

Bieren-DatLayer- EF Core - code first

We starten met de bestaande BierenApp op Github:

Unzip en open de solution in VS 2019

- **Voeg onder de folder Models in Bieren.DataLayer een nieuwe class DbUser toe:**

//EF Core - Code first: eerst Model classes in datalayer en daaruit Database genereren

```
public class DbUser
{
    public DbUser()
    {
        FavorieteBieren = new HashSet<DbBier>();
    }
    public int UserId { get; set; }
    public string Voornaam { get; set; }
    public string Familienaam { get; set; }
    public DateTime GeboorteDatum { get; set; }
    public string Email { get; set; }

    public virtual ICollection<DbBier> FavorieteBieren { get; set; }
}
```

- Voeg aan class DbBier een Property Users toe en initialiseer deze in de constructor:

```
public partial class DbBier
{
    public DbBier()
    {
        Users = new HashSet<DbUser>();
    }
    public int BierNr { get; set; }
    public string Naam { get; set; }
    public int? BrouwerNr { get; set; }
    public int? SoortNr { get; set; }
    public double? Alcohol { get; set; }

    public virtual DbBrouwer BrouwerNrNavigation { get; set; }
    public virtual DbSoort SoortNrNavigation { get; set; }

    public virtual ICollection<DbUser> Users { get; set; }
}
```

- Voeg aan class BierenDbContext een public property toe:

```
public virtual DbSet<DbUser> DbUsers { get; set; }
```

-

Maak een nieuwe database met naam BierenMetUsersDb en kopieer de connectionstring in BierenDbContext in de methode OnConfiguring:

Bv :

```
protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
{
    if (!optionsBuilder.IsConfigured)
    {
        optionsBuilder.UseSqlServer("Data Source=(LocalDB)\\MSSQLLocalDB;Initial
Catalog=BierenMetUsersDb;Integrated Security=True");
    }
}
```

-Er is een **Many-to-Many relatie** tussen dbUser en dbBier. Een User kan meerdere favoriete bieren hebben. En een bier kan het favoriete bier zijn van meerdere Users.

-We definiëren de Many-to-Many relatie dmv **entity.HasMany(e => e.Users);** en **entity.HasMany(e => e.FavorieteBieren);** bij entiteiten DbBier en DbUser in de methode OnModelCreating

-Voeg aan de methode OnModelCreating de volgende lijen toe (geel) :

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.HasAnnotation("Relational:Collation",
"SQL_Latin1_General_CP1_CI_AS");

    modelBuilder.Entity<DbBier>(entity =>
    {
        entity.HasKey(e => e.BierNr)
            .HasName("PK_Bieren");

        entity.ToTable("DbBier");

        entity.Property(e => e.Naam)
            .HasMaxLength(100)
            .IsUnicode(false);

        entity.HasOne(d => d.BrouwerNrNavigation)
            .WithMany(p => p.DbBiers)
            .HasForeignKey(d => d.BrouwerNr)
            .HasConstraintName("FK_Bieren_Brouwers");

        entity.HasOne(d => d.SoortNrNavigation)
            .WithMany(p => p.DbBiers)
            .HasForeignKey(d => d.SoortNr)
            .HasConstraintName("FK_Bieren_Soorten");
entity.HasMany(e => e.Users);

    });

    modelBuilder.Entity<DbBrouwer>(entity =>
    {
        entity.HasKey(e => e.BrouwerNr)
            .HasName("PK_Brouwers");

        entity.ToTable("DbBrouwer");

        entity.Property(e => e.Adres)
            .HasMaxLength(50)
            .IsUnicode(false);

        entity.Property(e => e.BrNaam)
            .HasMaxLength(50)
            .IsUnicode(false);

        entity.Property(e => e.Gemeente)
            .HasMaxLength(50)
            .IsUnicode(false);

    });

    modelBuilder.Entity<DbSoort>(entity =>
    {
        entity.HasKey(e => e.SoortNr)
            .HasName("PK_Soorten");

        entity.ToTable("DbSoort");

        entity.Property(e => e.Soort)
            .HasMaxLength(50)
            .IsUnicode(false);

    });
}
```

```

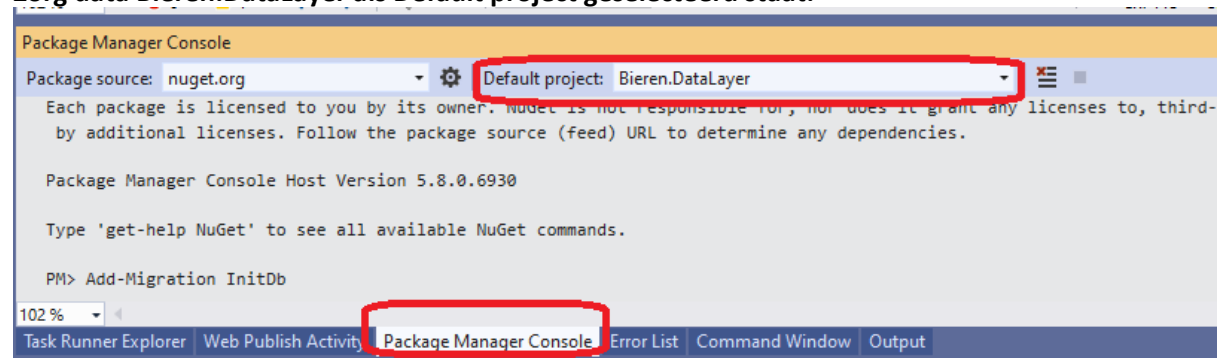
        modelBuilder.Entity<DbUser>(entity =>
        {
            entity.HasKey(e => e.UserId) //UserId van DbUser class wordt gemapt
op kolom met PK 'PK_Users'
                .HasName("PK_Users");
            entity.Property(e => e.UserId).ValueGeneratedOnAdd();
            entity.ToTable("DbUser"); //DbUser class wordt gemapt op tabel met
naam DbUser
            entity.Property(e => e.Voornaam) //Wordt gemapt op kolom varchar(50)
                .HasMaxLength(50)
                .IsUnicode(false);
            entity.Property(e => e.Familienaam)
                .HasMaxLength(50)
                .IsUnicode(false);
            entity.Property(e => e.Email)
                .HasMaxLength(20)
                .IsUnicode(false);
            entity.HasMany(e => e.FavorieteBieren);

