Basis standalone

Plannen en ontwerpen

hoofdstuk

3

Data analyse NORMA

Algemene informatie

Onderwerp(en)	NORMA
Leerdoel(en)	De student is in staat om een data analyse uit te werken door gebruik te maken van de NORMA extensie binnen Visual Studio.
Vereiste voorkennis	De student dient de stappen wat betreft data analyse in Excel te begrijpen.
Kwalificatiedossier	☐ B1-K1-W1: Plant werkzaamheden en bewaakt de voortgang ☒ B1-K1-W2: Ontwerpt software ☐ B1-K1-W3: Realiseert (onderdelen van) software ☐ B1-K1-W4: Test software ☐ B1-K1-W5: Doet verbetervoorstellen voor de software ☐ B1-K2-W1: Voert overleg ☐ B1-K2-W2: Presenteert het opgeleverde werk ☐ B1-K2-W3: Reflecteert op het werk

Inhoudsopgave

Wat is NORMA?

NORMA (Natural ORM Architect) is een Visual Studio Extensie waarmee je middels een ORM methode kunt komen tot een analyse van data. Om precies te begrijpen wat hiermee bedoeld wordt is het belangrijk om te weten wat dan ORM is.

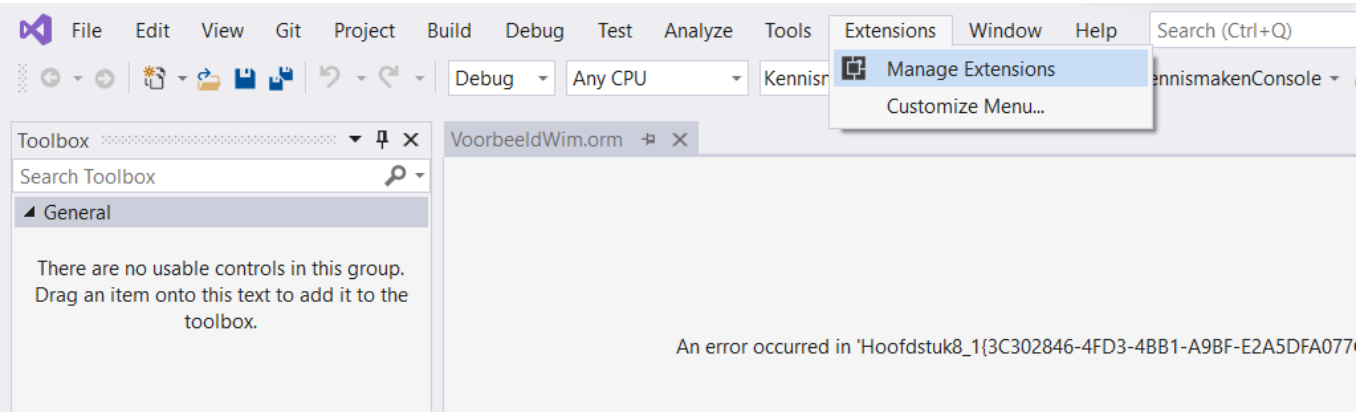
ORM (Object Role Modeling), niet te verwarren met Object Relational Mapping (dat ook een Software Developer term is), is een methode waaraan allerlei regels verbonden zijn om datamodellen te ontwerpen. Hierbij probeer je de werkelijke situatie zo te beschrijven dat ook niet-technische gebruikers begrijpen wat er wordt bedoeld. Dit is wat je in de zinnen die je gemaakt hebt in de Excel Sheet eigenlijk ook al gedaan hebt.

In dit hoofdstuk ga je leren hoe je tot een data analyse kunt komen door middel van de extensie NORMA, in plaats van de Excel Sheet. Echter zal je merken dat de denkwijze exact hetzelfde is. Je hebt nu dus voldoende kennis opgebouwd om deze tool te gaan gebruiken! Je zal merken dat het uitwerken van een data-analyse met deze tool een stuk sneller gaat!

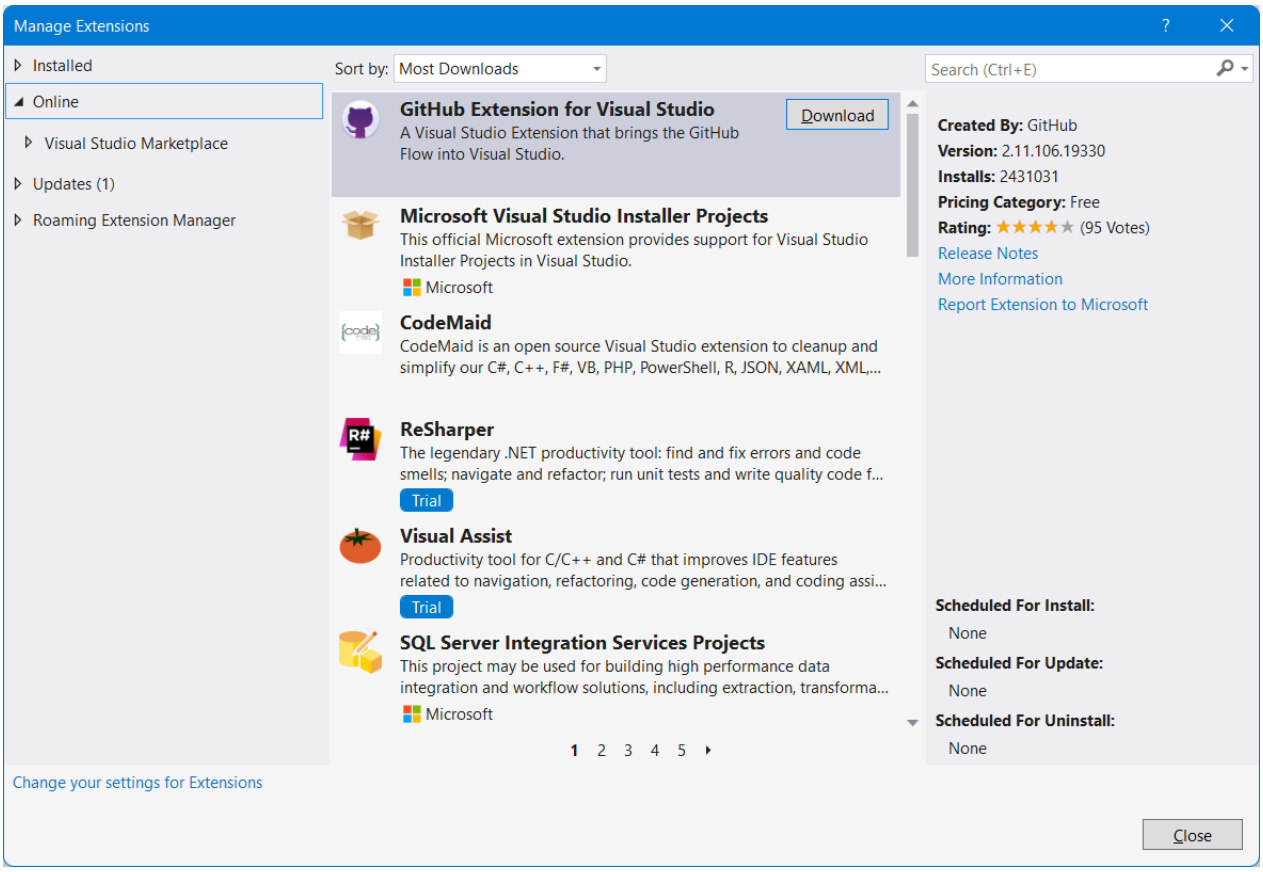
Installeren van NORMA

Voordat je NORMA kunt gebruiken ga je eerst de extensie installeren.

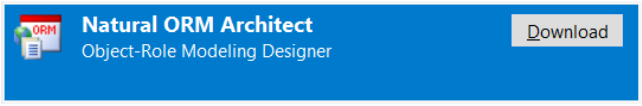
→ Klik in het menu op Extensions en klik daarna op Manage Extentions.



→ Klik links op Online en zoek daarna op norma.

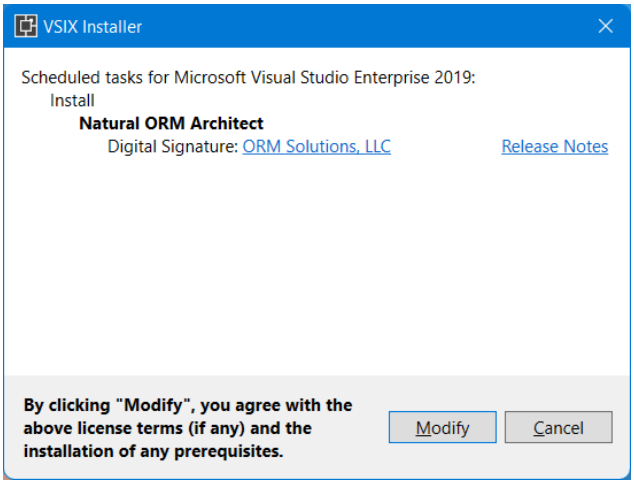


→ Klik op **Natural ORM Architect** en daarna op **Download**.

