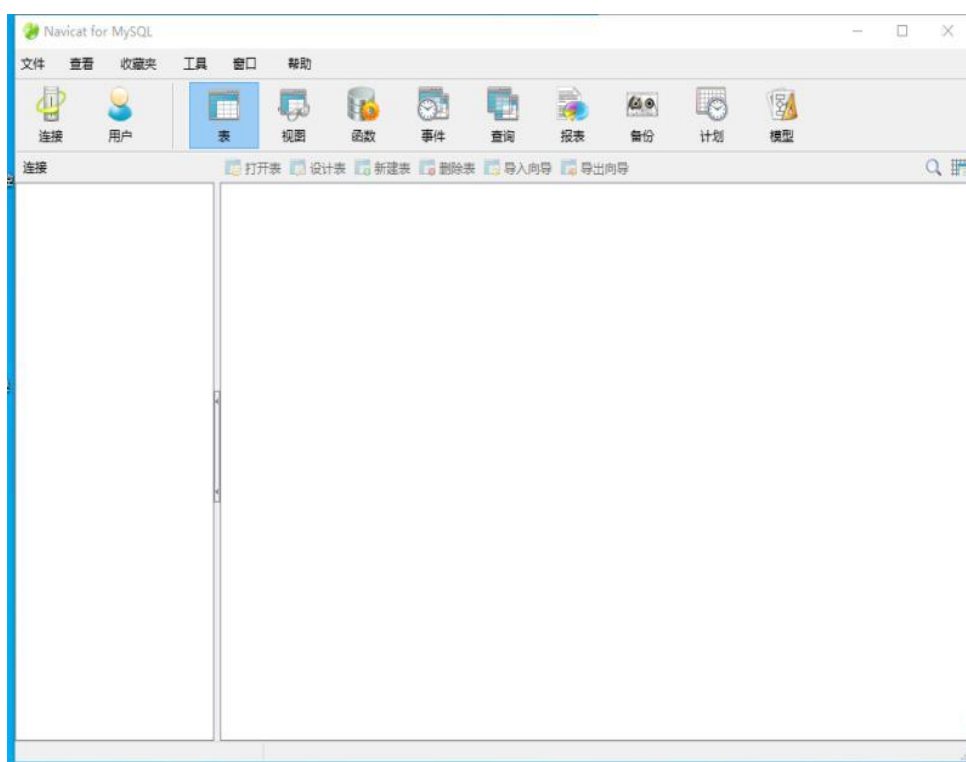


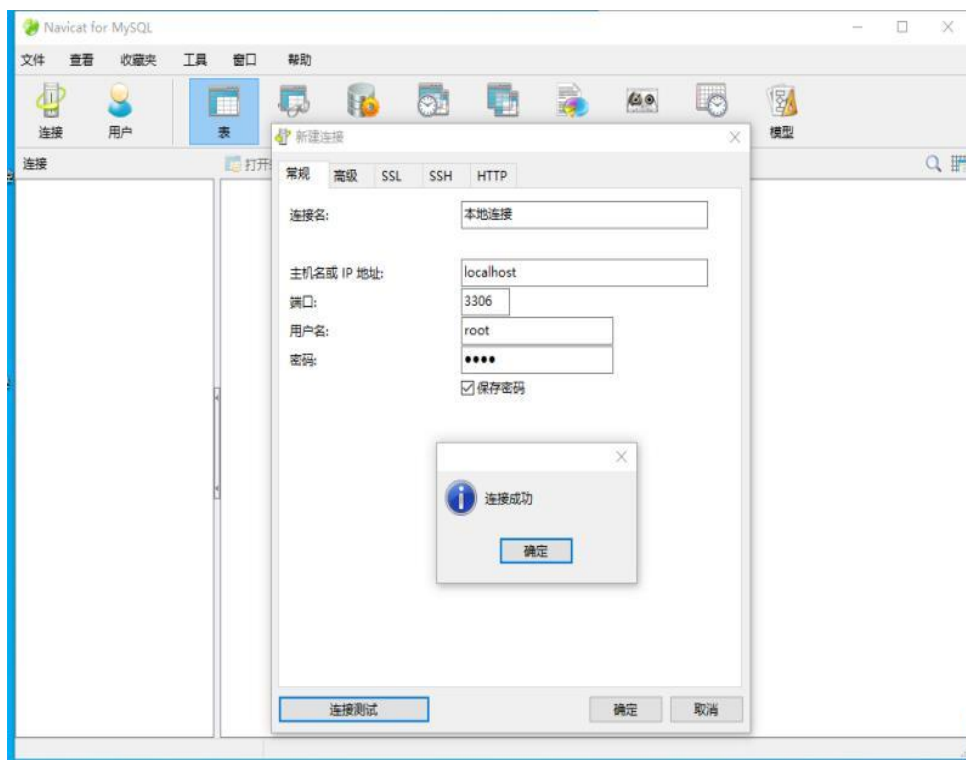




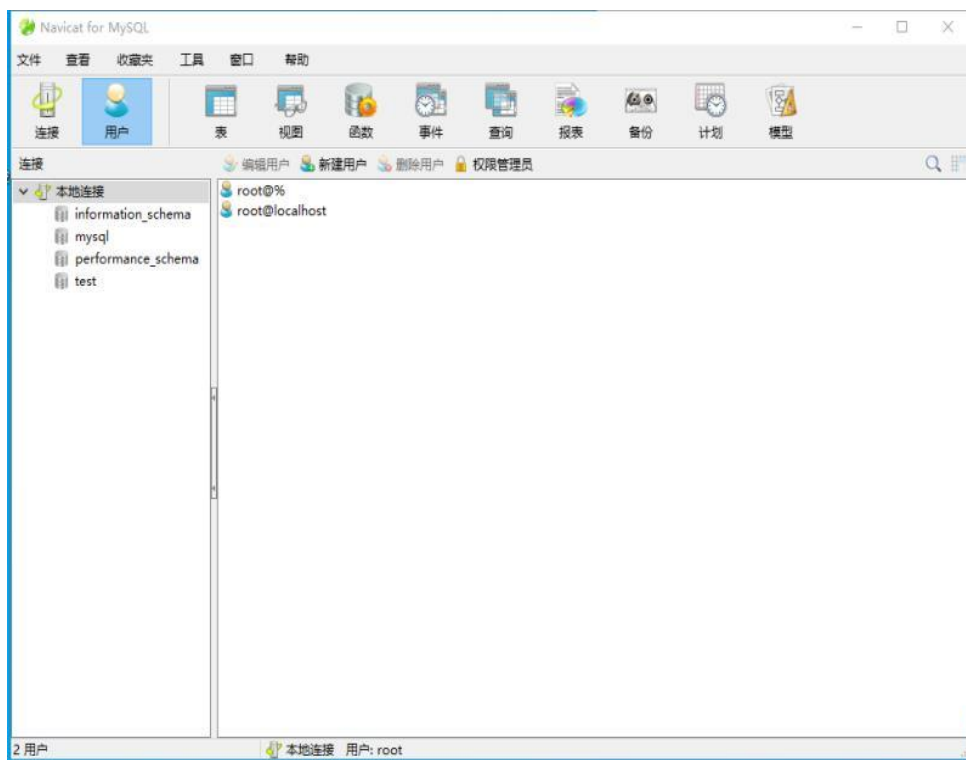


安装完成，在安装数据库的本地电脑上连接测试

