SQL Server 2005 - Hack encrypted data by password

As you know, password encryption is a basic data encryption method that only uses passwords and can decrypt with the same password. Now let's suppose we forgot the password we set and need to restore the data as it was.

In part 1 of this series, we introduced a password encryption and decoding method. This part 2 will go into how to hack that data back.

As you know, password encryption is a basic data encryption method that only uses passwords and can decrypt with the same password. Now let's suppose we forgot the password we set and need to restore the data as it was.

Step 1

Encrypt data by password encryption method

```
select EncryptedData = EncryptByPassPhrase ('MAK', '123456789')
```

Result

```
EncryptedData
---------------------------------------------- ----------------------
0x01000000F75D553409C74570F6DDBCADA53FD489DDD52D9277010050565ADF30F244F8CC
```

Step 2

Create user procedures to restore chained data. This procedure will use the DecryptByPassPhrase function to decrypt the data and display it on the password.

```
USE [Master]
GO

/ ****** Object: StoredProcedure [dbo]. [Hack_encryption] Script Date: 12/18/2007 18:18:36
****** /
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID (N '[dbo].
[Hack_encryption]')
AND type in (N'P ', N'PC'))
DROP PROCEDURE [dbo]. [Hack_encryption]
GO
```

```
??t nocount trên
SET CONCAT_NULL_YIELDS_NULL OFF
go
USE [Master]
GO

/ ****** Object: StoredProcedure [dbo]. [Hack_encryption] Script Date: 12/18/2007 18:18:55
****** /
SET ANSI_NULLS ON
GO

SET QUOTED_IDENTIFIER ON
GO

CREATE procedure [dbo]. [Hack_encryption] @encryptedtext varbinary (max)
as
declare @password varchar (6)
declare @i int
declare @j int
declare @k int
declare @l int
declare @m int
declare @n int


set @ i = -1
set @ j = -1
set @ k = -1
set @ l = -1
set @ m = -1
set @ n = -1
set @password = ''

while @i 255
begin
while @j 255
begin
while @k 255
begin
while @l 255
begin
while @m 255
begin
while @n = 255
begin
set @ password = isnull (char (@i), '')
+ isnull (char (@j), '')
+ isnull (char (@k), '') + isnull (char (@l), '')
```

```
+ isnull (char (@m), '') + isnull (char (@n), '')
if convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)),
@encryptedtext)) is not null
begin
print 'This is the Encrypted text:' + @ password
set @ i = 256; set @ j = 256; set @ k = 256; set @ l = 256; set @ m = 256; set @ n = 256;
print 'The actual data is:' + convert (varchar (100),
DecryptByPassPhrase (ltrim (rtrim (@password)), @encryptedtext))
end
--print 'A' + ltrim (rtrim (@password)) + 'B'
--print convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)), @ encryptedtext))
set @ n = @ n + 1
end
set @ n = 0
set @ m = @ m + 1
end
set @ m = 0
set @ l = @ l + 1
end
set @ l = 0
set @ k = @ k + 1
end
set @ k = 0
set @ j = @ j + 1
end
set @ j = 0
set @ i = @ i + 1
end


GO
```

Step 3

Assume that you forgot the password used to encrypt the data to '*0x01000000F75D553409C74570F6DDBCADA53FD489DDD52D9277010050565ADF30F244F8CC* '. We can retrieve the password and encrypted data using the following procedure

```
use master
go
select getdate () as StartingTime
go
declare @myencryptedtext varbinary (max)
set @ myencryptedtext = 0x01000000F75D553409C74570F6DDBCADA53FD489DDD52D9277010050565ADF30F244F8CC
print @myencryptedtext
exec hack_encryption @ encryptedtext = @ myencryptedtext
```

go
select getdate () as EndingTime
go

Result

StartingTime

2007-12-18 18:24: 10.843

0x01000000F75D553409C74570F6DDBCADA53FD489DDD52D9277010050565ADF30F244F8CC
This is the Encrypted text: MAK
T?p tin d? li?u là: 123456789

