



Using DBMaker with NHibernate Manual

Version: 01.00

Document No: 54/DBM54-T12182017-02-DBNH
Author: SYSCOM Computer Engineering CO.
Publication Date: Feb 10, 2017

Table of Content

1. Introduction	2
1.1 Additional Resources	2
1.2 Technical Support.....	3
2. Install NHibernate	5
3. Create Users table in Database	6
4. Build C# test AP Project.....	7
4.1 Build Dialect and Driver DLL.....	7
4.1.1 Add existed source code.....	7
4.1.2 Build NHibernate.DBMakerDriver.dll.....	7
4.2 Create Test Console AP project	7
4.2.1 Add Users.cs to map the table in database	7
4.2.2 Add Users.hbm.xml for Users.cs	8
4.2.3 Add hibernate.cfg.xml	8
4.2.4 Add Test.cs	9
4.3 Add depended DLL for Test AP.....	10
5. Run Test and check result	11
5.1 Build and run Test.cs	11
5.2 Check inserted data in DB Table.....	11
6. Limitation at present	12
6.1 Data type	12
6.2 SQL syntax	12
6.3 Other.....	12

1. Introduction

DBMaker is a powerful and flexible SQL Database Management System (DBMS) that supports an interactive Structured Query Language (SQL), a Microsoft Open Database Connectivity (ODBC) compatible interface, and Embedded SQL for C (ESQL/C). DBMaker also supports a Java Database Connectivity compliant interface and DBMaker COBOL interface (DCI). The unique open architecture and native ODBC interface give the user freedom to build custom applications using a wide variety of programming tools, or query a database using existing ODBC, JDBC, or DCI compliant applications.

From basic architectures of JDBC and AP Server, take Tomcat as the sample AP Server, we introduce Architectures that AP Server accesses DBMaker via JDBC on one machine and different machines. If you understand these two kinds architectures that DBMaker used, you will make your AP Server access DBMaker easily and do not make too much settings.

1.1 Additional Resources

DBMaster provides a complete set of DBMS manuals in addition to this one. For more detailed information on a particular subject, consult one of the books listed below:

- For an introduction to DBMaster's capabilities and functions, refer to the *DBMaster Tutorial*.
- For more information on designing, administering, and maintaining a DBMaster database, refer to the *Database Administrator's Guide*.
- For more information on the SQL language used in dmSQL, refer to the *SQL Command and Function Reference* manual.
- For more information on the UNLOAD/LOAD programming, refer to the *SQL Command and Function Reference* manual – dmSQL commands.
- For more information on stored procedure, refer to the *Stored Procedure User's Guide*.
- Document Conventions

This book uses a standard set of typographical conventions for clarity and ease of use. The NOTE, Procedure, Example, and Command Line conventions also have a second setting used with indentation.

CONVENTION	DESCRIPTION
<i>Italics</i>	Italics indicate placeholders for information that must be supplied, such as user and table names. The word in italics should not be typed, but is replaced by the actual name. Italics also introduce new words, and are occasionally used for emphasis in text.
Boldface	Boldface indicates filenames, database names, table names, column names, user names, and other database schema objects. It is also used to emphasize menu commands in procedural steps.
KEYWORDS	All keywords used by the SQL language appear in uppercase when used in normal paragraph text.
SMALL CAPS	Small capital letters indicate keys on the keyboard. A plus sign (+) between two key names indicates to hold down the first key while pressing the second. A comma (,) between two key names indicates to release the first key before pressing the second key.
NOTE	Contains important information.
Procedure	Indicates that procedural steps or sequential items will follow. Many tasks are described using this format to provide a logical sequence of steps for the user to follow
Example	Examples are given to clarify descriptions, and commonly include text, as it will appear on the screen. Other forms of this convention include Prototype and Syntax.
Command Line	Indicates text, as it should appear on a text-delimited screen. This format is commonly used to show input and output for dmSQL commands or the content in the dmconfig.ini file

Table 1-1 Document Conventions

1.2 Technical Support

DBMaster provides thirty days of complimentary email and phone support during the evaluation period. When software is registered an additional thirty days of support will be included. Thus extend the total support period for software to sixty days. However, DBMaster will continue to provide email support for any bugs reported after the complimentary support or registered support has expired (free of charges).

