

Базы данных

Юрий Литвинов
y.litvinov@spbu.ru

07.11.2024

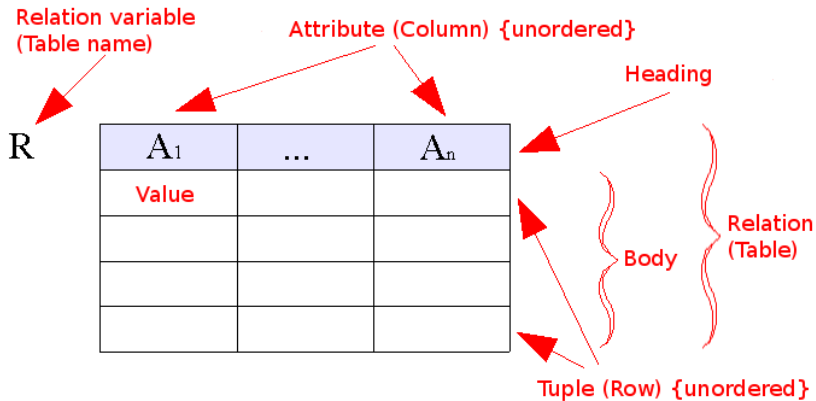
СУБД

- ▶ Реляционные
 - ▶ Отношения
 - ▶ Операции
- ▶ Объектно-ориентированные
 - ▶ Сериализованные объекты
- ▶ Иерархические
- ▶ ...

Реляционные vs ОО-СУБД

- ▶ Реляционные
 - ▶ Сложность интеграции с ОО-кодом
 - ▶ ORM (Microsoft Entity Framework, Hibernate, MyBatis, ...)
 - ▶ Эффективные и выразительные запросы
- ▶ Объектно-ориентированные
 - ▶ Проще, легче, не требуют ORM
 - ▶ “Бедный” язык запросов
 - ▶ Часто не умеют того, что для реляционных СУБД естественно (например, транзакций)

Реляционная модель данных



© Wikipedia

Пример таблицы

| CustomerID | TaxID | Name | Address |
|------------|-------------|-----------|------------------|
| 1234567890 | 555-5512222 | Munmun | 323 Broadway |
| 2223344556 | 555-5523232 | Wile E. | 1200 Main Street |
| 3334445563 | 555-5533323 | Ekta | 871 1st Street |
| 423242432 | 555-5325523 | E.F. Codd | 123 It Way |

Ключи

- ▶ Первичные (primary)
 - ▶ Естественные
 - ▶ Составные
 - ▶ Суррогатные
- ▶ Внешние (foreign)

CITY

| ID | Name |
|----|-----------------|
| 1 | Москва |
| 2 | Санкт-Петербург |
| 3 | Владивосток |

STREET

| ID | Name | ID_CITY |
|-----|------------------|---------|
| 181 | Малая Бронная | 1 |
| 182 | Тверской Бульвар | 1 |
| 183 | Невский проспект | 2 |
| 184 | Пушкинская | 2 |
| 185 | Светланская | 3 |
| 186 | Пушкинская | 3 |

Ограничения

- ▶ PRIMARY KEY
- ▶ FOREIGN KEY
- ▶ NOT NULL
- ▶ UNIQUE
- ▶ ...