EndingTime

2007-12-18 18:26: 36.080

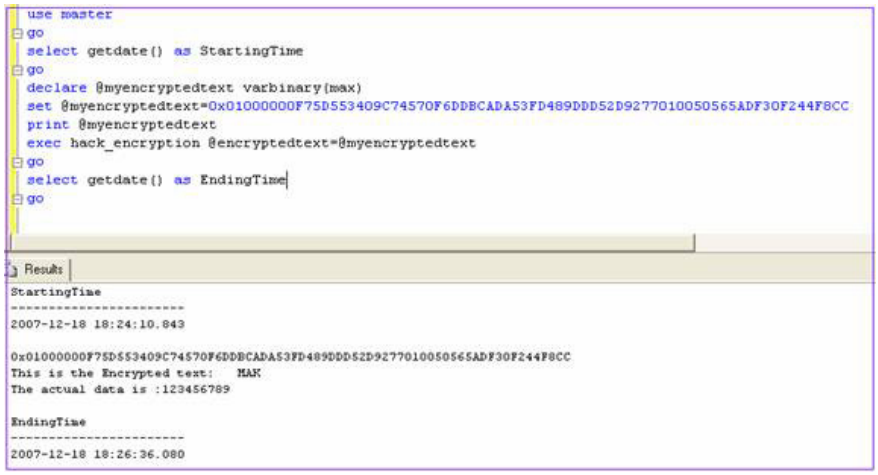


Figure 1

As you can see in the result (Figure 1), it only takes 2 minutes to retrieve the data and password. Basically, this procedure repeats all possible probabilities of ASCII characters over 6 characters long to find the password and use it to decrypt the data.

Creating a procedure will not help much when the encrypted data is in a table. So we have to change this procedure to a scalar function as shown below

Step 1

Create the procedure as follows

```
USE [master]
GO

/ ****** Object: UserDefinedFunction [dbo]. [Hack_encryption_password] Script Date: 12/18/2007 18:36:29 ****** /
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID (N '[dbo]. [Hack_encryption_password]')
AND type in (N'FN ', N'IF', N'TF ', N'FS', N'FT '))
DROP FUNCTION [dbo]. [Hack_encryption_password]
GO
use [Master]
go

CREATE function [dbo]. [Hack_encryption_password] (@encryptedtext varbinary (max))
returns varchar (6)
v?i th?c hi?n nh? m?t caller
as
begin
declare @password varchar (6)
declare @i int
declare @j int
declare @k int
declare @l int
declare @m int
declare @n int


set @ i = -1
set @ j = -1
set @ k = -1
set @ l = -1
set @ m = -1
set @ n = -1
set @password = ''

while @i 255
begin
while @j 255
begin
while @k 255
begin
while @l 255
begin
while @m 255
begin
while @n = 255
begin
```

```
set @ password = isnull (char (@i), '') + isnull (char (@j), '')
+ isnull (char (@k), '') + isnull (char (@l), '')
+ isnull (char (@m), '') + isnull (char (@n), '')
if convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)),
@encryptedtext)) is not null
begin
--print 'This is the Encrypted text:' + @ password
set @ i = 256; set @ j = 256; set @ k = 256; set @ l = 256; set @ m = 256; set @ n = 256;
--print 'The actual data is:' + convert (varchar (100),
DecryptByPassPhrase (ltrim (rtrim (@password)), @ encryptedtext))
end
--print 'A' + ltrim (rtrim (@password)) + 'B'
--print convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)), @ encryptedtext))
set @ n = @ n + 1
end
set @ n = 0
set @ m = @ m + 1
end
set @ m = 0
set @ l = @ l + 1
end
set @ l = 0
set @ k = @ k + 1
end
set @ k = 0
set @ j = @ j + 1
end
set @ j = 0
set @ i = @ i + 1
end

return @password
END
```