        });
    OnModelCreatingPartial(modelBuilder);
}

```

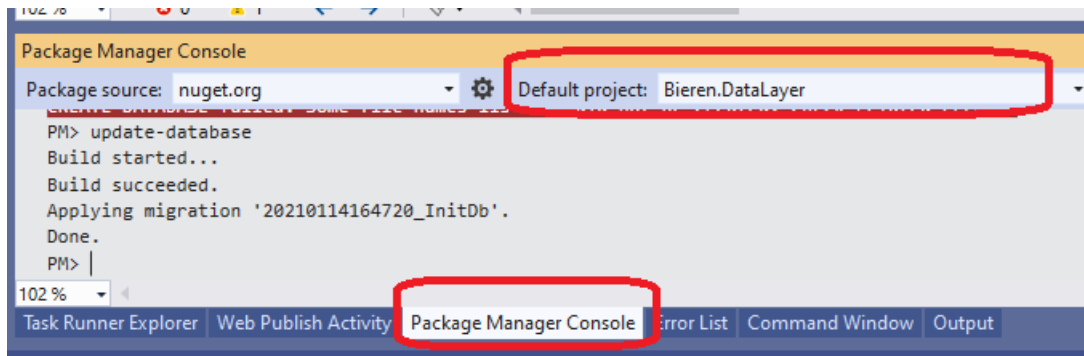
- Open de Package Manager Console en run de volgende instructie :
- **Add-Migration InitDb**

Zorg data **Bieren.DataLayer** als Default project geselecteerd staat:

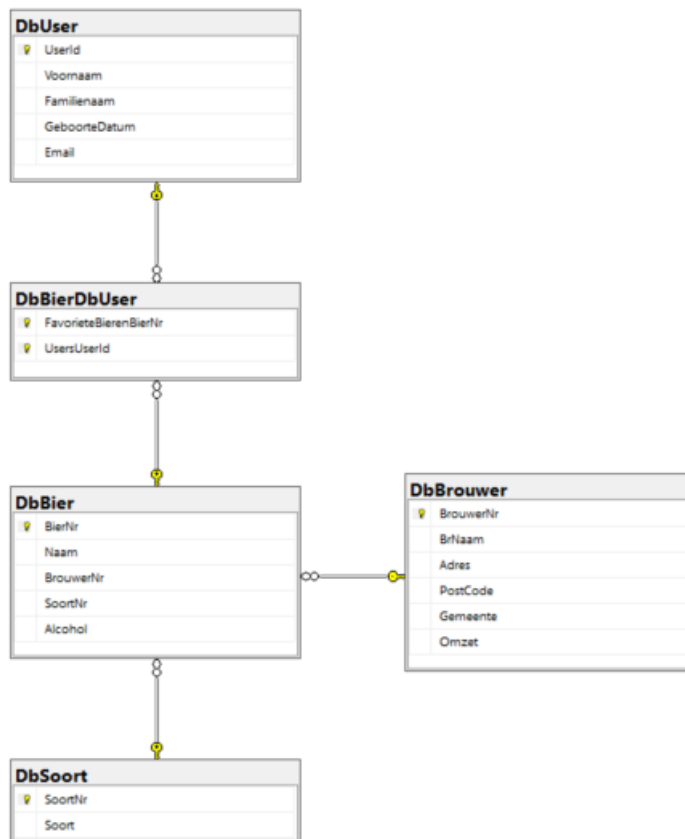


- Een Folder **Migrations** wordt automatisch aangemaakt in het project Bieren.DataLayer, daarin is een class xxxxx **InitDb** aangemaakt. Deze bevat een methode **Up** en **Down**. De **Up** methode wordt uitgevoerd indien we de instructie update-database uitvoeren (deze maakt de tabellen en hun relaties aan in de