Proceduri stocate

Crearea procedurilor stocate.

Varianta 1
În Management Studio se dă clic pe **New Query** ca în imaginea de mai jos:

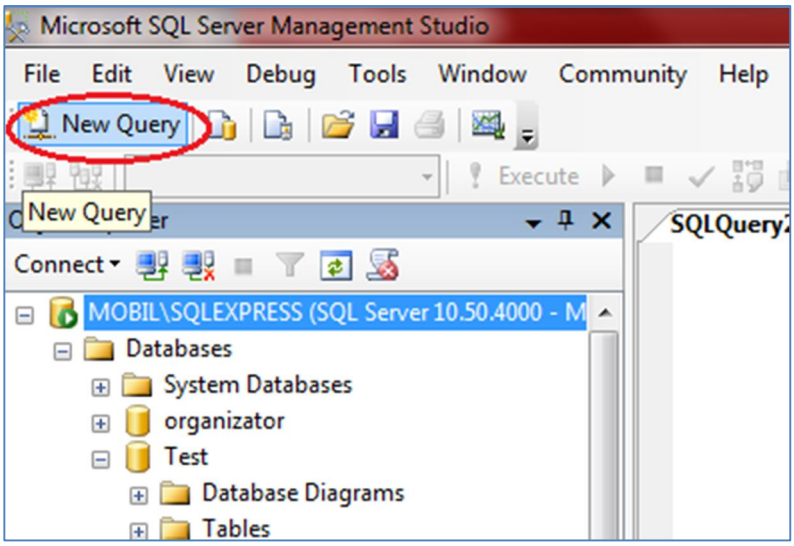

Fig. 1

Odată cu deschiderea editorului SQL, apare și bara de instrumente **SQL Editor** care conține butoane specifice editoarelor de cod.

Fig. 2 Bara SQL Editor

Aici trebuie să fie selectată baza de date în care dorim să creem procedurile stocate (**master**, în exemplul de mai sus). Pentru verificarea sintaxei se dă clic pe butonul **Parse** (bifa), iar pentru executarea codului se face clic pe butonul **Execute**.
Șablonul pentru scrierea unei proceduri stocate este:

```
CREATE PROCEDURE nume_procedura
      --listă parametri
AS
BEGIN
      --Corp de cod
END
```

Pentru a comenta o linie de cod se scrie ”--” (minus minus) în fața liniei. Pentru a comenta un bloc de cod (mai multe linii), blocul comentat se introduce între ”/*” și între ”*/” ca în C++.

Varianta 2

În **Object Explorer** se deschid, cu clic pe ”+” elementele bazei de date, apoi **Programmability**. Clic dreapta pe **Stored Procedures**, apoi clic pe **New Stored Procedure…**, acțiune în urma căreia se va deschide editorul de cod SQL în care vom gasi implementat șablonul pentru procedura stocată.