Step 2

Create a table with encrypted data

```
USE [tempdb]
GO
/ ****** Object: Table [dbo]. [MyTable] Script Date: 12/18/2007 18:44:40 ****** /
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID (N '[dbo]. [MyTable]') AND type in (N'U '))
DROP TABLE [dbo]. [MyTable]
GO
create table MyTable (int, encrypteddata varbinary (max))
go
```

```
insert into MyTable select 1, EncryptByPassPhrase ('Do', '1112228333')
insert into MyTable select 2, EncryptByPassPhrase ('Re', '1212223833')
insert into MyTable select 3, EncryptByPassPhrase ('Me', '1132223393')
insert into MyTable select 4, EncryptByPassPhrase ('Fa', '1114223383')
insert into MyTable select 5, EncryptByPassPhrase ('So', '1112523333')
insert into MyTable select 6, EncryptByPassPhrase ('La', '1112263373')
insert into MyTable select 7, EncryptByPassPhrase ('Si', '1112227338')
go
```

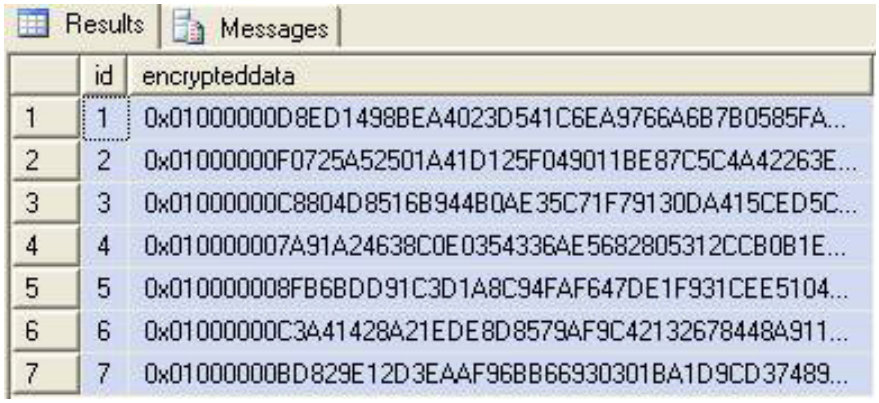
Step 3

Query data using the following SQL statement

```
Select * from MyTable
```

You will see the data displayed as follows (Figure 2)

```
1 0x01000000D8ED1498BEA4023D541C6EA9766A6B7B0585FAE91B942C88C23677550C6FD7FA
2 0x01000000F0725A52501A41D125F049011BE87C5C4A42263E7538B837B8278ADEE5FC2678
3 0x01000000C8804D8516B944B0AE35C71F79130DA415CED5CCF58E522692AC749115EEF0D9
4 0x010000007A91A24638C0E0354336AE5682805312CCB0B1E6BBACB6D9E65DC5D9DA73906E
5 0x010000008FB6BDD91C3D1A8C94FAF647DE1F931CEE5104045BD03DE4E809565E74604DF3
6 0x01000000C3A41428A21EDE8D8579AF9C42132678448A9113A31A869276A7631A58A32BE3
7 0x01000000BD829E12D3EAAF96BB66930301BA1D9CD3748946F354301922A03AE49047FE00
```



	id	encrypteddata
1	1	0x01000000D8ED1498BEA4023D541C6EA9766A6B7B0585FA...
2	2	0x01000000F0725A52501A41D125F049011BE87C5C4A42263E...
3	3	0x01000000C8804D8516B944B0AE35C71F79130DA415CED5C...
4	4	0x010000007A91A24638C0E0354336AE5682805312CCB0B1E...
5	5	0x010000008FB6BDD91C3D1A8C94FAF647DE1F931CEE5104...
6	6	0x01000000C3A41428A21EDE8D8579AF9C42132678448A911...
7	7	0x01000000BD829E12D3EAAF96BB66930301BA1D9CD37489...