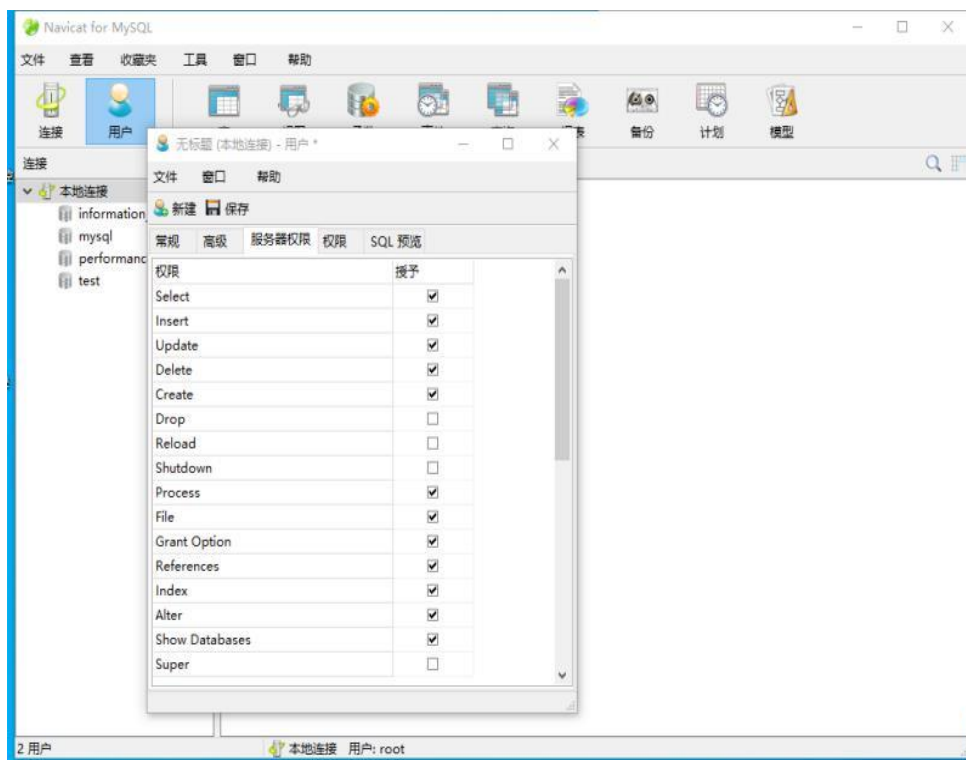




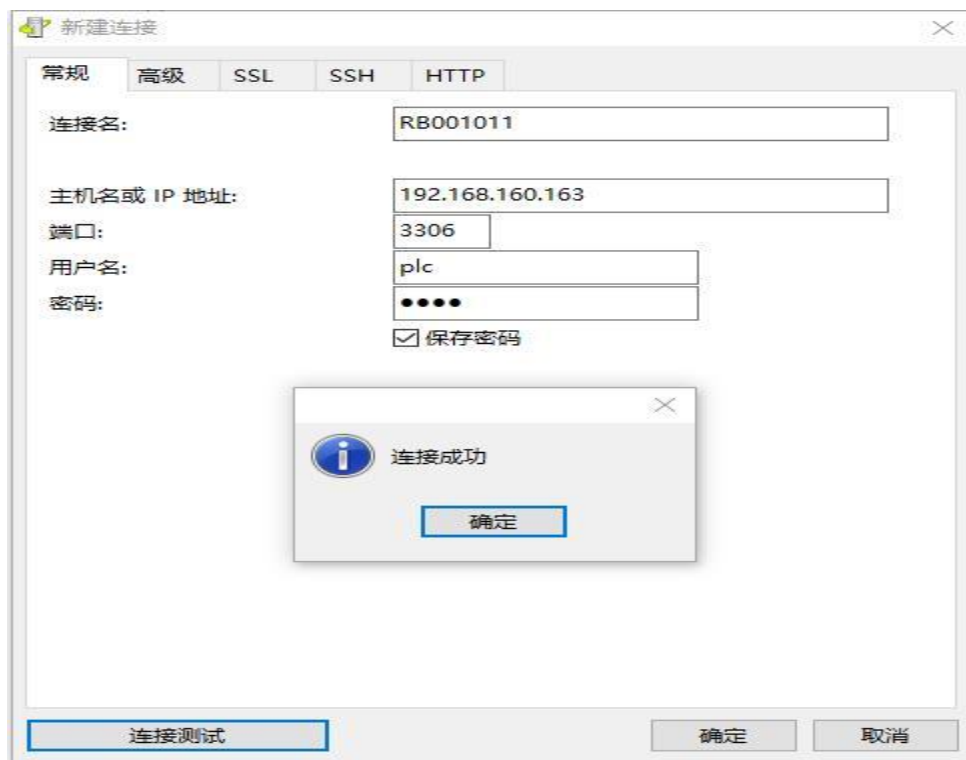
增加 PLC 连接所用的用户名和密码，测试程序中使用的用户名:plc,密码:root,注意主机为%，就是接受所有 IP 地址的连接。



