[1.summary 2](#_Toc8028772)

[2. Source path description 2](#_Toc8028773)

[3. Compile 3](#_Toc8028774)

[3.1.windows 3](#_Toc8028775)

[3.2.linux 3](#_Toc8028776)

[4.configuration 3](#_Toc8028777)

# 1.summary

This document primarily describes how to use test code. This code is suitable for compiling and running on linux and windows environments. It is compiled by executing make on linux. You need to install Visual Studio, VS2013, VS2015 or other versions on Windows. You can create new projects and then add source files to VS project.

This test program mainly sets the test options through the configuration file, so please pay attention to the parameter values set by the configuration file during use.

The design principle of this test code:

Test items and tables used for testing, table definitions, etc. are passed through the configuration file. The test table can be a single table or a multi-table, passing the fields [table], [table\_1], [table\_2]. . . The [table\_n] method defines multiple tables, and each [table] field defines a table, and the test is sequentially tested using the definitions of Table 1 to Table N.

All test results are saved to the log file. After the test is completed, the test result analysis should be read from the log file.

This document does not fully cover the implementation of the code, the implementation details also need to implement the reference code, the test documentation will continue to improve in the application.

# 2. Source

**bin:**

Mainly store the bin file and the configuration file used during the test. When running in the Linux environment, enter the directory and execute sh build.sh to complete the compilation. Compile and execute the make in the previous directory; run the test in the directory. Mainly to facilitate testing, management test logs, etc..

**case:**

The main implementation of the test case.

**config:**

The configuration file is mainly stored; when configuring the test configuration, refer to the configuration file in the directory.

**doc:**

Mainly store relevant documents, such as "Using Documents.docx".

**driver:**

The second encapsulation of the ODBC API implementation makes it easy to test the design and invocation of the case.

**main:**

The entry implementation of the test program.

**util:**

Public interface implementation.

**vs:**

Vs2013 project path.

# 3. Compile

## 3.1.windows

Open VS, add a new project, add all the .cpp, .c and .h files under the source path, click compile. When testing, please note that the configuration and executable files are in the same path.

## 3.2.linux

In the Linux environment, the supported driver managers include unixODBC and trafodion driver manager. The driver libraries used are different. Therefore, you need to specify the driver manager to be used during compilation. Please note that the configuration and executable files are in the same path during testing.

UnixODBC:

The default is to use unixODBC, vi to open the makefile, see if ODBC\_VERSION = UNIXODBC is open, if it is commented out, please remove "#".

Trafodion driver manger:

Vi open the makefile, comment out ODBC\_VERSION = UNIXODBC, add "#" at the beginning of the line, such as: #ODBC\_VERSION = UNIXODBC

Modify the dynamic library path used by the driver:

If the default installation path is /usr/lib64, modify it as follows:

LIBPATH+=-L/usr/lib64

# 4.configuration

The configuration file config.ini mainly contains two fields: [config] field is mainly used globally, including configuration case, data source information, etc.; [table], [table\_1], [table\_n] fields are mainly used to test the use table. Configuration information.