From: honda@luckstar.com.tw

Date: 2012.04.15

Subject: 如何建立只有日期沒有時間的欄位

說明

若(sql server的版本 >= 2008), 則(直接使用新資料型態date),

否則

可以自訂資料型態如下. 但建議還是

1 以8位數整數(INT)或文數字(CHAR 8)存檔. 雖然這樣比較占空間也比較容易出錯, 但是可以直覺處理.

2自行以 YYYY-MM-DD 00:00:00的格式定義處理.

/\* 自訂日期與時間型態, 其實都是dateTime \*/

create type Date from dateTime

create type Time from dateTime

/\* 自訂日期規則, 固定時間為00:00:00 \*/

create rule DateOnlyRule as dateAdd(dd,datediff(dd,0,@DateTime),0) = @DateTime

/\* 自訂時間規則, 固定為1900-01-01 \*/

create rule TimeOnlyRule as datediff(dd,0,@DateTime) = 0

/\* bind the new data types. \*/

EXEC sp\_bindrule 'DateOnlyRule', 'Date'

EXEC sp\_bindrule 'TimeOnlyRule', 'Time'

/\* 範例 \*/

create table Table1

 (

 FID int identity primary key,

 FAmount money not null,

 FDate Date not null,

 FTime Time not null

 )

insert into Table1 (FAmount, FDate, FTime)

 values (200, '2005-01-01', '09:00:00 AM')

insert into Table1 (FAmount, TDate, TTime)

 values (400, '2005-01-03', '7:50:30 PM')

參考:

In this post, I showed a simple way to create simple but useful Date and Time user-defined data types in SQL Server 2000. Here's how to do it in SQL Server 2005, without the need for CLR types or anything fancy.

First, we must create two user defined data types:

create type Date from dateTime

 create type Time from dateTime

So, internally (and externally to our clients), these types are really just DateTime. But, we will apply some rules to these types so that the Date data type will always be constrained to a time exactly at midnight, and the the Time data type will always be at the "base date", or 1900-01-01:

create rule DateOnlyRule as

 dateAdd(dd,datediff(dd,0,@DateTime),0) = @DateTime

go

create rule TimeOnlyRule as

 datediff(dd,0,@DateTime) = 0

go

Finally, we call sp\_bindrule to bind the rules to the data types:

EXEC sp\_bindrule 'DateOnlyRule', 'Date'

 EXEC sp\_bindrule 'TimeOnlyRule', 'Time'

That's it! Now, we can create a table that uses our brand-new data types:

create table Trans

 (

 TranID int identity primary key,

 TranAmount money not null,

 TranDate Date not null,

 TranTime Time not null

 )

Notice that TranDate and TranTime are two separate columns here. If we try to insert data, our rules will ensure that our TranDate and TranTime columns only contain the appropriate data, and by doing so we can simply calculate TranDate + TranTime to get the transactions DateTime.

The following inserts will succeed:

insert into Trans (TranAmount, TranDate, TranTime)

 values (200, '2005-01-01', '09:00:00 AM')

insert into Trans (TranAmount, TranDate, TranTime)

 values (400, '2005-01-03', '7:50:30 PM')

But this will fail, since our rules are violated (getdate() returns the current date and the time):

insert into Trans (TranAmount, TranDate, TranTime)

 values (200, getdate(), getdate())

Msg 513, Level 16, State 0, Line 1

 A column insert or update conflicts with a rule imposed by a previous CREATE RULE statement. The statement was terminated. The conflict occurred in database 'PlayGround', table 'dbo.Trans', column 'TranDate'.

 The statement has been terminated.

Finally, we can select from our table and simply add the two columns together to get each transaction's DateTime:

select \*, TranDate + TranTime as TranDateTime

 from Trans

TranID TranAmount TranDate TranTime TranDateTime

 ------- ----------- ------------------- ------------------- -------------------

 1 200.00 2005-01-01 00:00:00 1900-01-01 09:00:00 2005-01-01 09:00:00

 2 400.00 2005-01-03 00:00:00 1900-01-01 19:50:30 2005-01-03 19:50:30

(2 row(s) affected)

The nice advantage of doing this is that you can quickly and efficiently select all transactions for a single day like this:

select \* from Trans

 where TranDate = '2005-01-01'

If we stored the Date and Time in the same column, we'd need to use a less efficient range:

select \* from Trans

 where TranDate >= '2005-01-01' and TranDate < '2005-01-02'

SQL 2008 does finally implement true Date and Time data types, but until then, I hope this gives you some ideas to play with.

Notes:

 •If you do wish to break apart DateTime data into two columns, another option is to simply use a single, standard DateTime column but then define two extra computed columns in your table, one that returns just the Date, another that returns just the Time. Then, you can index those computed columns and reference them easily and efficiently. If you truly wish to store just a date or just a time, however, these user defined data types are a great way to do it. At the end of this SQLTeam article there's a good example demonstrating this.

•You could also just use the expressions for the rules in CHECK constrains on individual table columns, instead of creating the user defined data types and rules. As Books On Line states, CREATE RULE will at some point be removed from future versions of SQL Server, so you may wish to avoid using it.