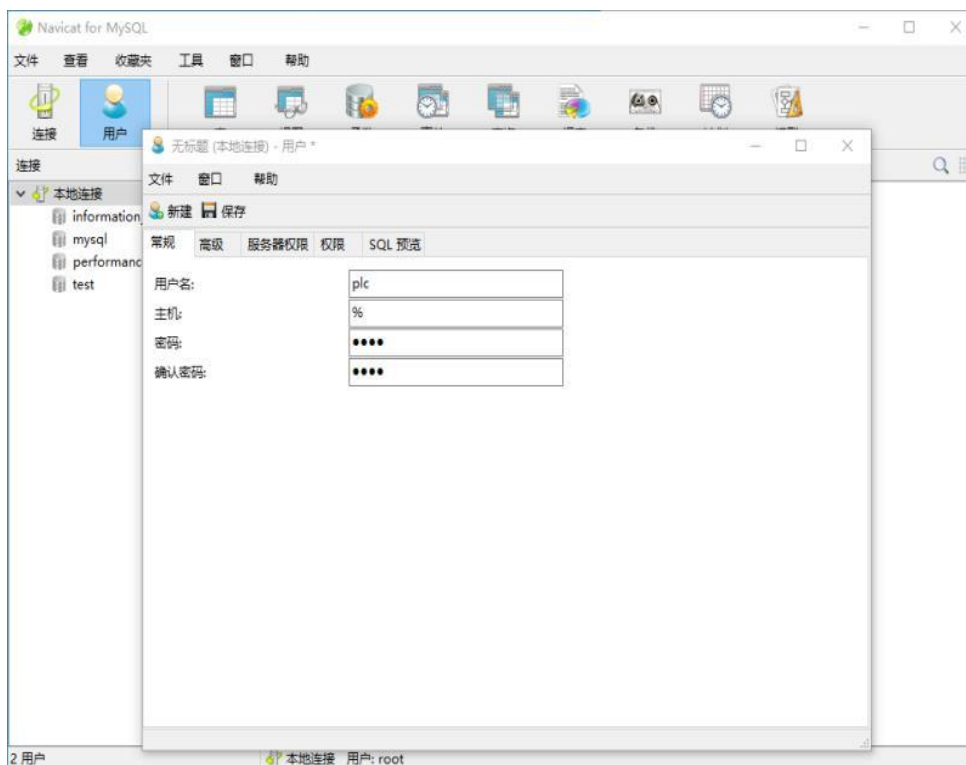
给定 plc 用户权限



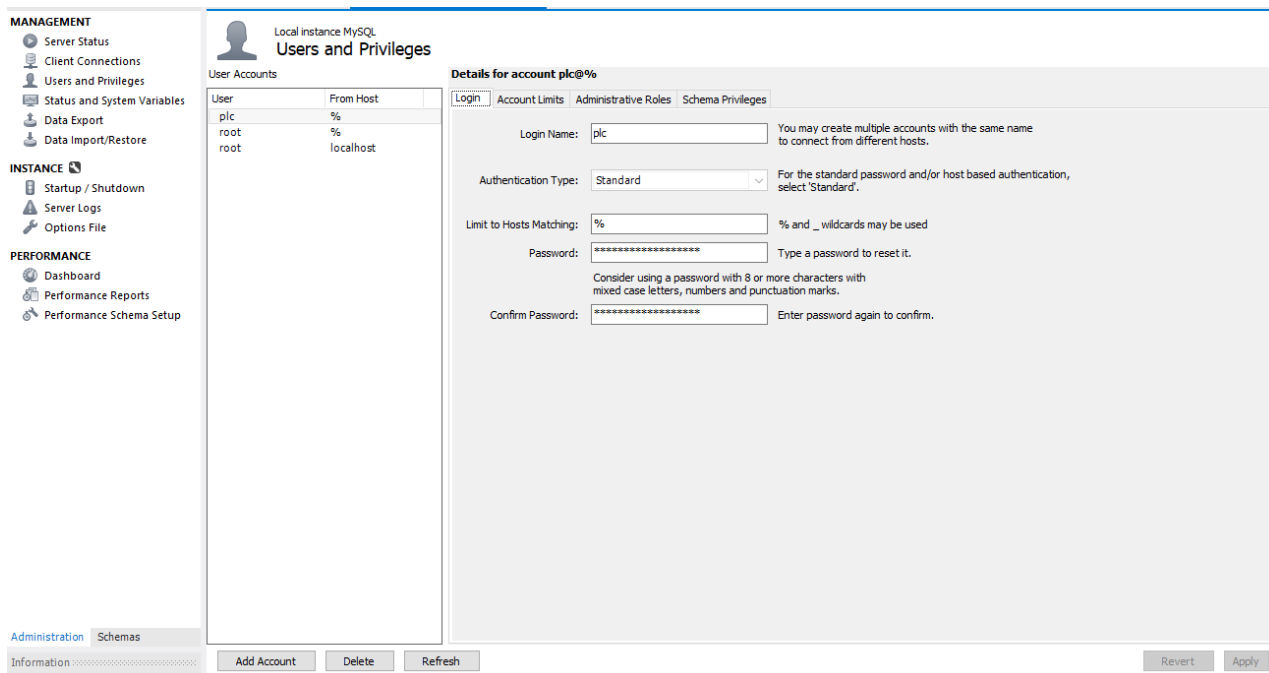
在网络中的其他电脑上，用新增用户和密码(plc，root)测试是否能够连接数据库