Figure 2

Step 4

Use *hack_encryption_password* function to recover all passwords from the encrypted data in MyTable table. Execute the following SQL statement

```
select ID, master. [dbo]. [hack_encryption_password] (encrypteddata) as Password from MyTable
```

You will see the following result (Figure 3)

1 Do
2 Re
3 Me
4 Fa
5 So
6 La
7 Si

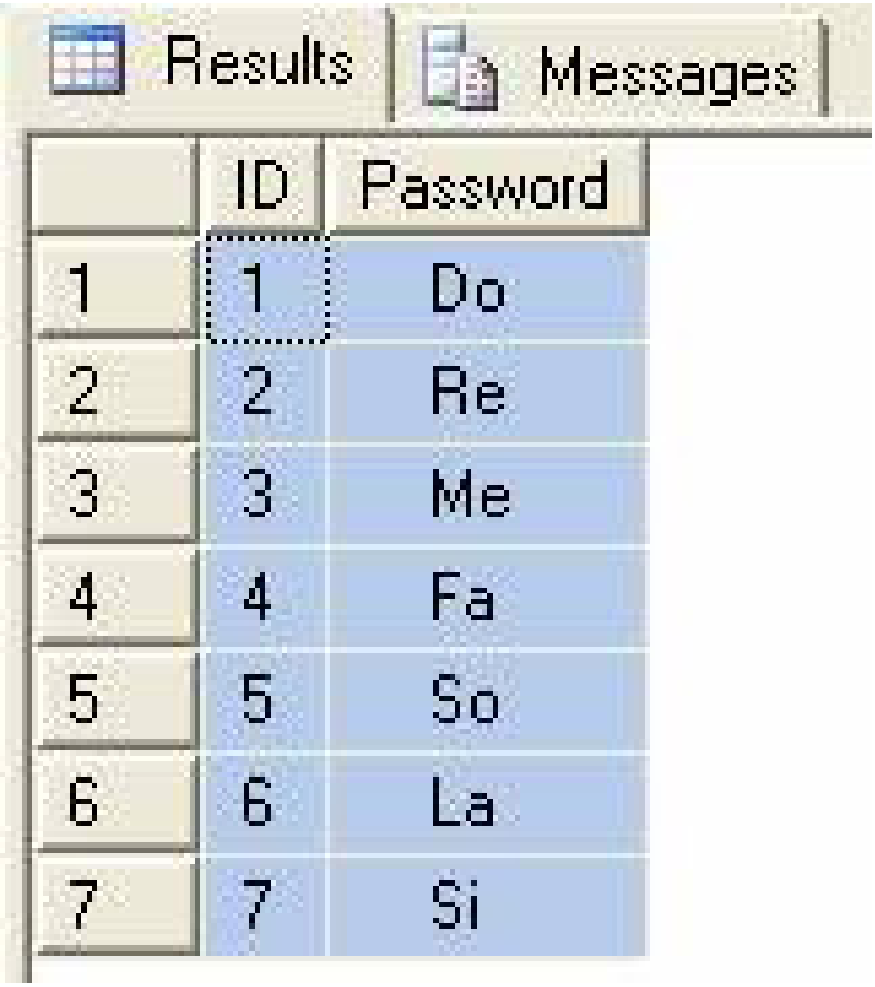


Figure 3

The above function can be edited to return the encrypted data, as follows

Step 1

Create the following function

```
USE [master]
GO

/ ****** Object: UserDefinedFunction [dbo]. [Hack_encryption_password] Script Date: 12/18/2007 18:36:29 ****** /
IF EXISTS (SELECT * FROM sys.objects WHERE object_id = OBJECT_ID (N '[dbo]. [Hack_encryption_data]')
AND type in (N'FN ', N'IF', N'TF ', N'FS', N'FT '))
DROP FUNCTION [dbo]. [Hack_encryption_data]
GO
use [Master]
go

CREATE function [dbo]. [Hack_encryption_data] (@encryptedtext varbinary (max))
returns varchar (8000)
v?i th?c hi?n nh? m?t caller
as
begin
declare @data varchar (8000)
declare @password varchar (6)
declare @i int
declare @j int
declare @k int
declare @l int
declare @m int
declare @n int


set @ i = -1
set @ j = -1
set @ k = -1
set @ l = -1
set @ m = -1
set @ n = -1
set @password = ''

while @i 255
begin
while @j 255
begin
while @k 255
begin
while @l 255
begin
while @m 255
begin
while @n = 255
```

