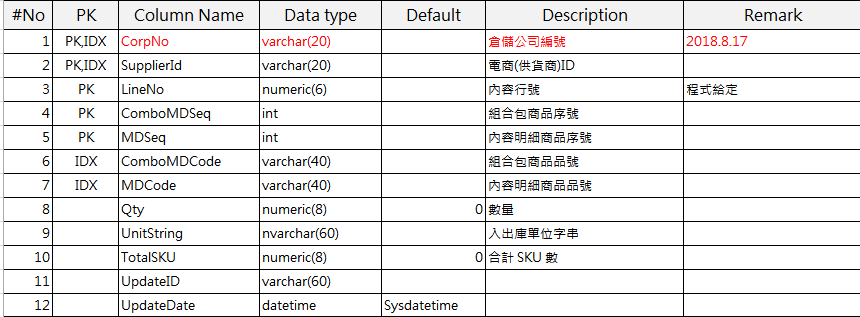
以下圖資料表為例

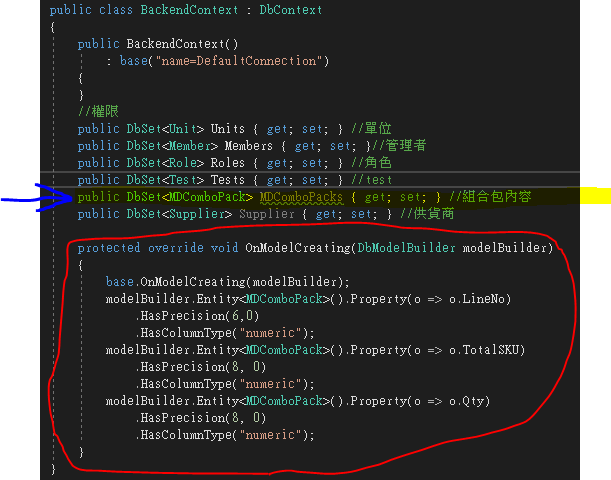


如何使用Code First在Model中建立此資料表

以每個欄位來介紹:

