Distinct in Linq based on only one field of the table

Ask Question

99

16

I am trying to use .distinct in Linq to get result based on one field of the table (so do not require a whole duplicated records from table).

I know writing basic query using distinct as followed:

```
var query = (from r in table1
orderby r.Text
select r).distinct();
```

but I need results where `r.text` is not duplicated.

c# sql linq

edited Jan 9 '15 at 0:18
Shimmy
51.4k 102 342 552

asked Jan 14 '13 at 15:07
Megha Jain
622 1 6 8

You need to specify what field you want to be distinct ,see msdn.microsoft.com/en-us/library/bb348436.aspx – Antarr Byrd Jan 14 '13 at 15:10

Home

PUBLIC

Stack Overflow

9 Answers

Jobs

Teams
Q&A for work

Learn More

244

Try this:

```
table1.GroupBy(x => x.Text).Select(x => x.FirstOrDefault()
```

This will group the table by `Text` and use the first row from each groups resulting in rows where `Text` is distinct.

edited Aug 23 '17 at 14:19

answered Jan 14 '13 at 15:09
Daniel Hilgarth
141k 33 251 360

What if groupby has more than 1 field? – user585440 Jan 6 '16 at 2:01

3 @user585440: In that case, you use an anonymous type like so: `table1.GroupBy(x => new { x.Text, x.Property2, x.Property3 }).Select(x => x.First());` – Daniel Hilgarth Jan 14 '16 at 12:58

2 Yes, you are right and I already found it. Thanks anyway. And I also find that Select(x => x.First()) can cause crash. It is better to change to Select(x => x.FirstOrDefault()); – user585440 Jan 14 '16 at 21:24

@user585440: In that particular scenario, `First` will never cause an exception, because it takes the first item of each group. And there only is a group, if there is at least one item in it. – Daniel Hilgarth Jan 15 '16 at 5:30

5 I had to use FirstOrDefault or else there was a runtime error – TruthOf42 Apr 27 '16 at 16:02

17

It will allow you to do:

```
var results = table1.DistictBy(row => row.Text);
```

The implementation of the method (short of argument validation) is as follows:

```
private static IEnumerable<TSource> DistinctByImpl<TSource
source,
    Func<TSource, TKey> keySelector, IEqualityComparer<TKey
{
    HashSet<TKey> knownKeys = new HashSet<TKey>(comparer);
    foreach (TSource element in source)
    {
        if (knownKeys.Add(keySelector(element)))
        {
            yield return element;
        }
    }
}
```

edited Apr 24 '17 at 0:49
Shiva
14.6k 10 64 95

answered Jan 14 '13 at 15:15
Servy
180k 18 242 356

sorry I wasnt keen to use equalityComparer. – Megha Jain Jan 14 '13 at 15:38

@MeghaJain Well, one will be used regardless, as `GroupBy` needs one as well. Both methods will use the default `EqualityComparer` if none is provided. – Servy Jan 14 '13 at 15:39

8 Well, correct me if I am wrong, but this distinct here is done in memory, not in DB ? Couldn't this lead to undesired full-scan ?

distinct element. Eventually, yes, you will load each key into the HashSet, but since it's IEnumerable in and IEnumerable out, you will only get those items. If you are talking about LINQ to SQL, then yes, this will do a table scan. – PRMan May 11 '17 at 16:00

10

> but I need results where r.text is not duplicated

Sounds as if you want this:

```
table1.GroupBy(x => x.Text)
      .Where(g => g.Count() == 1)
      .Select(g => g.First());
```

This will select rows where the `Text` is unique.

edited Mar 18 '14 at 18:59

answered Jan 14 '13 at 15:09

Rango
368k 46 479 745

3

There are lots of discussions around this topic.

You can find one of them [here](#):

One of the most popular suggestions have been the

The chief architect of C#, Anders Hejlsberg has suggested the solution here. Also explaining why the framework design team decided not to add an overload of Distinct method which takes a lambda.

edited May 23 '17 at 12:34
Community ♦
1 1

answered Jan 14 '13 at 15:46
TKharaishvili
1,304 16 26

3

Daniel Hilgarth's answer above leads to a `System.NotSupported` exception With **Entity-Framework**. With **Entity-Framework**, it has to be:

```
table1.GroupBy(x => x.Text).Select(x => x.FirstOrDefault()
```

edited Jun 11 '18 at 8:33
Roshna Omer
366 1 8 14

answered Sep 17 '16 at 23:44
Biraj Saha
31 3

2

From what I have found, your query is mostly correct. Just change "select r" to "select r.Text" is all and that should solve the problem. This is how MSDN documented how it

```
var query = (from r in table1 orderby r.Text select r.
```

answered Aug 22 '14 at 19:01

Josh Parks
29 3

you changed the "select" statement that may not be desired in this case – faza Sep 17 '18 at 7:45

1

```
data.Select(x=>x.Name).Distinct().Select(x => new SelectLi
```

edited Jan 18 '17 at 6:54

Pang
7,043 16 67 105

answered Jan 18 '17 at 6:26

bgS
101 5

-2

try this code :

```
table1.GroupBy(x => x.Text).Select(x => x.FirstOrDefault()
```

answered Jul 24 '17 at 10:18

HamidReza
581 4 11

-4

You can try this: `table1.GroupBy(t => t.Text).Select(shape => shape.r)).Distinct();`

answered Nov 19 '13 at 11:16

LucaGuerra
104 8