数据库部分安装成功。

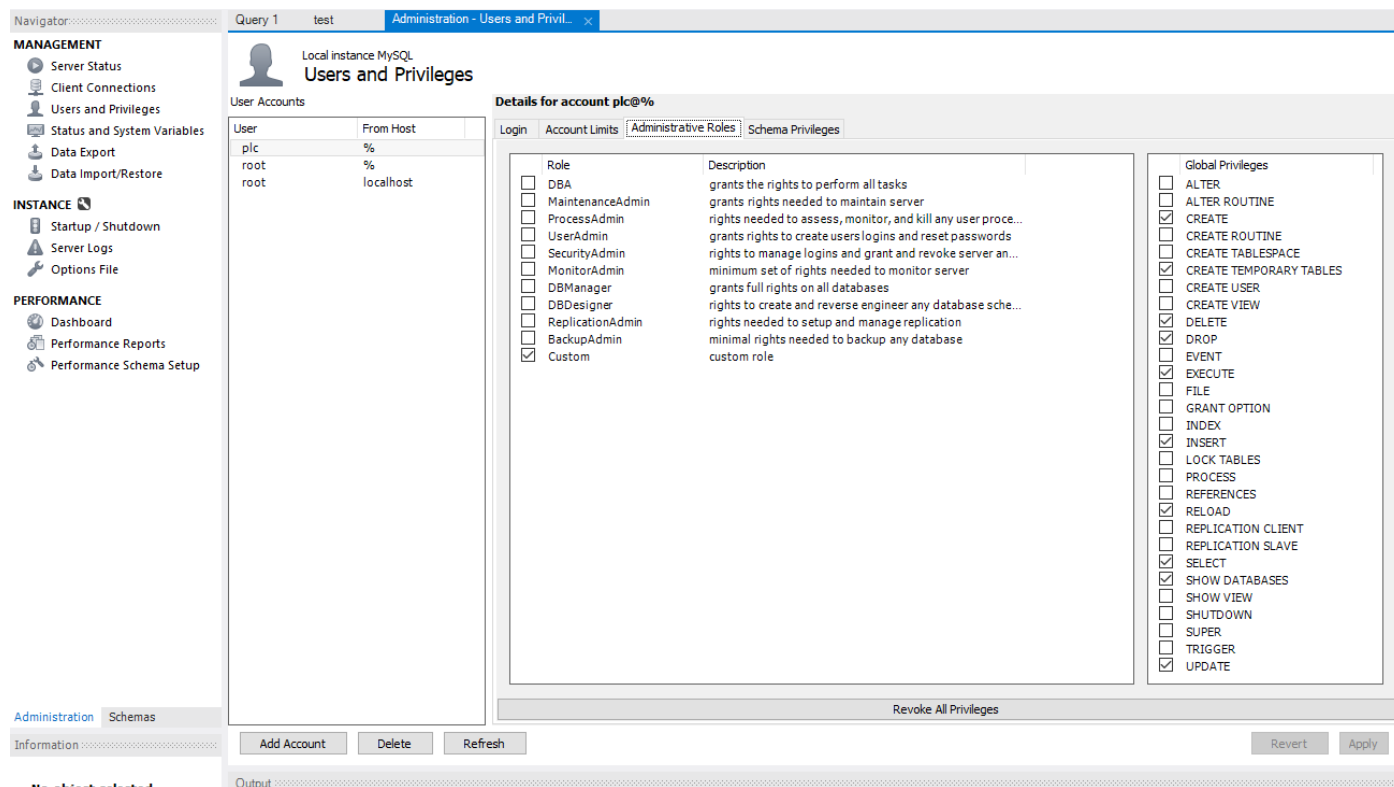


3.3 几点特别注意的地方



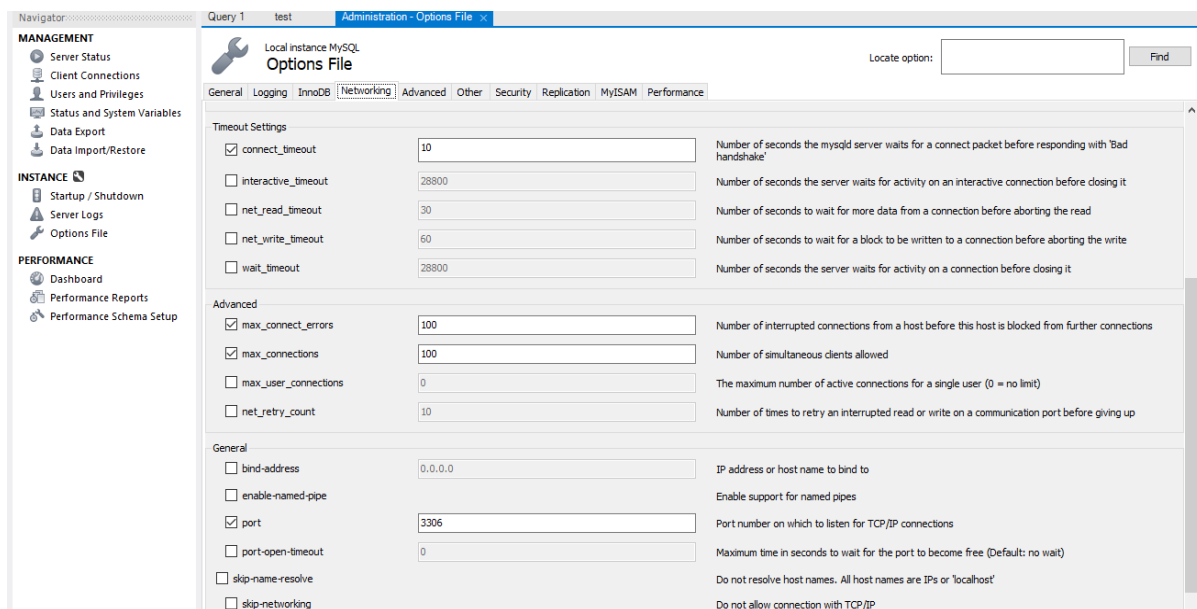
数据库安装完成后一般能通过本地访问连接，但是如果要在远程访问的话默认是连接不上的，这时候需要在用户管理中，**新增加一个用户，并给与相应的权限**，并将 Limit to local host 的值改为%，意思是 unlimited 对方的 IP 地址，并指定密码。

上图中就是新建了一个用户名为 plc 的账户。权限设置如下：



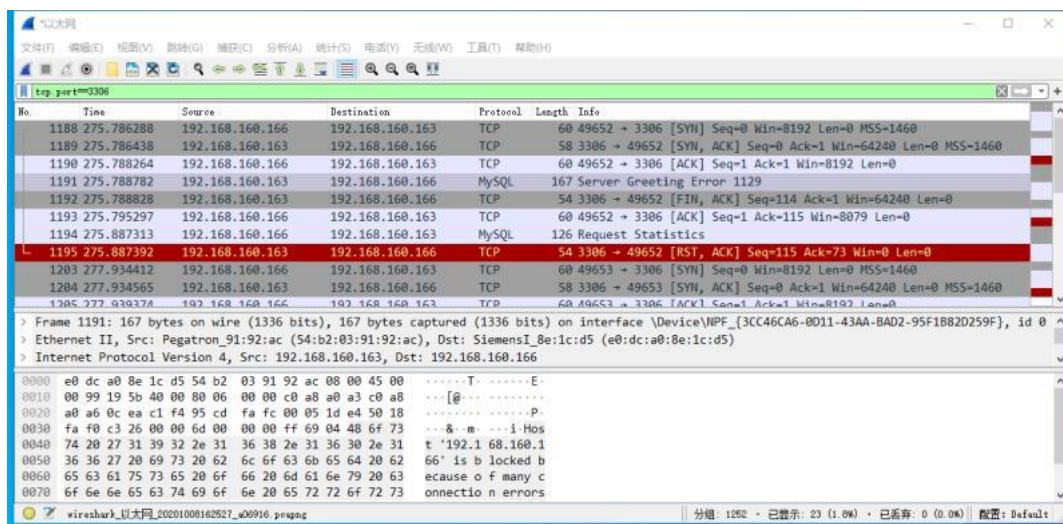
应用后重启 MySQL 服务。

另外：



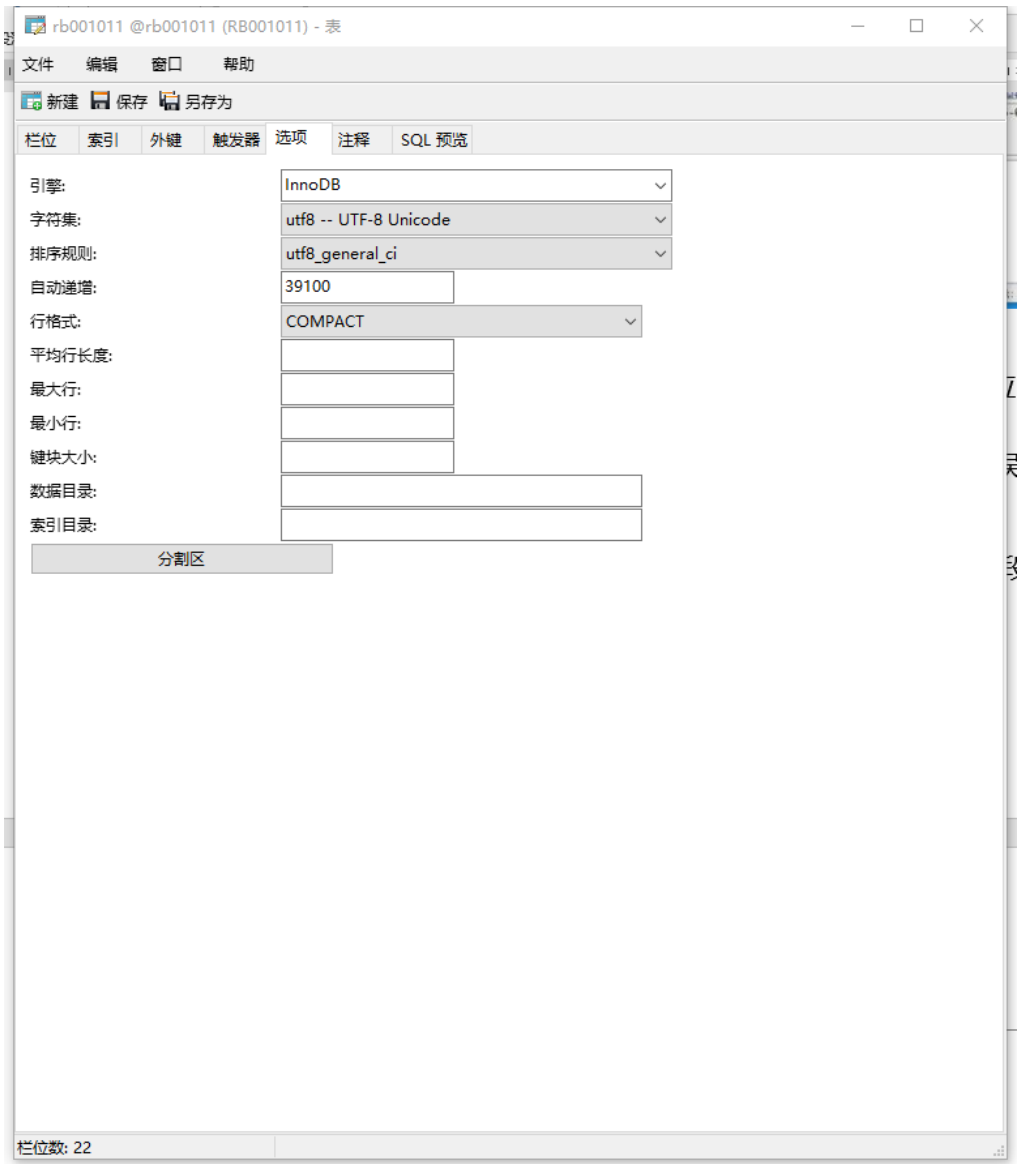
考虑 PLC 的连接可能没有电脑快速, 一般把 Connect timeout 设置为 10; max connect errors 设置为 100, max connections 设置为 100.

