CompareSQL&DBMaker

Reference document for T_SQL SP

Compare to DBMaker 5.4.0 SQL SP

Version: DBMaker5.4 & **SQL Server 2005**

Document No: **54/DBM54-T07222016-01-CSDB**

Author: **DBMaster Support Team & Production Team, Research & Development Division, SYSCOM Computer Engineering CO.**

Print Date: **July 07, 2016**

凌群電腦
THE SYSCOM GROUP

Table of Content

1. Data Type

1.1 SQL Support Data Type but DBMaker Unsupport

- tyinint
- money
- bit

1.2 IDENTIFY&SERIAL

SQL support definition: ID int IDENTITY(1,5) but DBMaker cannot support definition serial（1，5）, so DBMaker cannot support such operation that is the increase range is 5 when insert every one record.

1.3 Other Sources of Information

DBMaster standard version provides many other user's guides and reference manuals in addition to this reference. For more information on a particular subject, consult one of these books:

- For more information on the designing, administering, and maintaining a DBMaster database, refer to the "Database Administrator's Guide".
- For more information on the SQL language implemented by DBMaster, refer to the "SQL Command and Function Reference".
- For more information on the ESQL/C language implemented by DBMaster, refer to the "ESQL/C Programmer's Guide".
- For more information on error and warning messages, refer to the "Error and Message Reference".
- For more information on using stored procedure, refer to the "Stored Procedure user's guide".

1.4 Technical Support

CASEMaker 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, CASEMaker 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.

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

2.1 SQL&DBMaker

In SQL, you can write control condition in SQL script, such as if……else and so on, but DBMaker cannot support.

```
IF EXISTS (SELECT name FROM sysobjects WHERE name = 'au_info_all' AND type = 'P')
DROP PROCEDURE au_info_all
```

If judge before create procedure that will stricter.

2.2 SQL & DBMaker “;”

In SQL, both add “;” or not in the end of a complete SQL row are ok in the SQL scropt, but in DBMaker, you must add”;” in the end of a complete SQL statement for row.

2.3 INSERT & INSERT INTO

About INSERT statement, SQL support “insert or insert into”, DBMaker only support write to “insert into”.

2.4 DBMaker SQLSP should be Capital Letters

- DBMaker doesn’t use global variable, in SqlServer, @@variable_name define global variable.
- In SqlServer, the existed cursor can be modified.

3. Stored Procedure

3.1 Different Syntax

- DBMaker sp:

```
CREATE PROCEDURE project_name.module_name.sp_name [ arg_list ]

LANGUAGE SQL

BEGIN

[declare_list]

[statement_list]

END;
```

[MS1]: Must be shows that language is the SQL statement

- SqlServer sp:

```
CREATE PROC [EDURE] procdure_name

[WITH

{

           Recompile

           |Encryption

           |Recompile，Encryption

}

]

AS

Sql_statement [...n]
```

[MS2]: DBMaker doesn't has these option

[MS3]: Operating statements begin with AS.
In DBMaker, Operating statements were included between BEGIN ……AND.
In SQLserver, Operating statements can be written follow AS directly or an also be written BEGIN……END after the AS,then write the operationg statements between BEGIN……END.

3.2 Some difference on SP which Return Result Set by SELECT

- DBMaker:

```
CREATE PROCEDURE au_info_all

LANGUAGE SQL

BEGIN



SELECT au_lname, au_fname, title, pub_name
```

```
   FROM authors a
      INNER JOIN titleauthor ta
      ON a.au_id = ta.au_id
      INNER JOIN titles t
      ON t.title_id = ta.title_id
      INNER JOIN publishers p
      ON t.pub_id = p.pub_id;
END;
```

The Select statement will return a result set, the syntax and compile both will be well, the sql_sp also can be created, but it cannot return result set when call sp_name:

So we must use cursor to return a result set.

Solve method:

```
CREATE PROCEDURE au_info_all
RETURNS VARCHAR(40) VAL1,VARCHAR(20) VAL2 , VARCHAR(80) VAL3,VARCHAR(40) VAL4
LANGUAGE SQL
BEGIN
DECLARE cur CURSOR WITH RETURN FOR
SELECT au_lname, au_fname, title, pub_name
   FROM authors a
      INNER JOIN titleauthor ta
      ON a.au_id = ta.au_id
      INNER JOIN titles t
      ON t.title_id = ta.title_id
      INNER JOIN publishers p
      ON t.pub_id = p.pub_id;
OPEN cur;
END;
```

[MS4]: RETURNS keyword and WITH RETURN (declare cur) must be existed when return a query result set.

[MS5]: The return query list by RETURN_LIST must be defined here, then can return correct data.

- SqlServer:

```
IF EXISTS (SELECT name FROM sysobjects
         WHERE name = 'au_info_all' AND type = 'P')
   DROP PROCEDURE au_info_all
GO
CREATE PROCEDURE au_info_all
AS
SELECT au_lname, au_fname, title, pub_name
   FROM authors a
```

[MS6]: To judge if current creating sp had been existed in databse(optional).
In DBMaker, we don't have such method and don't have any similar replace method to judge, and cannot add if condition in script except SP internal.

```
      INNER JOIN titleauthor ta

      ON a.au_id = ta.au_id

      INNER JOIN titles t

      ON t.title_id = ta.title_id

      INNER JOIN publishers p

      ON t.pub_id = p.pub_id

GO
```

SqlServer doesn't need to declare any return value, call directly then can return query result set.