database). De **Down** methode is om een rollback te kunnen doen en deze zal de tabellen terug verwijderen uit de database.
- **Run in de Package Manager Console de volgende instructie** (controleer eerst dat je connectionstring goed staat in BierenDbContext class !!)

Update-database



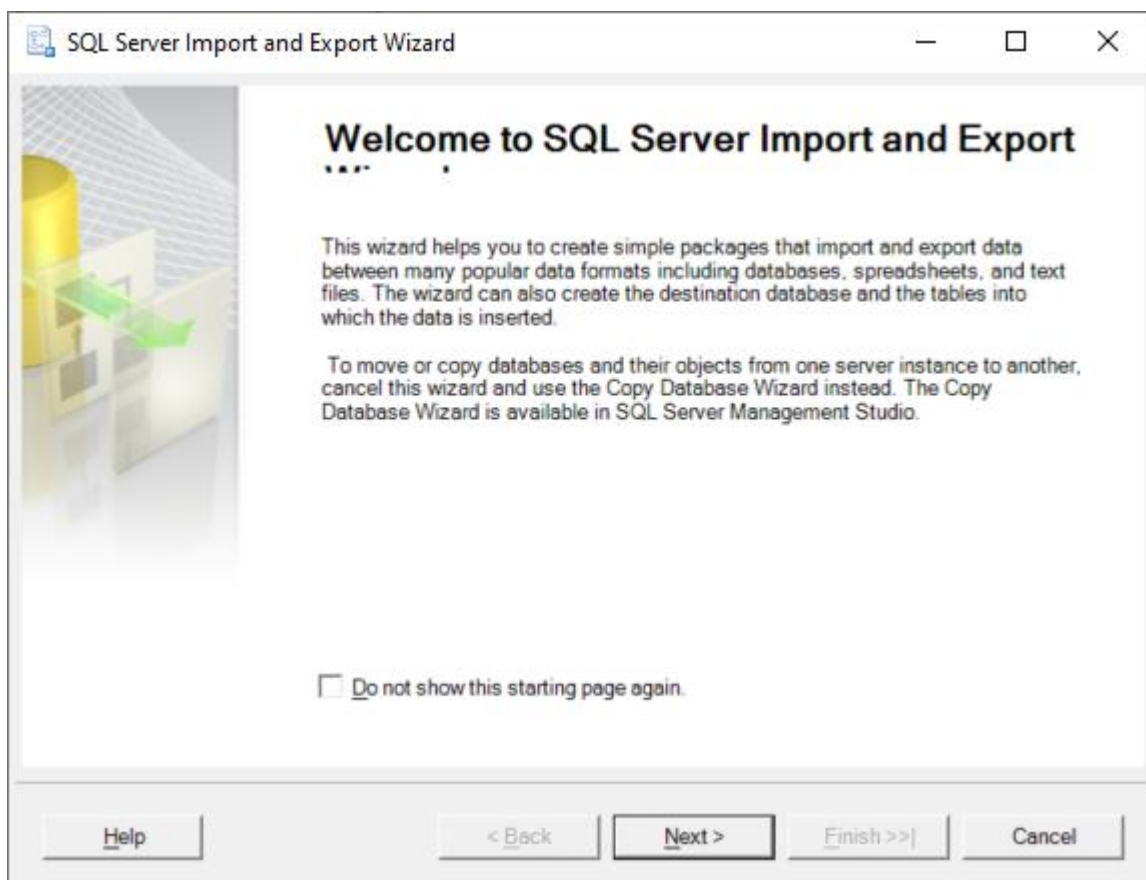
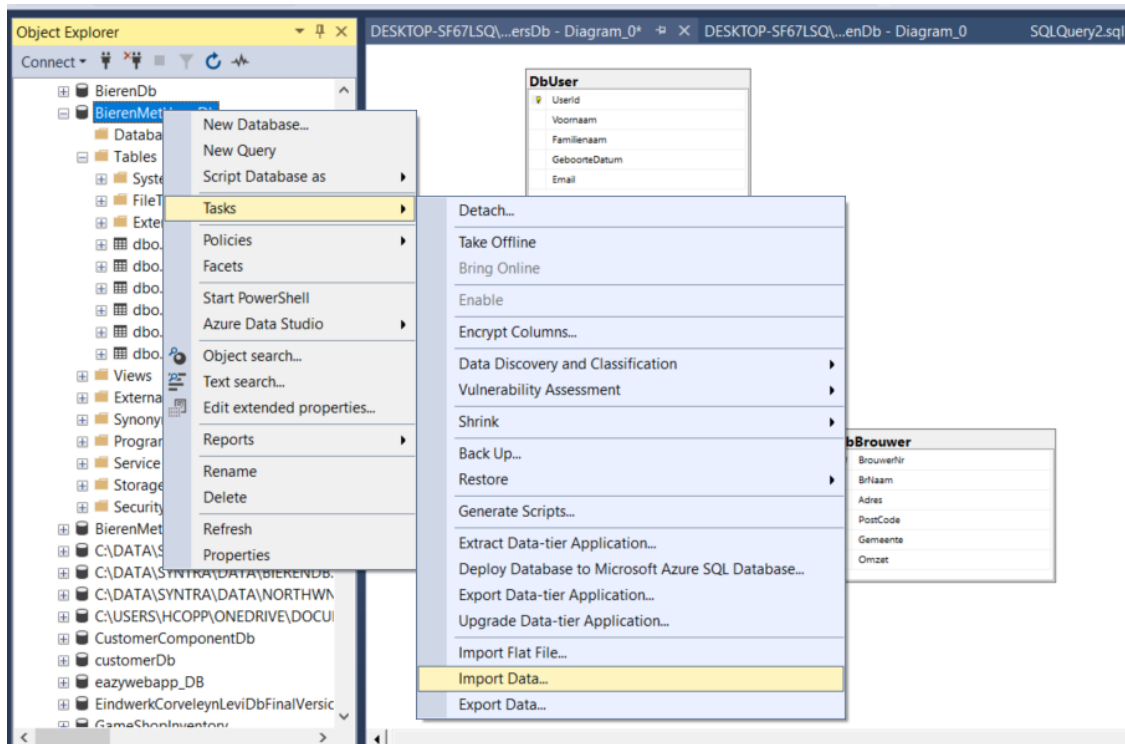
- Open Sql Server Management Studio en maak een diagram van de tabellen dbUser, dbBier, dbBrouwer en dbSoort. Er is een tussentabel gemaakt, dit is nodig vanwege de many-to-many relatie tussen DbUser en DbBier. Er zijn eveneens PK-FK relaties gelegd tussen de verschillende tabellen:



- Wanneer we de tabellen bekijken, zien we dat deze allemaal nog leeg zijn. We kunnen er manueel in elke tabel enkele rijen aan toevoegen, of we kunnen reeds de gegevens in DbBier, DbBrouwer en DbSoort importeren vanuit de BierenDb naar deze lege database:

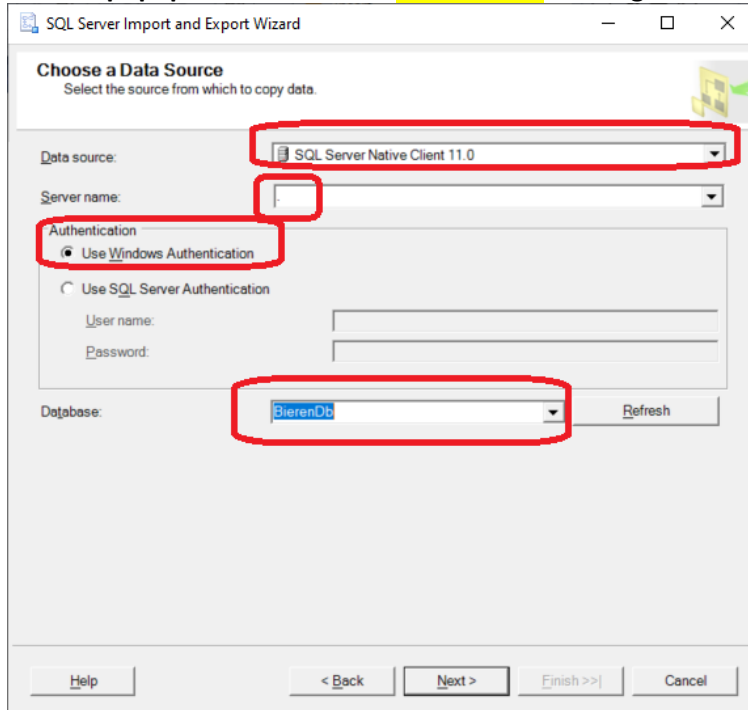
- In Sql Server Management studio kunnen gegevens geïmporteerd worden via het context menu:

Rechtsklik met de muis op BierenMetUsersDb en kies dan menuitem Tasks/import Data :



Klik op de button Next

- Een popup window voor de **Data Source** wordt getoond:



Kies als data Source: **Sql Server Native Client 11.0**