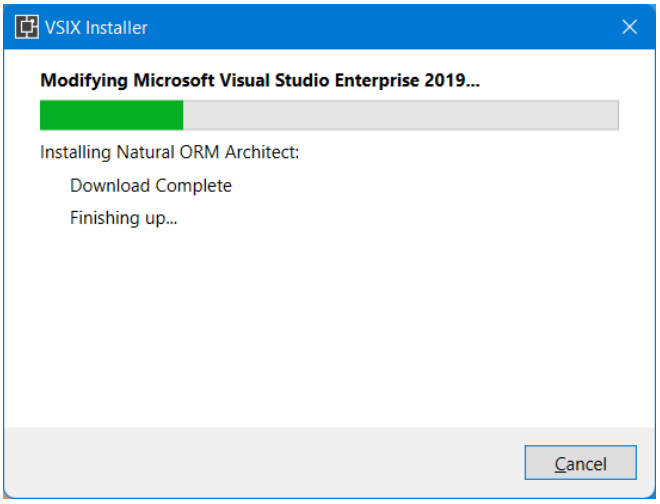


→ Sluit Visual Studio af.

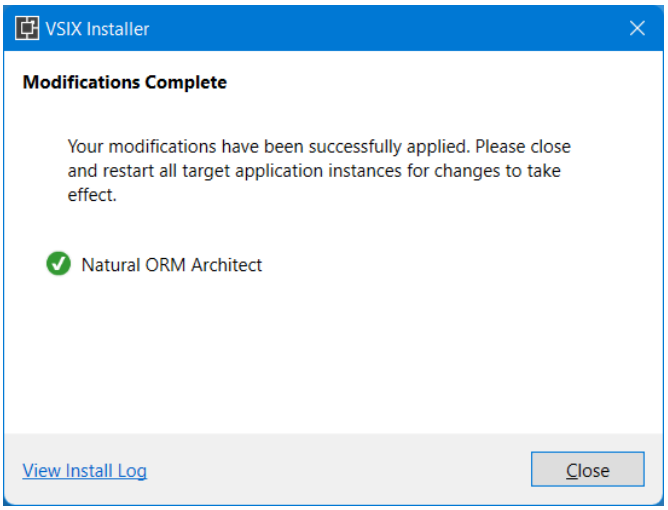
→ Laat de Visual Studio Installer even zijn werk doen en wacht op onderstaand venster. Klik daarna op **Modify**.



De extensie wordt nu geïnstalleerd.



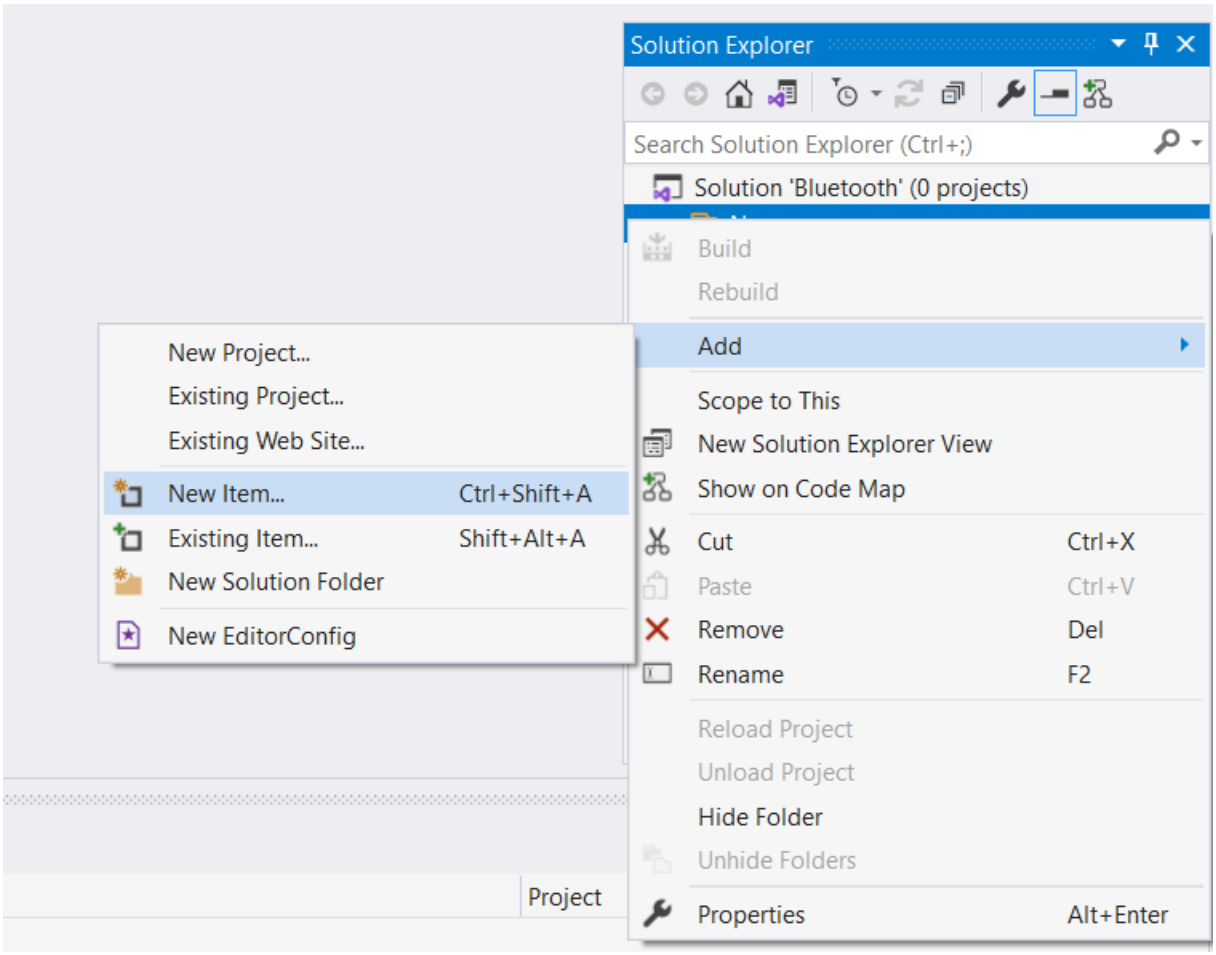
→ Klik na het voltooien van de installatie op **Close**.



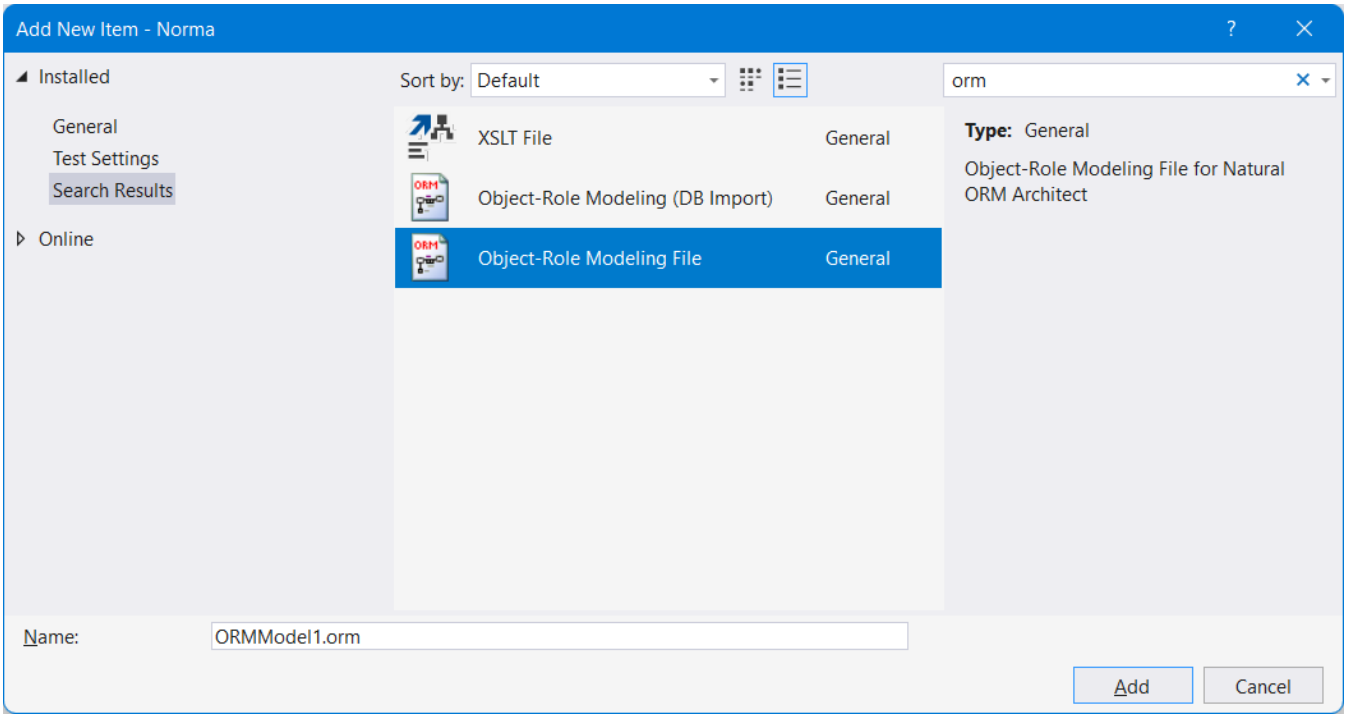
Aanmaken van een NORMA-bestand

Een NORMA-bestand kun je aanmaken binnen een project die in een bestaande solution staat of als een los bestand. Het is handig om het binnen een project aan te maken, zodat je daar later direct je applicatie in kunt realiseren.