3.3 Stored Procedure with Parameters

- DBMaker:

```
CREATE PROCEDURE au_info (firstname varchar(20),lastname varchar(40))

RETURNS VARCHAR (40) VAL1, VARCHAR (20) VAL2, VARCHAR (80) VAL3, VARCHAR (40) VAL4

LANGUAGE SQL

BEGIN

DECLARE cur CURSOR WITH RETURN FOR

SELECT au_lname, au_fname, title, pub_name

   FROM authors a INNER JOIN titleauthor ta

      ON a.au_id = ta.au_id INNER JOIN titles t

      ON t.title_id = ta.title_id INNER JOIN publishers p

      ON t.pub_id = p.pub_id

      WHERE au_fname = firstname

      AND au_lname = lastname;

OPEN cur;

END;
```

[MS7]: You must define input parameter here.

- SqlServer:

```
CREATE PROCEDURE au_info @lastname varchar (40), @firstname varchar (20)

AS

SELECT au_lname, au_fname, title, pub_name

   FROM authors a INNER JOIN titleauthor ta

      ON a.au_id = ta.au_id INNER JOIN titles t

      ON t.title_id = ta.title_id INNER JOIN publishers p

      ON t.pub_id = p.pub_id

      WHERE au_fname = @firstname

      AND au_lname = @lastname
```

[MS8]: @parameters

3.4 Call Stored Procedure

- DBMaker:

```
Call sp_name；
```

- SqlServer:

```
Exec (ute) sp_name
```

3.5 Comment Symbol is Different

- DBMaker: #
- SqlServer: Multi-line comment /**/ and Single-Line Comments--

3.6 Parameter Declaration is Different

- DBMaker: Input parameter：IN /INPUT(or omit) Output parameter：OUT/OUTPUT
- SqlServer: Input parameter @variable_name

3.7 Using Wildcard in Stored Procedure

- DBMaker: No
- SQLserver: OK and the example as below.

```
CREATE PROCEDURE au_info2

   @lastname varchar (30) = 'D%',

   @firstname varchar (18) = '%'

AS

SELECT au_lname, au_fname, title, pub_name

FROM authors a INNER JOIN titleauthor ta

   ON a.au_id = ta.au_id INNER JOIN titles t

   ON t.title_id = ta.title_id INNER JOIN publishers p

   ON t.pub_id = p.pub_id

WHERE au_fname LIKE @firstname

   AND au_lname LIKE @lastname
```

3.8 Temporary Stored Procedure

- DBMaker: No
- SqlServer:

The local stored procedure: #procedure_name

The global stored procedure: ##procedure_name

3.9 The Stored Procedure with Output Parameter

If want to get same result by call SqlServer sp and DBMaker sp, the corresponding wrtting as below, In SqlServer, get the result with select but in DBMaker should use cursor to get same result.

- DBMaker:

```
CREATE PROCEDURE T_SQL (IN lname varchar (40), OUT fname varchar (20))

LANGUAGE SQL

BEGIN

     DECLARE C CURSOR FOR SELECT au_fname FROM authors WHERE au_lname = lname;

     OPEN C;

     FETCH C INTO fname;

     CLOSE C;

END;
```

- SqlServer:

```
CREATE PROCEDURE t_sql

@lname varchar (40),

@fname varchar (20) OUT

AS

        select au_fname from authors

    where au_lname = @lname*/

/*

DECLARE @LNAME VARCHAR (40),

                @FNAME VARCHAR (20)

EXEC t_sql 'White', @FNAME OUTPUT

            SELECT @FNAME 'fname'
```

3.10 The Procedure with Stored Procedure

The usage of cursor

- DBMaker:

```
CREATE PROCEDURE Gettitles1 (OUT VAL1 VARCHAR (6), OUT VAL2 DOUBLE)

LANGUAGE SQL

BEGIN

     DECLARE mytitle varchar(6);

     DECLARE myprice double;


     DECLARE titleCursor CURSOR FOR SELECT title_id,price FROM TITLES;

     OPEN titleCursor ;
```

```
         WHILE (SQLSTATE = '00000')

              DO

              FETCH FROM titleCursor INTO mytitle, myprice;

         END WHILE;

     CLOSE titleCursor;

     SET VAL1 = mytitle;

     SET VAL2 = myprice;

END;
```