Als Server name: **.** of **localhost** of naam van je sqlSvrInstantie (bv desktop naam)

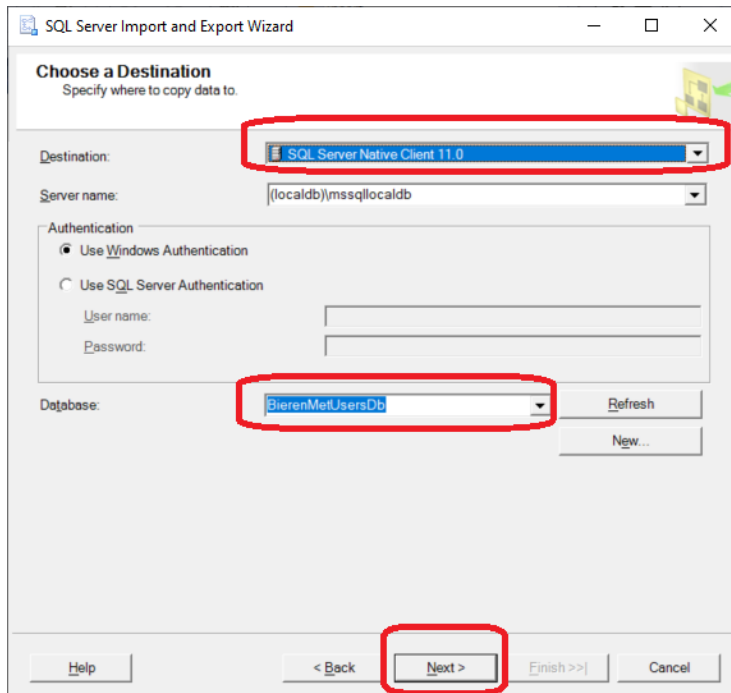
Als Database **BierenDb**

Klik op Next button

- Een popup window voor de **Database Destination** wordt getoond:

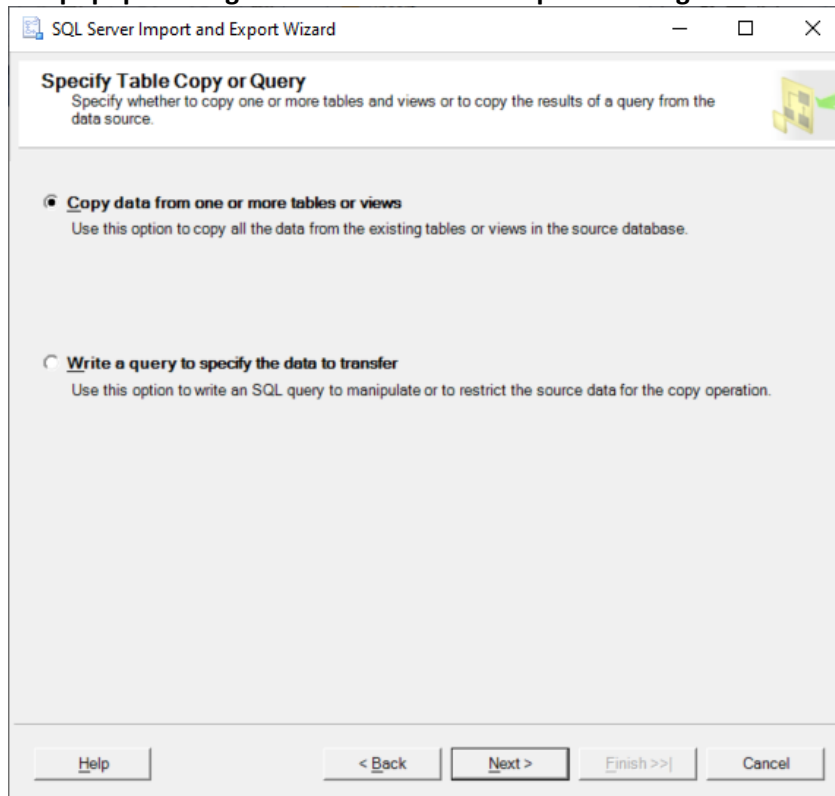
Kies opnieuw als data Source: **Sql Server Native Client 11.0**

Kies als Database **BierenMetUsersDb**



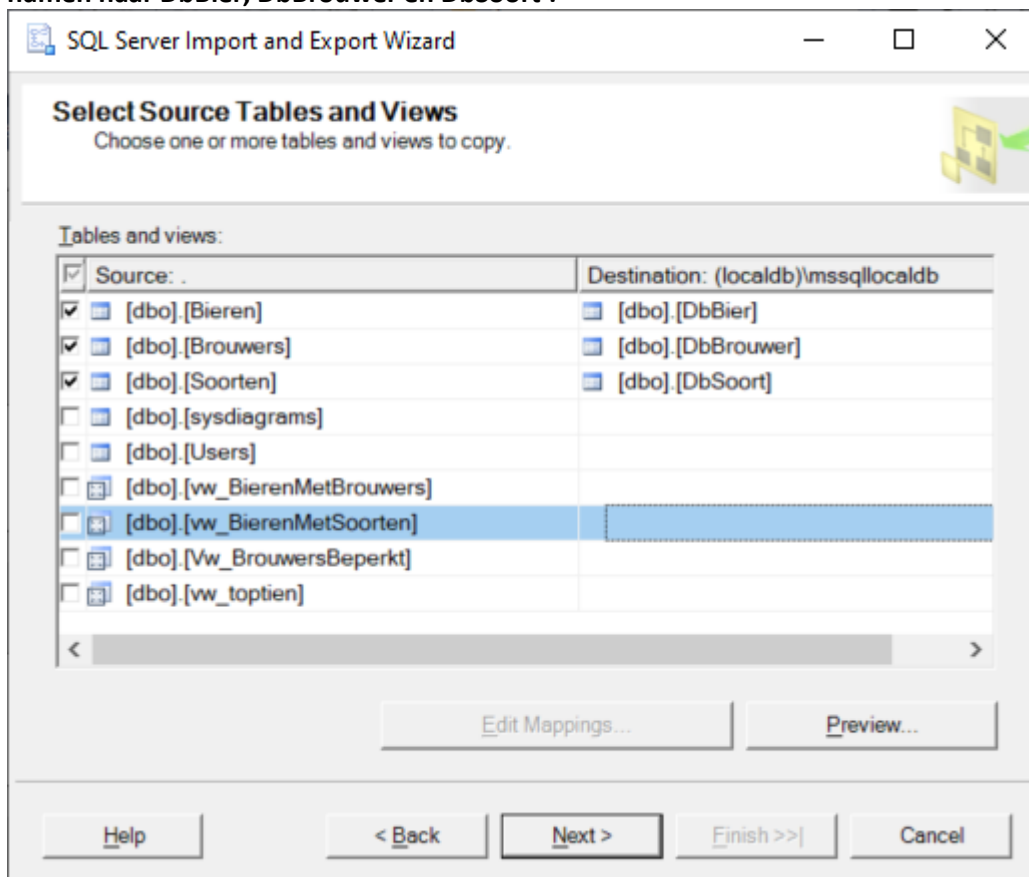
- Klik op Next button

- Een popup wordt getoond. Laat de eerste optie als aangeduid :

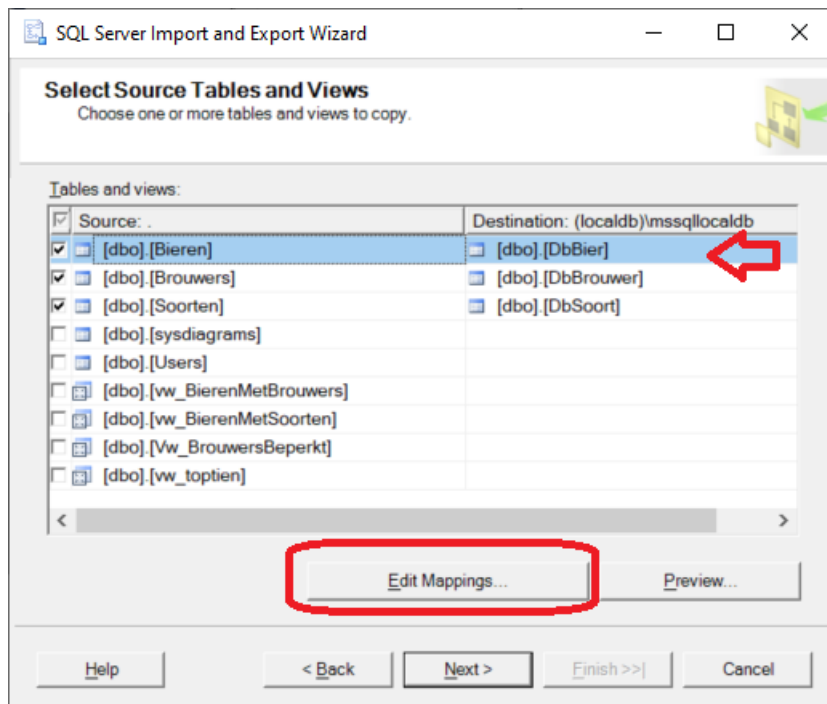


- Klik op Next Button

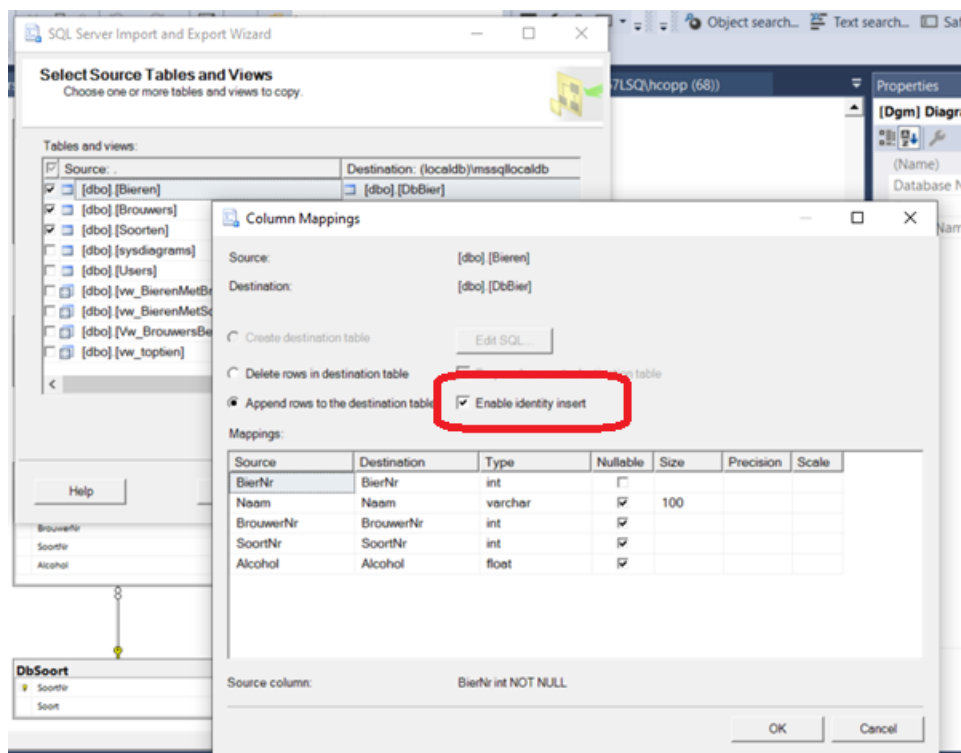
