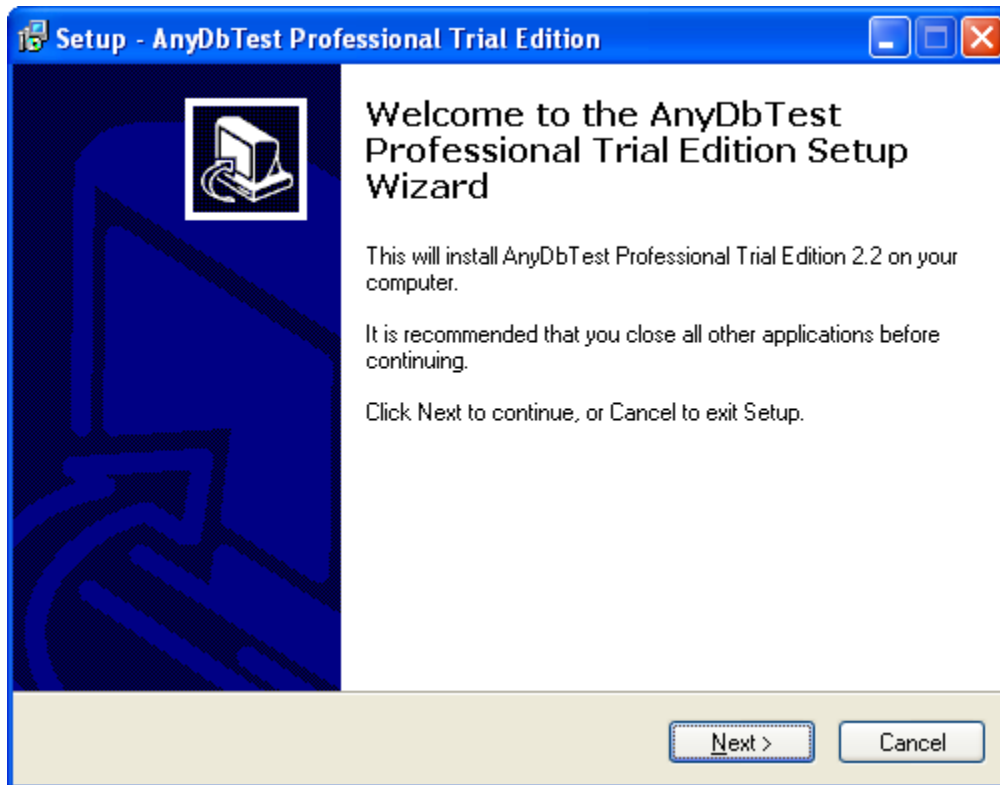


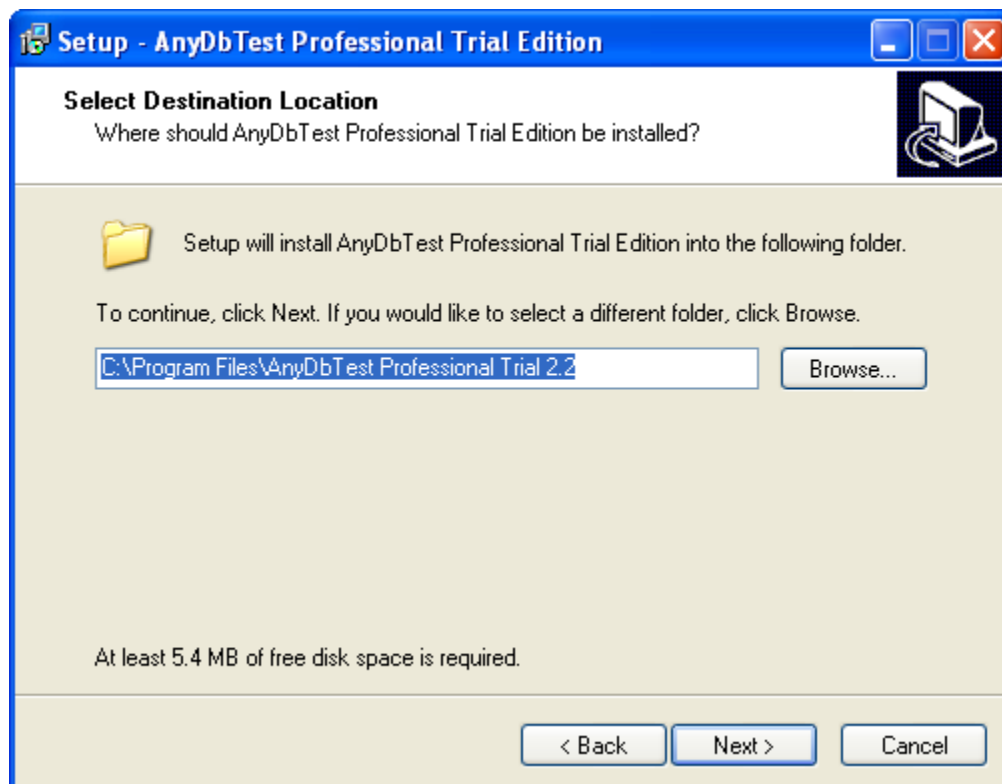
