APQ-2.1 Win32 Build Instructions

Warren W. Gay

September 24, 2003

Chapter 1

Getting Started

This document provides a guide to those people who prefer to compile products they install, from the source code. This guide specifically deals with compiling and installing APQ in a win32 environment.

What do you get from APQ in the win32 environment?

The following are the products of this compile and install procedure:

- apq_myadapter.dll for MySQL support
- apq-mysql.ads generated spec for MySQL support
- libapq.a static library for APQ client programs
- APQ static spec and body sources
- APQ-2.1.EXE Installer program

You may also choose which databases you wish to support. For example, you may only wish to include MySQL support for the win32 environment. Or instead, you may choose to support all products[1], since PostgreSQL databases may reside on hosts other than the win32 environment that you are using. The choice is yours.

1.1 Does APQ Need CYGWIN?

No. When APQ is built and installed using these instructions, is completely independent of any CYGWIN tools and libraries. You may run your APQ application in or out of a CYGWIN environment. It only requires the win32 environment, and those libraries that come installed with GNAT.

[1] At the present time, this is PostgreSQL and MySQL.

APQ does however, require the use of CYGWIN environment tools, to perform the build process. The compiler used is only GNAT and GNAT's underlying gcc.[2] However, the win32 environment is not rich in the way of scripted build processes, so the CYGWIN POSIX like environment is used instead.

1.2 Tools Needed

In order to build the APQ library from sources, you will need a number of tools installed:

- CYGWIN (see below for list)
- Microsoft Visual Studio (Version 6.0)[3]
- makensis (Nullsoft Scriptable Install System)[4]
- GNAT Compiler (3.14p was tested)

The makensis is actual somewhat optional. The actual build does not require this tool. If you are prepared to move the various components into the correct places (or custom locations), you do not need it. However, the built APQ-2.1.EXE install program is highly recommended, because it provides the following benefits:

- makes installation easy and foolproof
- allows you to choose what components to install
- registers its version in the registry
- is "GNAT aware"
- provides an uninstall capability[5]

1.3 CYGWIN Tools

In order to provide a POSIX like environment, the CYGWIN tools are necessary to facilitate the complex process of building the APQ library from sources. If binary versions of the library are available, you may want to use them instead.

The following list are the CYGWIN tools needed:

- bash (shell)
- sed

[2] Microsoft's tool set is also required to compile some components.
[3] It is possible that an earlier version such as 5.0 might be sufficient, but this is untested.
[4] Dowload it from nsis.sourceforge.net/site/index.php
[5] From the Control Panel, select Add/Remove Programs.

- sort
- ar
- tar
- cp
- rm
- chmod
- mkdir
- make
- cygpath
- uname

Chapter 2

PostgreSQL Preparation

This section shows you what you need to do to prepare the PostgreSQL components of APQ. If you do not plan to provide win32 support for PostgreSQL, then you can skip to the next chapter. Note that this chapter is only about preparing a PostgreSQL client facility. Installing and using a PostgreSQL database under Windows is well beyond the scope of this document.

2.1 Shell Sessions

This chapter will assume that you are using the cmd.exe native shell sessions (ie. not CYGWIN).

2.2 Source Code

This document will assume that your 3rd party source code will be downloaded and unpacked in the directory:

- c:\work

If you don't have this directory, then create one now.

2.2.1 Download Sources

Download the PostgreSQL sources that you plan to use. In this example the download file:

```
postgresql-base-snapshot.tar.gz
```

was used. You may want to choose a stable production quality release instead. Once you have downloaded your source file, unpack it to your work directory. Here we will use the file shown above.

2.2.2 Unpack Sources

Unpack your PostgreSQL sources to the work subdirectory. Using the file downloaded above, the sources unpacked into the directory named:

- c:\work\postgresql-snapshot

If you downloaded a production level release, then "snapshot" was likely replaced by a version number. Change to the src subdirectory, and then list the files there. You should find a file named win32.mak (using cmd.exe shell):

```
C:\work> cd postgresql-snapshot\src\interfaces\libpq
C:\work\postgresql-snapshot\src\interfaces\libpq> dir /w
```

2.3 Compile PostgreSQL Components

In that directory, you should see a file named win32.mak (the prompt here will be abbreviated):

```
C:> nmake /f win32.mak
```

Note: you should always check with the documentation that comes with your source code. These directions should work on all recent releases up to and including 7.4beta2. Build and install instructions often change, so check for them, with each new release.