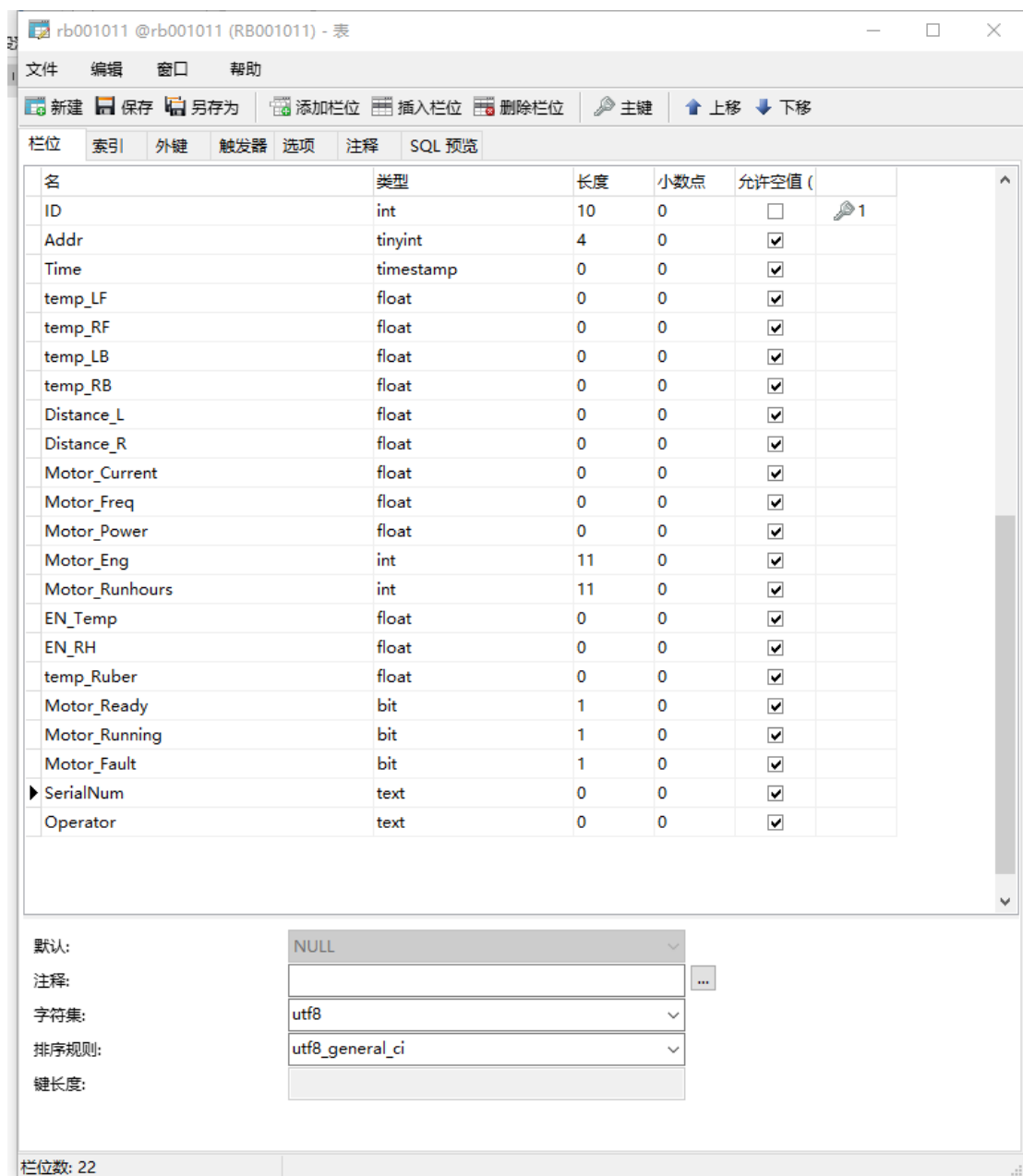
如果连接错误过多, 可以通过 wireshark 截取到错误代码 1129



数据库要提前建好需要写入的表, 这个表一定要和 PLC 执行的 SQL 语句对应, 可以先将 PLC 生成的 SQL 语句复制出来, 用 naticat 进行测试, 当执行无误时, 就可以判断语句拼接无误了。

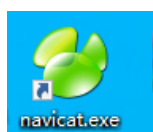
如果执行语句中有中文字符, 一定要把数据库中的字符编码改为 UTF-8, 字段设计中也要改为 UTF-8





