**Rukovanje uskladištenim procedurama**

**00:10:44**

Jedinica: 10 od 19

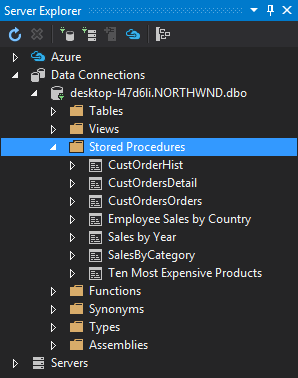
**+Rezime**

Uskladištene procedure su predefinisane, ponovo upotrebljive rutine koje se skladište u bazi podataka. Na taj način se kod za manipulaciju podacima centralizuje na serveru, što osigurava da će svaka aplikacija rukovati podacima na identičan način.

**Uskladištene procedure**

Već je rečeno da su uskladištene procedure objekti koji postoje na serveru. Stoga je sadržaj neke uskladištene procedure moguće videti korišćenjem nekog alata koji dozvoljava interakciju sa serverom baze podataka.

Korišćenjem razvojnog okruženja Visual Studio, tako nešto se može postići upotrebom Server Explorera. U kursu većina primera se temelji na bazi podataka *Northwind*. Ova baza poseduje nekoliko uskladištenih procedura, koje je moguće pronaći unutar foldera *Stored Procedures*, kao na slici 10.1.



*Slika 10.1 – Uskladištene procedure baze podataka Northwind*

Duplim klikom na neku od procedura, moguće je videti njen sadržaj.

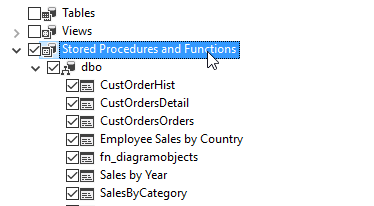
**Upotreba uskladištenih procedura**

Kako bi se uskladištene procedure koristile u kodu, prethodno moraju biti importovane na isti način kao i drugi objekti baze podataka. Dovoljno je pokrenuti opciju *Update Model from Database*, kao na slici 10.2.

https://www.link-elearning.com/linkdl/coursefiles/1142/MDA_10_02.jpg

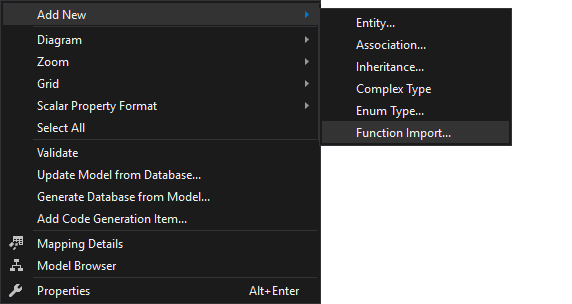
*Slika 10.2 – Update Model from Database opcija*

Unutar Update Wizard prozora potrebno je čekirati *Stored Procedures and Functions* opciju, kao na slici 10.3.



*Slika 10.3 – Uključivanje uskladištenih procedura u model*

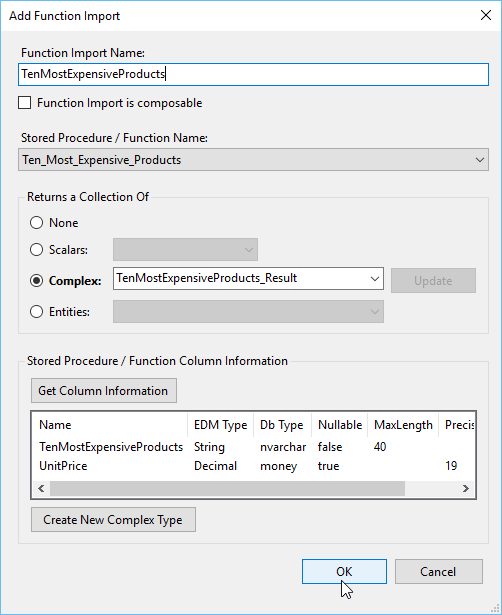
Importovanje uskladištenih procedura je samo prvi korak u omogućavanju njihovog nesmetanog korišćenja. Sledeći korak podrazumeva importovanje uskladištenih procedura u konceptualni model. Tako nešto se postiže korišćenjem prozora *Add Function Import*. On se može otvoriti na dva načina. Prvi način podrazumeva desni klik na dizajner površinu i odabir opcije *Add New -> Function Import*, kao što je prikazano na slici 10.4.