|  |  |  |
| --- | --- | --- |
| CorpNo | [Key, Column(Order = 1, TypeName = "VARCHAR")] [Display(Name = "倉儲公司編號")]  [Required(ErrorMessage = "{0}必填")]  [StringLength(20)]  public string CorpNo { get; set; } | 因為此欄位為此表的主鍵，所以要加入[Key]也就是第一行，但由於其資料表為複合主鍵，我們假設第一個欄位為主鍵順序1則如語法要加Order=1(複合主鍵，一定得加Order=順序)，在Model中默認string型態轉為SQL語法為nvarchar所以這裡我們要在後面再補上TypeName = "VARCHAR")告訴他我們要varchar，接著我們看以看到第二行是EXCEL裡的Description，接著第三行可以看到他是說必填欄位那妳會發現有ErrorMessage這是顯示錯誤訊息，不顯示則該標籤就寫[Required]，第四行[StringLength(20)]則是設定資料庫欄位大小，這裡也可以使用[MaxLength(20)]兩者皆可用於資料庫欄位大小，最後一行則是型態及欄位名稱當然如果對此欄位有其他需求是可以再get跟set裡面加入程式碼的。 |
| SupplierId | [Key, Column(Order = 2, TypeName = "VARCHAR")]  [Display(Name = "電商(供貨商)ID")]  [Required(ErrorMessage = "{0}必填")]  [StringLength(20)]  public string SupplierId { get; set; } | 基本上跟第一個欄位差不多， 但因為他是第二個所以Order=2若有順序按順序，無的話看要怎麼定義，這裡由上往下依序增加 |
| LineNo | [Key, Column(Order = 3)]  [Required(ErrorMessage = "{0}必填")] [Display(Name = "內容行號")]  public decimal LineNo { get; set; } | 此欄位numeric(6)這個在這裡要另外處理，因下方還有最後統一說明，首先我們看到其實就如同前面欄位一樣，第一行一樣設定主鍵，第二行必填，第三行描述，第四行型態及欄位名稱，你會發現需求為numeric而不是decimal，但code裡面沒有numeric  因為在此只能用decimal然後在去另一個地方設定後面統一說明。 |
| ComboMDSeq | [Key, Column(Order = 4)]  [Required(ErrorMessage = "{0}必填")]  [Display(Name = "組合包商品序號")]  public int ComboMDSeq { get; set; } | 此欄位很容易第一行為主鍵，第二行必填，第三行描述，第四行為型態及欄位名稱，因這裡為int他默認轉SQL也為int |
| MDSeq | [Key, Column(Order =5)]  [Required(ErrorMessage = "{0}必填")]  [Display(Name = "內容明細商品序號")]  public int MDSeq { get; set; } | 此欄位很容易第一行為主鍵，第二行必填，第三行描述，第四行為型態及欄位名稱，因這裡為int他默認轉SQL也為int |
| ComboMDCode | [Display(Name = "組合包商品品號")]  [Column(TypeName = "VARCHAR")]  [StringLength(40)]  public string ComboMDCode { get; set; } | 此欄位很容易第一行為描述，第二行為型態(此行為默認型態與所需型態不符才需要)，第三行為欄位大小，第四行為型態與名稱 |
| MDCode | [Display(Name = "內容明細商品品號")]  [Column(TypeName = "VARCHAR")] [StringLength(40)]  public string MDCode { get; set; } | 此欄位很容易第一行為描述，第二行為型態(此行為默認型態與所需型態不符才需要)，第三行為欄位大小，第四行為型態與名稱 |
| Qty | [Display(Name = "數量")]  public decimal Qty { get; set; } | 此欄位與前述的LineNo欄位如何轉為numeric後面依並介紹，，此欄位第一行為描述，第二行為欄位型態及名稱，至於他有預設值這點等等會與其他有預設值的一併在下面介紹 |
| UnitString | [Display(Name = "入出庫單位字串")]  [StringLength(60)]  public string UnitString { get; set; } | 此欄位就是第一行為描述，第二行為大小，第三行為型態跟名稱，前面有提到凡是string沒有在指定型態其轉換為SQL語法即為nvarchar |
| TotalSKU | [Display(Name = "合計 SKU 數")]  public decimal TotalSKU { get; set; } | 此欄位與前述的LineNo欄位如何轉為numeric後面依並介紹，，此欄位第一行為描述，第二行為欄位型態及名稱，至於他有預設值這點等等會與其他有預設值的一併在下面介紹 |
| UpdateID | [Column(TypeName = "VARCHAR")]  [StringLength(60)]  public string UpdateID { get; set; } | 此欄位與前面有提到的varchar欄位一樣都是要額外告知，型別如第一行，第二行為長度，第三行為型態及名稱 |
| UpdateDate | [DatabaseGenerated(DatabaseGeneratedOption.Computed)]  public DateTime UpdateDate { get; set; } | 此欄位有預設值下面一併介紹，首先針對第一行這是告知他限定該欄位會使用SQL裡的預設值，就絕對不會接受外給的值，那這個標籤還有其他用法下面在介紹，第二行為型態及名稱 |

此來介紹如何使用numeric由於此Type無法透過[DataType("numeric")]來指定其微度若只透過[DataType("numeric")]微度會是自動生成的(18,2)，故需要在Models中的BackendContext.cs中先註冊你建好的表以此為例，如下圖所示



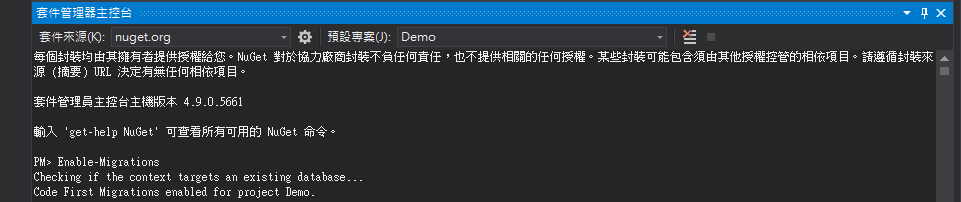
要先在藍色箭頭中註冊此資料表，然後在下方補上紅筆圈起來的語法才能，對於numeric的型態以及numeric的微度做一個設定才能使其轉換為SQL為numeric型態以LineNo為例欄位需求寫numeric(6)也就是numeric(6,0)所以需以此方法達成目的。

|  |
| --- |
| protected override void OnModelCreating(DbModelBuilder modelBuilder)  {  base.OnModelCreating(modelBuilder);  modelBuilder.Entity<MDComboPack>().Property(o => o.LineNo)  .HasPrecision(6,0)  .HasColumnType("numeric");  modelBuilder.Entity<MDComboPack>().Property(o => o.TotalSKU)  .HasPrecision(8, 0)  .HasColumnType("numeric");  modelBuilder.Entity<MDComboPack>().Property(o => o.Qty)  .HasPrecision(8, 0)  .HasColumnType("numeric");  } |