3.4 MySQL 数据库软件

本人非专业人士，对数据库软件不是很熟悉，所以使用了两个软件，一个是 navcat,一个是 MySQL Workbench CE。



MySQL Workbench 8.0 CE
桌面应用

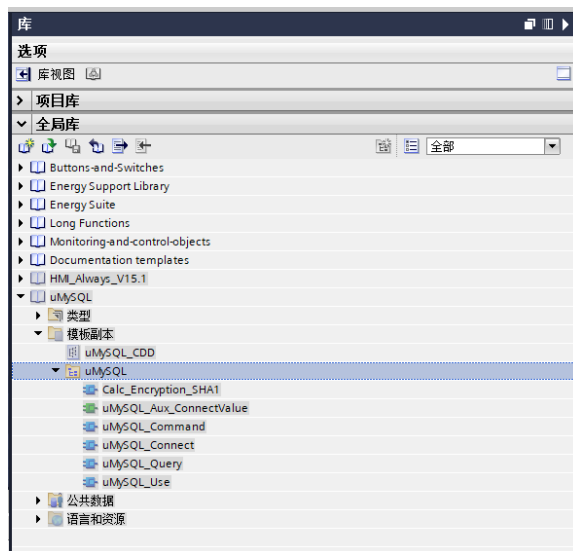
使用 Navicat 可以方便的连接数据库，设计表，查询数据。

使用 MySQL Workbench CE 可以方便的通过图形界面设置数据库的参数，不要敲那么多命令。

4、使用过程

4.1 博图部分

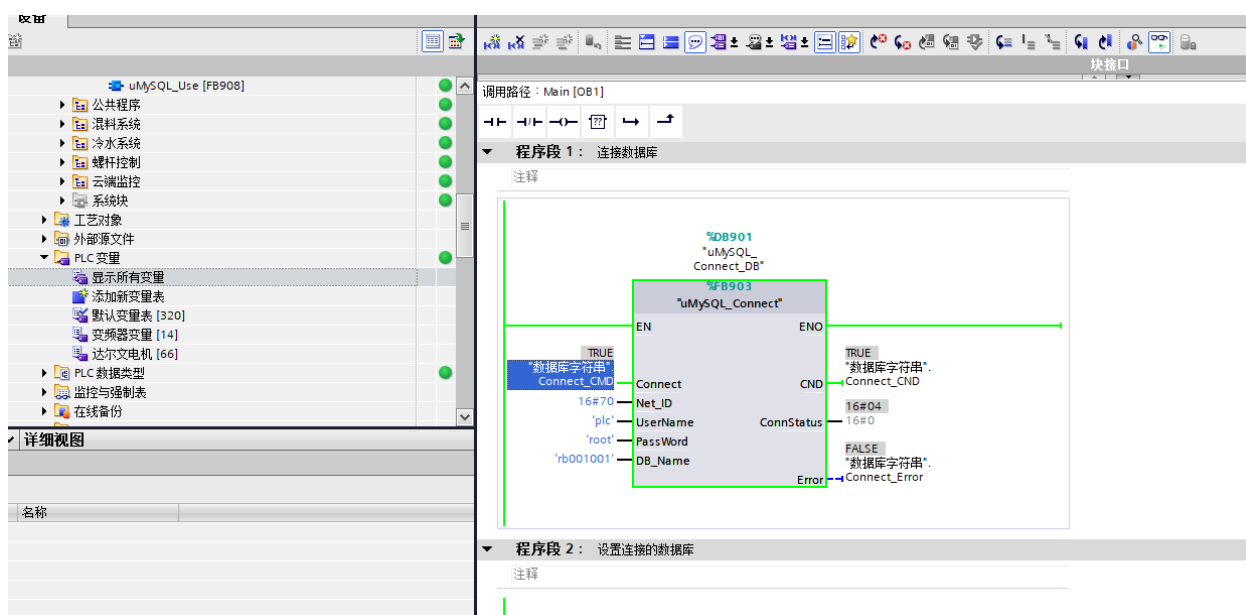
导入库文件



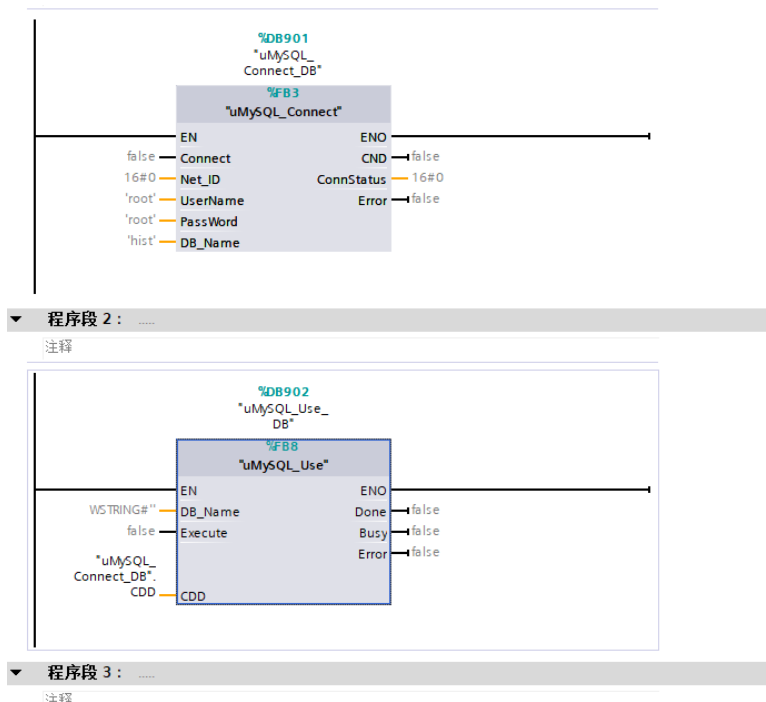
库文件拖入项目



uMySQL_Connect_DB										
名称	数据类型	起始值	保持	可从 HMI...	从 H...	在 HMI...	设定值	监控	注释	
1 Input										
2 Connect	Bool	false							发起连接	
3 Net_ID	CONN_OUC	16#0							连接ID	
4 UserName	String	'root'							数据库用户名	
5 PassWord	String	'root'							数据库密码	
6 DB_Name	String	'hist'							数据库名称	
7 Output										
8 CND	Bool	false							连接成功	
9 ConnStatus	Byte	16#0							连接状态	
10 Error	Bool	false							错误	
11 InOut										
12 Static										
13 IP	TCON_IP_V4									
14 Interfaceld	HW_ANY	0							HW-identifier of IE-interface submodule	
15 ID	CONN_OUC	16#0							connection reference / identifier	
16 ConnectionType	Byte	16#08							type of connection: 11=TCP/IP, 19=UDP (17=TCP)	
17 ActiveEstablished	Bool	true							active/passive connection establishment	
18 RemoteAddress	IP_V4								remote IP address (IPv4)	
19 ADDR	Array[1..4] of Byte								IPv4 address	
20 ADDR[1]	Byte	16#C0							IPv4 address	
21 ADDR[2]	Byte	16#A8							IPv4 address	
22 ADDR[3]	Byte	16#1F							IPv4 address	
23 ADDR[4]	Byte	16#A8							IPv4 address	
24 RemotePort	UInt	3306							remote UDP/TCP port number	
25 LocalPort	UInt	0							local UDP/TCP port number	
26 CDD	"uMySQL_CDD"									
27 TempBool	Array[0..19] of Bool									
28 REV_Buffer	Array[0..2048] of B...									
29 SND_Buffer	Array[0..2048] of B...									
30 NetConnect	TCON									
31 NetRcv	TRCV									
32 NetRcv_EN	Bool	false								
33 NetRcv_Len	Int	0								
34 NetRcv_Done	Bool	false								
35 Step	Int	0								
36 MessageLen	Int	0								



