SQL Server 2012 – Trabalhando com Sequences e comparações com IDENTITY

Posted by Dirceu Resende on 28 de abril de 2016

Olá pessoal!
Bom dia!

Hoje me deparei com uma situação em que a utilização das sequences, velhas conhecidas minhas do Oracle e que a partir do SQL Server 2012 foram introduzidas no SGBD da Microsoft, vi uma solução perfeita para o problema que eu estava enfrentando.

Introdução – Detalhando o Problema

Esse problema consistia em uma tabela com dezenas de milhões de registros e que possui um número sequencial para controle e unicidade do registro e agora surgiu uma necessidade de criar um outro número sequencial nessa mesma tabela, independente, onde os registros existentes deveriam receber NULL e que seria iniciado em 1, para controlar um outro tipo de informação e sendo preenchido apenas quando um determinado evento ocorrer (diferente do sequencial já existente, que é gerado a cada inserção na tabela)

Uma outra situação que deve ser evitado à todo custo, é que nunca duas sessões podem pegar o mesmo sequencial e gravar na tabela e essa possibilidade é bem possível, já que essa tabela possui várias inserções em sessões paralelas por segundo.

Nesse caso, a utilização do IDENTITY não seria possível, uma vez que o IDENTITY é aplicado em todos os registros da tabela, o que não seria o caso. A utilização de estruturas de ranking como o ROW_NUMBER até poderia ser viável, se não fosse tão pesado calcular esse ranking a cada nova inserção (além da possibilidade de duas sessões pegarem o mesmo sequencial).

Diferenças entre Sequence e IDENTITY

SEQUENCE	IDENTITY
É um objeto independente, que pode ser utilizado pra preencher qualquer coluna (inclusive, mais de uma na mesma tabela) do tipo numero inteiro (int, bigit, smallint, tinyint, decimal com escala 0 ou numeric com escala 0), de uma ou mais tabelas	É associado a uma coluna de uma tabela
É populada quando for chamada. Ou seja, pode ser populada a cada inserção ou apenas quando alguma condição for atendida	É populada em cada inserção
Ao ser iniciada em uma tabela já populada, os registros anteriores não serão alterados	Ao ser iniciada na tabela, a coluna inteira é populada com o sequencial
Deve ser chamado manualmente para gerar o sequencial	O sequencial é gerado automaticamente
Possui permissões à parte	Não requer permissões adicionais além da tabela
Pode ser definido valor mínimo e máximo (Ex: De 1 a 100)	O valor máximo é o limite do tipo de dado da coluna
O sequencial pode ser reiniciado automaticamente ao atingir o valor máximo (parâmetro CYCLE)	Ao atingir o valor máximo, não é possível inserir mais registros
Pode ser gerado um novo sequencial em comandos de UPDATE, caso necessário	O sequencial é gerado apenas no INSERT dos dados
Disponível a partir do SQL Server 2012	Disponível a partir do SQL Server 6.0 (SQL 95)
O valor atual do sequencial pode ser O valor atual do sequencial pode ser consultando através da view sys.sequences	consultando através da view sys.identity_columns
O valor da sequencia pode ser reiniciado	O valor da sequencia NÃO pode ser reiniciado

Como criar uma sequence

Para resolver o problema descrito acima, tive a ideia de criar uma sequence, que é um objeto do banco de dados especialmente criado pra esse tipo de necessidade. Diferente do IDENTITY, você pode utilizar mais de uma sequence na tabela e os registros anteriores não são alterados. Estes foram os motivos que me levaram a utilizar esse recurso do SQL Server para resolver essa situação.