語法很簡單，Entity<MDComboPack>()，Entity中為資料表名稱，Property(o => o.LineNo)

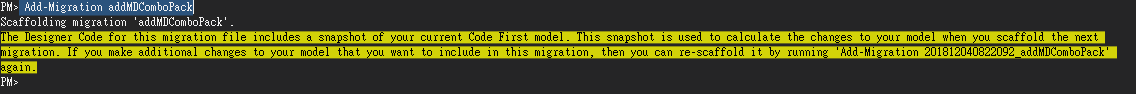
後面LineNo為欄位名稱.HasPrecision(6, 0)為numeric的微度，.HasColumnType("numeric")為欄位型態。

等到上述的都完成之後要來設定此資料表的預設值，首先因為你已經完成資料表的建置要將此轉換為SQL語法到SQL執行，所以假設我們今天是第一次做Code First需要先下初始化語法，才能使用Code first ，初始化的語法為Enable-Migrations在套件管理器主控台(工具->NuGet套件管理員->套件管理器主控台)下，打開後如下圖在PM>後面輸入上面的語法Enable-Migrations後Enter執行，成功專案會多一個資料夾Migrations，

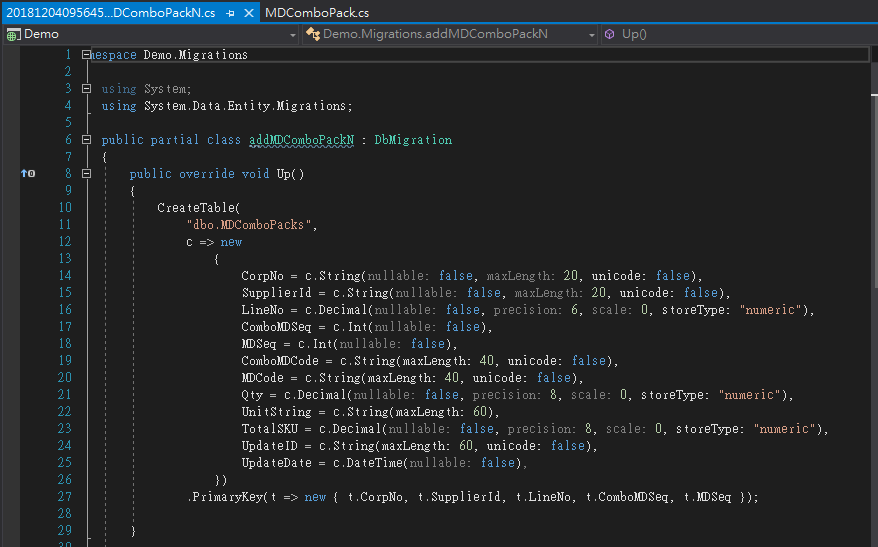


接著假設我們今天建了一張MDComboPack表，那我們接著下來就是下這個語法

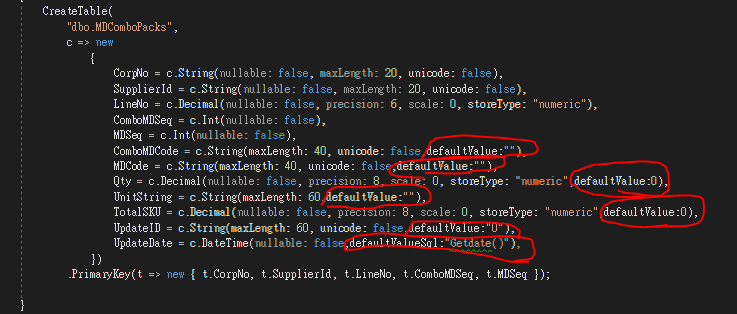
Add-Migration addMDComboPack，其組成為Add-Migration加上你對這此編輯的描述以此為例我新增了了MDComboPack表故我寫Add-Migration addMDComboPack，成功時你會看到如下圖



