Delete a single record from Entity Framework?

Ask Question

166

28

I have a SQL Server table in Entity Framework named `employ` with a single key column named `ID`.

How do I delete a single record from the table using Entity Framework?

c# sql-server entity-framework

edited Feb 2 '18 at 6:45
CharithJ
36.4k 16 93 112

asked Jul 18 '13 at 12:20
user2497476
883 2 7 6

db.employ.Remove(db.employ.Find(ID1)) – Carter Medlin Dec 1 '16 at 19:17

1 @CarterMedlin - while that will work, those are two database hits: one SELECT and one DELETE. Most people find that extremely wasteful, especially since select will probably take significantly more time than a delete. – Davor Jul 4 '17 at 13:48

I would not suggest to use entity

framework Remove or RemoveRange due to the performance issues. I would rather just use something super simple as following: var sql = "DELETE FROM YOUR_TABLE WHERE YOUR_FIELD= @your_parameter"; this.your_context.Database.ExecuteSqlCommand(sql, new SqlParameter("@your_parameter", yourParameter)); – curiousBoy Apr 24 '18 at 0:44

@curiousBoy I think that when you execute statements like you suggested, the EF6 cache doesn't reflect the change. – Yitzchak Jul 8 '18 at 10:09

12 Answers

313

✓

It's not necessary to query the object first, you can attach it to the context by its id. Like this:

```
var employer = new Emp
ctx.Employ.Attach(empl
ctx.Employ.Remove(empl
ctx.SaveChanges();
```

Alternatively, you can set the attached entry's state to deleted :

```
var employer = new Emp
ctx.Entry(employer).St
ctx.SaveChanges();
```

edited Feb 20 '18 at 18:26

Brian Webster
20.6k 40 130 208

answered Jul 18 '13 at 12:37

mt_serg
5,832 2 21 41

76 Alternatively, `ctx.Entry(employer).State = EntityState.Deleted` – Simon Belanger Jul 18 '13 at 12:44

11 this will only work if the relationships are defined as delete cascade. otherwise the code above will fail on an FK exception. – baruchl Sep 29 '14 at 19:08

5 @mt_serg, I'm looking 3 steps ahead. when was the last time you really had to remove such a simple record from the DB? usually you are dealing with more complex records that include FK relations. hence my comment. – baruchl Sep 30 '14 at 18:46

2 @IanWarburton The 2nd and 3rd line (Attach and Remove) – Simon Belanger May 25 '16 at 11:22

2 @PaulZahra: sometimes you have a list of IDs from some other query or source, and you need to delete one. Rather than loading up the objects just to delete them, this way you can delete

by ID. You know, that's how the DELETE statement works in SQL normally. – siride Jul 11 '16 at 18:18

70

You can use `SingleOrDefault` to get a single object matching your criteria, and then pass that to the `Remove` method of your EF table.

```
var itemToRemove = Con
item.

if (itemToRemove != nu
    Context.Employ.Rem
    Context.SaveChange
}
```

answered Jul 18 '13 at 12:24

Mansfield
7,847 14 58 95

2 this is not good way, because you are select all field from database! – Ali Yousefie May 12 '16 at 5:31

2 This is the way I do it. – Jack Fairfield Aug 31 '16 at 20:22

@JackFairfield checkout the accepted answer. There is no need to hit the database and retrieve data. EF only looks at the Id of the object being deleted anyway. – Chazt3n Dec 27 '16 at 23:34

4 @Ali, Jack - But I think this is preferable because it

first checks if the data you are trying to delete actually exists which can prevent any trouble. The accepted answer has no check as such. – Michael Philips Mar 20 '17 at 7:31

4 This is the better way. Think about it. What if John Smith is trying to remove an item with an id = 1 that Susie Smith removed 30 seconds ago but John doesn't know? You need to hit the database in that case. – Yusha Jan 9 '18 at 17:47

12

```
var stud = (from s1 i
            where s1.II
            select s1)

//Delete it from memo
entities.DeleteObject
//Save to database
entities.SaveChanges
```

edited Feb 21 '18 at 18:35

Brian Webster
20.6k 40 130 208

answered Jul 18 '13 at 12:27

Alex G
416 4 16

1 `FirstOrDefault` is dangerous. Either you know there's only one (so use `SingleOrDefault`), or there is more than one, and it should be done in a loop. – Mark Sowul Feb 6 '18 at 19:36

8

```
Employer employer = co

context.Customers.Dele
context.SaveChanges();
```

answered Jul 18 '13 at 12:24

Sam Leach
9,388 7 35 65

Does this protect if there is no object with Id 1? Wouldn't it throw an exception? – Jack Fairfield Aug 31 '16 at 20:22

@JackFairfield i think you should check for null object. and according to it perform remove. – Jawand Singh Apr 7 '17 at 13:38

`First` is dangerous. Either you know there's only one (so use `Single`), or there is more than one, and it should be done in a loop. – Mark Sowul Feb 6 '18 at 19:34

4

I am using entity framework with LINQ. Following code was helpful for me;

1- For multiple records

```
using (var dbContext =
{
    var allRec= dbCon
    dbContext.myEntit
    dbContext.SaveCha
}
```