Additional support is available beyond the sixty days for most products and may be purchased for twenty percent of the retail price of the product. Please contact sales@casemaker.com for more details and prices.

DBMaster support contact information for your area (by snail mail, phone, or email) can be located at: www.casemaker.com/support. It is recommended that the current database of FAQ's be searched before contacting DBMaster support staff.

Please have the following information available when phoning support for a troubleshooting enquiry or include the information with a snail mail or email enquiry:

- Product name and version number
- Registration number
- Registered customer name and address
- Supplier/distributor where product was purchased
- Platform and computer system configuration
- Specific action(s) performed before error(s) occurred
- Error message and number, if any
- Any additional information deemed pertinent

2. Install NHibernate

Download and unzip NHibernate package.

For example:

D:\NHibernate-4.1.1.GA-bin

<https://sourceforge.net/projects/nhibernate/files/NHibernate/4.1.1.GA/NHibernate-4.1.1.GA-bin.zip/download>

3. Create Users table in Database

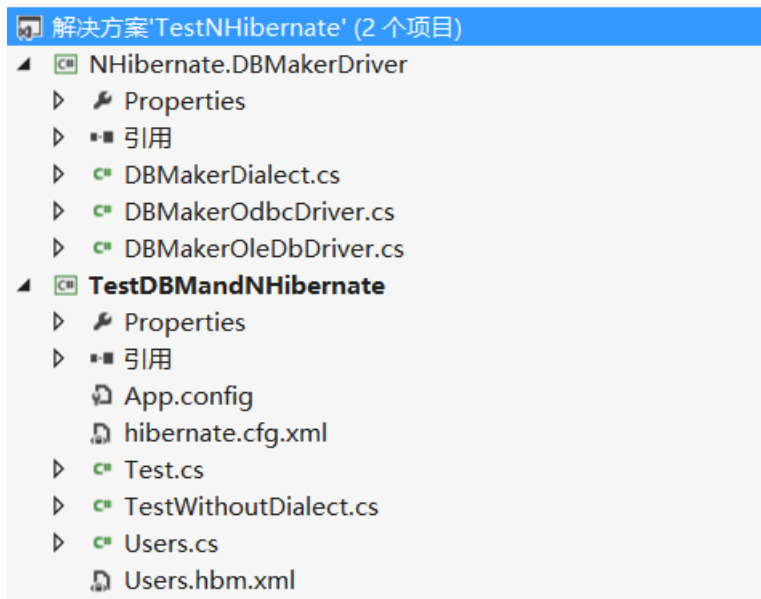
Connect to DB and create a table for running hibernate sample.

```
dmSQL> create table users (id int, username char(30), password char(30));
```

4. Build C# test AP Project

Create a new Solution – TestNHibernate in Microsoft Visual Studio, and add two Project (*Class Library Project* and *Console Application Project*).

For example: Solution and Projects name as following:



4.1 Build Dialect and Driver DLL

4.1.1 ADD EXISTED SOURCE CODE

Add DBMakerDialect.cs, DBMakerOdbcDriver.cs and DBMakerOleDbDriver.cs in *Class Library Project* - NHibernate.DBMakerDriver.

Note: You can get these source code files from DB Support Team.

4.1.2 BUILD NHIBERNATE.DBMAKERDRIVER.DLL

The depended DLL lib files are NHibernate.dll and log4net.dll (which are in D:\NHibernate-4.1.1.GA-bin\Required_Bins\ and D:\NHibernate-4.1.1.GA-bin\Tests\)

4.2 Create Test Console AP project

4.2.1 ADD USERS.CS TO MAP THE TABLE IN DATABASE

```
using System;
using System.Collections.Generic;
using System.Linq;
```



```

using System.Text;
using System.Threading.Tasks;
namespace TestNHibernate
{
    public class Users
    {
        public virtual int Id { get; set; }
        public virtual string Username { get; set; }
        public virtual string Password { get; set; }
    }
}

```

4.2.2 ADD USERS.HBM.XML FOR USERS.CS

```

<?xml version="1.0" encoding="utf-8" ?>
<hibernate-mapping xmlns="urn:hibernate-mapping-2.2"
    assembly="TestNHibernate"
    namespace="TestNHibernate">
    <!-- more mapping info here -->
    <class name="Users" table="Users">
        <id name="Id" column="Id">
            <generator class="assigned" />
        </id>
        <property name="Username" column="Username" type="String" length="30" />
        <property name="Password" column="Password" type="String" length="30" />
    </class>
</hibernate-mapping>

```

Note: Please modify the XML *File Property* as

1) Copy to output directory

copy always

2) Build Action

Embedded Resource.