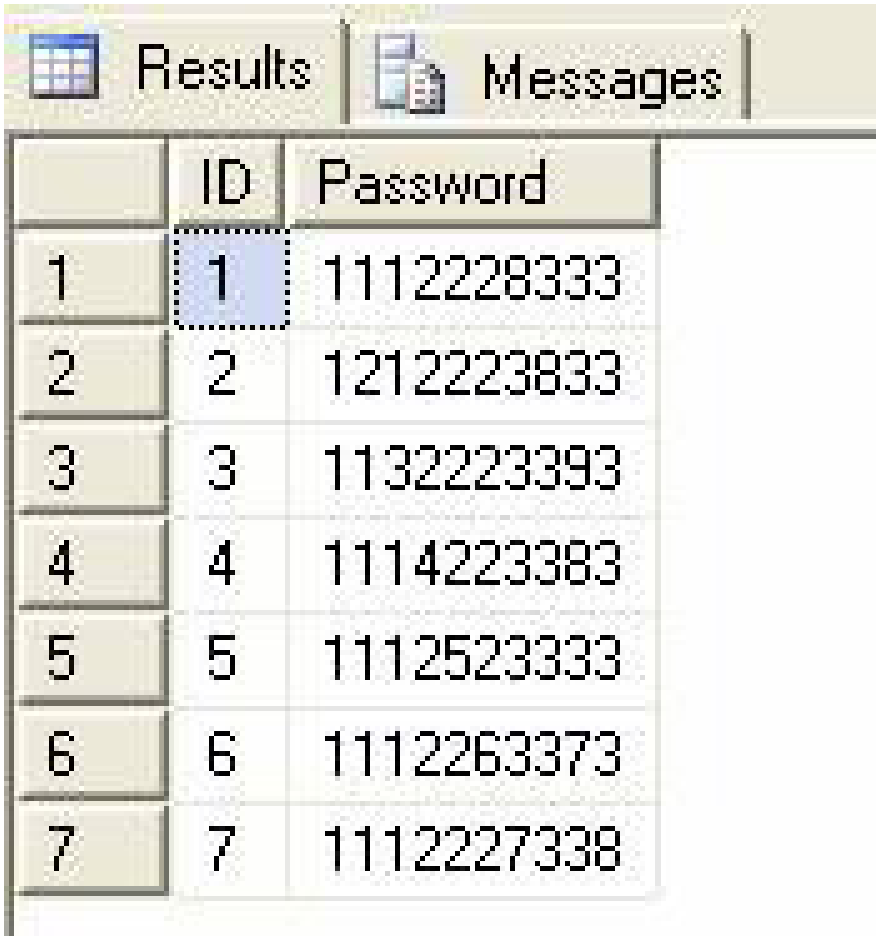
```
begin
set @ password = isnull (char (@i), '') + isnull (char (@j), '') + isnull (char (@k), '')
+ isnull (char (@l), '') + isnull (char (@m), '') + isnull (char (@n), '')
if convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)),
@encryptedtext)) is not null
begin
--print 'This is the Encrypted text:' + @ password
set @ i = 256; set @ j = 256; set @ k = 256; set @ l = 256; set @ m = 256; set @ n = 256;
set @data = convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)), @
encryptedtext))
end
--print 'A' + ltrim (rtrim (@password)) + 'B'
--print convert (varchar (100), DecryptByPassPhrase (ltrim (rtrim (@password)), @ encryptedtext))
set @ n = @ n + 1
end
set @ n = 0
set @ m = @ m + 1
end
set @ m = 0
set @ l = @ l + 1
end
set @ l = 0
set @ k = @ k + 1
end
set @ k = 0
set @ j = @ j + 1
end
set @ j = 0
set @ i = @ i + 1
end

return @data
END
```

Step 2

Decrypt the data using the created function

```
select ID, master. [dbo]. [hack_encryption_data] (encrypteddata) as Password from MyTable
```

The result is shown in Figure 4



	ID	Password
1	1	1112228333
2	2	1212223833
3	3	1132223393
4	4	1114223383
5	5	1112523333
6	6	1112263373
7	7	1112227338

Figure 4

Note :

1. Procedures and functions can only hack for 6-character passwords.
2. This procedure and function can take up a lot of CPU to retrieve data and retrieve the password