Prelucrarea datelor cu caracter temporar

Baza de date TempDB, creată la instalarea sistemului Microsoft SQL Server, este o resursă globală care disponibilă tuturor utilizatorilor conectați la instanța SQL

Tempdb este folositca bază de date de sistem pentru stocarea rezultatelor intermediare de sortare, pentru operațiuni precum crearea sau reconstruirea indexurilor, pentru stocarea rezultatelor intermediare pentru interogări ce folosesc clauzele GROUP BY, ORDER BY sau UNION, etc.

> Baza de date TempDB poate fi folosită de utilizatori pentru crearea în mod explicit a obiectelor temporare: tabele globale sau locale, proceduri stocate temporare, cursoare, variabile de tabel, tabele returnate în funcții ce au ca tip returnat variabile de tip tabel, cursoare.

Tabele temporare

Tabelele temporare sunt obiecte create de utilizator pentru prelucrarea datelor cu caracter temporar. Tabelele temporare sunt prelucrate similar tabelelor permanente relativ la comenzile CREATE TABLE, SELECT, INSERT, UPDATE, DELETE, DROP TABLE.
De asemenea sunt posibile operații ca indexarea, adăugarea sau eliminarea de coloane.
Sintaxa CREATE TABLE acceptă definiții de constrângere cu excepția constrângerilor de tip FOREIGN KEY.
Tabele temporare sunt create în baza de date de sistem *TempDb,* special creată de SQL Server ca să ofere spațiu de lucru pentru diverse obiecte temporare precum tabele temporare și proceduri stocate temporare. *TempDb* este o bază de date de sistem similară bazelor de date aplicative, cu diferența că datele stocate nu sunt persistente. Obiectele create în *TempDb* sunt automat eliminate când sesiunea clientului curent se termină.

Există două tipuri de tabele temporare:
Tabele temporare locale, create cu prefixul #. Domeniul lor de aplicare este limitat la sesiunea curentă a utilizatorului care le-a creat. Acest lucru înseamnă că în sesiuni (scripturi) diferite pot fi create tabele temporare cu aceeași denumire (unicitatea fiind asigurată prin asocierea unui număr de identificare).
Tabele temporare globale, create cu prefixul ##. Un astfel de tabel este vizibil din orice sesiune(script). Nu pot exista două tabele temporare globale cu aceeași denumire.
Tabelul temporar global va fi distrus când sesiunea de referință în care a fost creat este închisă și toate procesele care îl referă sunt terminate.

Exemplu de utilizare a tabelelor temporare locale:

```
use dbFacturare

create table #Clienti
(codClient char(10),nrFacturi int default 0)
insert into #Clienti values('100',0)
insert into #Clienti(codClient) values('101')

update #Clienti
  set nrFacturi=(select COUNT(NrFact) from tFacturi
                 where CodClient=#Clienti.codClient)
 select * from #Clienti
```

Exemple de utilizare a tabelelor temporare globale:
1)

```
use dbFacturare

if object_id('tempdb.dbo.##CentralizatorFacturi') is not null
  drop table ##CentralizatorFacturi

create table ##CentralizatorFacturi
   (nrFact char(10) primary key,
    CodClient char(10),
    ValFact numeric(7,2)
    )

insert into ##CentralizatorFacturi
select tFacturi.nrfact, codClient, SUM(Cantitate*pret)
from tFacturi inner join tDetaliiFact
       on tFacturi.nrFact=tDetaliiFact.NrFact
       inner join tProduse
       on tDetaliiFact.CodProd=tProduse.CodProd
group by tFacturi.NrFact,CodClient

select * from ##CentralizatorFacturi
```

sau

```
if object_id('tempdb.dbo.##CentralizatorFacturi') is not null
  drop table ##CentralizatorFacturi

select tFacturi.nrfact, codClient, SUM(Cantitate*pret)as valoare
into ##CentralizatorFacturi
from tFacturi inner join tDetaliiFact
       on tFacturi.nrFact=tDetaliiFact.NrFact
       inner join tProduse
       on tDetaliiFact.CodProd=tProduse.CodProd
group by tFacturi.NrFact,CodClient

select * from ##CentralizatorFacturi
```

2) Numerotarea randurilor

```
drop table ##Clienti
```

```
select IDENTITY(int,1,1) as NrCrt,
      CodJudet,Localitate, Nume
into ##Clienti
from tClienti
order by  CodJudet,Localitate, Nume

select NrCrt,CodJudet,Localitate,Nume,NrCrt
from ##Clienti
```