→ Klik met rechts op een project of map. Klik daarna op **Add** en **New Item**.



→ Zoek op **orm** en klik op **Object-Role Modeling File**.



→ Hernoem het bestand.

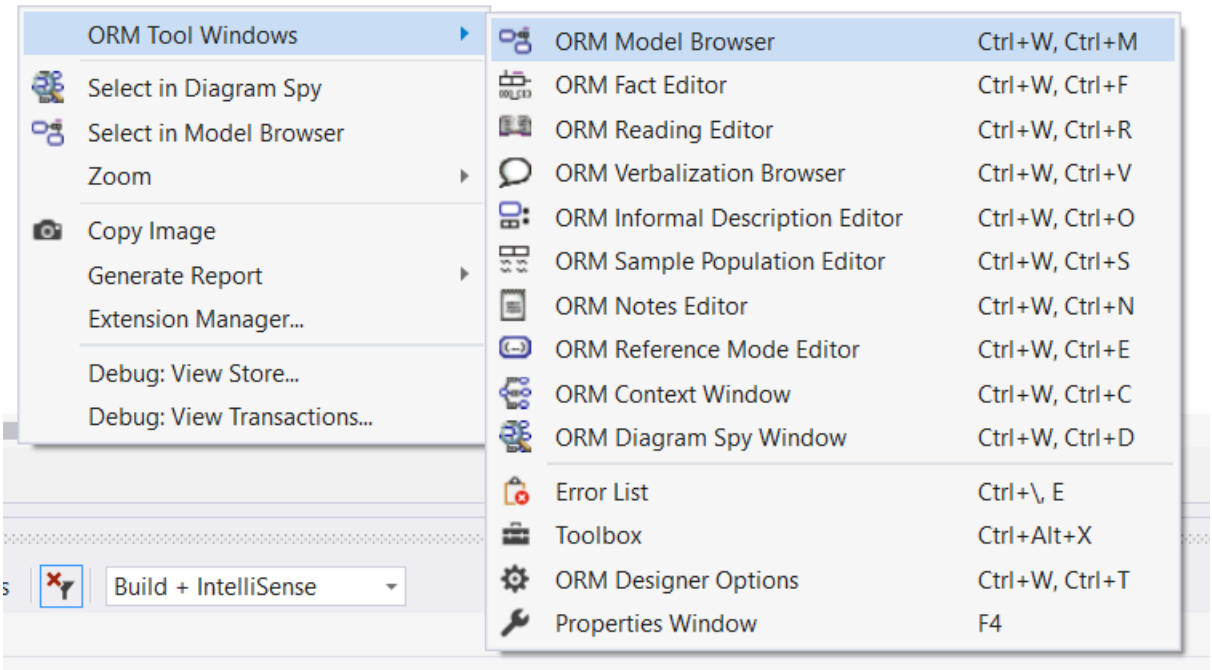
→ Klik op **Add**.

Configureren van NORMA

Nu je de NORMA extensie geïnstalleerd hebt krijg je de mogelijkheid om extra vensters aan te roepen in Visual Studio. Een aantal daarvan ga je vaak gebruiken als je met een NORMA-bestand bezig bent. Daarom gaan we deze vensters standaard openzetten, zodra je met een NORMA-bestand bezig bent. Het gaat om onderstaande vensters:

- ORM Model Browser
- ORM Fact Editor
- ORM Verbalization Browser
- ORM Sample Population Editor
- Properties Window
- Error List

Je kunt deze vensters zichtbaar maken door met rechts in het **Document Window** (werkveld) te klikken. **Via ORM Tool Window** kun je instellen welke vensters je nog meer zichtbaar wil maken.



Het stappenplan als je werkt met NORMA

Als je met NORMA gaat werken doorloop je een net iets andere volgorde dan in de ExcelSheet. Hieronder volgt een overzicht van de stappen. In de rest van de reader worden deze stappen verder toegelicht.

1. Zinnen maken
    a. Entiteit → attribuut
    b. Entiteit → entiteit
2. Datatypen opgeven
3. Data definitie opgeven
4. Aangeven of de entiteit een persoon is
5. Voorbeeld populatie opgeven
    a. Voor entiteiten
    b. Voor attributen
6. Uniciteiten opgeven
    a. Primaire uniciteiten
    b. Alternatieve uniciteiten die over slechts één attribuut liggen
7. Mandatories opgeven
8. Alternatieve uniciteiten die over meerdere attributen liggen
9. Voorbeeld populatie opgeven voor relaties
10. ERD genereren
11. Model aanpassen vanuit de Relational View
    a. Tabellen aanpassen
    b. Kolommen aanpassen

Zoals je ziet lijken dit meer stappen dan in de Excel-Sheet. Dat is ook zo, maar de hoeveelheid tijd die je nodig hebt om deze stappen te doorlopen is een stuk minder. Je bent namelijk geen tijd meer kwijt met Excel-oefeningen zoals kopiëren plakken, rijen toevoegen etc. De stappen in NORMA hebben zelfs grotere sprongen, waardoor het wel wat moeilijker kan zijn.

Zinnen maken

Net zoals in de Excelsheet begin je met het maken van zinnen. Je begint met het maken van de "heeft een" zinnen, oftewel de zinnen waarbij je een entiteit koppelt aan een attribuut. De identificatie geef je in NORMA ook direct mee.

Zinnen maken in Norma doe je in de **Fact Editor**. Alle woorden die je met een hoofdletter typt worden gezien als entiteit of attribuut. Iedere entiteit geef je ook gelijk zijn identificerende waarde mee tussen ronde haken. Alle woorden die niet met hoofdletters geschreven worden, worden als relatiebeschrijving beschouwd.

Zinnen maken (Entiteit → attribuut)

Hieronder zie je een voorbeeld van hoe je een entiteit-attribuut zin schrijft in de **Fact Editor**.

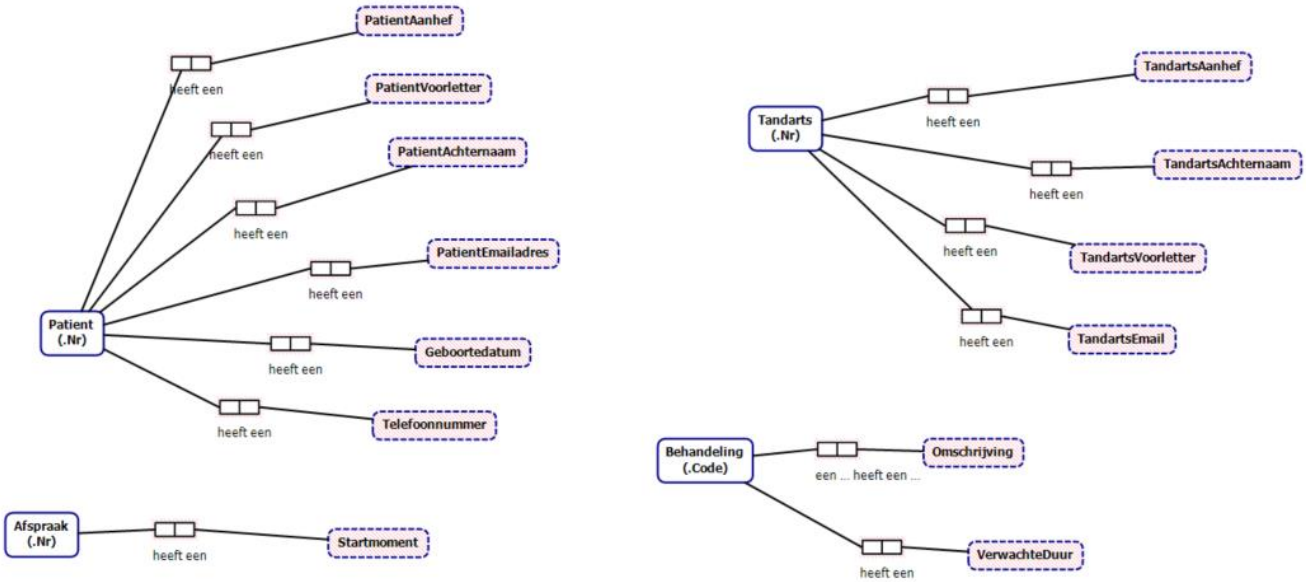
ORM Fact Editor

```
een Patient(.Nr) heeft een Aanhef()
```