2.3.1 Check for libpq.dll

This should now build the PostgreSQL client library libpq.dll. This process might fail at some point.[1] This can be ignored, if the important part (libpq.dll) was built. Change to the dll directory to check:

```
C:> cd interfaces\libpq\Release
C:> dir libpq.dll
 Volume in drive C has no label.
 Volume Serial Number is 587F-CB45
 Directory of C:\work\postgresql-snapshot\src\interfaces\libpq\Release
09/19/2003  10:42p              90,112 libpq.dll
               1 File(s)         90,112 bytes
               0 Dir(s)  32,435,912,704 bytes free
C:>
```

The example session above, shows that the libpq.dll file was successfully created.

[1] Release 7.4beta2 failed building a component after libpq.dll.

2.4 Create Staging Area

Now create a directory to place the important PostgreSQL files into. Create the following directories:

c:\postgresql top level staging directory

c:\postgresql\include directory of C header files for compiling

c:\postgresql\lib optional directory for DLL

These directories create a place for the PostgreSQL DLL file and the C header files.[2]

2.4.1 Install the DLL File

Copy the PostgreSQL DLL file into the staging area. You want to copy the file:

- c:\work\postgresql-snapshot\src\interfaces\libpq\release\libpq.dll

to the location:

- c:\postgresql\lib\libpq.dll

If this directory is not on your windows PATH, you will need to either put it on your path, or place it somewhere else that is on the path. One suggestion is GNAT's bin directory. On the author's system, this would be:

- c:\opt\gnat\bin\libpq.dll

2.4.2 Install C Header Files

Now it is necessary to copy over the C header files that APQ will need to examine. Copy all the files from:

- c:\work\postgresql-snapshot\src\interfaces\libpq*.h

to the directory:

- c:\work\postgresql\include

There are a few other C header files that APQ will need. Copy the header files from:

[2] It is possible to install everything from its original location, provided that you answer the configuration prompts correctly. But this install procedure has been designed to keep things simple.

- c:\work\postgresql-snapshot\src\include*.h

additionally to the directory:

- c:\work\postgresql\include

Now the PostgreSQL preparation is ready for APQ.

Chapter 3

MySQL Preparation

Like the PostgreSQL chapter, this chapter is optional. If you do not plan to include MySQL client support, then skip to the next chapter. Just be sure that you include support for at least one of PostgreSQL or MySQL!

3.1 Shell Sessions

Where shell sessions are required, this chapter is assuming that you will be using cmd.exe sessions (ie. not CYGWIN).

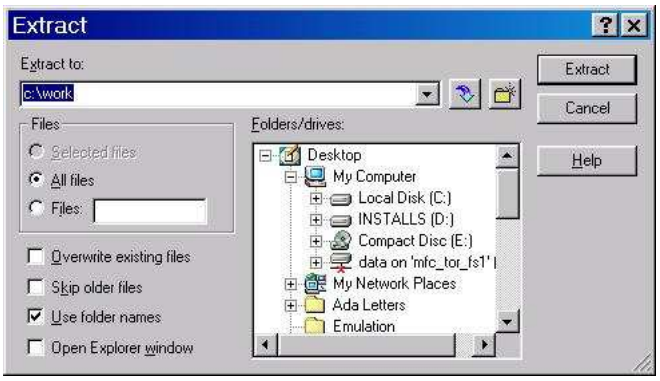
3.2 Create Work Directory

If you don't already have a C:\WORK directory, then create one now.

Note: numerous shortcuts are possible in this procedure if you know what you are doing. We will be taking a cautious approach to the procedure, so that the steps will be easy to understand, and to avoid confusion.