调用 uMySQL_Use



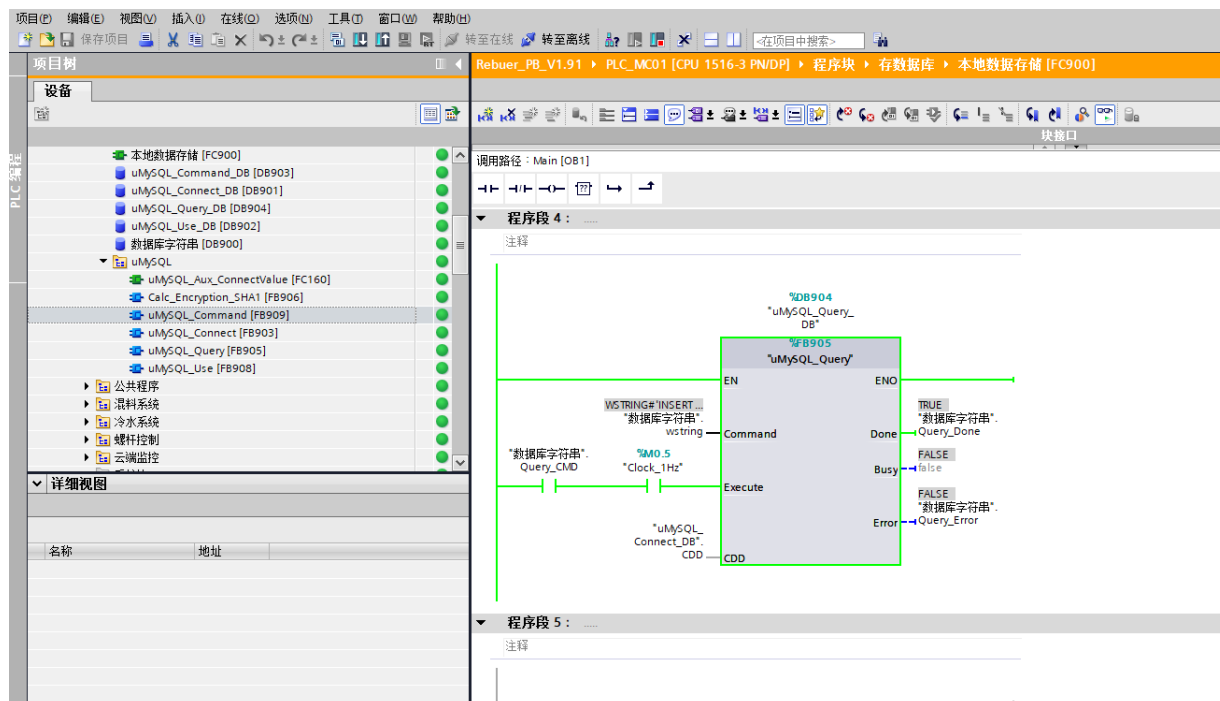
调用 uMySQL_Command



调整 uMySQL_Command 的 SQL 语句

```
1 "测试数据块"."wstring" := WSTRING#'INSERT INTO plcdata(T1,T2,T3,T4,T5,T6,T7,T8,T9,T10)VALUES(';
2 "uMySQL_Aux_ConnectValue"(Value:="测试数据块".T_Data[0],
3     PREC:=3,
4     Tail:=WSTRING#',',
5     CommandLine:="测试数据块"."wstring");
6
7 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[1],
8     PREC := 3,
9     Tail := WSTRING#',',
10    CommandLine := "测试数据块"."wstring");
11 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[2],
12    PREC := 3,
13    Tail := WSTRING#',',
14    CommandLine := "测试数据块"."wstring");
15 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[3],
16    PREC := 3,
17    Tail := WSTRING#',',
18    CommandLine := "测试数据块"."wstring");
19 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Data[4],
20    PREC := 3,
21    Tail := WSTRING#',',
22    CommandLine := "测试数据块"."wstring");
23 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_int[0],
24    PREC := 3,
25    Tail := WSTRING#',',
26    CommandLine := "测试数据块"."wstring");
27 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_int[1],
28    PREC := 3,
29    Tail := WSTRING#',',
30    CommandLine := "测试数据块"."wstring");
31 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_int[2],
32    PREC := 3,
33    Tail := WSTRING#',',
34    CommandLine := "测试数据块"."wstring");
35 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Bool[0],
36    PREC := 3,
37    Tail := WSTRING#',',
38    CommandLine := "测试数据块"."wstring");
39 "uMySQL_Aux_ConnectValue"(Value := "测试数据块".T_Bool[1],
40    PREC := 3,
41    Tail := WSTRING#',',
42    CommandLine := "测试数据块"."wstring");
```

调整 uMySQL_Query



4.2 数据库效果

主数据监控 @rb001001 (172) - 查询

文件(F) 编辑(E) 格式(O) 查看(V) 窗口(W) 帮助(H)

运行 停止 解释 导出向导 新建 载入 保存 另存为 查找 自动换行 网格查看 表单查看 备注 十六进制 图像

查询创建工具 查询编辑器

```

1 SELECT
2   rb_machine_realtime.ID,
3   rb_machine_realtime.Time,
4   rb_machine_realtime.BM00_M001_Current,
5   rb_machine_realtime.BM00_M001_Frequency,
6   rb_machine_realtime.BM20_M001_Current,
7   rb_machine_realtime.BM20_M001_Frequency,
8   rb_machine_realtime.BM00_M000_Frequency
9 FROM
10  rb_machine_realtime
11

```

信息 结果1 概况 状态

ID	Time	BM00_M001_Current	BM00_M001_Frequency	BM20_M001_Current	BM20_M001_Frequency	BM00_M000_Frequency
1027	2020-04-07 20:50:47	584	54.81	240	55.12	25
1028	2020-04-07 20:50:48	586	54.81	247	55.12	25
1029	2020-04-07 20:50:49	582	54.81	242	55.12	25
1030	2020-04-07 20:50:50	592	54.81	254	55.12	25
1031	2020-04-07 20:50:51	588	54.81	243	55.12	25
1032	2020-04-07 20:50:52	600	54.81	248	55.12	25
1033	2020-04-07 20:50:53	600	54.81	243	55.12	25
1034	2020-04-07 20:50:54	594	54.81	248	55.12	25
1035	2020-04-07 20:50:55	588	54.81	244	55.12	25
1036	2020-04-07 20:50:56	592	54.81	250	55.12	25
1037	2020-04-07 20:50:57	602	54.81	238	55.12	25
1038	2020-04-07 20:50:58	594	54.81	243	55.12	25
1039	2020-04-07 20:50:59	596	54.81	240	55.12	25
1040	2020-04-07 20:51:00	598	54.81	241	55.12	25
1041	2020-04-07 20:51:01	592	54.81	244	55.12	25
1042	2020-04-07 20:51:02	584	54.81	240	55.12	25
1043	2020-04-07 20:51:03	574	54.81	242	55.12	25
1044	2020-04-07 20:51:04	572	54.81	237	55.12	25
1045	2020-04-07 20:51:05	586	54.81	246	55.12	25
1046	2020-04-07 20:51:06	584	54.81	241	55.12	25
1047	2020-04-07 20:51:07	596	54.81	247	55.12	25
1048	2020-04-07 20:51:08	586	54.81	242	55.12	25
1049	2020-04-07 20:51:09	580	54.81	255	55.12	25
1050	2020-04-07 20:51:10	588	54.81	248	55.12	25
1051	2020-04-07 20:51:11	580	54.81	248	55.12	25
1052	2020-04-07 20:51:12	582	54.81	243	55.12	25
1053	2020-04-07 20:51:13	584	54.81	246	55.12	25
1054	2020-04-07 20:51:14	596	54.81	244	55.12	25

◀ ▶ 🔍 ⌂

库已经在实际项目中进行测试，有任何疑问欢迎大家讨论，斧正。

谢谢！

名称	数据类型	默认值	注释
1 Input			
2 Value	Variant		
3 PREC	USInt		
4 Tail	WString		
5 Output			
6 <新增>			
7 InOut			
8 CommandLine	WString[16382]		
9 Temp			
10 Value_Real	Real		
11 Value_Int	Int		
12 Value_Bool	Bool		
13 Temp_Str	WString		
14 Temp_Command	WString		
15 Value_M	Int		
16 Value_FP	USInt		

IF... CASE... OF... FOR... TO DO... WHILE... DO... (*...*) REGION

```
1 #Value_Bool:=FALSE;  
2 #Value_Real := 0.0;  
3 #Temp_Command:=WSTRING#'';  
4 IF TypeOf(#Value_Bool) = TypeOf(#Value) THEN  
5     VariantGet(SRC := #Value,  
6         DST => #Value_Bool);  
7     IF #Value_Bool THEN  
8         #Temp_Command := WSTRING# '1';  
9     ELSE  
10        #Temp_Command := WSTRING# '0';  
11 FND IF;
```