- In de volgende popup vink je Bieren, Brouwers en Soorten aan. Wijzig in de 2de kolom de Tabel namen naar DbBier, DbBrouwer en DbSoort :



-Selecteer de eerste lijn in de grid en klik op de button 'Edit Mappings' :

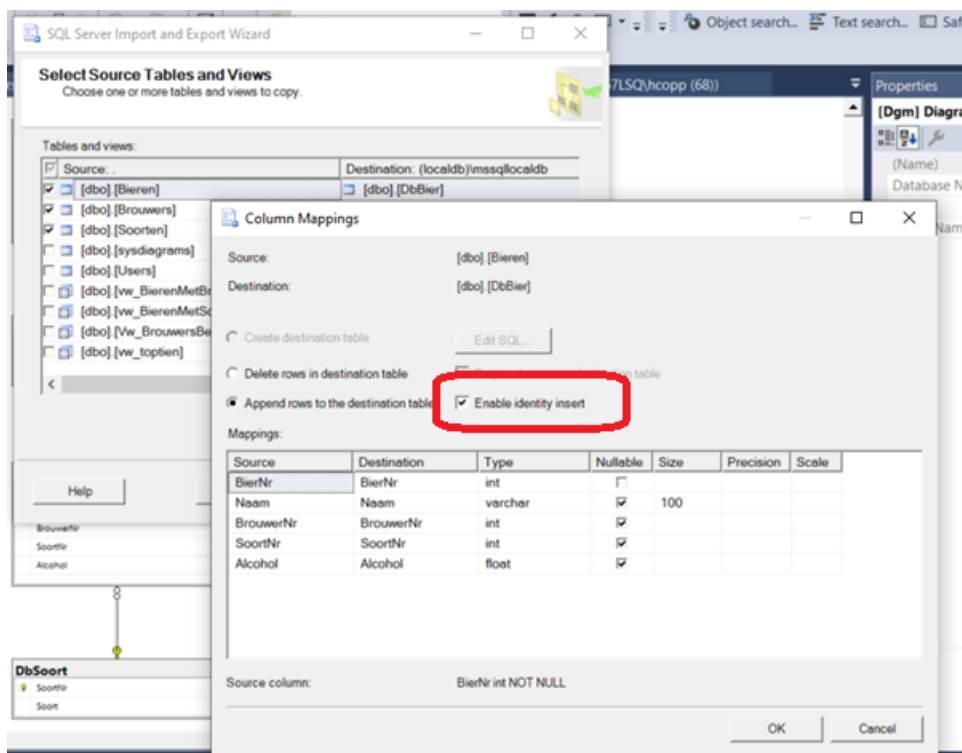
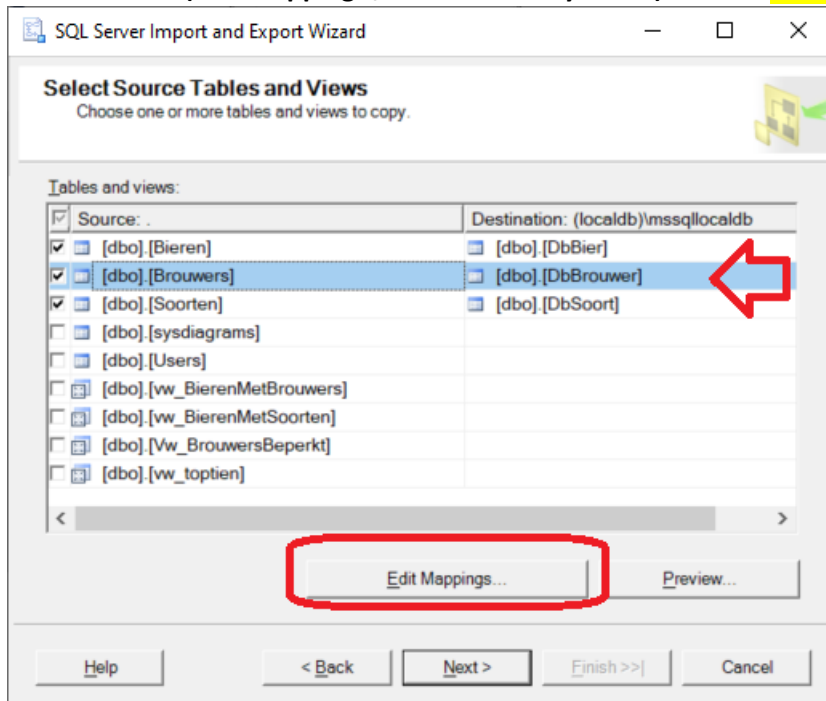


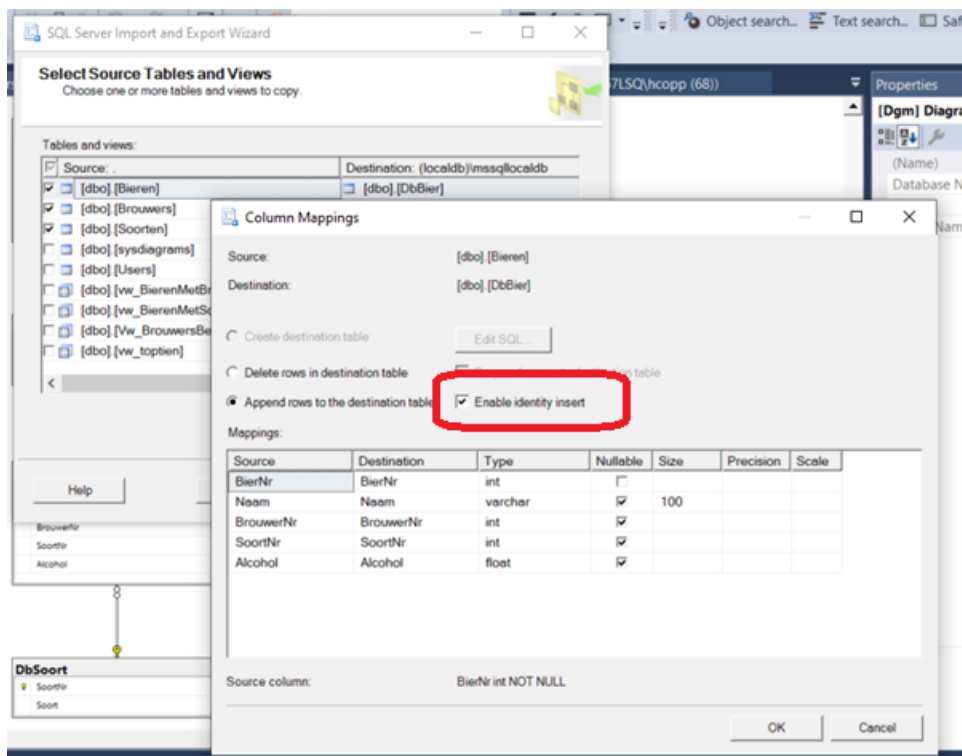
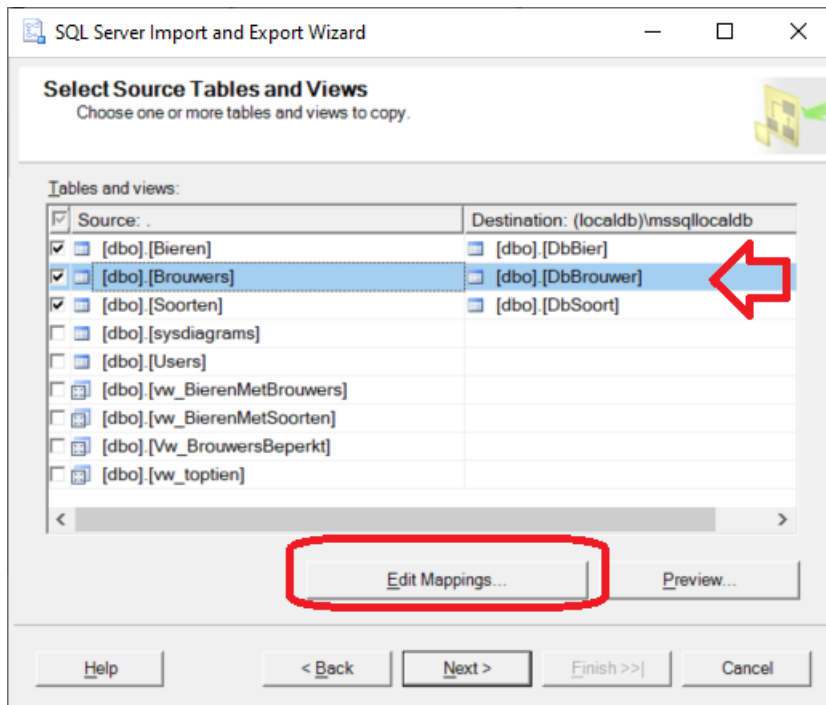
In de Column Mappings popup vink je "Enable identity insert" aan:



Klik op Ok button

Doe hetzelfde (Edit Mappings / Enable Identity Insert) voor de **Brouwers en Soorten**:





-Klik op Next button nadat alle 3 (Bieren, Brouwers en Soorten) Mappings goed gezet zijn