Vamos ver agora como criar uma sequence:

```
1 CREATE SEQUENCE dbo.[seq_Teste]
2 AS [INT]
3     START WITH 1
4     INCREMENT BY 1
5     MINVALUE 1
6     MAXVALUE 999999999
7     CYCLE
8     CACHE
9 GO
```

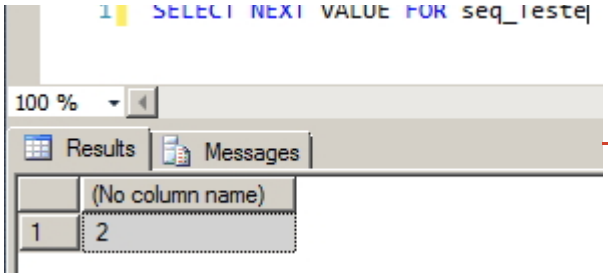
Os parâmetros explicados:

- START WITH: Define qual o número inicial da sequencia
- MINVALUE e MAXVALUE: Delimitam o limite da SEQUENCE com seu respectivo valor máximo e mínimo. Caso o valor não seja definido, será atribuído o valor máximo e míni do tipo de dado escolhido.
- INCREMENT BY: Define a quantidade que será incrementada na sequencia. No exemplo acima, será realizado o incremento de 1 em 1.
- CYCLE: A propriedade CYCLE permite começar novamente um ciclo a partir do momento que a propriedade MINVALUE e MAXVALUE for atingida. Ou seja, ao atingir o valor definido em MAXVALUE, a sequencia será iniciada novamente no valor do parâmetro MINVALUE (quando isso ocorre, serão gerados valores duplicados na sequencia, pois o range todo já foi percorrido)
- CACHE: Ao utilizar esse parâmetro, o SQL Server pré-aloca os números sequencias pela propriedade CACHE, sendo que o valor padrão para esta é 15, significando que valores os próximos 15 valores disponíveis serão alocados na memória até que sejam utilizados e a sequence já trata esses números como utilizados. Quando todos os números do CACHE são utilizados, 15 novos valores são alocados na memória novamente e assim segue o ciclo. Vale lembrar que se a instância for reiniciada, os números que estão no cache são perdidos e fica esse "buraco" na sequencia.

Como retornar o próximo número de uma sequence

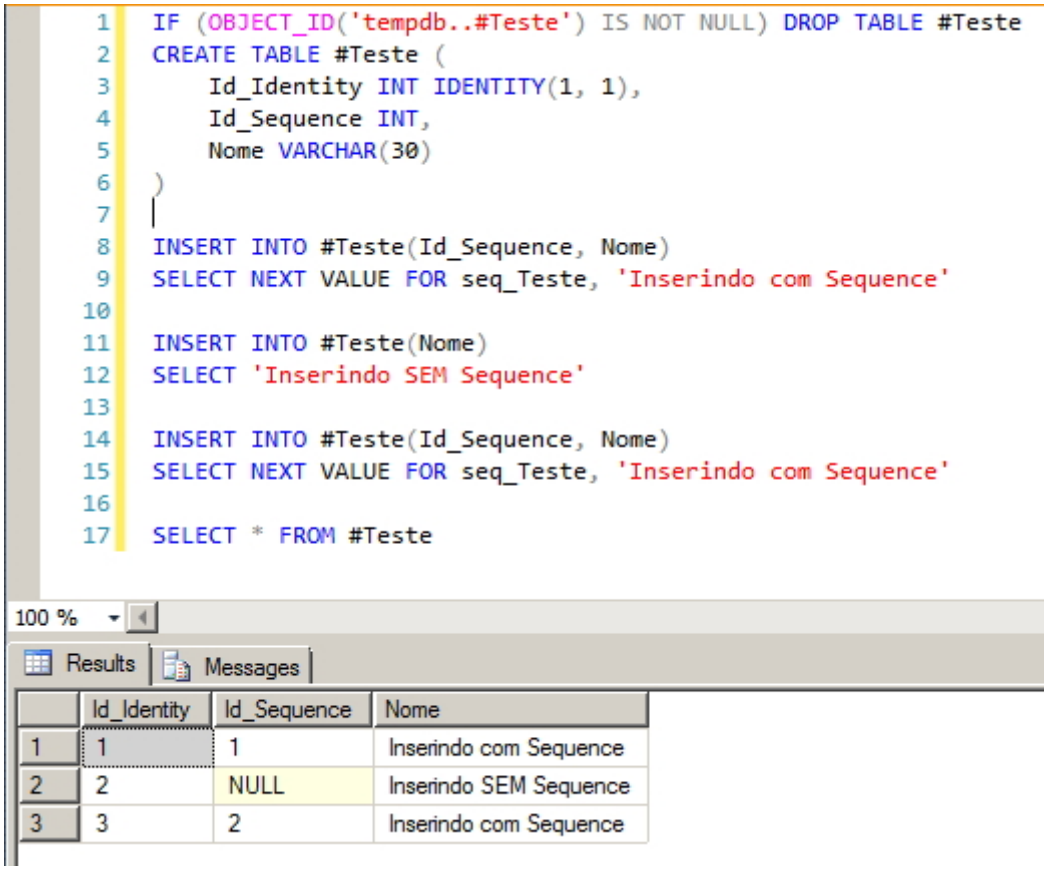
Retornar o próximo número de uma sequence é uma tarefa muito simples:

```
1 SELECT NEXT VALUE FOR seq_Teste
```



