

# Unique key with EF code first



I have a following model in my project

62



25

```
public class Category
{
    public Guid ID { get; set; }
    [Required(ErrorMessage = "Title cannot be empty")]
    public string Title { get; set; }
}
```

and I'm trying to make `Title` as unique key, I googled for the solution, but couldn't find any. Can any suggest me how to do it, please?

c#

ef-code-first

entity-framework-4.1

unique-key

edited Apr 18 '11 at 10:43



Ladislav Mrnka

331k 55 622 646

asked Apr 18 '11 at 10:33



Prashant C

8,133 18 84 155

## 5 Answers



108



Unfortunately you can't define it as unique key in code first because EF doesn't support unique keys at all (it is hopefully planned for next major release). What you can do is to create custom database initializer and add unique index manually by calling SQL command:

```
public class MyInitializer : CreateDatabaseIfNotExists<MyContext>
{
    protected override void Seed(MyContext context)
    {
        context.Database.ExecuteSqlCommand("CREATE UNIQUE INDEX IX_Category_Title ON
Categories (Title)");
    }
}
```

Join Stack Overflow to learn, share knowledge, and build your career.

[Sign up with email](#)[Sign up with Google](#)[Sign up with Facebook](#)

```
Database.SetInitializer<MyContext>(new MyInitializer());
```

## Edit

Now (EF 6.1 onwards )you can easily have unique constrains ,

```
[Index("TitleIndex", IsUnique = true)]
public string Title { get; set; }
```

edited Jun 9 '14 at 5:27



Community ♦

1 1

answered Apr 18 '11 at 10:42



Ladislav Mrnka

331k 55 622 646

I work with MVC 3 and EF 4 and the code don't recognize ExecuteSqlCommand in context.Database.ExecuteSqlCommand("CREATE UNIQUE INDEX IX\_Category\_Title ON Categories (Title)"); is this about version or otherthing? – [Saeid](#) Nov 16 '11 at 10:48

1 @Saeid: This is for DbContext API (EFv4.1). There is no database initializer in EFv4.ObjectContext API offers its own methods to execute SQL directly - ExecuteStoreCommand . – [Ladislav Mrnka](#) Nov 16 '11 at 11:01

1 Also a great way to add default constraints (e.g. GETDATE(), etc.) – [John Laffoon](#) Sep 28 '12 at 13:49

2 Seed is executed multiple time - would this not error out as the index (or function/stored procedure/or whatever) already exists in the database? – [codeputer](#) Feb 22 '13 at 22:30

@codputer: In this case the Seed is executed only once because it doesn't use migrations. In case of migrations you can create index directly in Up method. – [Ladislav Mrnka](#) Feb 25 '13 at 9:06

First create the custom attribute class:

22

```
[AttributeUsage(AttributeTargets.Property, AllowMultiple = false, Inherited = true)]
public class UniqueAttribute : ValidationAttribute
{
    public override Boolean IsValid(Object value)
    {

```

Join Stack Overflow to learn, share knowledge, and build your career.

Sign up with email



Sign up with Google

Sign up with Facebook



Then add to your classes:

```
public class Email
{
    [Key]
    public int EmailID { get; set; }

    public int PersonId { get; set; }

    [Unique]
    [Required]
    [MaxLength(100)]
    public string EmailAddress { get; set; }
    public virtual bool IsDefault { get; set; }
    public virtual Boolean IsApprovedForLogin { get; set; }
    public virtual String ConfirmationToken { get; set; }

    [ForeignKey("PersonId")]
    public virtual Person Person { get; set; }
}
```

Then add a Initializer on your DbContext:

```
public class Initializer : IDatabaseInitializer<myEntities>
{
    public void InitializeDatabase(myEntities context)
    {
        if (System.Diagnostics.Debugger.IsAttached && context.Database.Exists() &&
!context.Database.CompatibleWithModel(false))
        {
            context.Database.Delete();
        }

        if (!context.Database.Exists())
        {
            context.Database.Create();

            var contextObject = context as System.Object;
            var contextType = contextObject.GetType();
            var properties = contextType.GetProperties();
        }
    }
}
```

Join **Stack Overflow** to learn, share knowledge, and build your career.