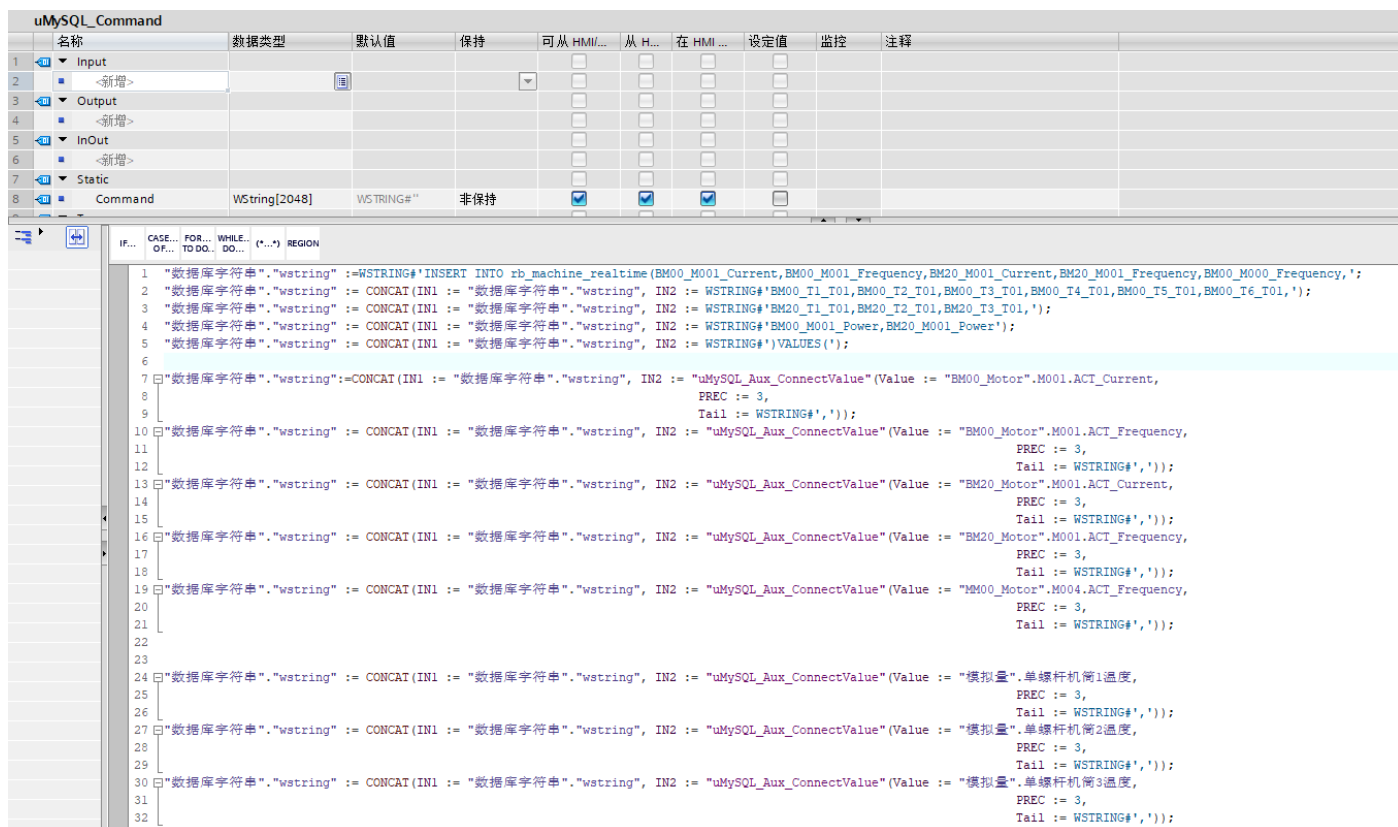
5、使用问题及网友反馈

5.1 问题反馈

- 1、原字符串拼接过程，如果字符串总长度超过 255 个字符，会超出 WString 默认长度，所以如果 MySQL 语句长于 255 字符，会写入失败，解决方法是降字符串拼接改在外部。
- 2、如果发现测试连接数据库时，无法连通，确认在 MYSQL 安装过程中，密码校验界面选择第二个选项，native password（网友 SuperTai）。
- 3、在执行命令前，需要激活一下 uMySQL_Use 命令。（网友 SuperTai）
- 4、如果使用系统提供的格式转换函数，需要特别注意数据库中字段的长度设置。

5.2 关于字符串拼接

- 1、新的字符串拼接可以解决 MySQL 命令长度超过 255 会出错的问题。



名称	数据类型	默认值	保持	可从 HMI...	从 H...	在 HMI...	设定值	监控	注释
Input									
Output									
InOut									
Static									
Command	WString[2048]	WSTRING#	非保持	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

```
1 "数据库字符串".wstring := WSTRING# INSERT INTO rb_machine_realtime(BM00_M001_Current, BM00_M001_Frequency, BM20_M001_Current, BM20_M001_Frequency, BM00_M000_Frequency,
2 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := WSTRING# BM00_T1_T01, BM00_T2_T01, BM00_T3_T01, BM00_T4_T01, BM00_T5_T01, BM00_T6_T01,
3 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := WSTRING# BM20_T1_T01, BM20_T2_T01, BM20_T3_T01,
4 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := WSTRING# BM00_M001_Power, BM20_M001_Power);
5 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := WSTRING#) VALUES(');
6
7 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "BM00_Motor".M001.ACT_Current,
8 PREC := 3,
9 Tail := WSTRING#, '));
10 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "BM00_Motor".M001.ACT_Frequency,
11 PREC := 3,
12 Tail := WSTRING#, '));
13 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "BM20_Motor".M001.ACT_Current,
14 PREC := 3,
15 Tail := WSTRING#, '));
16 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "BM20_Motor".M001.ACT_Frequency,
17 PREC := 3,
18 Tail := WSTRING#, '));
19 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "BM00_Motor".M004.ACT_Frequency,
20 PREC := 3,
21 Tail := WSTRING#, '));
22
23
24 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "模拟量".单螺杆机筒1温度,
25 PREC := 3,
26 Tail := WSTRING#, '));
27 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "模拟量".单螺杆机筒2温度,
28 PREC := 3,
29 Tail := WSTRING#, '));
30 "数据库字符串".wstring := CONCAT(IN1 := "数据库字符串".wstring, IN2 := "uMySQL_Aux_ConnectValue"(Value := "模拟量".单螺杆机筒3温度,
31 PREC := 3,
32 Tail := WSTRING#, '));
```

2、由于 PLC 内中文是采用两字节的 Wchar 表示的，但是发送给数据库的要采用 UTF-8 表示，所以在发送过程中要将 WCHAR 转换为 UTF-8，因此增加了 WCHAR_To_UTF8 的函数。

华文博

镇江明润信息科技有限公司

MingRun Info. Tech. (MRIT)

Add: 江苏省镇江市丹徒新城工业园区盛园路 24 号

P.C.: 212028

Item: uMySQL 库文件说明书
Date: 2020 年 4 月 7 日
Revision: V2.1

E-Mail: Mr@MrShip.com

Tel:+86-511-85968799, Fax:: +86-511-85968768

Mobile: +86-13952850491