Execute Statement

```
Call titleCursor (?,?);

SqlServer：

If Object_ID ('dbo.Gettitles') is Not Null

       Drop Proc dbo.Gettitles

 Go


 Create Proc Gettitles

 @MyCursor Cursor Varying Output

 As

        Set @MyCursor = Cursor

        For

               Select title_id, price from titles

 Open @MyCursor

 Go

Execute Procedure () （The variable must be declared if used）

Declare @titleID varchar (6)

Declare @bookprice money

Declare @titleCursor Cursor

Exec Gettitles @titleCursor out

Fetch next from @titleCursor

Into @titleID, @bookprice

While (@@Fetch_Status = 0)

Begin

       Begin

              Print @titleID

              Print @bookprice

       End
       Fetch next from @titleCursor

       Into @titleID, @bookprice

End

Close @titleCursor

Deallocate @titleCursor

Go
```

```
NEXT, LAST, PRIOR, FIRST SP

DBMaker

CREATE PROCEDURE FETCH_TEST (OUT FHTY_1 CHAR (20), OUT FHTY_2 CHAR (20), OUT FHTY_3
CHAR (20), OUT FHTY_4 CHAR (20))

LANGUAGE SQL

BEGIN

        DECLARE V1 CHAR (20);

        DECLARE V2 CHAR (20);

        DECLARE V3 CHAR (20);

        DECLARE V4 CHAR (20);

        DECLARE CUR CURSOR FOR SELECT * FROM TB_1;

        OPEN CUR;

        FETCH FIRST FROM CUR INTO V1;

                IF V1 = 'FIRST' THEN

                        SET FHTY_1 = V1;

                ELSE

                        SET FHTY_1 = 'NULL';

                END IF;

        FETCH NEXT FROM CUR INTO V2;

                IF V2 = 'NEXT' THEN

                        SET FHTY_2 = V2;

                ELSE

                        SET FHTY_2 = 'NULL';

                END IF;

        FETCH NEXT FROM CUR INTO V3;

        FETCH PRIOR FROM CUR INTO V3;

                IF V3 = 'PRIOR' THEN

                        SET FHTY_3 = V3;

                ELSE

                        SET FHTY_3 = 'NULL';
                END IF;

        FETCH LAST FROM CUR INTO V4;

                IF V4 = 'LAST' THEN

                        SET FHTY_4 = V4;

                ELSE

                        SET FHTY_4 = 'NULL';

                END IF;

        CLOSE CUR;

END;
```

- SqlServer:

```
Create Proc sp_move

 @MyCursor Cursor Varying Output

 As

        Declare mycur SCROLL CURSOR FOR select * from tb_move

        set @MyCursor = mycur

 Open @MyCursor

 Go


Declare @val1 char (10)

Declare @titleCursor Cursor

Exec sp_move @titleCursor out


Fetch next from @titleCursor InTo @val1

if (@val1! = NULL)

print 'Fetch the next row in the cursor：'+@val1

else

fetch FIRST from @titleCursor into @val1

print 'Fetch the first row in the cursor：'+@val1


fetch LAST from @titleCursor into @val1

print 'Fetch the last row in the cursor：'+@val1

fetch PRIOR from @titleCursor into @val1

print 'Fetch the row immediately prior to the current row in the cursor：'+@val1

fetch RELATIVE -2 from @titleCursor into @val1
```

[MS9]: In SqlServer, just the cursor which was declared by keyword SCROLLcan move to other direction except NEXT.

[MS10]: SqlServer is much than DBMaker 2 absolute value

```
print 'Fetch the row that is two rows prior to the current row：'+@val1

fetch ABSOLUTE 2 from @titleCursor into @val1

print 'Fetch the second row in the cursor'+@val1


Close @titleCursor

Deallocate @titleCursor

Go
```

3.11 Using try……catch Statement Block in SP

- DBMaker:
- SqlSever:

```
IF OBJECT_ID ('usp_MyError', 'P') IS NOT NULL

    DROP PROCEDURE usp_MyError;

GO

CREATE PROCEDURE usp_MyError

AS

    SELECT * FROM NonExistentTable;

GO

BEGIN TRY

    EXECUTE usp_MyError;

END TRY

BEGIN CATCH

    SELECT

        ERROR_NUMBER () AS ErrorNumber,

        ERROR_MESSAGE () AS ErrorMessage;

END CATCH;

GO
```

3.12 Parameter Data Type

- DBMaker: Support 14 data types.

SMALLINT, INTEGER, BIGINT, FLOAT, DOUBLE, DECIMAL, DATE, TIME, TIMESTAMP, BINARY, CHAR, VARCHAR, NCHAR, NVARCHAR.

- SqlServer: Support 27 data types, text, ntext and image data type cannot be used as OUTPUT parameter.

All of the data types（include text, ntext and image）can be used as stored procedure parameter. But cursor data type just can be used as OUTPUT parameter. If specify the data type is cursor，that must specify the keywords VARYING and OUTPUT at the same time.

The included data type as below:

Precise figures

Integer

Bigint, int, smallint, tinyint, Bit, decimal, numeric, money, smallmoney

Approximate numbers

Float, real, datetime, smalldatetime

Character string

Char, varchar, text, nchar, nvarchar, ntext

Binary string

Binary, varbinary, image

Other data types:

Cursor

sql_variant:

The data type which used to storage SQL Server supports various data type (except text, ntext, timestamp and sql_variant) value.

Table, timestamp, uniqueidentifier