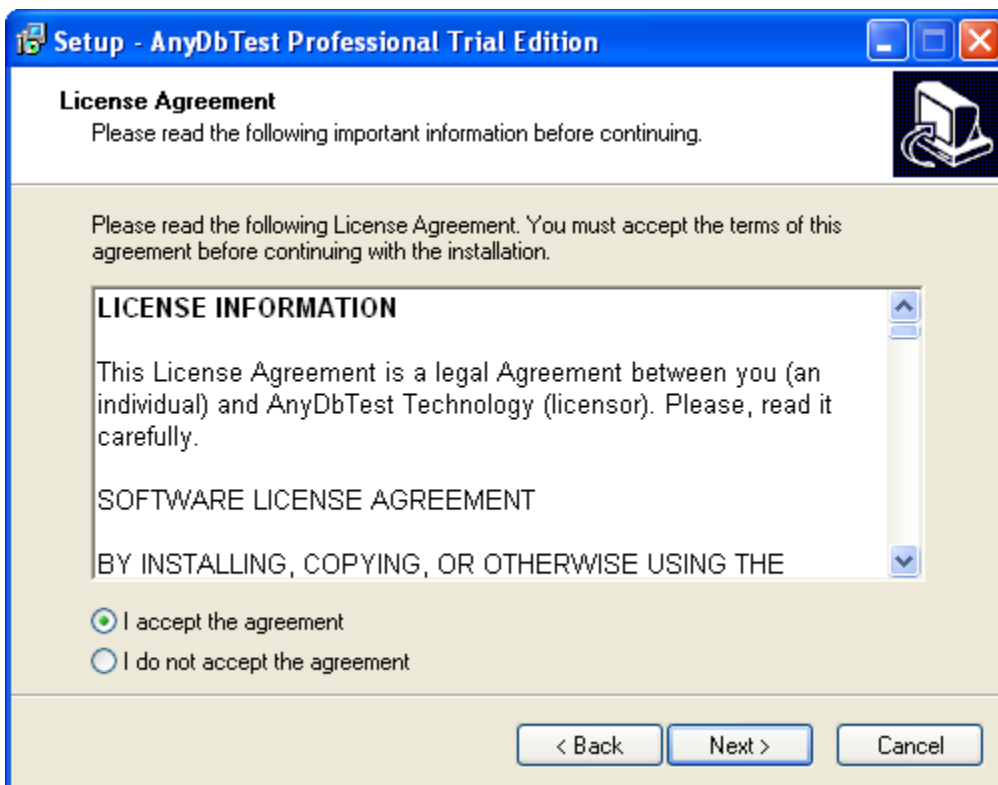
Installation manual