*Slika 10.4 – Opcija za uključivanje uskladištenih procedura u konceptualni model*

Drugi način za otvaranje prozora *Add Function Import* podrazumeva da se unutar Model Browsera izvrši klik na folder *Stored Procedure*, a potom desni klik na proceduru za koju je potrebno dodati referencu. Iz kontekstnog menija, potrebno je odabrati opciju *Add Function Import*.

Kao što je rečeno, oba načina rezultuju otvaranjem prozora *Add Function Import*, koji donosi različite opcije za podešavanje procesa uključivanja uskladištene procedure. Slika 10.5 prikazuje ovaj prozor.



*Slika 10.5 – Prozor za konfigurisanje uključivanja uskladištenih procedura*

Add Function Import prozor omogućava definisanje naziva funkcije, kao i definisanje povratnih vrednosti.

U primeru sa slike 10.5 izvršeno je uključivanje uskladištene procedure sa nazivom *Ten\_Most\_Expensive\_Products*, a za naziv funkcije koja će je predstavljati, postavljeno je *TenMostExpensiveProducts*.

Dugme *Get Column Information* omogućava čitanje šeme uskladištene procedure. Stoga je prvo potrebno kliknuti na ovo dugme, a potom na *Create New Complex Type*, pri čemu će se kreirati novi kompleksni tip povratne vrednosti ove funkcije.

Ovim je završeno dodavanje reference na uskladištenu proceduru unutar konceptualnog modela. Naredni korak je njeno korišćenje.

**Korišćenje uskladištenih procedura**

Nakon dodavanja reference na uskladištenu proceduru, nju je moguće pozivati iz koda. Sledi primer koji to ilustruje:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | using (var ctx = new NorthwindEntities())              {                  var topTenMostExpensiveProducts = ctx.TenMostExpensiveProducts();                  foreach (var expensiveProduct in topTenMostExpensiveProducts)                  {                      Console.WriteLine(expensiveProduct.UnitPrice);                  }              } |

U prikazanom primeru prvo je, kao i obično, kreirana instanca NorthwindEntities klase. Zatim je upućen poziv funkciji *TenMostExpensiveProducts*, a rezultat je skladišten u promenljivoj *topTenMostExpensiveProducts*. Na kraju je, u okviru jedne foreach petlje, izvršen prikaz svih proizvoda.

Ovo je bila jednostavna uskladištena procedura koja nije imala ulazne parametre. Stoga će sledeća tema biti rad sa uskladištenim procedurama koje imaju ulazne parametre.

**Uskladištene procedure sa ulaznim parametrima**

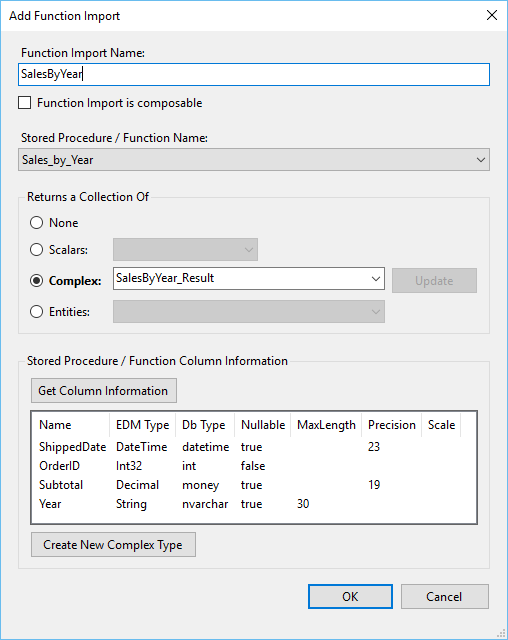
U narednom primeru biće korišćena uskladištena procedura *Sales\_By\_Year* koja zahteva *Begining\_date* i *Ending\_date* ulazne parametre.