(https://www.dirceuresende.com/wp-content/uploads/2016/04/SQL-Server-Sequence-Next-Value-For.jpg)

Entretanto, você deve ter em vista que diferente do IDENTITY, a cada inserção você deve chamar a sequence para retornar o sequencial e assim inserir na tabela ou então criar uma default constraint para automatizar essa tarefa.



(https://www.dirceuresende.com/wp-content/uploads/2016/04/SQL-Server-Sequence-x-Identity.jpg)

Como retornar o próximo número de uma sequence de forma automática

Embora a maioria das pessoas (eu, inclusive) utilize a sequence manualmente a cada inserção para retornar o próximo sequencial, isso pode ser automatizado utilizando uma DEFAULT CONSTRAINT na tabela:

```
CREATE TABLE dbo.Teste_Sequence (
    Id INT DEFAULT NEXT VALUE FOR dbo.seq_Teste,
    Nome VARCHAR(100)
)
```

Exemplo:

```
IF (OBJECT_ID('dbo.Teste_Sequence') IS NOT NULL) DROP TABLE dbo.Teste_Sequence
IF (OBJECT_ID('dbo.seq_Teste') IS NOT NULL) DROP SEQUENCE dbo.seq_Teste

CREATE SEQUENCE seq_Teste START WITH 1 INCREMENT BY 1

CREATE TABLE dbo.Teste_Sequence (
    Id INT DEFAULT NEXT VALUE FOR dbo.seq_Teste,
    Nome VARCHAR(100)
)

INSERT INTO dbo.Teste_Sequence (Nome)
SELECT 'Inserção automática'

INSERT INTO dbo.Teste_Sequence (Id, Nome)
SELECT NULL, 'Id NULL inserido manualmente'

INSERT INTO dbo.Teste_Sequence (Nome)
SELECT 'Inserção automática'

INSERT INTO dbo.Teste_Sequence (Nome)
SELECT 'Inserção automática'

SELECT * FROM dbo.Teste_Sequence
```

100 %

Results | Messages

	Id	Nome
1	1	Inserção automática
2	NULL	Id NULL inserido manualmente
3	2	Inserção automática
4	3	Inserção automática

(https://www.dirceuresende.com/wp-content/uploads/2016/04/SQL-Server-Sequence-NEXT-VALUE-FOR-Automatic.jpg)

Como reiniciar o valor da sequence

Em alguns momentos, é necessário que o contador da sequence seja reiniciado ou alterado para um determinado valor especifico. Para isso, podemos utilizar o ALTER SEQUENCE para essa tarefa:

```
ALTER SEQUENCE seq_Teste RESTART WITH 1
```

No exemplo acima, estamos reiniciando o valor da sequencia para 1.