且此時螢幕上會打開一個code頁面這是剛剛執行Add-Migration addMDComboPack產生的頁面，那預設值在此頁面新增。



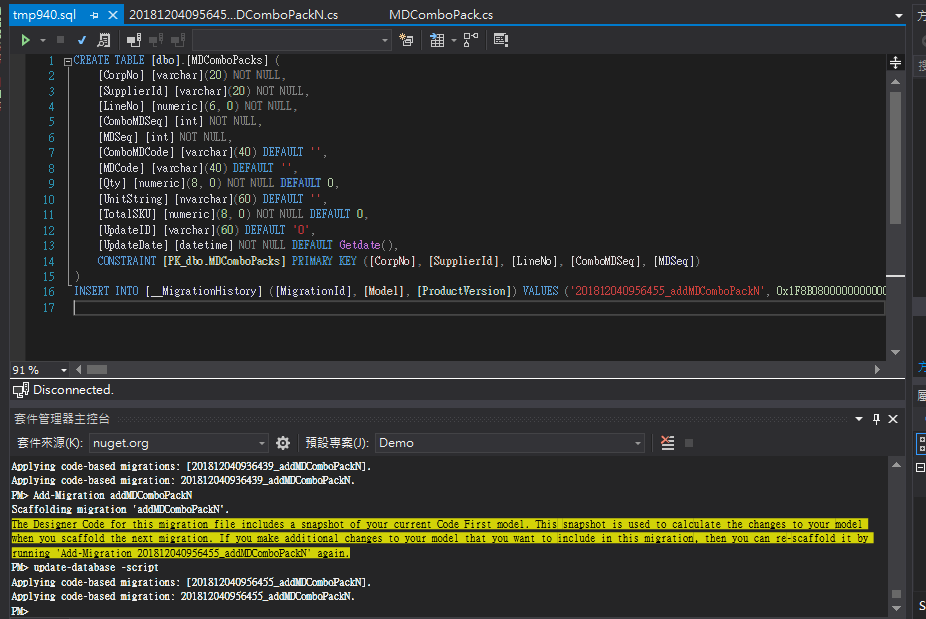
以剛剛的資料表來說凡是字串格式不管是nvarchar還是varchar都沒值時，都填入空字串，主鍵跟外來鍵除外，數值的欄位都填入0(指要型態是int不允許NULL的話不用額外設定都會自動填0)，其餘的話要設定，那現在來看如何設定首先我們先設定的事numeric要求預設為0，但前面有說到前五個欄位皆為主鍵與外來鍵所以這裡就不去設定值了，其餘的我們要在上圖頁面中，如下圖



這裡我們可以發現我們在這些需要加入預設值的欄位加入了   
defaultValue: ""，defaultValue:0，defaultValueSql: "Getdate()"以下說明:  
**defaultValue: ""**：此為加入預設值為空字串

**defaultValue:0**：此為加入預設值為0

**defaultValueSql: "Getdate()"**：這個比較不一樣因為他要加入的是SQL的函式所以妳不能用defaultValue而是要用defaultValueSql然後再加入參數，這裡來說的話他要的是SQL的參數設定完成預設值之後。

在回到套件管理器主控台中下update-database –script則會產生SQL語法 ，然後會產生，如下圖的資料庫語法，在將此語法貼到資料庫去執行即可建立完成。

**標籤介紹**

1. [DatabaseGenerated(DatabaseGeneratedOption.Computed)]其中DatabaseGeneratedOption有三個屬性

Computed 就是限定該欄位會使用SQL裡的預設值，就絕對不會接受外給的值

Identity 就是KEY值自動編號，也不接受外來的值，可以保護KEY值

None 不處理

1. [Key]:主鍵；如需複合主鍵則[Key, Column(Order = 順序)]一定得加Column(Order = 順序)，那如果還有型態的話假設是varchar則寫法為[Key, Column(Order =順序, TypeName = "VARCHAR")]，那如果今天他不是主鍵要寫型態的話[Column(TypeName = "VARCHAR")]可能會覺得為什麼主鍵要加再後面不是的話要另外寫，因為如果你是主鍵又要加型態的話會對一個欄位重複使用Column這個[Attribute](http://msdn.microsoft.com/zh-tw/library/system.componentmodel.dataannotations.keyattribute)，所以才會有差異。
2. [Display(Name = "欄位描述")]:欄位描述
3. [[MaxLength](http://msdn.microsoft.com/zh-tw/library/system.componentmodel.dataannotations.maxlengthattribute)(長度)]:允許的最大長度，可視為是欄位大小；語之相近的有

[StringLength(長度)]一樣是欄位大小；但兩者差異在**MaxLength**屬性中，你只能提供**最大**需求範圍而不是**最小**需求範圍。 而在**StringLength中，**你可以提供**兩者**