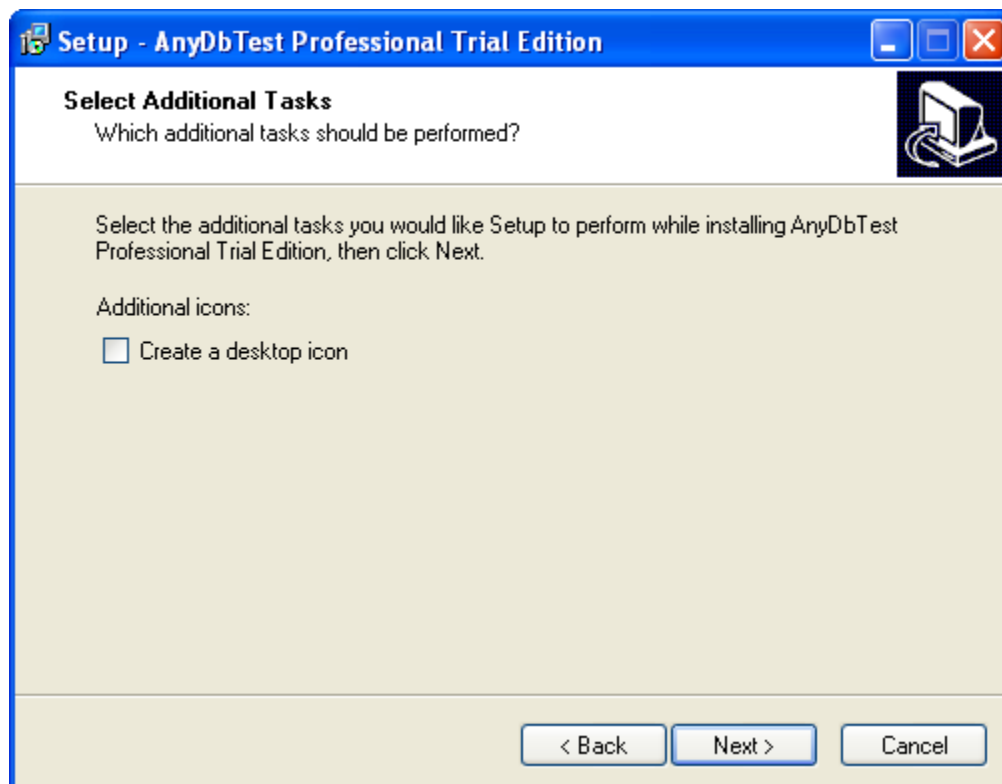
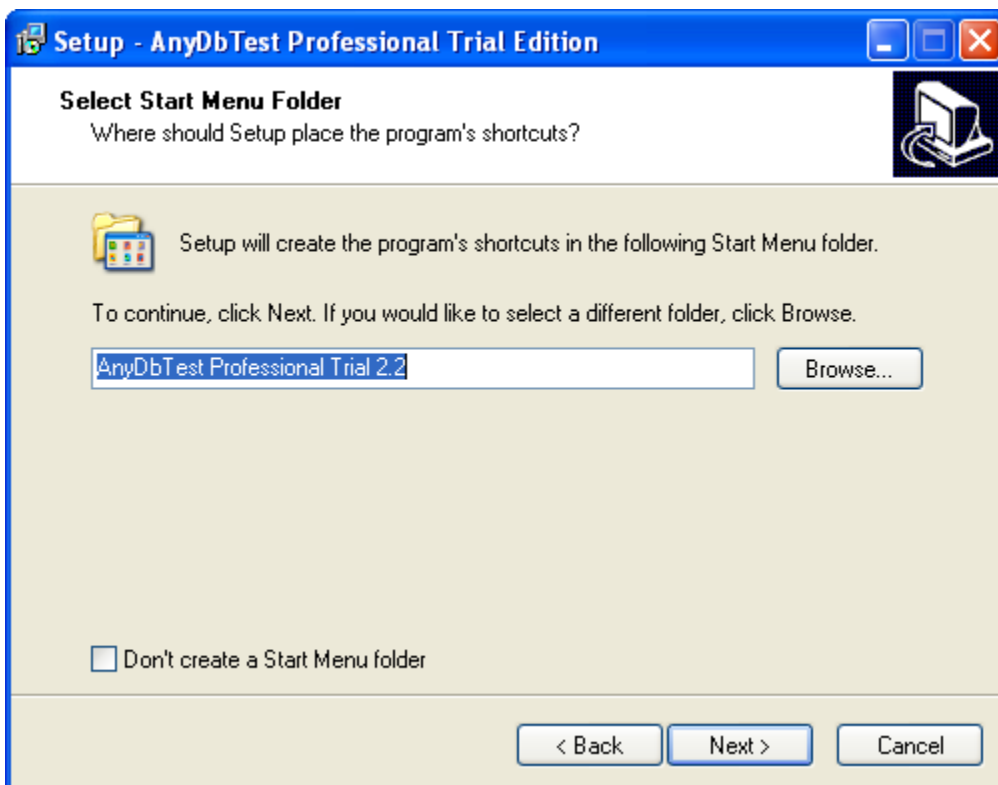
Prerequisites: Windows machine with .Net Framework 3.5. Click [here to get .Net Framework 3.5](#). Click [here to verify if you have the right version](#).