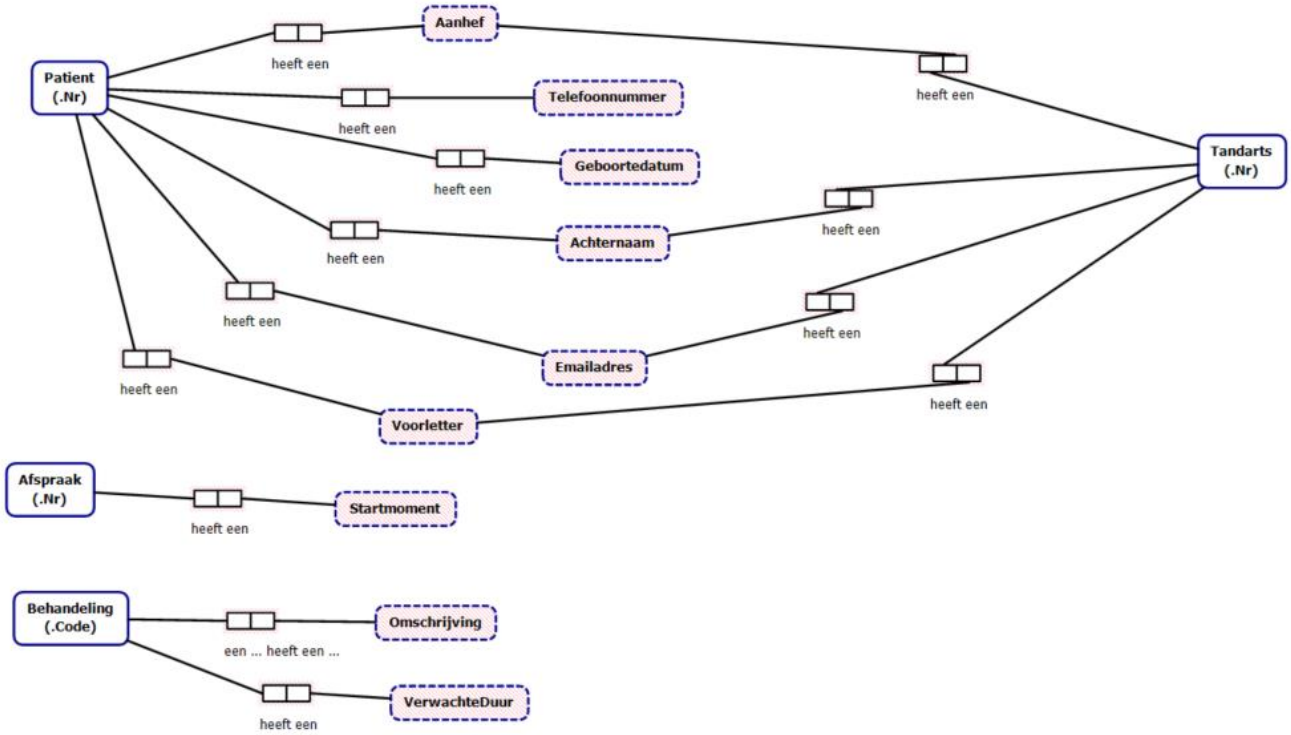
Hierbij is .Nr de identificerende waarde voor de Patient. Als je de ronde haken intypt kun je kiezen uit een aantal veelvoorkomende identificerende attributen. Echter kun je ze ook een eigen naam geven. Je zet dan achter de punt je eigen attribuutnaam.

Nadat je de zin getypt hebt druk je op **Ctrl + Enter**. De visuele weergave hiervan wordt nu zichtbaar in het Document Window.

Nadat je alle entiteit-attribuut zinnen gemaakt hebt kun je een soortgelijk resultaat verwachten:

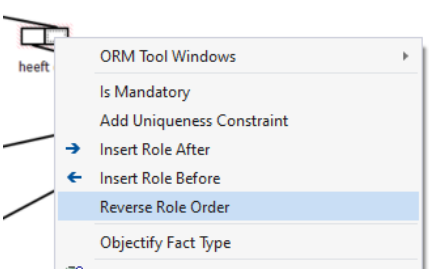


Je hebt bij de attributen van de patiënt steeds de naam van de entiteit (patiënt) ervoor gezet. Dit heb je gedaan omdat de tandarts dezelfde soort attributen heeft. Als je dat er nu niet steeds bij zet dan zou je uitkomen op de visuele weergave zoals op de volgende bladzijde.

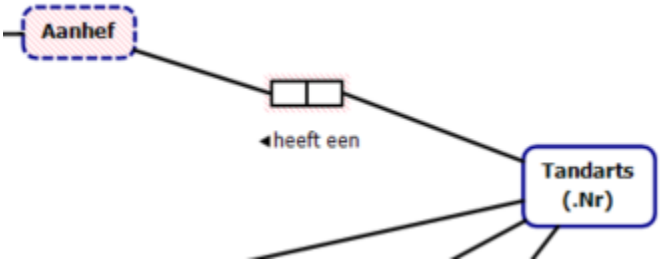


Uiteindelijk zal je straks uitkomen op hetzelfde eindresultaat. Er zit in bovenstaand diagram echter wel een omgekeerde relatie aanduiding in. De kant waarop aanhef gekoppeld zit op het blokje is rechts, terwijl aanhef links van het blokje staat.

Zodra je later aan de slag gaat met de uniciteiten is dit erg onhandig. Daarom is het handig om de relatie om te draaien. Dit doe je door er rechts op te klikken en te kiezen voor Reverse Role Order.



Je ziet nu dat de relatie wel op een nette manier is getekend.



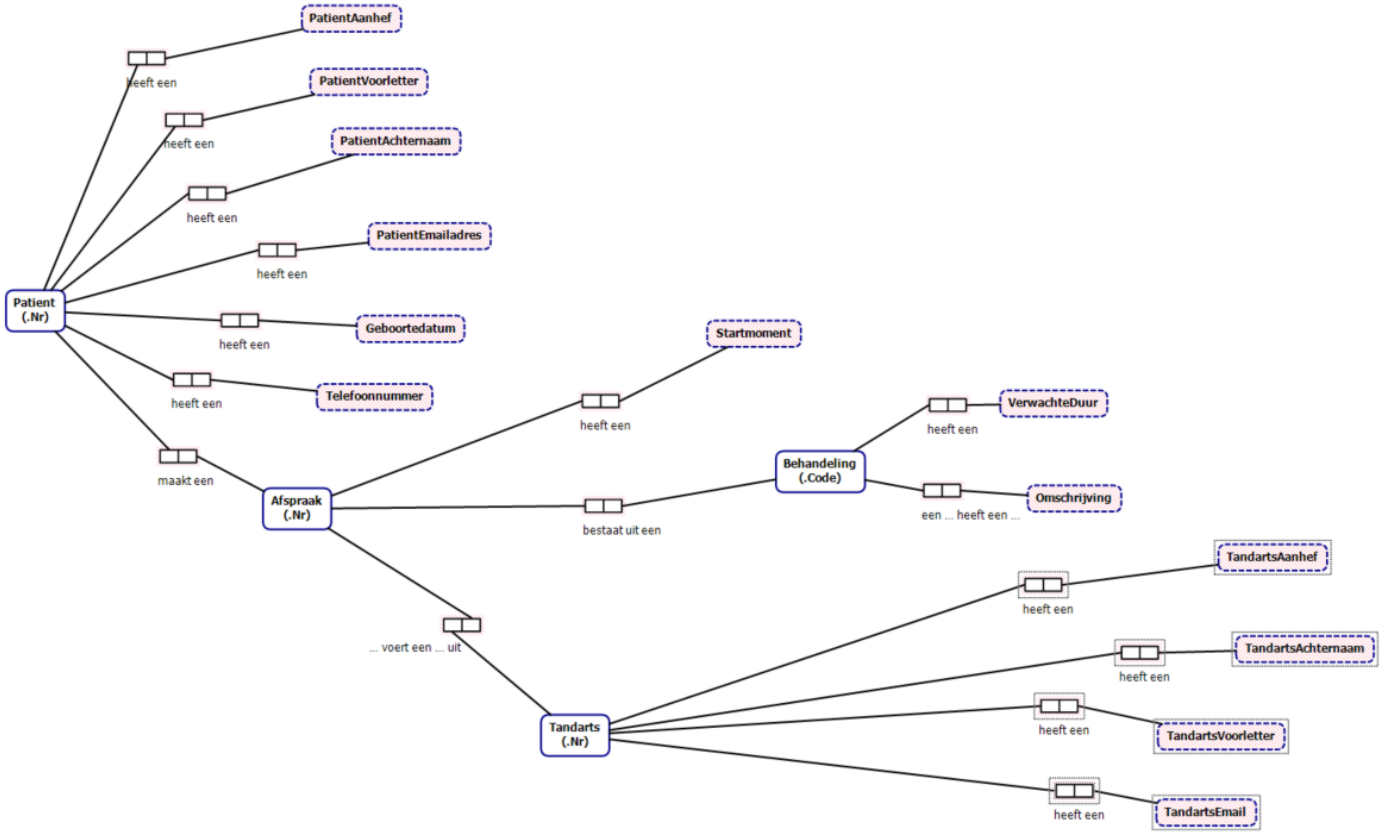
Bij "heeft een" is nu een pijltje ontstaan om aan te geven dat de relatie omgekeerd is. Het is ook een Tandarts heeft een Aanhef en niet een Aanhef heeft een Tandarts. Logisch toch?