3.3 Unpack MySQL Sources

Unpack (or unzip) the tar/zip file into the C:\WORK directory. Here we used WinZip to unpack the sources into a subdirectory.



Note carefully that the "Use folder names" checkbox is checked.

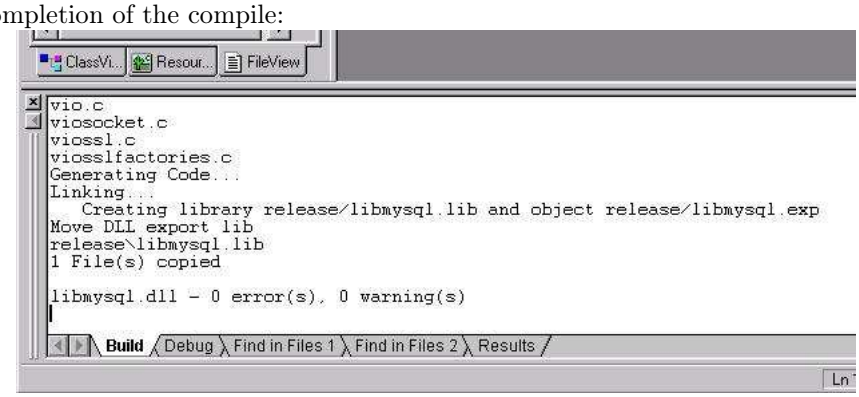
3.4 Build MySQL DLL

You will need to start the Microsoft Visual Studio 6.0 at this point.[1] Select File->Open Workspace, and then browse in the unpacked subdirectory for a file named mysql.dsw :

Next, from the Build->Set Active Configuration Menu, choose libmsql - Win32 Release (or Win32 Debug if you are debugging changes).

[1] Check with the MySQL documentation for the necessary build instructions. This procedure was effective for MySQL version 4.0.14b. If their instructions vary from these, follow theirs.

Then compile by pressing F7. If all went well, you should see a successful completion of the compile:



3.5 Create MySQL Staging Area

You may find that you already have a MySQL lib\debug and lib\opt library directory. The former has components for debugging while the later contains optimized libraries.

This APQ build procedure will assume that you have the following directories in place for the client MySQL components:

c:\mysql top level MySQL directory

c:\mysql\include containing all client MySQL C header files

c:\mysql\bin containing the libmySQL.dll DLL file

c:\mysql\lib containing the libmySQL.lib import file

Your directory structure can vary from this theme, but we'll refer to the directories in this document as noted above.

You may have installed MySQL from binary distributions. If it included the install of a libmySQL.dll client library (as it should), you might already have include files, that were installed with the dll. Note however, the author has found that you need more than the include files that were provided, for the compile of the APQ component. So it is recommended that you follow this procedure, including the staging area.

3.5.1 Staging Directories

Create the c:\mysql staging area directories and subdirectories as shown in the prior section, if they don't already exist. If you are using other directories, you may want to write your notes on this document page.

3.5.2 Copy Include Files

Copy the include files from:[2]

[2]The directory name obviously reflects MySQL version 4.0.14b, which may be different in your case.

- c:\work\mysql-4.0.14b\include*.h

to the directory:

- c:\mysql\include

3.5.3 Copy the libmySQL.dll File

If you built the libmySQL.dll file from sources, you will want to install that. Copy the file:

- c:\work\mysql-4.0.14b\lib_release\libmysql.dll

to your directory:

- c:\mysql\bin\libmysql.dll

You will need to make certain that this DLL is on the PATH, in order to be executable. You may prefer to put the DLL with the GNAT binary directory such as:

- c:\opt\gnat\bin\libmysql.dll

instead. The choice is up to you.

3.5.4 Copy the Import Library

You will need the import library to assist with the linking of the APQ DLL file. Copy the file:

- c:\work\mysql-4.0.14b\lib_release\libmysql.lib

to your directory:

- c:\mysql\lib\libmysql.lib

Now your staging area for MySQL is ready for APQ.

Chapter 4

Compiling APQ

Now we are ready to start into building the APQ components. The first step is one of configuration.