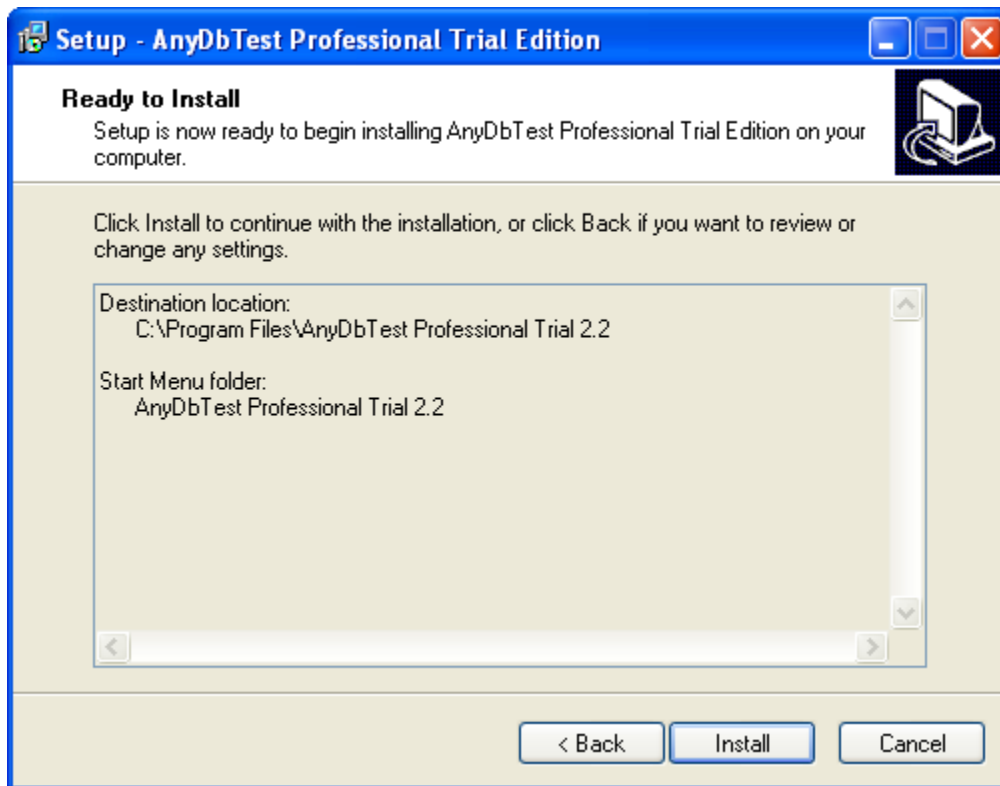
Procedures:

1. Download the latest AnyDbTest from <http://anydbtest.com/download.html>;
2. If you downloaded the zip file, extract it to C:\Program Files\AnyDbTest, and you are ready to go;
3. If you downloaded the msi file, follow the on screen instruction to install the software, create shortcut, and desktop link to it. For completeness, listed below are screen shots of each step.