2- For Single record

```
  using (var dbContext =
  {
      var singleRec = dl
object your want to de
      dbContext.ChatUse
      dbContext.SaveCha
  }
```

edited Feb 7 '18 at 15:38

answered Jun 23 '17 at 9:28

Baqer Naqvi
1,815 2 23 36

For Single record why not use `SingleOrDefault` instead of `FirstOrDefault` ? – Mark Sowul Feb 6 '18 at 19:35

Whenever you use SingleOrDefault, you clearly state that the query should result in at most a single result. On the other hand, when FirstOrDefault is used, the query can return any amount of results but you state that you only want the first one stackoverflow.com/a/1745716/3131402 – Baqer Naqvi Feb 7 '18 at 10:47

1 Yes, so why would it be correct to delete an arbitrary record, if there is more than one? Particularly in this case the id is the key, so there should be one: if there is more than one, it is a bug (which Single would detect) – Mark Sowul Feb 7 '18 at 14:56

@MarkSowul you are right. I have edited the answer to use FirstOrDefault. – Baqer Naqvi Feb 7 '18 at 15:39

@BaqerNaqvi RemoveRange is terrible way to remove entity from the performance perspective.. Especially when your entity is heavy with all the navigational properties by foreign keys. I would rather use var sql = "DELETE FROM YOUR_TABLE WHERE YOUR_FIELD= @your_parameter"; this.your_context.Database.ExecuteSqlCommand(sql, new SqlParameter("@your_parameter", yourParameter)); – curiousBoy Apr 24 '18 at 0:43

1

More generic approuch

```
public virtual void De
{
    T instance = Activ
    instance.Id = id;
    if (dbContext.Entr
    {
        dbContext.Set<
    }

    dbContext.Set<T>()
}
```

answered Feb 11 '18 at 10:52

valentasm
375 2 9

u can do it simply like

1

this

```
   public ActionResult
    {
        using (var db =
        {
            Models.Reg
            Registratio
db.RegisterDbTable.Find
            if (persona
            {
                return
            }
            else
            {
                Obj.Use
                Obj.Fi
                Obj.La
                Obj.Ci

            }
            return View
        }
    }


    [HttpPost, ActionNa

    public ActionResult
    {
        using (var db =
        {
            Registratio
db.RegisterDbTable.Find
            db.Register
            db.SaveChan
            return Red
        }
    }
```

model

```
  public class Register
 {

    public int UserID
    { get; set; }


    public string First
    { get; set; }


    public string LastN
    { get; set; }


    public string Passw
    { get; set; }


    public string City
    { get; set; }
```

```
}
```

view from which u will call it

```
 <table class="table">
    <tr>
        <th>
            FirstName
        </th>
        <th>
            LastName
        </th>

        <th>
            City
        </th>
        <th></th>
    </tr>

    @foreach (var item
    {
        <tr>
            <td> @item
            <td> @item
            <td> @item
            <td>
                <a hre
})">Edit</a> |
                <a hre
})">Details</a> |
                <a hre
})">Delete</a>

            </td>
        </tr>

    }

</table>
```

i hope this will be easy for u to understand

answered Oct 3 '18 at 13:18

Sikander Iqbal
58 7

0

For generic DAO my work finnaly this:

```
    public void Detele
    {
        db.Entry(entity
        db.SaveChanges
    }
```

answered May 25 '18 at 20:19

Tom Trnka
1 2

0

Using [EntityFramework.Plus](#) could be an option:

```
dbContext.Employ.Where
```

More examples are available [here](#)

answered Aug 18 '18 at 12:07

Mohammad Reza Sadreddini
158 1 10

0

You can do something like this in your click or celldoubleclick event of your grid(if you used one)

```
if(dgEmp.CurrentRow.In
 {
    employ.Id = (Int32
    //Some other stuff
 }
```

Then do something like this in your Delete Button:

```
using(Context context 
{
     var entry = conte
     if(entry.State ==
     {
        //Attached it 
        context.Employ
     }
     //Use Remove meth
     context.Employee.
```

```
        //Finally, execute
        //to the actual ta
        context.SaveChange

        //Some stuff here
    }
```

Alternatively, you can use a LINQ Query instead of using LINQ To Entities Query:

```
var query = (from emp :
where emp.Id == employ
select emp).Single();
```

employ.Id is used as filtering parameter which was already passed from the CellDoubleClick Event of your DataGridView.

edited Dec 21 '18 at 5:12

answered Dec 21 '18 at 3:19

a arvin aquio
1 3

The Idea behind the code is you wire the id(employ.Id) of the record you want to delete to the model(Employee Class) and then attach it to the actual Table from the Context then execute in-memory Remove() Method then finally execute actual saving to the database using SaveChanges() Method. Though the LINQ Query also works fine but I don't like the idea of querying to the table just to get the id of the record. –

arvin aquio Dec 21 '18 at 5:53

0

With Entity Framework 6, you can use `Remove` . Also it 's a good tactic to use `using` for being sure that your connection is closed.

```
using (var context = n
{
    Employ emp = conte
    context.Employ.Rem
    context.SaveChange
}
```

answered Dec 3 '18 at 14:57

Gizmo
11 3

0

```
    [HttpPost]
    public JsonResult
    {
        using (Mycased
        {
            Contact ro



            dbde.Conta
            dbde.SaveC

            return Jso
        }
    }
```

What do you think of this, simple or not, you could also try this:

```
        var productrow
        cnn.Product.Re
        cnn.SaveChange
```

edited Apr 18 '18 at 14:50

answered Apr 18 '18 at 14:03

Namroy

1 1