APQ has been designed to allow PostgreSQL and/or MySQL support to be included. The preceeding chapters documented preparations for both, but either of those database products can been omitted.[1] We will assume in this document, that you want all the support you can build, and so will build both. However, you can easily bypass one or the other, with the appropriate answers to the scripted prompts.

4.1 Shell Sessions

The APQ work will primarily use a CYGWIN shell session using the bash shell. Open a CYGWIN bash session now, and make a work subdirectory there, and then unpack APQ and finally change to that directory. In this section you may need to switch between this CYGWIN bash session, and a cmd.exe session window.

If you downloaded APQ-2.1, then your unpacked sources should probably be unpacked to a directory named:

- c:\work\apq-2.1 (ie. /cygdrive/c/work/apq-2.1 for CYGWIN)

4.2 Configuration

Before you can build the APQ sources, you must configure them. Run the shell script ./configure in the CYGWIN bash session:

```
$ ./configure
THIS IS A WIN32 INSTALL OF APQ-2.1 :
```

[1] As long as at least one is chosen.

```
Building gnat_info command, to obtain some registry
information about your installed GNAT compiler..

You have GNAT 3.14p installed :
```

Initially the script should identify this as a WIN32 configure step[2]. Then the program gnat_info.adb is compiled to allow the configuration script to access the registry. It locates where your GNAT compiler is installed, and later tests to see if APQ has already been installed. If APQ is found to be installed, you will need to uninstall it to compile APQ successfully.[3]

4.2.1 Configure Edit Mode

The script will ask for a number of pathnames for input. This is more easily done, if editing is permitted at the input prompts. But to do this, the script must know which edit mode the user prefers:

```
  Please choose your preference for input editing
  bindings:
-
  e - emacs mode
  v - vi mode
  n - neither
-
Choose e/v/n ?
```

So the script prompts you for the editing mode you want to use. Choose:

e for emacs editor bindings

v for vi editor bindings

n for no special editing

If you are not in the habit of using emacs or vi UNIX/Linux editors, then you might prefer to use "n". This document will assume that you have chosen "e" for emacs mode, because this permits some special features to be highlighted, for the convenience of power users.

4.2.2 Microsoft Toolset Check

After answering the edit mode prompt, you'll get an assessment made of the Microsoft Toolset. If you have Microsoft Visual Studio 6.0 installed, you should see:

[2]The script eventually execs the script win32_config.

[3]If you installed APQ manually, this test will likely fail to detect that APQ is already installed.

```
Choose e/v/n ? e
-
You seem to have the Microsoft Toolset available..
```

Here the script is confirming that you have the necessary tools to build the APQ_MYADAPTER.DLL file. Certainly if you built the MySQL or PostgreSQL DLL files, you should receive a favourable report here. However, it is possible that those components be downloaded in binary form instead and then placed into the work directories indicated.

If you have the Visual Studio installed, and you don't receive confirmation of this from the script, then check your PATH variable within the CYGWIN environment. The PATH setting probably needs adjustment.

The Microsoft Toolset required by APQ includes:

CL the C/C++ Compiler

LIB the library utility for creating export and import files

LINK the Microsoft C/C++ linker utility

Before you despair of not owning the Microsoft toolset, consider:

- If you are not building support for MySQL, you do not need them.
- If you have downloaded a binary form of APQ_MYADAPTER.DLL you do not need them.

If on the other hand, you are planning to compile APQ, with MySQL support, and lack the Microsoft toolset above, then you must now locate and download a binary copy of apq_myadapter.dll. Otherwise, you will be required to use the Microsoft toolset to create that dll file. This file may be available in your package, or perhaps made available separately, to reduce download file sizes.

4.2.3 Choosing MySQL Support

Now you must decide whether or not you want your installed copy of APQ to include MySQL support. The only real reason not to include it is to save the installer time and effort. Otherwise, I suggest you include it for flexibility.

```
Questions for MySQL :
-
Installing MySQL database support ? (Yes/No)? Yes
```

I'm going to assume that you are answering yes, to get your money's worth from APQ.