Exemplo:

```
1  IF (OBJECT_ID('tempdb..#Teste') IS NOT NULL) DROP TABLE #Teste
2  CREATE TABLE #Teste (
3      Id_Identity INT IDENTITY(1, 1),
4      Id_Sequence INT,
5      Nome VARCHAR(30)
6  )
7
8  INSERT INTO #Teste(Id_Sequence, Nome)
9  SELECT NEXT VALUE FOR seq_Teste, 'Inserindo com Sequence'
10
11 INSERT INTO #Teste(Id_Sequence, Nome)
12 SELECT NEXT VALUE FOR seq_Teste, 'Inserindo com Sequence'
13
14 INSERT INTO #Teste(Id_Sequence, Nome)
15 SELECT NEXT VALUE FOR seq_Teste, 'Inserindo com Sequence'
16
17 ALTER SEQUENCE seq_Teste RESTART WITH 1
18
19 INSERT INTO #Teste(Id_Sequence, Nome)
20 SELECT NEXT VALUE FOR seq_Teste, 'Inserindo com Sequence'
21
22 INSERT INTO #Teste(Id_Sequence, Nome)
23 SELECT NEXT VALUE FOR seq_Teste, 'Inserindo com Sequence'
24
25 SELECT * FROM #Teste
```

100 %

Results | Messages

	Id_Identity	Id_Sequence	Nome
1	1	1	Inserindo com Sequence
2	2	2	Inserindo com Sequence
3	3	3	Inserindo com Sequence
4	4	1	Inserindo com Sequence
5	5	2	Inserindo com Sequence

(https://www.dirceuresende.com/wp-content/uploads/2016/04/SQL-Server-Sequence-Restart.jpg)

Recuperando o valor atual da sequence, sem aumentar a sequencia

Para recuperar o valor atual da sequence basta realizar uma consulta da view sys.sequences:

```
1 SELECT current_value
2 FROM sys.sequences
3 WHERE name = 'seq_Teste'
```

```
SELECT current_value
FROM sys.sequences
WHERE name = 'seq_Teste'
```

100 %

Results | Messages

	current_value
1	2

(https://www.dirceuresende.com/wp-content/uploads/2016/04/SQL-Server-Sequence-Current-Value.jpg)

Como alterar uma sequence

A alteração de uma sequence segue os mesmos parâmetros da criação, podendo ser alterada a qualquer momento.

Exemplos:

```
ALTER SEQUENCE seq_Teste MAXVALUE 99999
ALTER SEQUENCE seq_Teste CACHE
```

Como apagar uma sequence

A remoção de uma sequence no banco de dados SQL Server é simples como a de qualquer outro objeto de banco de dados e pode ser feita utilizando a instrução DROP:

```
DROP SEQUENCE dbo.Sua_Sequence
GO
```

Permissões da sequence

Como já comentado, a sequence são objetos independentes no banco de dados e por tanto, possuem permissões independentes também:

CREATE SEQUENCE: Para se criar uma sequence, é necessário ter permissão de CREATE SEQUENCE, ALTER ou CONTROL no schema. Os usuários da role db_owner e db_ddladmin podem criar, alterar e dropar sequences e os usuários das roles db_owner e db_datawriter podem utilizar a sequence para retornar o próximo número da sequência.

Exemplo de grant:

```
GRANT CREATE SEQUENCE ON SCHEMA::dbo TO [DOMINIO\usuario]
```

ALTER SEQUENCE: Para se alterar uma sequence, é necessário ter permissão de ALTER no schema.

Exemplo de grant:

```
GRANT ALTER ON OBJECT::dbo.Sua_Sequence TO [DOMINIO\usuario]
```

DROP SEQUENCE: Para se apagar uma sequence, é necessário ter permissão de ALTER ou CONTROL no schema.