Zinnen maken (Entiteit → entiteit)

Hieronder zie je een voorbeeld van hoe je een entiteit-entiteit zin schrijft in de **Fact Editor**.

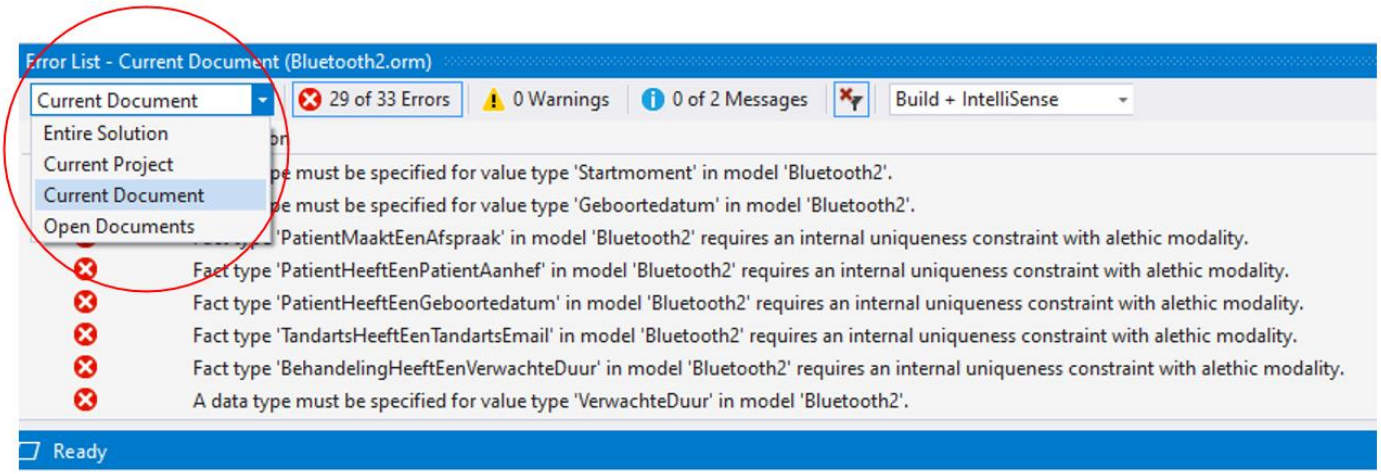
Let er hierbij op dat je de entiteiten dezelfde identificerende attributen geeft als het identificerende attribuut dat je gebruikt hebt bij de entiteit-attribuut zinnen.

Nadat je alle entiteit-entiteit zinnen gemaakt hebt kun je een soortgelijk resultaat verwachten:

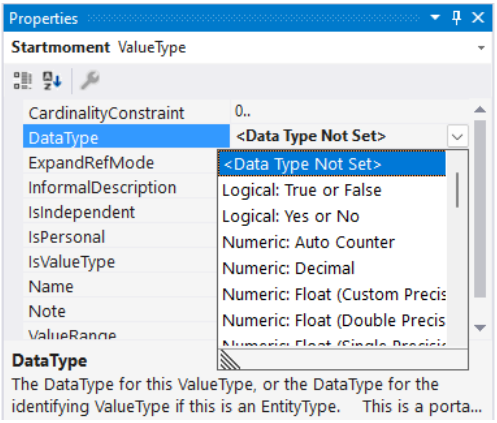


Datatypen opgeven

Nadat je de zinnen gemaakt hebt zie je in de errorlist allerlei foutmeldingen over datatypes en constraints. Filter de foutmeldingen zodat je alleen foutmeldingen ziet van het openstaande bestand.



Zodra je nu op een foutmelding klikt over een datatype, kom je direct uit bij je **Properties Window**. Je kunt eventueel ook het **Properties Window** handmatig open zetten en een attribuut selecteren die een rode achtergrondkleur heeft.



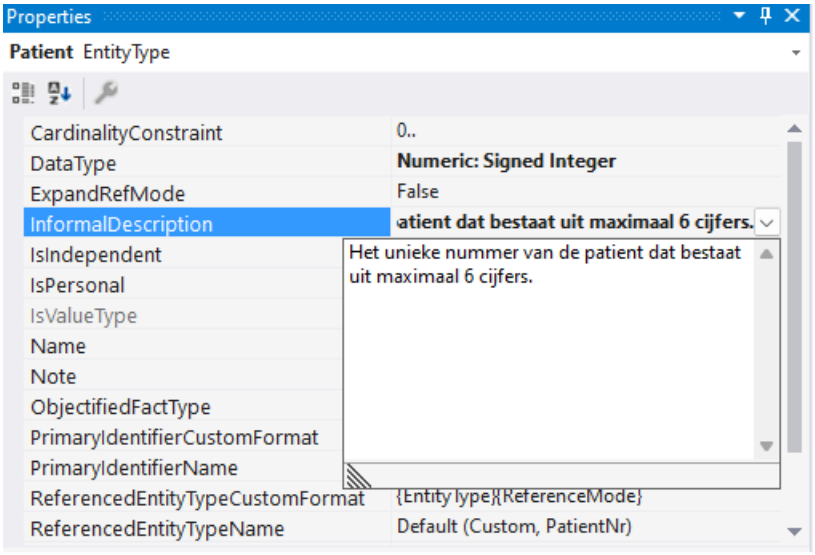
Kies hieruit het geschikte datatype. Houd er rekening mee dat je dit later ook opneemt in je datadictionary en je uiteindelijke create-script. Zodra je het datatype hebt opgegeven wordt de achtergrondkleur van dat attribuut wit.

Let op:
Bij sommige datatypen verschijnt er opeens een extra property. Bijvoorbeeld bij het datatype Text: fixed length ook de property DatatTypeLength. Bij een Numeric: Float (Custom Precision) krijg je de extra property DataTypePrecision. Je vult dan ook de waarde van die property in.

Je kunt ook hier gelijk kiezen voor een automatische nummering (handig voor bijvoorbeeld een afspraaknummer).

Data definitie opgeven

Geef voor iedere entiteit en attribuut aan wat de definitie er van is. Klik hiervoor op de entiteit of attribuut en stel de property **InformalDescription** in.

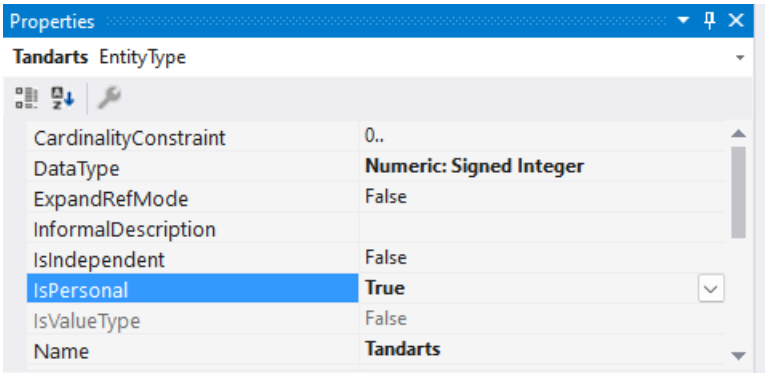


Probeer de definities zo duidelijk mogelijk te omschrijven.