Quickstart

In this example, we will use **SQL Server 2005** as our database server. If you have not yet, you can get Express edition (free of charge) from Microsoft, please click [here](#). We will also use **AdventureWorks** sample database shipped with SQL Server 2005. If you have not this sample database yet, click [here](#) to download on Microsoft CodePlex Site (or click [here](#) to download it from AnyDbTest).

Suppose that we need to create two stored procedures to retrieve state/province information. Procedure `usp_GetStatesInCountry` is to fetch states in a specific country; another procedure `usp_GetAllStates` is to fetch all states in the database. T-SQL code listed below,

```
USE [AdventureWorks]
GO

/* Purpose: To get all state in this database */
IF EXISTS (SELECT * FROM sys.objects WHERE
object_id = OBJECT_ID(N'[Person].[usp_GetAllStates]') AND type in (N'P', N'PC'))
DROP PROCEDURE [Person].[usp_GetAllStates]
GO

CREATE PROCEDURE [Person].[usp_GetAllStates]
AS
BEGIN
    SELECT name from Person.StateProvince ;
END;
GO

/* Purpose: To get all state in specific country */
```

```

IF EXISTS (SELECT * FROM sys.objects WHERE
object_id = OBJECT_ID(N'[Person].[usp_GetStatesInCountry]') AND type in (N'P',
N'PC'))
DROP PROCEDURE [Person].[usp_GetStatesInCountry]
GO

CREATE PROCEDURE [Person].[usp_GetStatesInCountry]
    @CountryCode nchar(3)
AS
BEGIN
    SELECT name FROM Person.StateProvince WHERE
    CountryRegionCode=@CountryCode;
END;

GO

```

As DB developers, we should test our code after we implemented them. For example, we can use the following test cases to verify our logic,

- 1. The count of states in USA should be 53, including Guam, District of Columbia, Puerto Rico.
- 2. Every state in the USA also should be in usp_GetAllStates.

Actually, we can design more test cases. But in this example, we only use these two cases for demonstrating how to use AnyDbTest.

Authoring test cases for AnyDbTest includes the following steps,

- 1. Create one blank test case XML file;
- 2. Declare database connection;
- 3. Declare procedure/SQL statement definition;
- 4. Write unit test/performance test.

Step 1 Create template test case XML file

We can use any text editor, even NotePad, to create a blank test file. But we recommend you strongly to use those advanced XML editors with code completion assistance. e.g. [XmlPad](#) is a free and powerful XML editor. These XML editors will be helpful to author test file from scratch.

One XML schema file namely **TestFileSchema2.xsd** has been packed in AnyDbTest distribution package. Suppose that this schema file is stored in the root folder of C drive. Now we have one blank test file displayed below,

```

<?xml version="1.0" encoding="utf-8"?>
  <dbTestFixture
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="file:/C:/TestFileSchema2.xsd">
    <globalSetting>
    </globalSetting>

    <procDeclarations>
    </procDeclarations>

    <sqlDeclarations>