4.2.4 Specifying MySQL Pathnames

Once you answer *Yes* to the MySQL support prompt, you are prompted for two more pathnames:

```
Questions for MySQL :
-
Installing MySQL database support ? (Yes/No)? yes
Windows pathname to MySQL include directory : c:\mysql\include
Windows path of MySQL DLL file is           : c:\MySQL\BIN\libmysql.dll
Windows path to libmysql.lib file           : c:\mysql\lib\libmysql.lib
```

While the session appears to show three prompts, the pathname of the MySQL DLL is determined by the script. It does this by looking for libmySQL.dll on your PATH. If your PATH is not correctly set, or the DLL not correctly installed, then this step will fail. Correct the PATH and try again, if necessary.

The two important inputs are:

1. Directory pathname to MySQL include files
2. Pathname of the MySQL DLL import file (libmysql.lib)

Your pathnames may vary from the example, but these are the ones I am using for illustration purposes.

4.2.5 Choosing PostgreSQL Support

Like MySQL, you can choose whether or not you want to build support for it:

```
Questions for PostgreSQL :
-
Installing PostgreSQL database support ? (Yes/No)? yes
```

Here, once again, we are assuming you want to get your money's worth.

4.2.6 Specifying PostgreSQL Pathnames

Currently, PostgreSQL only needs to know the location of the C header (include) files:

```
Questions for PostgreSQL :
-
Installing PostgreSQL database support ? (Yes/No)? yes
Windows pathname to PostgreSQL include directory : c:\postgresql\include
Windows path of PostgreSQL DLL file is           : c:\OPT\GNAT\bin\libpq.dll
```

The script locates the PostgreSQL DLL file libpq.dll, by searching the PATH variable. If the script fails to locate the DLL file, then you may need to change the PATH (under CYGWIN) and/or make the DLL file in a directory that is searched by the PATH variable setting. Then try the configure step again.

In this example, I will just hit return and accept the default. But you may specify a different directory from this.

4.2.7 Concluding the Configuration

The configuration session should look something like the following:

```
You seem to have the Microsoft Toolset available..
-
Questions for MySQL :
-
Installing MySQL database support ? (Yes/No)? yes
Windows pathname to MySQL include directory : c:\mysql\include
Windows path of MySQL DLL file is           : c:\MySQL\BIN\libmysql.dll
Windows path to libmysql.lib file           : c:\mysql\lib\opt\libmysql.lib
-
Questions for PostgreSQL :
-
Installing PostgreSQL database support ? (Yes/No)? yes
Windows pathname to PostgreSQL include directory : c:\postgresql\include
Windows path of PostgreSQL DLL file is           : c:\OPT\GNAT\bin\libpq.dll
-
    You are now ready to 'make'

To proceed with the build, perform the following:

$ make(do 'make clean' prior to rebuilds)
$ make installer(optional)
$ make install
```

You may of course, choose different directories and preferences. This document is just a guideline.

At this point, you should receive a message indicating that you are ready to perform a "make". If you are making changes and need to rebuild the sources, perform a "make clean" first. If you are making changes to the installer programm, then you can repeat builds of it by simply doing a "make installer".

A "make install" will make the installer program, and then execute it.

4.3 Building APQ

If you have all of the CYGWIN components required, then this should be the easiest part of the WIN32 build process.

4.3.1 Win32 Makefile

The APQ components are compiled using a Makefile that is tailored to the Windows environment. The actual make file used is named:

- Makefile.win32

When the configure script ran[4], it also created a symlink to the Makefile.win32 file. The symlink is named:

[4]The configure script invokes script win32_config under Windows.

- GNUmakefile

This symlink permits the make command to reference Makefile.win32 by default. If you must customize the make file, be sure to edit Makefile.win32.

The configuration process also created another file:

- config.win32

The configured values for the build are included in the config.win32 file. This is the file that is included from Makefile.win32. If you need edit configuration values, edit that file. Just be aware that those changes will be lost if the configure script is successfully rerun.