Aangeven of de entiteit een persoon is

Zodra je zo aan de slag gaat met de uniciteiten is het handig om aan te geven of een entiteit een persoon is ja of nee. Bij de controle van de uniciteiten worden namelijk automatisch zinnen gemaakt. Door aan te geven of de entiteit een persoon is, worden de zinnen beter opgesteld. Dat maakt het controleren van je uniciteiten straks makkelijker.

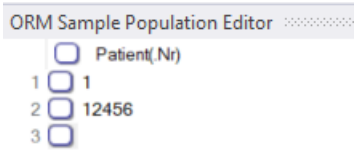
Dit stel je in door op de entiteit die een persoon representeert te klikken en de property **IsPersonal** op **true** te zetten.



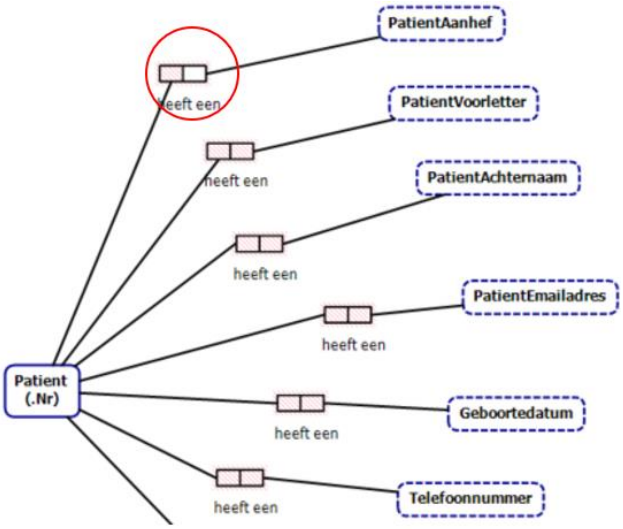
Voorbeeld populatie opgeven voor entiteiten en attributen

Geef voor iedere entiteit en attribuut minimaal 2 waarden op als voorbeeldpopulatie. Houd er hierbij rekening mee dat dit realistische waarden zijn. Ze moeten ook voldoen aan het opgegeven datatype (en eventueel de opgegeven lengte).

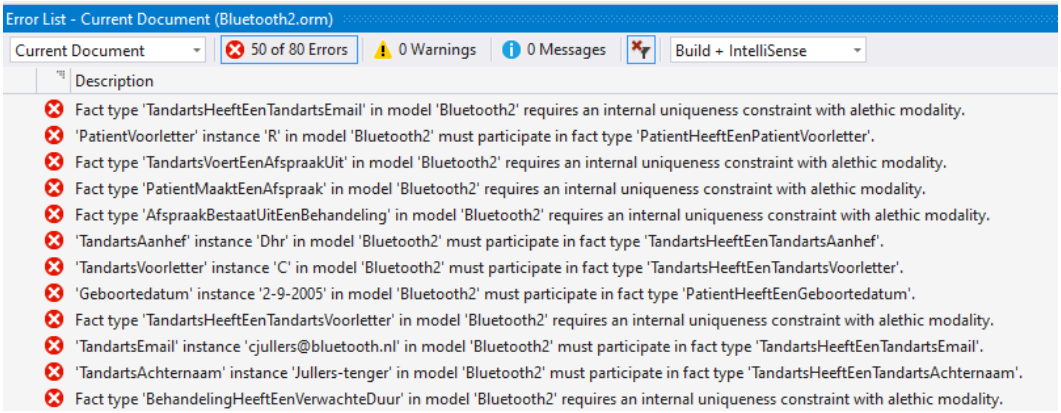
Klik op een entiteit of attribuut en ga naar de **ORM Sample Population Editor**. Vul hier minimaal 2 waarden in.



Tip: Zodra je ergens een waarde vergeten bent zul je een wit blokje zien.



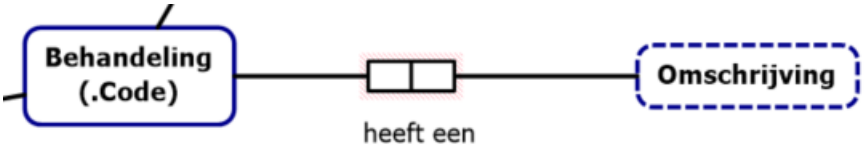
Ondertussen is de Error List gevuld met een heleboel fouten. Dat is normaal in dit stadium van je data-analyse. Echter gaan we ze nu wegwerken door uniciteiten te gaan bepalen en vervolgens de populatie in te vullen voor de relatie.



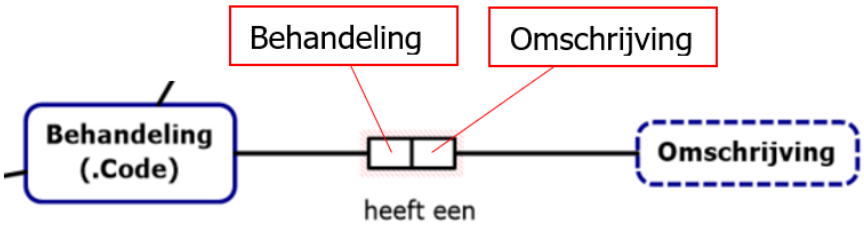
Als je goed kijkt, zie je ook dat dit de enige twee soorten foutmeldingen zijn in de afbeelding hierboven.

Uniciteiten (constraints) opgeven

Het toevoegen van constraints, waaronder ook de primaire uniciteit valt, wordt via het **Document Window** gedaan.



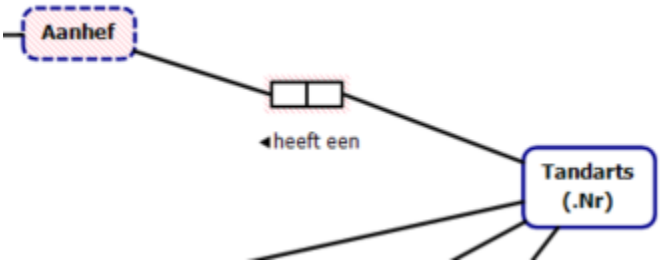
In bovenstaand voorbeeld zie je dat de entiteit behandeling gekoppeld is aan het attribuut omschrijving door een tweetal blokjes. Deze twee blokjes staan gelijk aan de behandeling en de omschrijving.



Bepaal nu aan de hand van de welbekende methode "Heeft een en slecht één" wat de uniciteit is. Kom je er niet aan uit, gebruik dan de techniek die je geleerd is in de ExcelSheet:

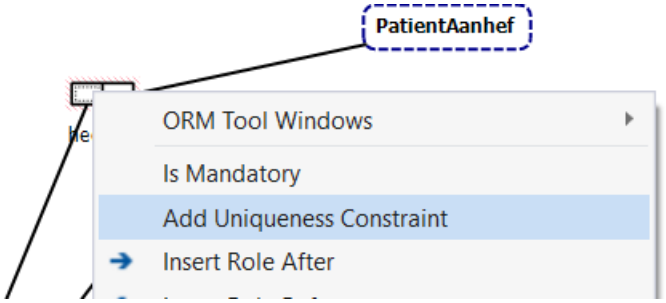
Afspraak(.Nr)	Behandeling(.Code)
321	A64
321	B12
654	A64

Let hierbij wel op de volgorde waarin je dit controleert.



Hier vraag je jezelf af "Een tandarts heeft slechts één aanhef", omdat je de relatie hebt omgedraaid. Dit zie je aan het pijltje voor "heeft een".

De uniciteit kun je plaatsen door op een van de blokjes rechts te klikken en te kiezen voor **Add Uniqueness Constraint**.

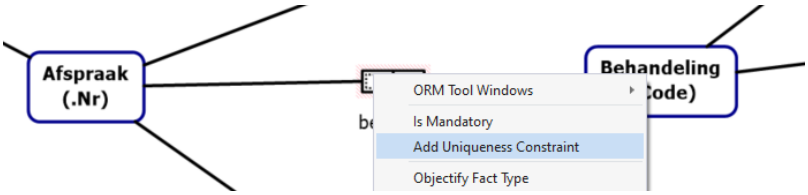


Daarna wordt de constraint toegevoegd. Je kunt dit zien doordat er een paarse streep boven komt te staan.

Samengestelde uniciteit opgeven

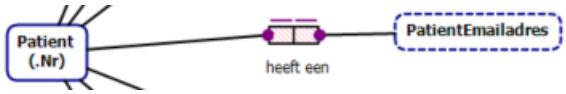
Wil je een samengestelde uniciteit plaatsen, dan selecteer je beide blokjes met behulp van de control toets.

Vervolgens kies je weer voor **Add Uniqueness Constraint**.



Alternatieve uniciteit opgeven die op één attribuut liggen

Dit doe je door ook op dat attribuut rechts te klikken en te kiezen voor **Add Uniqueness Constraint**. Het toevoegen van een alternatieve uniciteit die over meerdere attributen liggen,, geef je aan in je datadictionary.