Uskladištena procedura Sales\_By\_Year izgleda ovako:

|  |  |
| --- | --- |
| 1  2  3  4  5 | create procedure "Sales by Year"  @Beginning\_Date DateTime, @Ending\_Date DateTime AS  SELECT Orders.ShippedDate, Orders.OrderID, "Order Subtotals".Subtotal, DATENAME(yy,ShippedDate) AS Year  FROM Orders INNER JOIN "Order Subtotals" ON Orders.OrderID = "Order Subtotals".OrderID  WHERE Orders.ShippedDate Between @Beginning\_Date And @Ending\_Date |

Kao što se vidi, ova procedura ima dva ulazna parametra @Beginning\_Date i @Ending\_Date, oba tipa DateTime.

Dakle, prvo je potrebno importovati uskladištenu proceduru u funkciju, na isti način kao u prethodnim pasusima. Taj proces ilustrovan je slikom 10.6.



*Slika 10.6 – Konfigurisanje uključivanja uskladištene procedure sa ulaznim parametrima*

Nakon dodavanja reference, uskladištenu proceduru je moguće pozvati kao funkciju, što je ilustrovano sledećim primerom.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | using (var ctx = new NorthwindEntities())              {                  var salesByYear = ctx.SalesByYear(DateTime.Now.AddYears(-20), DateTime.Today);                  foreach (var salesByYearResult in salesByYear)                  {                      Console.WriteLine("{0} - {1}", salesByYearResult.Year,                          salesByYearResult.Subtotal);                  }              } |

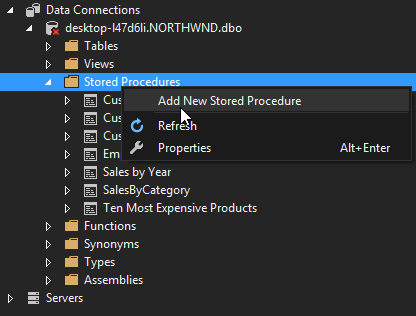
U primeru se poziva *SalesByYear* metoda i njoj se prosleđuju dva parametra – jedan koji predstavlja *Beginning\_Date* (koji ima vrednost današnjeg datuma, samo pre dvadeset godina) i *Ending\_Date* (koji se odnosi na današnji datum). Rezultat se skladišti u promenljivoj *salesByYear*. Na kraju se prolazi kroz petlju kako bi se prikazale sve prodaje koje su se odigrale u definisanom vremenskom rasponu.

**Uskladištene procedure sa izlaznim parametrima**

Uskladištene procedure mogu imati i izlazne parametre. Izlazni parametar u uskladištenim procedurama ima OUTPUT oznaku.

Da bi se upravljalo izlaznim parametrima, potrebno je koristiti ObjectParametar klasu. ObjectParametar klasa reprezentuje parametar upita, koji se prosleđuje upitu nad objektima iz baze. Da bi se koristio, neophodno je uključiti prostor imena System.Data.Objects.

Za demonstraciju upotrebe uskladištene procedure, koja ima izlazne parametre, biće kreirana nova procedura. Za tako nešto je dovoljno u Server Exploreru kliknuti desnim tasterom miša na folder Stored Procedures i iz padajućeg menija odabrati opciju *Add New Stored Procedure*, kao na slici 10.7.

