



Управление транзакциями

Последнее обновление: 31.10.2015



Когда мы вызываем при добавлении, обновлении, удалении метод `SaveChanges()`, то фактически Entity Framework проводит транзакцию. В приложении на C# мы также управлять транзакциями. Когда может быть полезно ручное управление транзакциями? Транзакции применяются чаще всего для того, чтобы выполнить последовательность операций, которые должны отличаться высокой согласованностью, и при этом иметь возможность откатить все сделанные из этих операций назад, если какая-нибудь из этих операций завершилась с ошибкой.

Рассмотрим на примере. Например, у нас есть в базе данных человек по имени Bob. У него родился сын, которого тоже назвали Bob. И теперь, чтобы их разграничить, отцу мы присваиваем имя Bob Senior, а сыну - Bob Junior:

```
using(UserContext db = new UserContext())
{
    using(var transaction = db.Database.BeginTransaction())
    {
        try
        {
            Person p1 = db.People.FirstOrDefault(p => p.Name == "Bob");
            p1.Name = "Bob Senior";
            db.Entry(p1).State = EntityState.Modified;
            Person p2 = new Person { Name = "Bob Junior", Age = 1 };
            db.People.Add(p2);
            db.SaveChanges();
            transaction.Commit();
        }
        catch(Exception ex)
        {
            transaction.Rollback();
        }
    }

    foreach(Person p in db.People.ToList())
        Console.WriteLine("Name: {0} Age: {1}", p.Name, p.Age);
}
```

Для создания транзакции используется выражение `var transaction = db.Database.BeginTransaction()`, и так как класс **DbContextTransaction** реализует интерфейс `IDisposable`, то весь код транзакции обертывается в конструкцию `using`.

Далее производятся все те же обычные операции редактирования и добавления. После операций вызывается метод `transaction.Commit()` для коммита транзакции.

Однако если, допустим, у нас возникнет исключение параллелизма или любое другое исключение при редактировании или добавлении, то есть одна из операций (или обе) завершатся неудачно, то они в целом смысла уже не будут иметь. Поэтому надо будет откатить все сделанные изменения с помощью метода `transaction.Rollback()`.

[Назад](#) [Содержание](#) [Вперед](#)



Стилизатор изображений Ашманова. ▾

 ashmanov.net



Яндекс.Директ

Курсы Learn Python в Минске ▾

 itstep.by



Яндекс.Директ