Observatie. In baza de date de sistem Tempdb, pot fi create și tabele permanente (care nu sunt eliminate automat la terminarea conexiunii utilizatorului cu baza de date).
Pentru aceasta, comanda va fi ca mai jos:
1)
```
create table tempdb.dbo.t1(x int)
```
2)
```
create table tempdb..t2(x int)
```
3)
```
use tempdb
create table t3(x int)
```
Eliminarea acestor tabele se poate face prin utilizarea explicită a comenzii DROP TABLE:
```
drop table tempdb.dbo.t1
drop table tempdb.dbo.t2
drop table tempdb.dbo.t3
```

Variabile de tip tabel

Începând cu SQL Server 2000, Microsoft a introdus un nou tip de date: *variabile de tip tabel,* ca o alternativă la utilizarea tabelelor temporare. Pentru seturile de rezultate mici variabilele de tip tablou sunt mai rapide și mai flexibile decât tabele temporare.
O variabilă de tip tabel se declară ca în exemplele următoare:

```
declare @t table
( x int identity(1,1) primary key,
  y int CHECK (y <1000)
 )
insert into @t values (5),(7),(9)
select * from @t
```

x	y
1	5
2	7
3	9

```
declare @t2 table(codProd char(10),pret numeric(7,2))
insert into @t2 select codProd,pret from tProduse

select codProd ,pret  from @t2
```

O funcție poate returna o variabilă de tip tabel:

```
create function centralizator()
 returns @Centralizator table(NrFact char(10),Valoare numeric(8,2))
 As
  begin
   insert into @Centralizator
   select NrFact,SUM(Cantitate*Pret) as ValoareFacturata
   from tDetaliiFact as A inner join tProduse as B
         on A.CodProd=B.CodProd
   group by NrFact

   return
  end

  go

  select * from dbo.centralizator()
```

Variabile de tip tabel sunt stocate în memorie și parțial stocate pe disc. Deoarece acestea sunt parțial stocate în memorie, timpul de acces pentru o variabilă tabel poate fi mai rapid decât timpul necesar pentru a accesa un tabel temporar.

EXPRESII TABEL

O expresie tabel (CTE, Common Table Expression) poate fi considerata un set temporar de rezultate definit in sfera de executie a unei singure comenzi SELECT, INSERT, UPDATE, DELETE sau CREATE VIEW. O expresie CTE reprezintă un tabel virtual, in sensul ca nu este stocat sub forma de obiect si nu dureaza mai mult decat comanda pentru care este definită.
O expresie CTE se compune din urmatoarele elemente: un nume reprezentand expresia CTE, o lista facultativa de coloane si o interogare SELECT al cărui set de rezultate populează expresia CTE. Din momentul in care expresia CTE a fost definita, se poate face referire la ea, similar unui tabel sau vederi, intr-o comanda SELECT, INSERT, UPDATE, DELETE sau CREATE VIEW.
O expresie CTE poate face trimitere catre ea insasi si poate fi referită de mai multe ori in comanda asociată.

Exemple
1) Să se determine clienții pentru care numărul de facturi emise este mai mare decât media numărului de facturi emise la nivel de client.

```
with
CTE(CodClient, nrFacturi)              -- definirea expresiei tabel
as
 (
```

```
  select   CodClient,count(NrFact)     -- comanda select ce populează cu date expresia CTE
  from     tFacturi
  group by CodClient
 )
Select CTE.CodClient,nume,nrFacturi     -- comanda care foloseste expresia tabe

(poate fi  seect, insert, update, delete)
from   CTE inner join tClienti
       on CTE.CodClient=tClienti.CodClient
where  nrFacturi>(select AVG(nrFacturi) from CTE)
```

2) Exemplu de utilizare a unei expresii tabel într-o comandă CREATE VIEW

```
create view v
as
with
CTE(CodClient, nrFacturi)
as
 (
  select   CodClient,count(NrFact)
  from     tFacturi
  group by CodClient
 )
select -- comanda ce foloseste expresia tabel
  CTE.CodClient,nume,nrFacturi
from   CTE inner join tClienti
       on CTE.CodClient=tClienti.CodClient
where  nrFacturi>(select AVG(nrFacturi) from CTE)
```

Observatie
Clauza *with* poate conține mai multe expresii CTE separate prin virgula.