SQL SELECT

| Таблица «Т» | Запрос | Результат | | | | | | | | | | | | |
|--|--------|-----------|---|---|---|---|--|--|----|----|---|---|---|---|
| <table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table> | C1 | C2 | 1 | a | 2 | b | <pre>SELECT * FROM T;</pre> | <table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table> | C1 | C2 | 1 | a | 2 | b |
| C1 | C2 | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |
| 2 | b | | | | | | | | | | | | | |
| C1 | C2 | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |
| 2 | b | | | | | | | | | | | | | |
| <table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table> | C1 | C2 | 1 | a | 2 | b | <pre>SELECT C1 FROM T;</pre> | <table><tr><th>C1</th></tr><tr><td>1</td></tr><tr><td>2</td></tr></table> | C1 | 1 | 2 | | | |
| C1 | C2 | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |
| 2 | b | | | | | | | | | | | | | |
| C1 | | | | | | | | | | | | | | |
| 1 | | | | | | | | | | | | | | |
| 2 | | | | | | | | | | | | | | |
| <table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table> | C1 | C2 | 1 | a | 2 | b | <pre>SELECT * FROM T WHERE C1 = 1;</pre> | <table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr></table> | C1 | C2 | 1 | a | | |
| C1 | C2 | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |
| 2 | b | | | | | | | | | | | | | |
| C1 | C2 | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |
| <table><tr><th>C1</th><th>C2</th></tr><tr><td>1</td><td>a</td></tr><tr><td>2</td><td>b</td></tr></table> | C1 | C2 | 1 | a | 2 | b | <pre>SELECT * FROM T ORDER BY C1 DESC;</pre> | <table><tr><th>C1</th><th>C2</th></tr><tr><td>2</td><td>b</td></tr><tr><td>1</td><td>a</td></tr></table> | C1 | C2 | 2 | b | 1 | a |
| C1 | C2 | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |
| 2 | b | | | | | | | | | | | | | |
| C1 | C2 | | | | | | | | | | | | | |
| 2 | b | | | | | | | | | | | | | |
| 1 | a | | | | | | | | | | | | | |

SELECT, вложенные запросы

```
SELECT isbn,  
        title,  
        price  
FROM Book  
WHERE price < (SELECT AVG(price) FROM Book)  
ORDER BY title;
```

INNER JOIN

City (Города)

| <u>Id</u> | Name |
|-----------|-----------------|
| 1 | Москва |
| 2 | Санкт-Петербург |
| 3 | Казань |

Person (Люди)

| <u>Name</u> | CityId |
|-------------|--------|
| Андрей | 1 |
| Леонид | 2 |
| Сергей | 1 |
| Григорий | 4 |

```
SELECT *
FROM
  Person
  INNER JOIN
  City
  ON Person.CityId = City.Id
```

| Person.Name | Person.CityId | City.Id | City.Name |
|-------------|---------------|---------|-----------------|
| Андрей | 1 | 1 | Москва |
| Леонид | 2 | 2 | Санкт-Петербург |
| Сергей | 1 | 1 | Москва |

© Wikipedia

OUTER JOIN

City (Города)

| <u>Id</u> | Name |
|-----------|-----------------|
| 1 | Москва |
| 2 | Санкт-Петербург |
| 3 | Казань |

Person (Люди)

| <u>Name</u> | CityId |
|-------------|--------|
| Андрей | 1 |
| Леонид | 2 |
| Сергей | 1 |
| Григорий | 4 |

```
SELECT *
FROM
  Person
  LEFT OUTER JOIN
  City
  ON Person.CityId = City.Id
```

| Person.Name | Person.CityId | City.Id | City.Name |
|-------------|---------------|---------|-----------------|
| Андрей | 1 | 1 | Москва |
| Леонид | 2 | 2 | Санкт-Петербург |
| Сергей | 1 | 1 | Москва |
| Григорий | 4 | NULL | NULL |

© Wikipedia

CROSS JOIN

City (Города)

| <u>Id</u> | Name |
|-----------|-----------------|
| 1 | Москва |
| 2 | Санкт-Петербург |
| 3 | Казань |

Person (Люди)

| <u>Name</u> | CityId |
|-------------|--------|
| Андрей | 1 |
| Леонид | 2 |
| Сергей | 1 |
| Григорий | 4 |



```
SELECT *
FROM
  Person,
  City
```



| Person.Name | Person.CityId | City.Id | City.Name |
|-------------|---------------|---------|-----------------|
| Андрей | 1 | 1 | Москва |
| Андрей | 1 | 2 | Санкт-Петербург |
| Андрей | 1 | 3 | Казань |
| Леонид | 2 | 1 | Москва |
| Леонид | 2 | 2 | Санкт-Петербург |
| Леонид | 2 | 3 | Казань |
| Сергей | 1 | 1 | Москва |
| Сергей | 1 | 2 | Санкт-Петербург |
| Сергей | 1 | 3 | Казань |
| Григорий | 4 | 1 | Москва |
| Григорий | 4 | 2 | Санкт-Петербург |
| Григорий | 4 | 3 | Казань |