Je hebt nu aangegeven dat ook het attribuut PatientEmailadres uniek moet zijn.

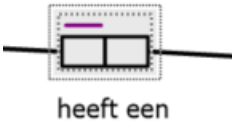
Mandatories

Nadat je de uniciteiten geplaatst hebt is het belangrijk om op te geven welke attributen mandatory zijn. Dit heeft ook invloed op het soort relatie. Als voorbeeld bekijken we een eenvoudige relatie tussen Patient(.Nr) en de GeboorteDatum.

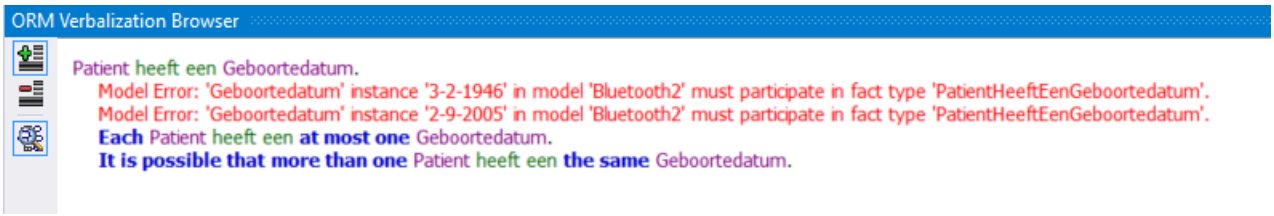
Relatiebeschrijvingen bekijken met de Verbalization Browser

Via de verbalization browser kun je de relatie in een zin in de gaten houden. Handig om te bepalen of de relatie zo geplaatst is zoals jij dat wil. Je krijgt deze zin te zien door de relatie te

selecteren (het geheel van de twee blokjes samen).



Klik vervolgens op de **ORM verbalization Browser** om te bijbehorende zin te kunnen zien.



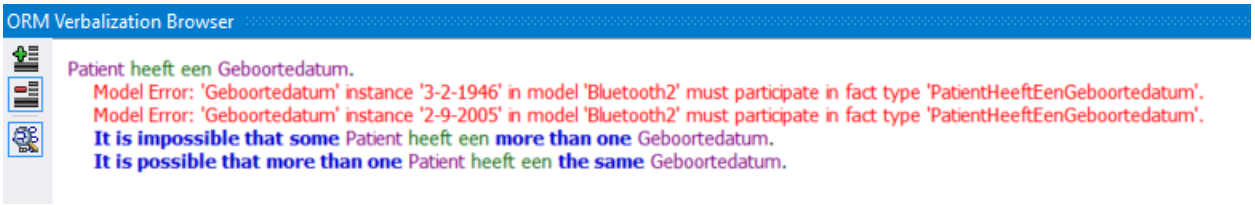
(De foutmeldingen mag je nog even negeren, die los je later op)

Hieruit is op te maken dat de patient minstens één geboortedatum heeft en dat het mogelijk is dat meer dan één patient dezelfde geboortedatum heeft.

Dit klopt niet helemaal. Want een patient heeft ALTIJD maar één geboortedatum. Dit kun je afvangen door gebruik te maken van een mandatory.

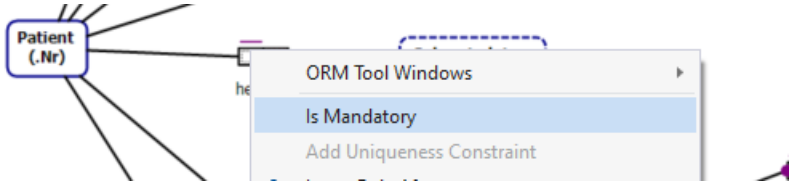
Voor extra controle kun je ook de negatieve relatiebeschrijving bekijken. Relatiebeschrijvingen kunnen best lastig te lezen zijn, waardoor het bekijken vanuit een negatief scenario handig kan zijn om de relatiebeschrijving alsnog te begrijpen.

Je klikt daarvoor op het  logo in de **ORM Verbalization Browser**.

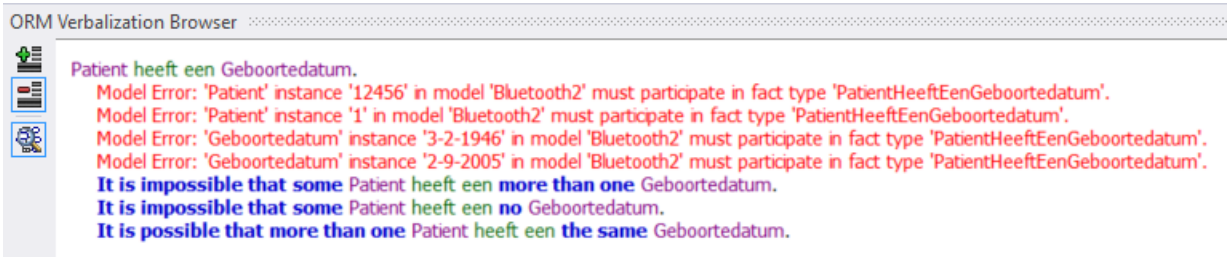


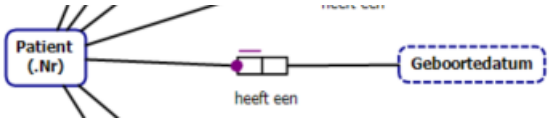
Mandatory opgeven

Door op een blokje te klikken van een relatie kun je aangeven dat het attribuut mandatory is. Blijf vervolgens controleren of de relatiebeschrijving in de **ORM Verbalization Browser** klopt.



Nadat het attribuut mandatory is gemaakt zie je dat de relatiebeschrijving in de **ORM Verbalization Browser** is aangepast naar:

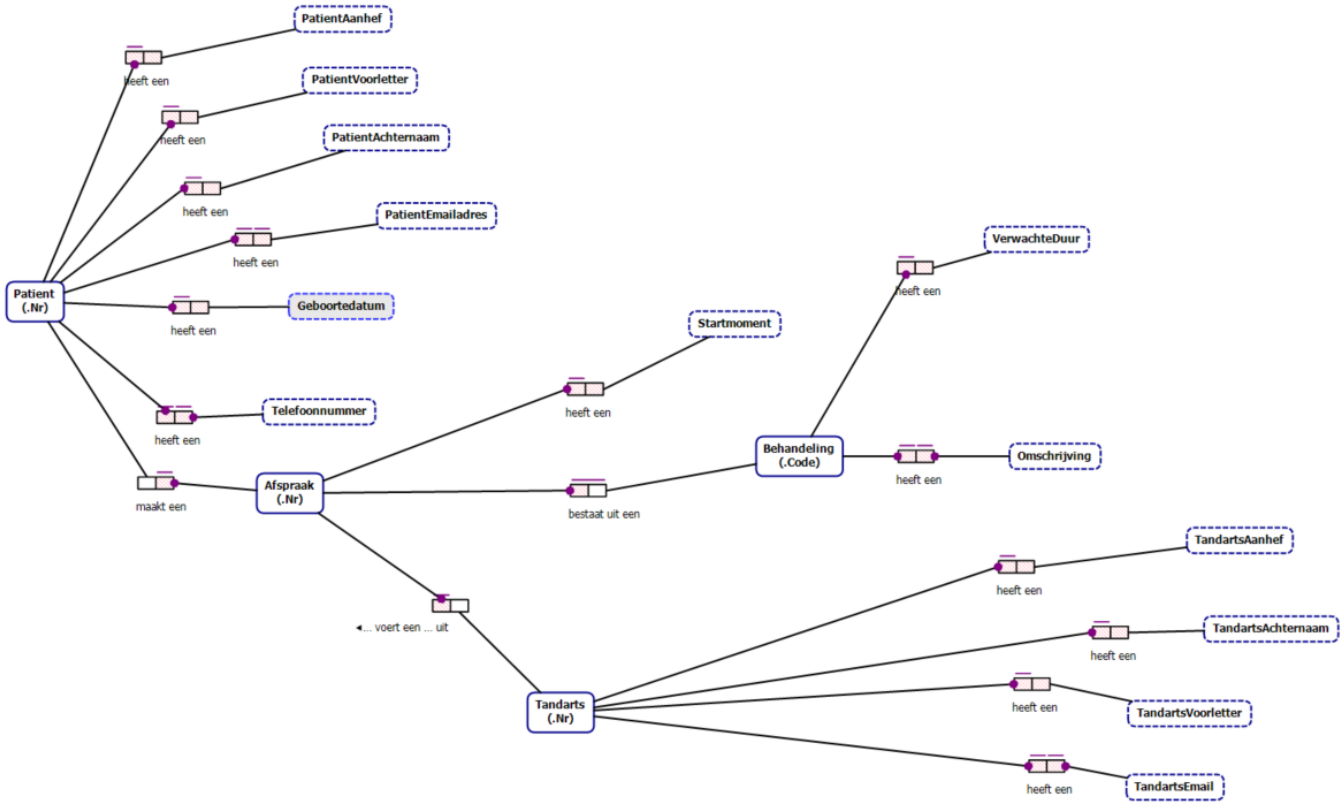




Met mandatory op Patient: betekent dat een Patient altijd een geboortedatum moet hebben.