3) schemas

D:\NHibernate-4.1.1.GA-bin\Required_Bins\hibernate-mapping.xsd

4.2.3 ADD HIBERNATE.CFG.XML

Add hibernate.cfg.xml for URL/connection string and so on.

```

<?xml version="1.0" encoding="utf-8" ?>
<hibernate-configuration xmlns="urn:hibernate-configuration-2.2">
    <!-- an ISessionFactory instance for DBMaker-->
    <session-factory>
        <!-- properties -->
        <property name="hbm2ddl.keywords">none</property>
        <property name="connection.provider">NHibernate.Connection.DriverConnectionProvider</property>
        <property name="dialect">NHibernate.Dialect.DBMakerDialect,
NHibernate.DBMakerDriver</property>
        <property name="query.substitutions">>true 1, false 0, yes 1, no 0</property>
        <property name="connection.driver class">NHibernate.Driver.DBMakerOdbcDriver,
NHibernate.DBMakerDriver</property>
    </session-factory>
</hibernate-configuration>

```

```

    <property name="connection.connection string">Driver={DBMaker 5.4
Driver};DATABASE=dbsample5;uid=SYSADM;pwd=;</property>
    <property name="show sql">true</property>
    <!-- mapping files -->
    <mapping assembly="TestNHibernate" />
</session-factory>
</hibernate-configuration>

```

Note: Please modify the XML *File Property* as

1) Copy to output directory

copy always

2) Build Action

Embedded Resource.

3) schemas

D:\NHibernate-4.1.1.GA-bin\Required_Bins\hibernate-configuration.xsd

4.2.4 ADD TEST.CS

Add main function in Test.cs for testing DB with NHibernate.

```

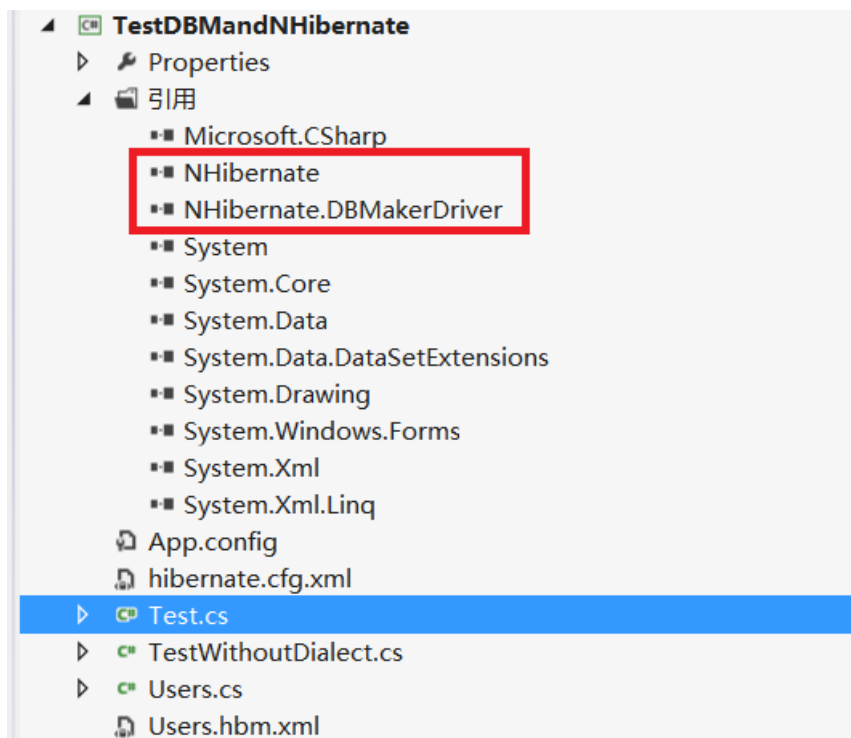
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using NHibernate;
using NHibernate.Cfg;
using NHibernate.Dialect;
using NHibernate.Driver;
using NHibernate.Mapping.ByCode;
using System.Reflection;
namespace TestNHibernate
{
    class Test
    {
        static void Main(string[] args)
        {
            try
            {
                Configuration cfg = new Configuration().Configure("hibernate.cfg.xml");
                ISessionFactory factory = cfg.BuildSessionFactory();
                ISession session = factory.OpenSession();
                ITransaction transaction = session.BeginTransaction();
                Users newUser = new Users();
                newUser.Id = 2;
                newUser.Username = "nuser2";
                newUser.Password = "npasswd2";
                session.Save(newUser);
                transaction.Commit();
                session.Close();
            } catch (Exception ex)
            {