Sign up with email

 Sign up with Google

Sign up with Facebook





- 1 Couple of corrections. 1. tableName should be bracket enclosed during ExecuteSqlCommand 2. if you are using non-pluralized names, use else { tableName = t.Name } – [James](#) Oct 25 '12 at 21:29 

Here is the VB.Net version - note the implementation of generics that is a little different, at the class level.

2

```
Public Class MyInitializer(Of T As DbContext)
    Inherits CreateDatabaseIfNotExists(Of T)
    Protected Overrides Sub Seed(context As T)
        context.Database.ExecuteSqlCommand("CREATE UNIQUE INDEX IX_Category_Title ON
Categories (Title)")
    End Sub
End Class
```

answered Nov 12 '11 at 9:26



[GilShalit](#)

3,182 8 34 52

oh come on - what's wrong with adding a concise VB version, for vb users with the exact same issue? isn't this the point of SO - providing a resource not only for the original poster? Additionally, as noted, the implementation is somewhat different. – [GilShalit](#) Jun 23 '14 at 4:44

0

I create this class (which ws enhanced from another Stackoverflow answer - [Execute a large SQL script \(with GO commands\)](#)), which allows me to drop in the SQL scripts into a directory, and have them all executed each time they are required (Seed, or Migration). I'm not going to leave this open after I deploy to production, but during development it makes it easy to apply scripts each time the DB is recreated.

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

Join Stack Overflow to learn, share knowledge, and build your career.

Sign up with email



Sign up with Google

Sign up with Facebook



```
using Monitor.Common;

namespace MonitorDB.DataLayer.Migrations
{
    public class ExecuteSQLScripts : Monitor.Common.ExceptionHandling
    {
        public ExecuteSQLScripts()
        {
        }

        public bool ExecuteScriptsInDirectory(DBContext.SolArcMsgMonitorContext context, string
scriptDirectory)
        {
            bool Result = false;
            try
            {
                SqlConnection connection = new
SqlConnection(context.Database.Connection.ConnectionString);
                Server server = new Server(new ServerConnection(connection));

                DirectoryInfo di = new DirectoryInfo(scriptDirectory);
                FileInfo[] rgFiles = di.GetFiles("*.sql");
                foreach (FileInfo fi in rgFiles)
                {

                    FileInfo fileInfo = new FileInfo(fi.FullName);
                    string script = fileInfo.OpenText().ReadToEnd();

                    server.ConnectionContext.ExecuteNonQuery(script);
                }
                Result = true;
            }
            catch (Exception ex)
            {
                CatchException("ExecuteScriptsInDirectory", ex);
            }
            return Result;
        }
    }
}
```

Here is what the VS Solution looks like:

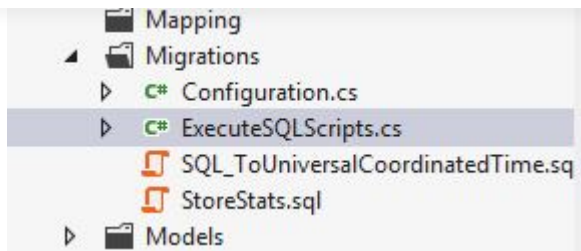
**Join Stack Overflow** to learn, share knowledge, and build your career.

Sign up with email

 Sign up with Google

Sign up with Facebook





edited May 23 '17 at 11:33



Community ♦

1 1

answered Feb 26 '13 at 16:03



codeputer

828 2 12 39

I found this solution which although not creating a unique-key in the SQL level, it uses DataAnnotations validation, check it out:

<http://blogs.microsoft.co.il/blogs/shimmy/archive/2012/01/23/validationattribute-that-validates-a-unique-field-against-its-fellow-rows-in-the-database.aspx>

answered Apr 14 '13 at 13:58



Shimmy

52.6k 107 349 557

Join Stack Overflow to learn, share knowledge, and build your career.

[Sign up with email](#)[Sign up with Google](#)[Sign up with Facebook](#)