To compile the APQ components, in a CYGWIN bash session, using the APQ source directory[5] as the current directory, you just type make:

```
$ make
```

This should successfully build everything that is required, depending upon the options you selected in the configuration process. The following products should come out of the build:

apq-mysql.ads This is generated for MySQL support only

apq_myadapter.dll This is for MySQL support only

libapq.a A static GNAT library for APQ

Additionally, all static APQ spec and body sources must also be installed.

4.4 Rebuilding/Configuring

Sometimes things don't go right on the first try. To allow for that possibility, the following tools are at your disposal:

1. make clean
2. make clobber

4.4.1 Make Clean

Use this when you want to delete all compiled objects, binary components and generated sources (like apq-mysql.ads). This allows you to tweak as necessary, but rebuild again without redoing the configuration process.

4.4.2 Make Clobber

This goes one step further than "make clean", in that it destroys the configuration information that was gathered. Use this procedure when you want to start over from scratch.

[5] Using our example this would be /cygdrive/c/work/apq-2.1.

4.5 Installing APQ

Once you have successfully built APQ, you need to install its components. If it were only a few binary files, you could do this by hand. However, in addition to the library components, the Ada specification and body source files must be installed in the correct place.[6]

If you perform "make installer", this will build the installer program for you. If you perform a "make install", it will build the installer program and invoke it.

```
$ make install
```

The installer used is a nice GUI installer, provided complements of the Nullsoft Scriptable Installer System folks. Visit their website at:

```
nsis.sourceforge.net/site/index.php
```

When the installer is run, simply follow the GUI prompts.

4.6 Uninstalling APQ

If you are making changes, you'll frequently need to uninstall and re-install. This is easily provided for in the Control Panel. Choose "Add/Remove Programs" and look for the entry named "APQ-2.1 (remove only)". Click on the button "Change/Remove" to proceed. The APQ_Uninstall.exe program will then be launched to guide you through the process.

4.7 Conclusion

If you reached this point, you have successfully installed database support, which is now at your Ada95 finger tips! Congratulations! The next chapter will show you how to test your APQ library.

[6] Some of the Ada bodies are mandatory because of the generics that must be compiled by GNAT. So APQ just installs them all.

Chapter 5

Testing APQ

Naturally, you want to test out your shiny new APQ library, now that you can connect to databases both on your win32 platform, and remotely over the network. The example programs in the ./eg and the ./eg2 are UNIX flavoured tests and are not suitable.

Starting with APQ-2.1, there is now a template program named win32_test.adb. This chapter will describe how you can use that source file to perform some simple tests.

5.1 Creating the Test Program

First you need to create a file named win_test.adb from the win32_test.adb source file provided. Do not modify the original win32_test.adb file. You may need it for testing another database, later.

Create the win_test.adb by copying the win32_test.adb file (bash):

```
$ cp win32_test.adb win_test.adb
```

But don't compile that file yet! You need to supply some changes.

> Note: if you are testing a binary installed release, look for the win32_test.adb program in the APQ bindings directory. If GNAT is installed as C:\OPT\GNAT, then look for the file named:
>
> C:\OPT\GNAT\Bindings\APQ\win32_test.adb
>
> Copy this file to your work area, and name it win_test.adb to match the procedure name within it.

5.1.1 Editing the win_test.adb Source

The test file win_test.adb is nearly complete. But it still needs you to define:

1. Which database product to use

2. What host name (if not local) to use

3. What account name (userid) to use

4. What password to use (if any)

Once these are defined, the program is ready to compile and test.

The unmodified start of the win_test.adb program should look like this:

```
 1  -- Very Simple Connect to Database Test
 2
 3  -- Uncomment one of the following:
 4  -- with APQ.PostgreSQL.Client; use APQ, APQ.PostgreSQL, ...
 5  -- with APQ.MySQL.Client; use APQ, APQ.MySQL, APQ.MySQL.Client;
 6
 7  with Ada.Text_IO; use Ada.Text_IO;
 8
 9  procedure Win_Test is
10     C : Connection_Type;
11     Q : Query_Type;
12  begin
13
14     Put_Line("Win32_Test Started:");
15
16  -- Set_Host_Name(C,"<hostname>");
17     Set_User_Password(C,"<userid>","<password>");
18     Set_DB_Name(C,"<database_name>");
```

The line numbers are not part of the source code however, since they are here only for our ease of reference. Now perform the following edits:

1. Uncomment line 4 if you are testing PostgreSQL. Else uncomment line 5 for MySQL.

2. If your database is a remote database, uncomment line 16. Then replace <hostname> with the host name or IP number of the database server.