```

```
        Console.WriteLine(ex.Message);
        Console.WriteLine(ex.InnerException);
    }
    Console.WriteLine("Press ENTER to exit...");
    Console.Read();
}
}
```

4.3 Add depended DLL for Test AP

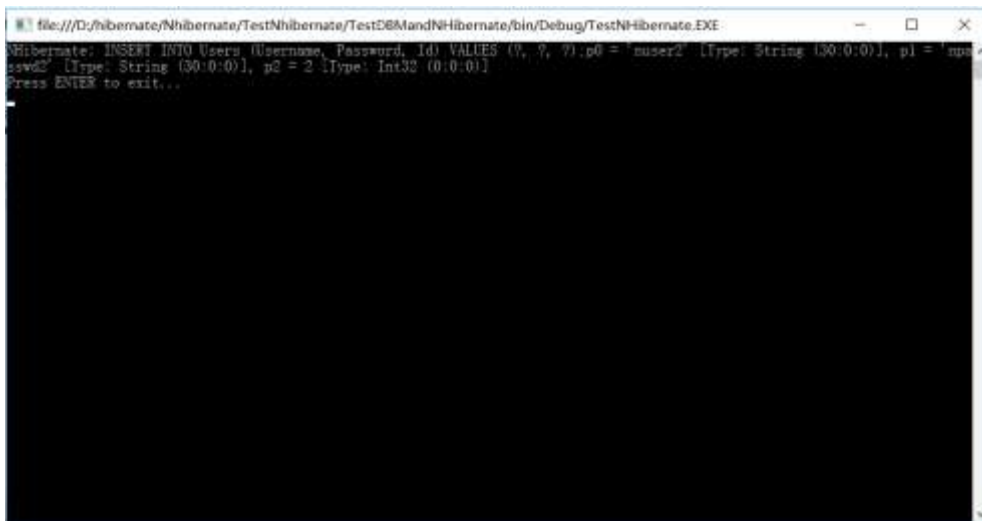
You must add some required library DLL for building and running this test case.



5. Run Test and check result

5.1 Build and run Test.cs

After building, if run it successfully, you can AP console as following:



```
file:///D:/hibernate/NHibernate/TestNHibernate/TestDBMandNHibernate/bin/Debug/TestNHibernate.EXE
NHibernate: INSERT INTO Users (Username, Password, Id) VALUES (?, ?, ?) p0 = 'nuser2' [Type: String (30:0:0)], p1 = 'npa
sswd2' [Type: String (30:0:0)], p2 = 2 [Type: Int32 (0:0:0)]
Press ENTER to exit...
```

5.2 Check inserted data in DB Table

At last, you can check the inserted data in database:

```
dmSQL> select * from users;
```

ID	USERNAME	PASSWORD
1	name1	pass1
2	nuser2	npasswd2

```
2 rows selected
```

6. Limitation at present

6.1 Data type

- GUID is not supported.
- VarBinary is mapping to BLOB, but not support comparison operation.

6.2 SQL syntax

- Not support SELECT in projection list, such as:

```
select name, (select avg(score) from sc where sc.sid = s.sid) as avg_score from s;
```

- Not support LIMIT in sub query.

6.3 Other

- Not support IDENTITY keyword.
- Not support SEQUENCE object.
- Not support batch SQL statements.