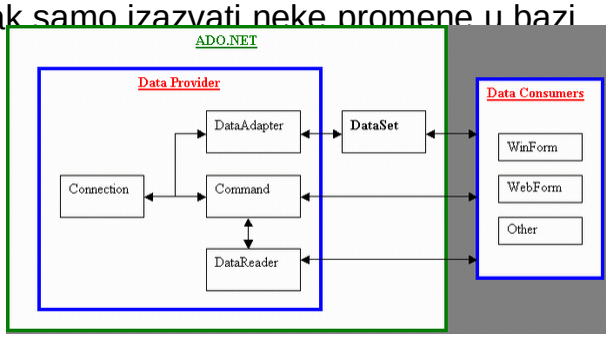
Izvršavanje SQL komandi

ČAS 3.

Komanda

- Komanda je služi da se izvrši SQL naredba ili uskladištena procedura (*store procedure*) koristeći objekat *Connection*.
- Komanda može kao rezultat imati vraćeni niz zapisa ili pak samo izazvati neke promene u bazi



Kreiranje

1. Samostalno:
   1. SqlCommand **cmd** = new SqlCommand();
2. Koristeći *Connection* objekat
   1. SqlCommand **cmd** = conn.CreateCommand();

- U prvom slučaju se mora obezbediti naknadno povezivanje sa nekim konekcijskim objektom. Zašto? Jer komanda mora da se izvršava preko neke (otvorene) konekcije.
- *cmd.Connection =conn;*

Vrste konstruktora

- new xxCommand(),
- new xxCommand*(string komanda),*
- new xxCommand*(string komanda, xxConnection konekcija),*
- *new xx*Command*(string komanda, xxConnection konekcija, Transakcija).*

- Dodeljivanje komandnog teksta tj. sql naredbe:
- cmd.**CommandText** = s*qlNaredba; //string*
- Treba da bude u ***try-catch*** bloku.

Svojstva

CommandText	SQL naredba ili uskladistena procedura koja treba da se izvršdi.
CommandTimeout	Vreme (u sekundama) za cekanje na odgovor od izvora podataka.
CommandType	Označava kako bi svojstvo CommandText trebalo da se protumaci. Podrazumeva se vrednost Text
Connection	Objekat Connection na kojem komanda za podatke treba da se izvrsi.
Parameters	Kolekcija Parameters.
Transaction	Transakcija u kojoj ce se komanda izvrsavati.

Opis...

- **CommandText**, je znakovni niz, sadrži ili stvarni tekst komande koja treba da se izvrsi na konekciji ili ime uskladištene procedure iz izvora podataka.
- **CommandTimeout** odreduje vreme koje ce komanda cekati na odgovor od servera pre nego sto generise gresku. Uzmite u obzir da je ova vreme koje protekne pre nego sto objekat Command pocne da prima rezultate, a ne vreme koje je potrebno komandi da se izvrši. Izvoru podataka moze biti potrebno 10 ili 15 minuta da vrati sve redove neke ogromne tabele, ali pod uslovom da je prvi red vracen u toku zadatog perioda CommandTimeout nece se generisati nikakva greSka.
- **CommandType** govori objektu Command na koji nacin treba da protumači sadržaj svojstva CommandText.
  - Vrednost TableDirect podržana je za objekat OleDbCommand, ali ne i za objekat SqlCommand, a ekvivalentna je naredbi SELECT * FROM <ime_abele>, pri cemu je <ime_tabele> ime tabele specifikovano u svojstvu CommandText

- Svojstvo **Parameters** objekta Command sadrži kolekciju parametara za SQL naredbu ili uskladistenu proceduru navedenu u svojstvu ConunandText.
- Ovu kolekciju cemo detaljno ispitati u nastavku.

- Svojstvo **Transaction** sadrži referencu na objekat Transaction i sluzi pri obradi transakcija. Ovo svojstvo cemo detaljno ispitati kasnije

Izvršavanje

1. Iskaz nije upit – ***ExecuteNonQuery***
2. Jedna vrednost – ***ExecuteScalar***
3. Jedan ili više redova – ***ExecuteReader***
4. XML - *ExecuteXMLReader*

Primer

- Testiranje komandi sa više vraćenih redova – **ExecuteReader**
- Vratiti imena i prezimena iz tabele *Employees.*

Izvršavanje komandi koje nisu upit: **ExecuteNonQuery**

- Pogledati tabelu Employes.
- Dodati 1 red:

1. ***insert into*** *Employees (FirstName, LastName) values ('Pera', 'Peric');*
2. ***delete*** *from Employees where FirstName='Pera' and LastName = 'Peric'*

Kreiranje baze i tabele

1. Otvaranje konekcije
2. “create database *imeNoveBaze*”
3. Promena baze u objektu konekcije
4. “create table *imeNoveTabele* (*imekol tip*)”

- Primer (na vežbama)

Parameters

- Šta je to kolekcija?