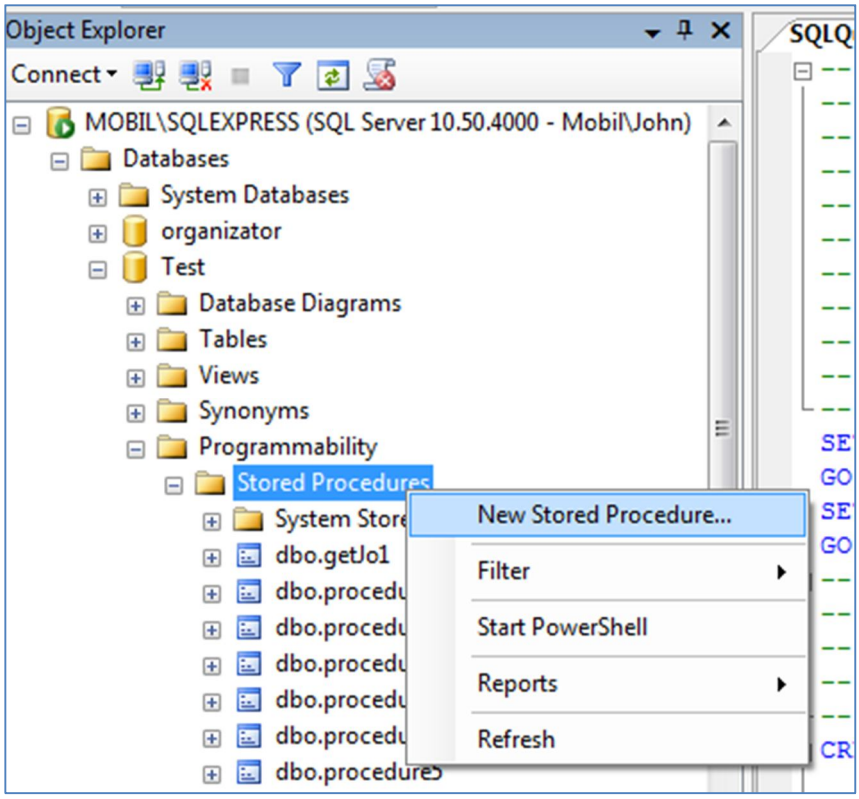

Fig. 3 Proceduri stocate

Salvarea și compilarea procedurilor stocate

După scrierea codului avem posibilitatea să-l salvăm ca script într-un fișier text, cu extensia **sql** sau, în cazul în care am scris o procedură, să o salvăm ca procedură stocată. Pentru aceasta, după verificarea sintaxei (clic pe butonul **Parse** din bara de instrumente **SQL Editor**), se face clic o singură dată pe butonul **Execute**. Această acțiune salvează și compilează procedura stocată.

Dacă se încearcă o nouă salvare, prin clic pe **Execute**, vom primi un mesaj de eroare în care ni se spune că deja există o procedură cu denumirea dată.

Verificare dacă procedura a fost salvată

Pentru a verifica dacă salvarea a avut loc se caută procedura în lista de proceduri stocate ale bazei de date, ca în fig 3. În cazul în care nu apare se face clic dreapta pe **Stored Procedures** și apoi clic pe **Refresh**.

Executarea procedurilor stocate

Se deschide o nouă fereastră de cod cu **New Query** și putem executa codul procedurii în oricare din cele 3 variante de mai jos:

- `EXEC numeprocedură`, apoi clic pe butonul Execute din bara **SQL Editor** (sau tasta F5)

sau

- `EXECUTE numeprocedură`, idem sus

sau

- `Numeprocedură`, idem sus

Modificarea procedurilor stocate

Pentru a modifica o procedură stocată, în fereastra **Object Explorer** se dă clic dreapta pe numele ei, apoi clic pe **Modify.** Vom observa că în codul procedurii cuvântul cheie CREATE a fost substituit cu cuvântul cheie ALTER, ceea ce e normal pentru că dorim să modificăm o procedură existentă, nu să o creem.

După scrierea modificărilor, e foarte important să nu uităm să dăm clic pe **Execute**; doar așa are loc o nouă compilare a procedurii și salvarea sa cu noile modificări. Când suntem în modul ALTER (de modificare) putem să facem clic de oricîte ori dorim pe butoul **Execute** pentru că nu vom mai primi nici un fel de mesaj de eroare (cîtă vreme codul e corect).

Interogarea obiectelor bazei de date

Pentru a verifica dacă un obiect există în baza de date avem mai multe metode. Interogarea obiectului **sys** sau a obiectului **INFORMATION_SCHEMA**.
Exemple:
1.

```
SELECT name FROM sys.default_constraints WHERE name='nume_constrângere')
```

Returnează un rând dacă găsește o constrângere de tip DEFAULT (implicit) în baza de date

2.

```
SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS
            where TABLE_NAME = 'nume_tabela'
            AND column_name = 'nume_coloana'
```

Returnează un rând dacă găsește ”nume_coloana” în ”nume_tabela”.

Generarea scripturilor

De cele mai multe ori atunci când creem o procedură stocată ne folosim de porțiuni de cod pe care le generează Management Studio-ul.
Pentru a genera un script se face clic dreapta, în fereastra **Object Explorer**, pe diferite obiecte (numele bazei de date, numele tabelului, a vederii, a procedurii stocate, etc), apoi se deschid submeniurile ca în fig. 4.

În acest mod putem să generăm scripturi pentru creare (**CREATE**), modificare (**ALTER**) sau ștergere (**DROP**) de obiecte care schimbă structura bazei de date (LDD –limbaj de definire a datelor, resp. DDL). Deasemenea, putem genera scripturi pentru manipularea datelor (LMD – limbaj de manipulare a datelor, resp. DML) cum ar fi adăugarea unui rând nou (**INSERT**), modificarea (**UPDATE**) sau ștergerea (**DELETE**) unuia existent, precum și pentru extragerea informațiilor după criteriile dorite (**SELECT**).

Scripturile generate pot fi deschise într-o nouă fereastră a editorului SQL, într-un fișier sau în memoria Clipboard pentru un eventual Paste viitor.