© Wikipedia

INSERT, UPDATE, DELETE

INSERT:

```
INSERT INTO phone_books VALUES ('Peter Doe', '555-2323');
```

UPDATE:

```
UPDATE persons SET  
    street = 'Nissestien 67',  
    city = 'Sandnes',  
WHERE lastname = 'Tjessem' AND firstname = 'Jakob';
```

DELETE:

```
DELETE ab, b  
FROM Authors AS a, AuthorArticle AS ab, Articles AS b  
WHERE a.AuthID = ab.AuthID AND ab.ArticleID = b.ArticleID  
    AND AuthorLastName = 'Henry';
```

Работа с метайнформацией

CREATE TABLE:

```
CREATE TABLE Students (  
  Code INTEGER NOT NULL,  
  Name NCHAR(30) NOT NULL,  
  Address NVARCHAR(50),  
  Mark DECIMAL);
```

DROP TABLE:

```
DROP TABLE Students;
```



Работа с метайнформацией

ALTER TABLE:

ALTER TABLE Students **ADD** email **VARCHAR(MAX)**;

ALTER TABLE Students **DROP COLUMN** email;

ALTER TABLE Students **ADD PRIMARY KEY** (Code);

Низкий уровень работы с данными, ADO.NET

- ▶ Возможность исполнять SQL-запросы для разных источников данных
- ▶ Data Provider обеспечивает общение с конкретной СУБД
- ▶ Connection String описывает, как подключиться к СУБД
- ▶ Command представляет абстракцию команды в СУБД
- ▶ DataSet обеспечивает более-менее высокоуровневое представление данных
- ▶ Может работать даже с XML или таблицами Excel
- ▶ Пространство имён System.Data
- ▶ Лучше не использовать в современном коде

Пример, чтение из базы

```
public static void Main()
{
    using (var connection = new MySqlConnection(
        "database=cities;server=localhost;user id=root;" +
        "Password=my-secr3t-p4ssw0rd;SslMode=none"))
    {
        var command = new MySqlCommand("SELECT Id, Name FROM City", connection);
        connection.Open();
        var reader = command.ExecuteReader();
        while (reader.Read())
        {
            Console.WriteLine($"Id: {reader.GetInt32(0)}\tName:{reader.GetString(1)}");
        }
    }
}
```

Пример, добавление в базу

```
public static async Task Main()
{
    using (var connection = new MySqlConnection(
        "database=cities;server=localhost;user id=root;" +
        "Password=my-secr3t-p4ssw0rd;SslMode=none"))
    {
        var command = new MySqlCommand(
            "INSERT INTO City (name) VALUES (@name)", connection);
        command.Parameters.AddWithValue("@name", "Peterhof");
        connection.Open();
        await command.ExecuteNonQueryAsync();
        Console.WriteLine($"Done, inserted row id = {command.LastInsertedId}");
    }
}
```

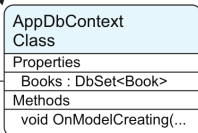
Высокий уровень работы с данными, Entity Framework

- ▶ ORM (Object-Relational Mapping) системы — преобразуют реляционные данные в объекты
- ▶ ORM-библиотека берёт на себя общение с базой
 - ▶ И даже генерацию SQL-запросов
 - ▶ Типобезопасность
- ▶ Entity Framework Core — одна из реализаций для .NET (.NET Core)

Как она работает

1. Looks at all the DbSet properties.

Your Application



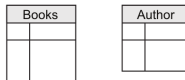
The EF Core library

Model the database

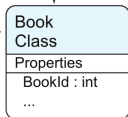
1. Look at DbSet<T> properties
2. Look at the class for columns
3. Inspect linked classes
4. Run OnModelCreating method

