SQL SERVER – USING THE TABLEDIFF COMMAND LINE UTILITY

The **tablediff** command line utitlity provides us with the ability to compare the content of two tables in a database. It not only tells us which ***records / columns*** are different but can also generate a **SQL** script to update the second table to make it the same as the first table.

The tablediff utility is new to **SQL Server 2005** and was orginally designed to help ***verify / troubleshoot*** **SQL Server** replication routines. It can however be used in any scenario where the data in two tables should be the same or very similar.

The syntax for ***tablediff*** looks quite daunting so we will first create an example from scratch and then at the end of the article give you the full syntax so you can explore further.

tablediff example from scratch

First lets create a basic table and add some dummy data to it:

```
CREATE TABLE Customers (
 CustomerID int IDENTITY(1,1) NOT NULL,
 Name varchar(50),
 Telephone varchar(30)
)
```

Add some records to the table:

```
INSERT INTO Customers VALUES ('Joe Bloggs', '020 2999999')
INSERT INTO Customers VALUES ('Bob Marley', '032 6595959')
INSERT INTO Customers VALUES ('Jimi Hendrix', '020 1141414')
```

Make a copy of this table and give it the name of **CustomersBackup** using the **SELECT INTO** statement:

```
SELECT *
INTO CustomersBackup
FROM Customers
```

Now change some data in the **CustomersBackup** [gs table]:

```
UPDATE CustomersBackup SET Name = 'Bob Hope' WHERE Name = 'Bob Marley'
UPDATE CustomersBackup SET Telephone = '999 838383' WHERE Name = 'Jimi Hendrix'
```

Now we have two tables with the same structure that each hold 3 records. 2 of the records in **CustomersBackup** have had the data in them changed. Now lets use **tablediff** to:

- Compare the two tables
- Tell us where any differences occur in the data
- Generate the **SQL** to bring the data in the two tables the same - note SQL code is not executed, it will just be written for us to execute if we wish.

First in the command prompt navigate to the directory where the tablediff executable file resides. You may need to change the drive letter below::

```
C:\Windows>cd C:\Program Files\Microsoft SQL Server\90\COM
```

Now execute the following command (**note that the following command should all be on one line of the command prompt**, it is split onto different lines below to help readability).

```
tablediff -sourceserver "MYSERVER\SQL2005" -sourcedatabase "MyTestDB"
-sourcetable "Customers" -destinationserver "MYSERVER\SQL2005"
-destinationdatabase "MyTestDB" -destinationtable "CustomersBackup"
-f C:\tablediff_Results.txt
```

You can see above how the majority of the parameters to the tablediff command specify the source and destination (the location of the two tables to compare). The **-f** parameter specifies the filename of the output file that is generated that contains the **SQL** statements that can be run to '*synchronise*' the data in the two tables.

Navigate to the file that you specified in the **-f** parameter above and when you open it you should see the following code which you can run if you wish from a query window in **SSMS** to update the **CustomersBackup** table so that it contains the same data as the **Customers** table.

```
-- Host: MYSERVER\SQL2005
-- Database: [MyTestDB]
-- Table: [dbo].[CustomersBackup]
SET IDENTITY_INSERT [dbo].[CustomersBackup] ON
UPDATE [dbo].[CustomersBackup] SET [Name]='Bob Marley' WHERE [CustomerID] = 2
UPDATE [dbo].[CustomersBackup] SET [Telephone]='020 1141414' WHERE [CustomerID]
= 3
SET IDENTITY_INSERT [dbo].[CustomersBackup] OFF
```

tablediff syntax

The full syntax for the tablediff command is as follows:

```
tablediff
```

```
[ -? ] |
{
        -sourceserver source_server_name[\instance_name]
        -sourcedatabase source_database
        -sourcetable source_table_name
    [ -sourceschema source_schema_name ]
    [ -sourcepassword source_password ]
    [ -sourceuser source_login ]
    [ -sourcelocked ]
        -destinationserver destination_server_name[\instance_name]
        -destinationdatabase subscription_database
        -destinationtable destination_table
    [ -destinationschema destination_schema_name ]
    [ -destinationpassword destination_password ]
    [ -destinationuser destination_login ]
    [ -destinationlocked ]
    [ -b large_object_bytes ]
    [ -bf number_of_statements ]
    [ -c ]
    [ -dt ]
    [ -et table_name ]
    [ -f [ file_name ] ]
    [ -o output_file_name ]
    [ -q ]
    [ -rc number_of_retries ]
    [ -ri retry_interval ]
    [ -strict ]
    [ -t connection_timeouts ]
}
```

For more details on the various parameters listed above see Microsoft Technet.

Conclusion

In this article we have demonstrated the purpose of the tablediff command line utitlity that is new to **SQL Server 2005**. We have demonstrated from scratch, a real world example of the command in use and then seen the full syntax of the command which shows many additional optional parameters.