Demonstração de uma única sequence para mais de uma tabela

Código-fonte do teste

```
CREATE SEQUENCE dbo.[seq_Pessoa]
AS [INT]
    START WITH 1
    INCREMENT BY 1
    MINVALUE 1
    MAXVALUE 999999999
    CYCLE
    CACHE
GO

CREATE TABLE dbo.Pessoa_Fisica (
    Id INT DEFAULT NEXT VALUE FOR dbo.seq_Pessoa,
    Nome VARCHAR(100),
    CPF VARCHAR(11)
)

CREATE TABLE dbo.Pessoa_Juridica (
    Id INT DEFAULT NEXT VALUE FOR dbo.seq_Pessoa,
    Nome VARCHAR(100),
    CNPJ VARCHAR(14)
)

INSERT INTO dbo.Pessoa_Fisica (Nome, CPF)
VALUES('Dirceu Resende', '11111111111')

INSERT INTO dbo.Pessoa_Juridica (Nome, CNPJ)
VALUES('Dirceu Resende Ltda', '22222222222222')

INSERT INTO dbo.Pessoa_Fisica (Nome, CPF)
VALUES('Dirceu Resende 2', '33333333333')

INSERT INTO dbo.Pessoa_Juridica (Nome, CNPJ)
VALUES('Dirceu Resende ME', '44444444444444')

SELECT * FROM dbo.Pessoa_Fisica
SELECT * FROM dbo.Pessoa_Juridica
```

Resultado

Results | Messages

	Id	Nome	CPF
1	1	Dirceu Resende	11111111111
2	3	Dirceu Resende 2	33333333333

	Id	Nome	CNPJ
1	2	Dirceu Resende Ltda	22222222222222
2	4	Dirceu Resende ME	44444444444444

(https://www.dirceuresende.com/wp-content/uploads/2018/12/SQL-Server-2012-Trabalhando-com-Sequences-e-comparações-com-IDENTITY-1.png)

É isso aí, pessoal!
Espero que tenham gostado e até o próximo post.

Obs: Precisa utilizar uma sequence em uma user defined function, seja scalar, aggregate ou table-valued e não está conseguindo ? Veja a solução no post Utilizando sequences em user defined functions no SQL Server (https://www.dirceuresende.com/blog/utilizando-sequences-em-user-defined-functions-no-sql-server/) ☺

Atenciosamente,

(https://www.youracclaim.com/user/dirceu-resende)

Dirceu Resende
Consultor de Banco de Dados e BI
Microsoft MVP, MCSE, MCSA, MTA, MCP
Whatsapp: @dirceuresende (http://bit.ly/dirceuresende)
Telegram: @dirceuresende (https://t.me/dirceuresende)
Skype: @dirceuresende (skype:dirceuresende?chat)
Consultoria: consultoria@dirceuresende.com
(mailto:consultoria@dirceuresende.com)

sequences / sql / sql server

3 Comments

Esse artigo foi bem interessante para a minha pesquisa. Obrigado pela ajuda!

Joaquim 2 anos ago

Excelente dica, Vithor! Vou adicionar no post!
Obrigado pela visita

Dirceu Resende 3 anos ago

Dirceu, Muito bom!

Creio que você pode complementar dizendo que é possível dar um CREATE TABLE onde um determinado campo ele utiliza uma sequence como o NEXT VALUE, isto é sensacional! Elimina a necessidade de no INSERT ter que fazer a consulta do próximo valor.
Exemplo:

```
ALTER TABLE Test.MyTable
ADD
DEFAULT N'AdvWorks_' +
CAST(NEXT VALUE FOR Test.CounterSeq AS NVARCHAR(20))
FOR IDColumn;
GO

INSERT Test.MyTable (name)
VALUES ('Larry') ;
GO
```

https://msdn.microsoft.com/en-us/library/ff878370.aspx (https://msdn.microsoft.com/en-us/library/ff878370.aspx)

vithorsilva 3 anos ago