1. Jedna vrsta niza (definisan u .NET)
2. Veličina nije ograničena (samo memorijom na računaru)
3. Podrška različitih metoda za baratanje sa nizom.

- Kako se koristi?

1. Specificirati parametre kroz SQL komandu u upitima ili uskladištenim procedurama
2. Definisati odgovarajuće parametre u kolekciji Parameters
3. Dodeliti vrednosti

1. Specifikacija parametara kolekcije *parameters* kroz SQL komande

1. Postoje dve različite vrste oznaka u zavisnosti od vrste dobavljača
   1. OleDbCommand
   2. SELECT * FROM Customer WHERE CustomerID = **?**
2. SqlCommand
   1. SELECT Count(*) AS OrderCount FROM OrderTotals WHERE
   2. (EmployeeID = **@emplD**) AND (CustomerID = **@custID**)

Obratite pažnju: ***Objekat OleDbCommand ne podržava imenovane parametre! Redosled je od presudnog značaja***!

2. Definisanje kolekcije

- 1. Ukoliko koristite IDE ova kolekcija će biti automatski formirana.
- 2. Ukoliko sami u kodu kreirate kolekciju, nove parametre dodajete pomoću metode **Add*(noviParametar)***
- Napomena: Za sve kolekcije važi isti metod (isto ime metoda) za dodavanje novog elementa.

Metode kolekcije (1)

- **Add(Value)** Na kraj kolekcije dodaje novi parametar cija je vrednost *Value.*
- **Add(Parameter)** Na kraj kolekcije dodaje objekat Parameter.
  - *.... Još par varijanti metode Add() ...*
- **Clear** Uklanja sve parametre iz kolekcije.
- **Insert(Index, Val)** Umece novi parametar cija vrednost je *Value* na mesto odredeno indeksom *Index* koji se racuna pocevsi od nule.

Metode kolekcije (2)

- **Remove(Value)** Iz kolekcije uklanja parametar cija vrednost je *Value.*
- **RemoveAt(Index)** Iz kolekcije uklanja parametar koji se nalazi na mestu odredenom indeksom *Index* koji se racuna pocevsi od nule.
- **RemoveAt (Name)** Iz kolekcije uklanja parametar cije ime je specifikovano u znakovnom nizu *Name.*

Konfigurisanje parametara

```
int cnt;
string strMsg;
System.Data.DataRowView drv;
drv = (System.Data.DataRowView) this.lbEmployees.SelectedItem;

this.cmdOrderCount.Parameters["@empID"].Value =
    drv["EmployeeID"];

drv = (System. Data.DataRowView) this.lbClients.SelectedItem;

this.cmdOrderCount.Parameters["@custID"].Value =
    drv["CustomerID"];
```

Metode objekta Command

- **Cancel** - Otkazuje izvrsenje komande za podatke.
- **CreateParameter** - Pravi novi parametar.
- **ExecuteNonQuery** - Izvršava komandu na objektu Connection i vraca broj izmenjenih redova. Koristi se u slučajevima kada SQL naredba ili SP ne vraća ni jedan red (INSERT, DELETE, UPDATE)

Još metoda objekta Command

- **Prepare -** Pravi pripremljenu (kompaj]iranu) verziju komande na izvoru podataka.
- **ResetCommandTimeout -** Svojstvo CommandTimeout ponovo postavlja na podrazumevanu vrednost.

Testiranje parametara:

- 1. Kreirajte tabelu:
- *“create table mytable (myname varchar(30), mynumber integer)”*
- 2. Insert komanda*:*
- *“insert into mytabel values(@myname, @mynumber)”*
- 3. Dodavanje parametara
- *cmd.Parameters.Add(“@myname”, SqlDbType.VarChar, 30);*
- *cmd.Parameters.Add(“@mynumber”, SqlDbType.Int);*

…

- 4. Priprema komande
- cmd.Prepare();
- 5. Dodeljivanje vrednosti i izvršavanje
- string[] names={“pera”, “aca”, “caca”, “mica”};
- for(int i=0; i<names.length; i++){
  - cmd.Parameters[“@myname”].Value = names[i];
  - cmd.Parameters[“@mynumber”].Value = i;
  - cmd.ExecuteNonQuery();
- }

A DataReader?

- DataReader se koristi kada komanda vraća skup zapisa.
- Tok podataka koji vraća DataReader je takav da se može **samo čitati** i kretati **samo unapred.**
- U memoriji se istovremeno nalazi samo jedan red podataka!