SQL Server Import and Export Wizard

Save and Run Package

Indicate whether to save the SSIS package.

☒ **R**un immediately

☐ **S**ave SSIS Package

☒ **S**QL Server

☐ **F**ile system

Package protection level:

Encrypt sensitive data with user key

Password:

Retype password:

Help < Back Next > Finish >>| Cancel

Klik op Next button

SQL Server Import and Export Wizard

Complete the Wizard

Verify the choices made in the wizard and click Finish.

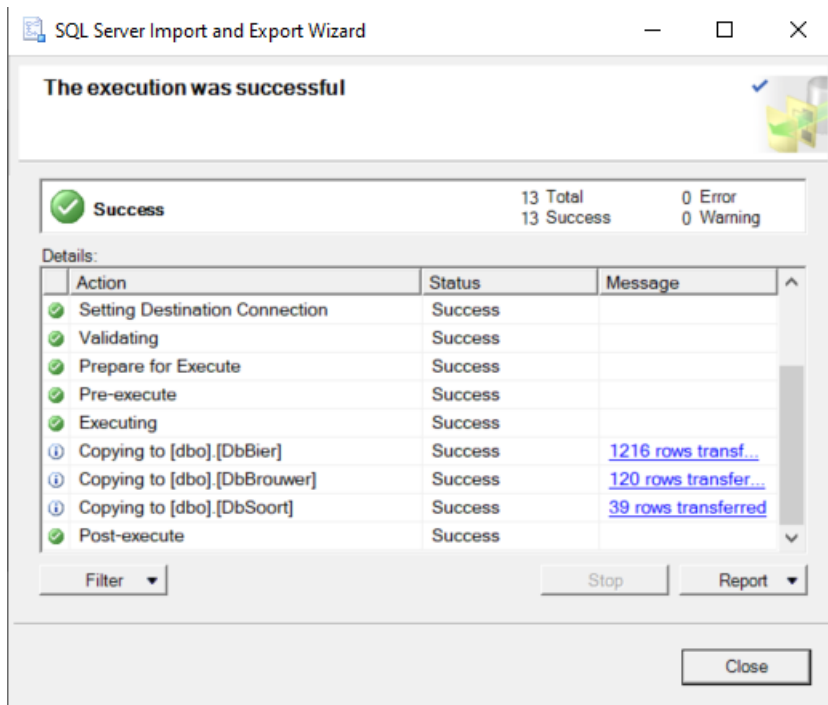
Click Finish to perform the following actions:

Source Location : .
 Source Provider : SQLNCLI11
 Destination Location : (localdb)\mssqllocaldb
 Destination Provider : SQLNCLI11

- Copy rows from [dbo].[Bieren] to [dbo].[DbBier]
The new rows will be appended to the existing table.
- Copy rows from [dbo].[Brouwers] to [dbo].[DbBrouwer]
The new rows will be appended to the existing table.
- Copy rows from [dbo].[Soorten] to [dbo].[DbSoort]
The new rows will be appended to the existing table.
- The package will not be saved.
- The package will be run immediately.

Help < Back Next > Finish Cancel

Klik op Finish button



Klik op Close

Open de Tabellen DbBier, DbBrouwer en DbSoort van de BiernMetUsersDb en controleer of de rijen zijn toegevoegd.