Output

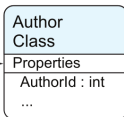
Database model (cached)



2. Looks at the properties in the class.



3. Does the same to any linked classes.



4. Runs OnModelCreating, if present.

5. The final result: a model of the database.

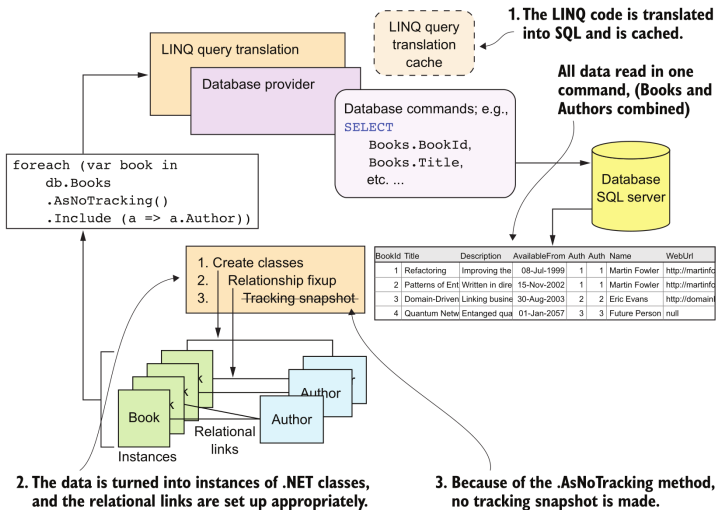
© J. Smith, "Entity Framework Core In Action"

Как это примерно выглядит в коде

Запрос к базе

```
public static void ListAll()
{
    using (var db = new AppDbContext())
    {
        foreach (var book in db.Books.AsNoTracking().Include(a => a.Author))
        {
            var webUrl = book.Author.WebUrl == null
                ? "- no web URL given -"
                : book.Author.WebUrl;
            Console.WriteLine($"{book.Title} by {book.Author.Name}");
            Console.WriteLine("Published on "
                + $"{book.PublishedOn:dd-MMM-yyyy}"
                + $" {webUrl}");
        }
    }
}
```

Что в это время делает EF



© J. Smith, "Entity Framework Core In Action"

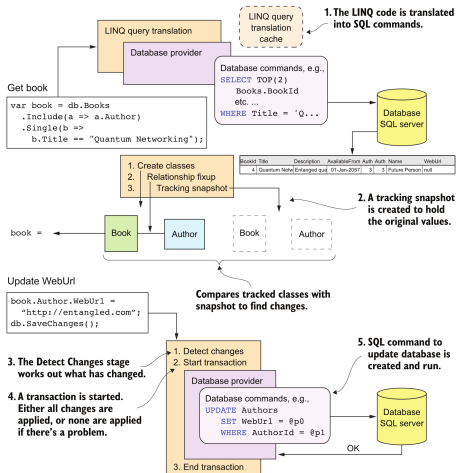
Сгенерированный SQL

```
SELECT b.BookId,  
        b.AuthorId,  
        b.Description,  
        b.PublishedOn,  
        b.Title,  
        a.AuthorId,  
        a.Name,  
        a.WebUrl  
FROM Books AS b  
INNER JOIN Author AS a ON  
        b.AuthorId = a.AuthorId
```

Обновление данных

```
public static void ChangeWebUrl()
{
    Console.WriteLine("New Quantum Networking WebUrl > ");
    var newWebUrl = Console.ReadLine();
    using (var db = new AppDbContext())
    {
        var book = db.Books.Include(a => a.Author)
            .Single(b => b.Title == "Quantum Networking");
        book.Author.WebUrl = newWebUrl;
        db.SaveChanges();
        Console.WriteLine("... SavedChanges called.");
    }
}
```


Что в это время делает EF



© J. Smith, "Entity Framework Core In Action"