SQL в EntityFramework

Работа с SQL

Последнее обновление: 31.10.2015

В большинстве случае разработчики смогут создать эффективные запросы с помощью методов и операторов LINQ. Однако в Entity Framework доступно также прямое выполнение sql-запросов.

Для осуществления прямых sql-запросов к базе данных можно воспользоваться свойством **Database**, которое имеется у класса контекста данных. Данное свойство позволяет получать информацию о базе данных, подключении и осуществлять запросы к БД. Например, получим строку подключения:

```
using(PhoneContext db = new PhoneContext())
{
    Console.WriteLine(db.Database.Connection.ConnectionString);
}
```

Непосредственно для создания запроса нам надо использовать метод **SqlQuery**, который принимает в качестве параметра sql-выражение.

Для примера возьмем базу данных, созданную в прошлой теме, которая описывается следующими моделями и контекстом данных:

```
class PhoneContext : DbContext
{
    public PhoneContext() :base("DefaultConnection")
    {}

    public DbSet<Company> Companies { get; set; }
    public DbSet<Phone> Phones { get; set; }
}

public class Company
{
    public int Id { get; set; }
    public string Name { get; set; }

    public ICollection<Phone> Phones { get; set; }
    public Company()
    {
        Phones = new List<Phone>();
    }
}

public class Phone
{
    public int Id { get; set; }
    public string Name { get; set; }
```

```
    public int Price { get; set; }

    public int CompanyId { get; set; }
    public Company Company { get; set; }
}
```

Итак, получим все модели из таблицы Companies:

```
using(PhoneContext db = new PhoneContext())
{
    var comps = db.Database.SqlQuery<Company>("SELECT * FROM Companies");
    foreach (var company in comps)
        Console.WriteLine(company.Name);
}
```

Выражение SELECT извлекает данные из таблицы. Так как эта таблица сопоставляется с моделью Company и хранит объекты этой модели, то данный вызов типизируется классом Company: db.Database.SqlQuery<Company>()

Другая версия метода SqlQuery() позволяет использовать параметры. Например, выберем из бд все модели, которые в названии имеют подстроку "Samsung":

```
using(PhoneContext db = new PhoneContext())
{
    System.Data.SqlClient.SqlParameter param = new System.Data.SqlClient.SqlParameter("@name", "%Samsung%");
    var phones = db.Database.SqlQuery<Phone>("SELECT * FROM Phones WHERE Name LIKE @name",param);
    foreach (var phone in phones)
        Console.WriteLine(phone.Name);
}
```

Класс **SqlParameter** из пространства имен System.Data.SqlClient позволяет задать параметр, который затем передается в запрос sql.

Метод SqlQuery() осуществляет выборку из БД, но кроме выборки нам, возможно, придется удалять, обновлять уже существующие или вставлять новые записи. Для этой цели применяется метод **ExecuteSqlCommand()**, который возвращает количество затронутых командой строк:

```
// вставка
int numberOfRowInserted = db.Database.ExecuteSqlCommand("INSERT INTO Companies (Name) VALUES ('HTC')");
// обновление
int numberOfRowUpdated = db.Database.ExecuteSqlCommand("UPDATE Companies SET Name='Nokia' WHERE Id=3");
// удаление
int numberOfRowDeleted = db.Database.ExecuteSqlCommand("DELETE FROM Companies WHERE Id=3");
```

Назад **Содержание** **Вперед**