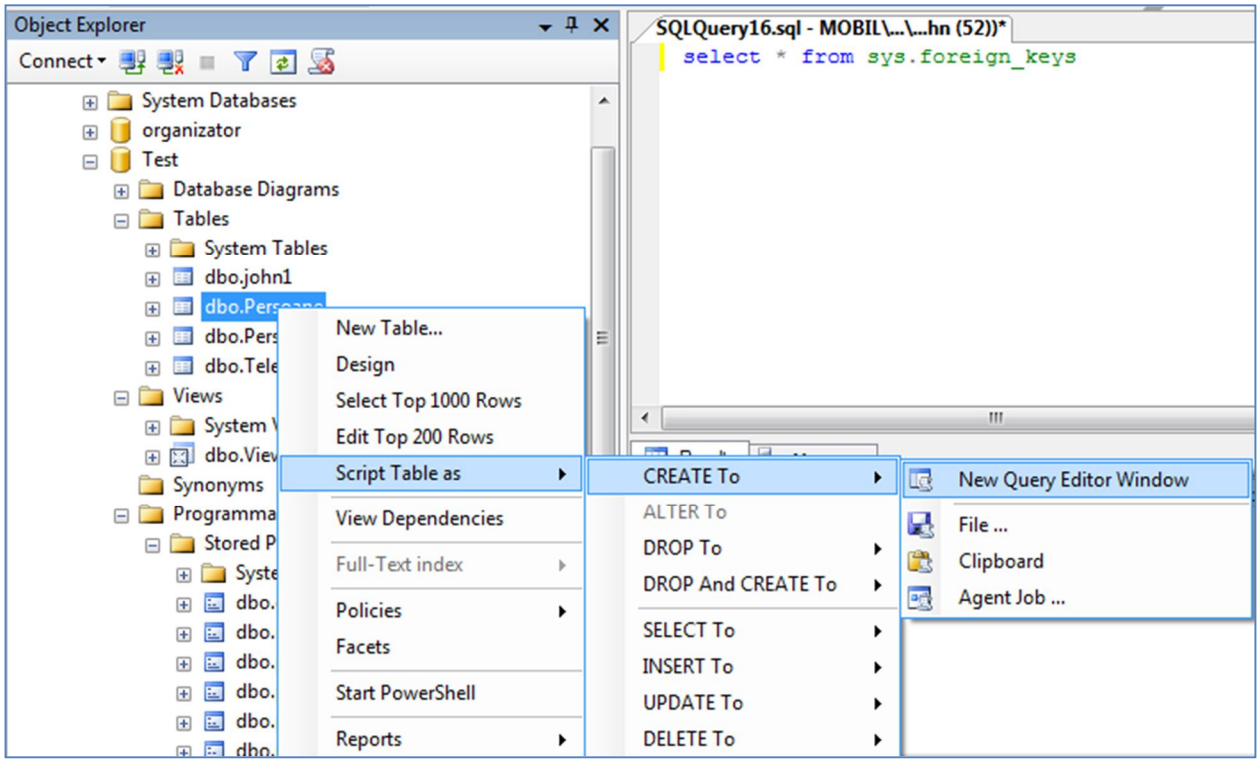

Fig 4

Pentru a genera scriptul pentru crearea unei constrângeri de tip ”cheie străină” (FOREIGN KEY) se intră în diagramă, se crează cu mouse-ul relația dintre cele două tabele, apoi clic dreapta pe una din tabele si clic pe **Generate Change Script...** ca în fig 5.

În fereastra modală care apare se poate identifica următoarea porțiune de cod, importantă pentru procedura 5:

```
ALTER TABLE dbo.Telefoane
ADD CONSTRAINT nume_constrangere FOREIGN KEY (ID)
    REFERENCES dbo.Persoane(id)
    ON UPDATE  NO ACTION
    ON DELETE  NO ACTION
```

Așadar, pentru a crea o constrângere de tip ”cheie străină”, se începe cu modificarea tabelei ”secundare” (cea în care se găsește cheia străină), apoi se adaugă o constrângere care face referință la tabela ”primcipală” (cea în care se găsește cheia primară). Deasemenea se setează constrângerile pentru modficare și ștergere. Opțiunile sunt, la fel ca în modul Design:

- NO ACTION
- CASCADE

- SET NULL
- SET DEFAULT

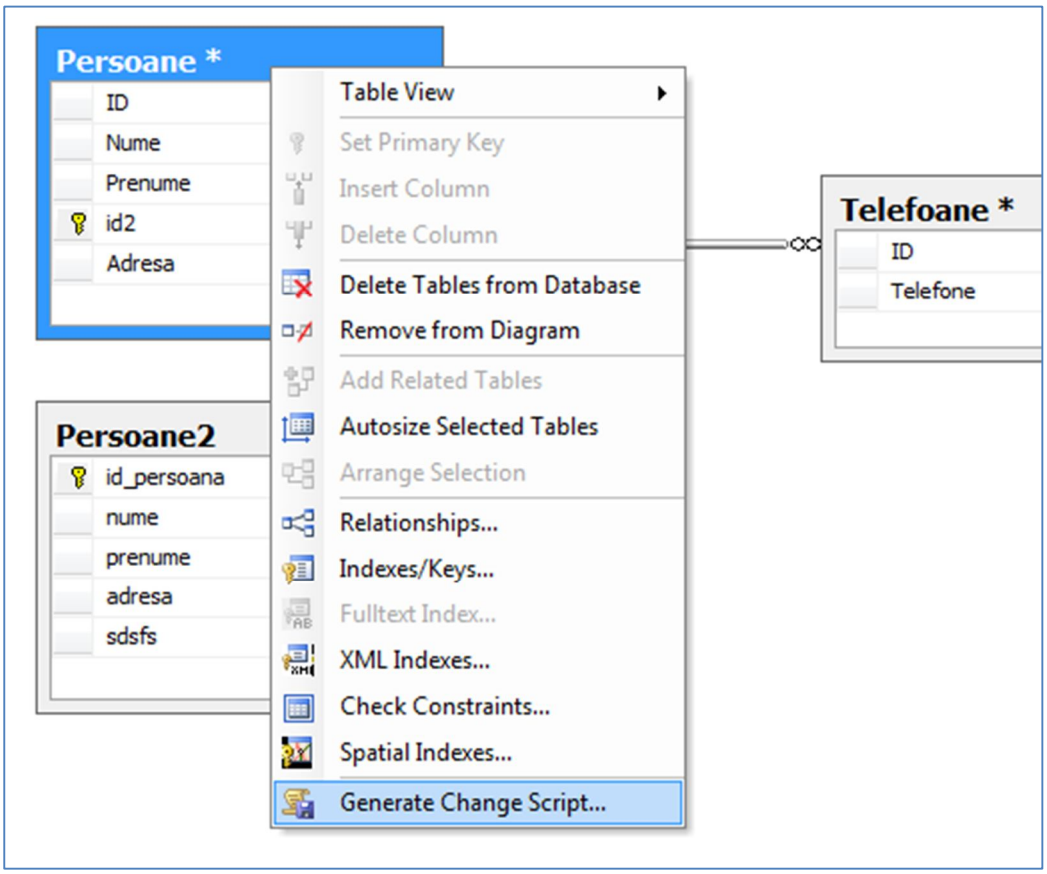

Fig. 5

Exemple de proceduri stocate

a) Procedura pentru modificarea tipului unei coloane

```
CREATE PROCEDURE nume_procedura
AS
BEGIN
	ALTER TABLE nume_tabela
	ALTER COLUMN nume_coloana nvarchar(200)—noul tip de coloană, inclusiv mărimea unde e cazul
END
```

b) Procedura pentru crearea unei constrângeri de tip DEFAULT

```
CREATE PROCEDURE nume_procedura
AS
	if exists(SELECT *
		--FROM sysobjects WHERE xtype='D' AND name='nume_constrangere')
		FROM sys.default_constraints WHERE name='nume_constrangere')
```

```
            print 'E deja definita o constrangere implicita cu numele '
      else
      begin
            ALTER TABLE nume_tabela
            ADD CONSTRAINT nume_constrângere DEFAULT 'implicit' FOR nume_camp
            --ALTER COLUMN nume add DEFAULT "jOH"
            print Constrangere creeata'
      end
```

Pentru a șterge procedura:

```
            ALTER TABLE nume_tabela
            DROP CONSTRAINT nume_constrângere
```

Bineînțeles după ce se verifică dacă există sau nu.

c) Procedura pentru crearea unei tabele noi cu un câmp cheie primară și două constrângeri, una de tip Default și una de tip Check.

```
            IF NOT EXISTS(SELECT * FROM sys.tables
                              WHERE name='nume_tabela')
                  BEGIN
                  CREATE TABLE tabela2(
                        id bigint NOT NULL PRIMARY KEY,
                        nume_t nvarchar(50) NOT NULL,
                        pren_t nchar(20) CONSTRAINT nume1 DEFAULT 'JO',
                        pret money,
                        cantitate smallint CONSTRAINT nume2
CHECK(cantitate>2)
                  )
                  PRINT 'Tabela creata!'
                  END
            ELSE
                  PRINT 'Exista Tabela2'
```

d) Procedura pentru adăgarea unui câmp (coloane) – doar esența

```
                  ALTER TABLE nume_tabela
                  ADD nume_coloana int --(tipul coloanei)
```

Pentru verificarea existenței câmpului aici e mai ușor de folosit obiectul INFORMATION_SCHEMA, ca mai jos:

```
SELECT COLUMN_NAME FROM INFORMATION_SCHEMA.COLUMNS Where TABLE_NAME =
'nume_tabela' AND column_name = 'nume_coloana'
```

De reținut:
Codul Transact-SQL nu este Case Senzitiv (poate fi scris și cu litere mari și cu litere mici).