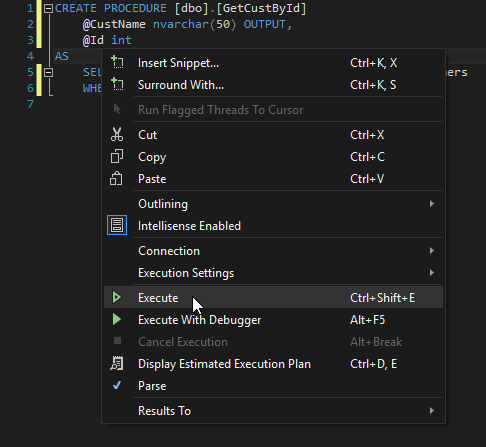


*Slika 10.7 – Dodavanje nove uskladištene procedure korišćenjem Server Explorera*

Ovo rezultuje otvaranjem prozora za pisanje procedure. Kod procedure može biti sledeći:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | CREATE PROCEDURE [dbo].[GetCustById]      @CustName nvarchar(50) OUTPUT,      @Id nchar(5)  AS      SELECT @CustName = Customers.ContactName FROM Customers      WHERE Customers.CustomerID = @Id |

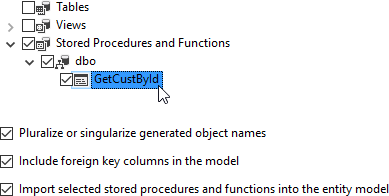
Ovo je jednostavna procedura sa dva parametra, i to jednim ulaznim i jednim izlaznim. Na osnovu ulaznog parametra, koji predstavlja id mušterije, vrši se selektovanje imena mušterije. Da bi se skripta izvršila na serveru baze, i da bi ovakva procedura bila kreirana, potrebno je pokrenuti opciju *Execute* iz menija koji se dobija kada kliknemo desnim testerom miša na radnu površinu, kao na slici 10.8.



*Slika 10.8 – Izvršavanje skripte za dodavanje nove uskladištene procedure u bazu*

Na ovaj način procedura će biti kreirana u bazi. Naredni korak jeste ažuriranje modela pokretanjem opcije *Update Model from Database*.

Prilikom ažuriranja modela moguće je čekirati opciju *Import selected stored procedures and functions into the entity model*, kao na slici 10.9.



*Slika 10.9 – Čekiranje opcije Import selected stored procedures and functions into the entity model*

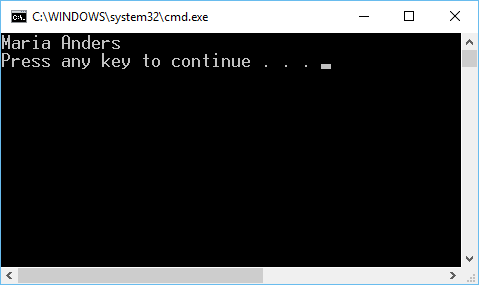
Na ovaj način će automatski biti izvršeno dodavanje reference na uskladištenu proceduru, i sve što preostaje jeste da se ona iskoristi u kodu.

Sledeći primer to ilustruje.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | using (var ctx = new NorthwindEntities())              {                  ObjectParameter name = new ObjectParameter("CustName", "");                  ctx.GetCustById(name, "ALFKI");                  Console.WriteLine(name.Value);              } |

Da bi se uhvatio izlazni parametar, u kodu se koristi instanca ObjectParameter klase. Konstruktor ove klase prihvata kao prvi parametar naziv parametra, a kao drugi njegovu početnu vrednost.

Zatim se GetCustById funkciji prosleđuju dva parametra – jedan koji predstavlja izlazni parametar i drugi koji predstavlja ID kupca. Na osnovu prosleđenog ID-ja, ispisaće se ime sa pripadajućom vrednošću navedenog ID-ja. Na kraju, ime kupca se ispisuje na standardni izlaz, kao na slici 10.10.



*Slika 10.10 – Izvršavanje skripte za dodavanje nove uskladištene procedure u bazu*

**Izlazni parametar uskladištenih procedura ima oznaku:**

 OUTPUT

 INPUT

 OUT

 Uskladištene procedure ne mogu imati izlazni parametar