```

```

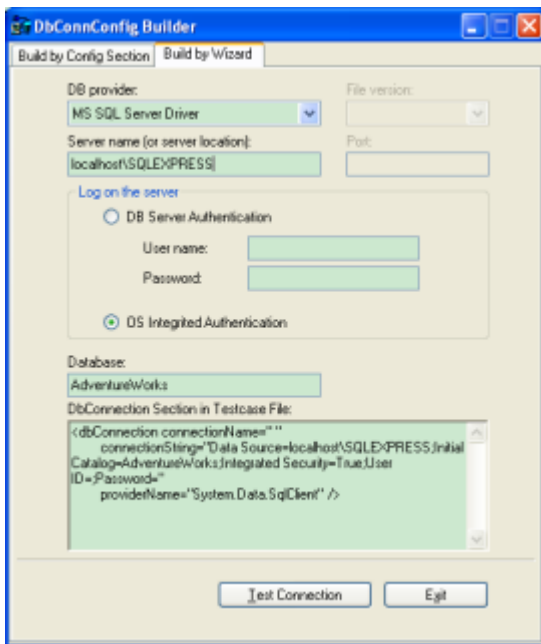
</sqlDeclarations>

<unitTestCases>
</unitTestCases>
</dbTestFixture>

```

Step 2 *Create database connection*

We can use DbConnectionBuilder application to help us to create database connection, it is a GUI wizard. We can input the SQL Server location and database name and SQL Server user/password. Notice it also supports integrated Windows authentication for SQL Server, which is what we used in the screenshot below.



The wizard will output one string. In our example, the result string is,

```

<dbConnection connectionName=""
  connectionString="Data Source=localhost\SQLEXPRESS;
  Initial Catalog=AdventureWorks; Integrated Security=True;
  User ID=;Password="
  providerName="System.Data.SqlClient" />

```

After that, copy the connection string above into globalSetting section of the test file, and give a name for this connection like db1. The globalSetting section now is,

```

<globalSetting>
  <dbConnection connectionName="db1"
    connectionString="Data Source=localhost\SQLEXPRESS;
    Initial Catalog=AdventureWorks; Integrated Security=True;
    User ID=;Password="
    providerName="System.Data.SqlClient" />
</globalSetting>

```

Step 3 Declare procedure/SQL statement definition

Now we need to tell AnyDbTest, what are our stored procedures, and where? With code completion of XML editor(like [XmlPad](#)), actually, this process also is very easy. The procDeclarations section is as follows,

```
<procDeclarations>
<procDeclaration alias="proc_Get_state_in_country" dbConnection="db1"
  name="usp_GetStatesInCountry" namespace="Person">
  <remark>
    This is declaration for usp_GetStatesInCountry
  </remark>
  <arguments>
    <argument name="@CountryCode" direction="Input"
      type="MSSQL_NCHAR"/>
    <argument name="@Return_Table" direction="AnonymousOutput"
      type="MSSQL_RECORDSET"/>
  </arguments>
</procDeclaration>

<procDeclaration alias="proc_Get_all_state" dbConnection="db1"
  name="usp_GetAllStates" namespace="Person">
  <remark>
    This is declaration for "[Person].[usp_GetAllStates]"
  </remark>
  <arguments>
    <argument name="@Return_Table" direction="AnonymousOutput"
      type="MSSQL_RECORDSET"/>
  </arguments>
</procDeclaration>
</procDeclarations>
```

Additionally, in sqlDeclarations section we add one SQL statement declaration for getting a constant number from SQL Server. It is used in this test file.

```
<sqlDeclarations>
<sqlDeclaration alias="sql_get_const_number" dbConnection="db1">
  <remark>to get constant number from database server
</remark>
<sql><![CDATA[select @constant]]></sql>
  <arguments>
    <argument name="@constant" direction="Input" type="MSSQL_INT"/>
    <argument name="@result" direction="Return" type="MSSQL_INT"/>
  </arguments>
</sqlDeclaration>
</sqlDeclarations>
```

Step 4 Write unit/performance test

Now it is time to create our test cases. As mentioned above, we will create two unit test cases: the first case is to verify states amount of USA is 53; the second is to verify that every state of USA also should be in the usp_GetAllStates().

One unit test case includes 3 mandatory parts, they are assertion type, targetResultset,

referenceResultset. We will explain them one by one.

Assertion type is to tell AnyDbTest how to judge if the test is successful. AnyDbTest contains the various standard assertions such as RecordCountEqual and IsSubsetOf and so on. Section targetResultset is our testing target; and section referenceResultset is our reference result set.