3. Edit your userid value in place of <userid> on line 17.

4. Edit your password value in place of <password> on line 17. If there is no password, supply a null string like “”.

5. Edit the database name in place of <database_name> on line 18.

Save the file to win_test.adb and compile it as follows:

```
$ gnatmake win_test
```

You will not need to specify any linking arguments, since the GNAT feature:

```
pragma Linker_Options(“...”);
```

was used in the APQ package specs to avoid this inconvenience.

5.2 Running the Test

Before the test run can be successful, there are some obvious prerequisites on the database side:

- The database server must accept connections from you
- The database server userid and password must be acceptable
- The database named must already exist (the test program does not create it)

The test program win_test.exe, first attempts to drop a table TEST_TBL, and then creates a new one. This is done so that you can run the test multiple times. This has the implication that:

- The userid must have table create privileges
- The userid must have table drop privileges
- The userid must have sequence drop privileges, for PostgreSQL only

Configuring the security and the privileges of the user accounts, is outside the scope of this document. Note however, that you can test the connectivity and rights with the mysql command for MySQL, and psql for PostgreSQL. Once those work, you should have success with win_test.exe as well.

The test results look something like the following for MySQL:

```
$ ./win_test
Win32_Test Started:
My Userid is 'wwg'
My Password is ''
My Database is 'wwg'
My Host Name is ''
My Engine is ENGINE_MYSQL
Connecting..
Connected!
-
DROP TABLE TEST_TBL
-
Table was dropped.
-
CREATE TABLE TEST_TBL (
  NAME   VARCHAR(32) NOT NULL,
  ID     INTEGER NOT NULL AUTO_INCREMENT PRIMARY KEY
)
-
Table created.
-
INSERT INTO TEST_TBL ( NAME )
  VALUES( 'ONE' )
-
```

```
OID= 1
INSERT INTO TEST_TBL ( NAME )
  VALUES( 'TWO' )
-
OID= 2
-
INSERT INTO TEST_TBL ( NAME )
  VALUES( 'THREE' )
-
OID= 3
-
SELECT NAME,ID
FROM TEST_TBL
-
Got 3 Tuples..
NAME='ONE', ID='1'
NAME='TWO', ID='2'
NAME='THREE', ID='3'
<END>
Test Completed.
```

Note that this win_test.exe can run independently of CYGWIN as well, in a cmd.exe session. APQ is not dependent upon the CYGWIN infrastructure. CYGWIN is only used as a tool to build APQ from sources.

5.3 Congratulations!

If you successfully built and installed APQ on your win32 environment, then you are ready to branch out into entire new horizons with APQ. This software has been developed and provided to you for free. One way you can help with the development is to provide support in the way of:

- bug reports
- user experiences with APQ (good and bad)
- suggestions for enhancements
- testimonials about the APQ library to encourage others to try it
- financial support

APQ is Open Sourced, and so financial support is not required. However, financial support is always welcome, and may help with future releases. An example of where financial support may be necessary is to fund the purchase of newer Microsoft development tools. The success of win32 ports of any package, depend upon access to these tools. So if you use APQ in a business setting, please consider providing some level of support to the APQ project.

Testimonials and bug reports are another major area of support that don't cost anything. Testimonials help others realize that they need to take the time

to test drive it. If you want to provide a testimonial, just let me know if you want it kept anonymous, or the email address removed. I understand the pain of spam. Bug reports help eliminate problems for everyone, including yourself in future revisions of APQ.

User experiences and suggestions are always welcome. Just drop a line or three, and let me know how APQ can be improved, or how it has made your projects successful.

Thanks for using APQ!

Warren W. Gay VE3WWG

ve3wwg@cogeco.ca