1. [Required]:必填欄位，但如果您今天有想顯示錯誤提示訊息可以改成  
   [Required(ErrorMessage = "{0}必填")]，加上錯誤訊息那裏面的{0}他會去抓你的欄位描述
2. 由於很多型態轉換成SQL時會有默認的型態，如常用的String->nvarchar；int->int，故有時有特殊需求可能必須加[Column(TypeName = "型態")]，去設定需求，但如果有特殊的如此表有numeric的話則需要先查找一下資料，那以他為例得到的方法是在欄位的型態及名稱的code裡面指定型態為decimal，接著到BackendContext中加入下方語法，如前面所說到的以numeric(6)，

|  |
| --- |
| protected override void OnModelCreating(DbModelBuilder modelBuilder)  {  base.OnModelCreating(modelBuilder);  modelBuilder.Entity<資料表名稱>().Property(o => o.該欄位)  .HasPrecision(6,0)  .HasColumnType("numeric");  } |

1. [DataType(型態)]：也能透過這個來設定型態，一般常見的用法為驗證email格式[DataType(DataType.EmailAddress)]或是表示此欄位為密碼欄位，所以輸入時會產生隱碼[DataType(DataType.Password)]；其中也都是可以加入錯誤提示訊息的[DataType(DataType.EmailAddress, ErrorMessage = "請輸入正確的電子信箱")]，但[DataType(DataType.EmailAddress)]僅侷限在client端驗證，如需要Server端驗證的話則使用此標籤[EmailAddress]看個人需求。

**建立關聯**

下面以新聞跟新聞類別為例，假設我們定義一個新聞只會有一個新聞類別，一個類別有多個新聞

那要如何建立起關聯以下不針對所有欄位探討只針對要如何讓兩張表達到關聯

|  |  |
| --- | --- |
| 新聞類別(NewsCategory) | 新聞(News) |
| **public** **virtual** ICollection<News> Newses { **get**; **set**; } | [Display(Name = "類別")]  **public** **int**? CategoryId { **get**; **set**; }  [ForeignKey("CategoryId")]  [JsonIgnore]  **public** **virtual** NewsCategory Category{ **get**; **set**; } |

如上表所示，關聯的建法很簡單，類別中(NewsCategory)加入  
**public** **virtual** ICollection<資料表> 您的命名 { **get**; **set**; }

然後再新聞中(News)加入外來鍵，

[Display(Name = "類別")]

**public** **int**? CategoryId { **get**; **set**; }

[ForeignKey("CategoryId")]

[JsonIgnore]

**public** **virtual** NewsCategory Category{ **get**; **set**; }

那上面兩行是你對這個欄位的描述及型態名稱的宣告，下來會看到[ForeignKey("CategoryId")]指定說我的CategoryId欄位他是外來鍵，[JsonIgnore]指是不希望其序列化，最後

**public** **virtual** NewsCategory Category{ **get**; **set**; }

很明顯就是告訴他與哪一張表關聯及對這個關聯的表命名，此命名會在程式中取用News表時，可點到Category去或得他的資訊，就是取得類別，反之類別資料表在程式中可以取得新聞的集合。

**Code First語法**

1. Enable-Migrations：會產生初始化的資料夾Migrations只有第一次要做，一但做過後就不用了
2. Add-Migration 此次編輯的描述：會在Migrations夾中生成一個文件，那其內容如有預設值等等的可在裡面編輯後，方法如上面提到的，待修改完成之後再執行最後一個動作產生SQL語法
3. update-database -script：會產生您此次編輯的SQL語法
4. 每次產生SQL語法會有一段

INSERT INTO [\_\_MigrationHistory] ([MigrationId], [Model], [ProductVersion]) VALUES ('201812040956455\_addMDComboPackN', …………(很長的亂碼), '5.0.0.net45')如果你今天是已經在線上WORK了，那當有資料庫的變更，你開發時要加上INSERT INTO [\_\_MigrationHistory] ([MigrationId], [Model], [ProductVersion]) VALUES ('201812040956455\_addMDComboPackN', …………(很長的亂碼), '5.0.0.net45')這一段去執行，但是你再正式的Server需拿掉這一段，他這一段新增後他會驗證資料庫的資料表等等的相關描述是否與你的專案中的Migrations夾相符所以最後Server上也不用複製此資料夾。

1. 那如果你今天執行到第二步產生文件時，發現錯誤你要更改後重新產生需要先將Migrations夾中你此次產生的文件刪除後再重新產生，否則驗證會錯誤，那假設你執行到資料那一段都已經新增完畢後，還要修改你可以再對此表編輯後，重新執行第二步跟第三步。