For test case 1, we choose the **RecordCountEqual** as assertion type. In this case, targetResultset comes from proc_Get_state_in_country. proc_Get_state_in_country is statement alias for stored procedure usp_GetStatesInCountry. Let us assign "US" to the value of @CountryCode input-type argument. The referenceResultset is constant number 53.

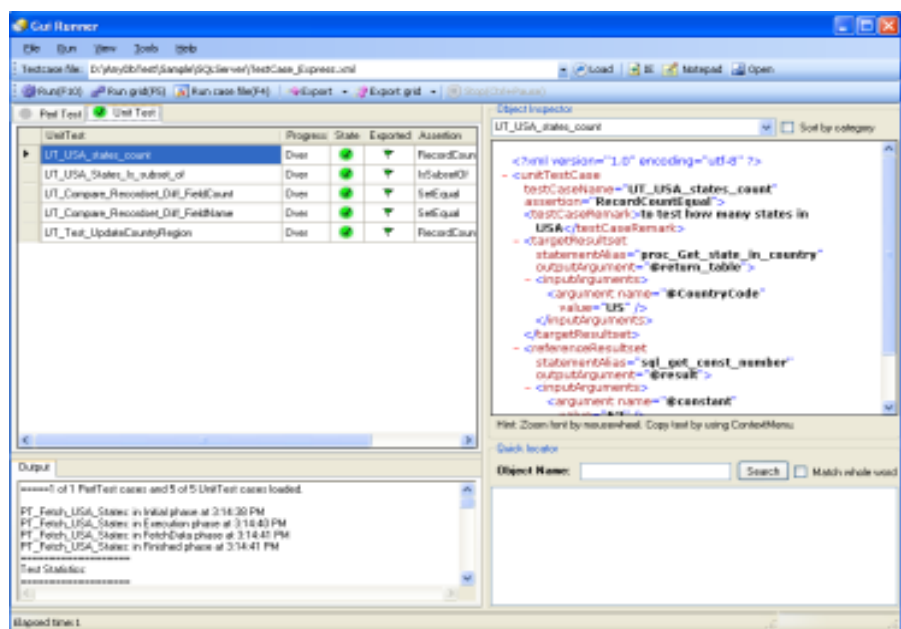
Finally, the unit test case section is as follows,

```
<unitTestCase testCaseName="UT_USA_states_count"
               assertion="RecordCountEqual">
  <testCaseRemark>to test how many states in USA
</testCaseRemark>
  <targetResultset statementAlias="proc_Get_state_in_country"
                   outputArgument="@return_table" >
    <inputArguments>
      <argument name="@CountryCode" value="US"/>
    </inputArguments>
  </targetResultset>
  <referenceResultset statementAlias="sql_get_const_number"
                     outputArgument="@result">
    <inputArguments>
      <argument name="@constant" value="53"/>
    </inputArguments>
  </referenceResultset>
</unitTestCase>
```

For test case 2, In fact, it is to judge one result set is subset of another result set. We choose another standard assertion type **IsSubsetOf** from available types. The entire unit test section is as follows,

```
<unitTestCase testCaseName="UT_USA_States_Is_subset_of"
               assertion="IsSubsetOf">
  <testCaseRemark>to judge states in USA appear in
result set of usp_GetAllStates</testCaseRemark>
  <targetResultset statementAlias="proc_Get_state_in_country"
                   outputArgument="return_table">
    <inputArguments>
      <argument name="@CountryCode" value="US"/>
    </inputArguments>
  </targetResultset>
  <referenceResultset statementAlias="proc_Get_all_state"
                     outputArgument="return_table">
    </referenceResultset>
</unitTestCase>
```

It is time to launch AnyDbTest to run our test file. Launch the AnyDbTest, and load the test case file. Then click the button 'Run case file'. You will see the familiar Red-Green Light icon just like other unit testing tool.



The whole process is easy, isn't it?