Zonder mandatory op Patient: betekent dat een Patient niet altijd een geboortedatum moet hebben.

Bij de relatiebeschrijving tussen een entiteit en een attribuut is dit niet heel erg lastig, maar bij een relatiebeschrijving tussen twee entiteiten kan dit een stuk lastiger worden. Lees dan altijd zowel de positieve- als negatieve relatiebeschrijvingen in de **ORM Verbalization Browser**.



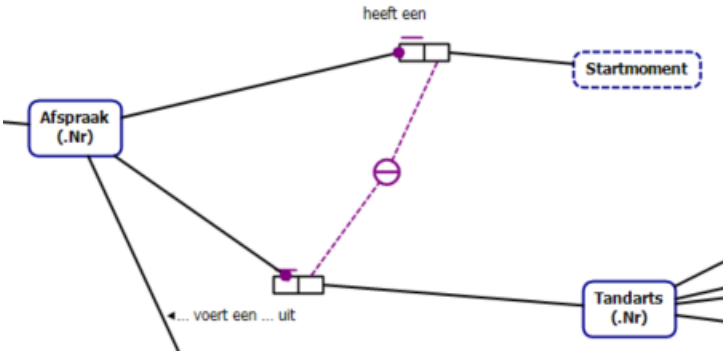
Alternatieve uniciteit opgeven die over meerdere attributen liggen

Je hebt in de Excelsheet ook alternatieve uniciteiten geplaatst die over meerdere attributen liggen.

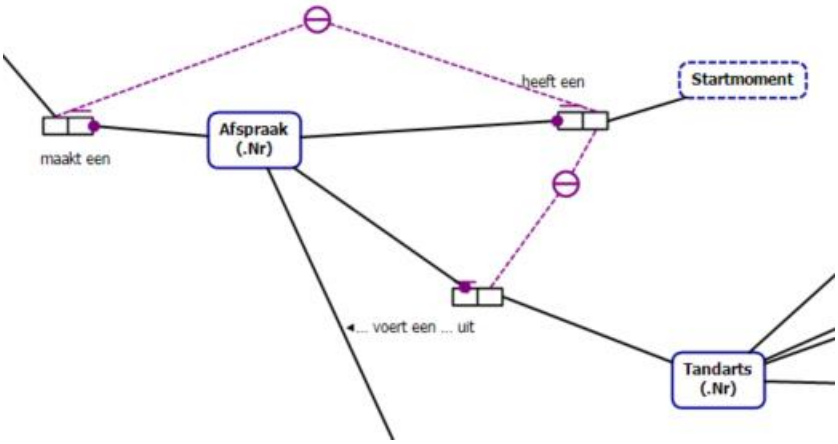
2	Afspraak			
		A3		A3
		A4		
	H2	**invoer**	**D1**	**D3**
	Afspraaknummer	**Startmoment**	**Patientnummer**	**Personeelsnummer**
	1	12-1-2020 09:00	201	9001
	5004	13-1-2020 10:00	202	9008

Ook dit kun je met NORMA aangeven. Dit doe je door gebruik te maken van een **External Uniqueness Constraint**. Klik deze control aan vanuit de **Toolbox** en klik vervolgens in je Document Window. Dubbelklik daarna op het attribuut wat uniek moet zijn.

Om de alternatieve uniciteit A3 van het voorbeeld te maken, leg je de **External Uniqueness Constraint** op deze manier:

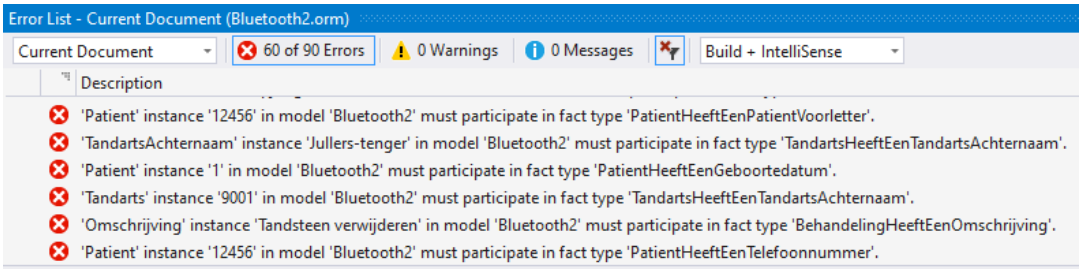


Om de alternatieve uniciteit A3 van het voorbeeld te maken, leg je de **External Uniqueness Constraint** op deze manier:



Voorbeeld populatie opgeven voor relaties

Je hebt nu nog steeds een aantal foutmeldingen in **je Error List**. Dit heeft er mee te maken dat je in de relaties ook nog een populatie moet opgeven.

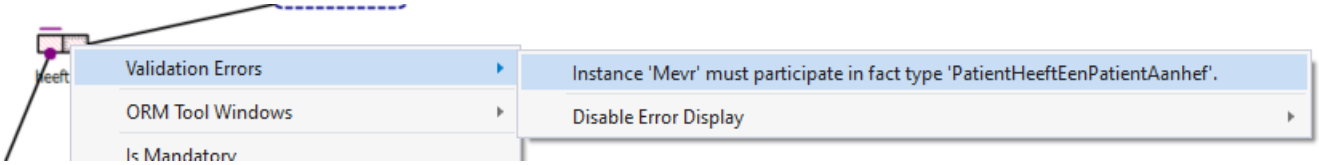


Klik hiervoor op de relatie (de twee blokjes samen) en ga naar de **ORM Sample Population Editor**. Vul hier een geldige populatie in.

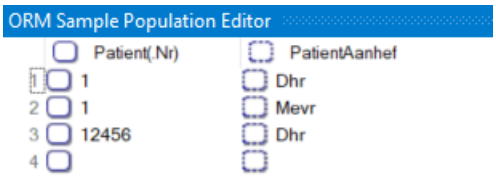
Zodra de rode blokjes wit zijn geworden, weet je dat je een geldige populatie hebt opgegeven.

Je ziet hier geen rode blokjes meer, dus de ingevoerde populatie is geldig.

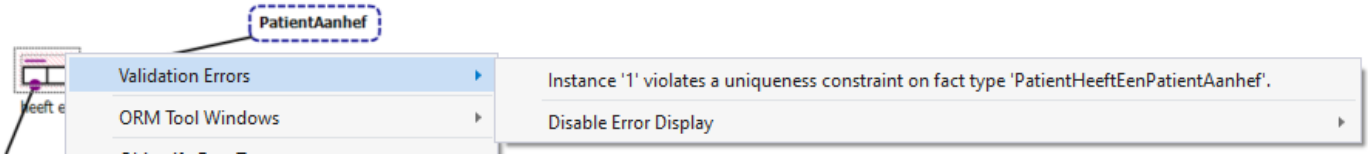
Let op dat je in een relatiebeschrijving altijd de volledige populatie moet gebruiken van hetgeen waar de relatie is opgelegd. Doe je dit niet krijg je een foutmelding.



Ook als je je niet houdt aan de gestelde uniciteiten, ga je een foutmelding krijgen.



Je hebt op Patient(.Nr) een uniciteit geplaatst, dus de ingevoerde populatie is ongeldig:



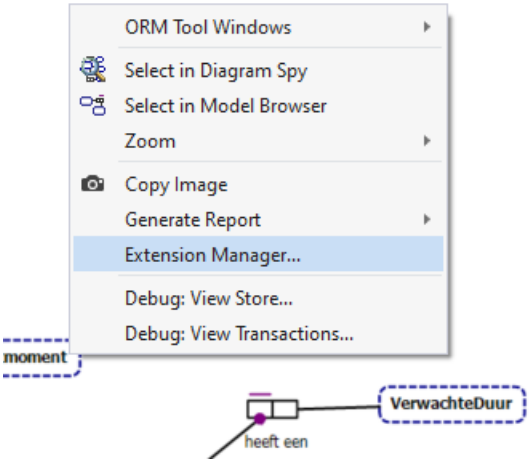
Door je populatie bij relaties in te voeren kun je goed testen of je de uniciteiten goed hebt geplaatst. Wellicht is het dan ook handig om soms meer dan twee voorbeeld populaties toe te voegen!

Na het uitvoeren van deze stap dient de Error List leeg te zijn.

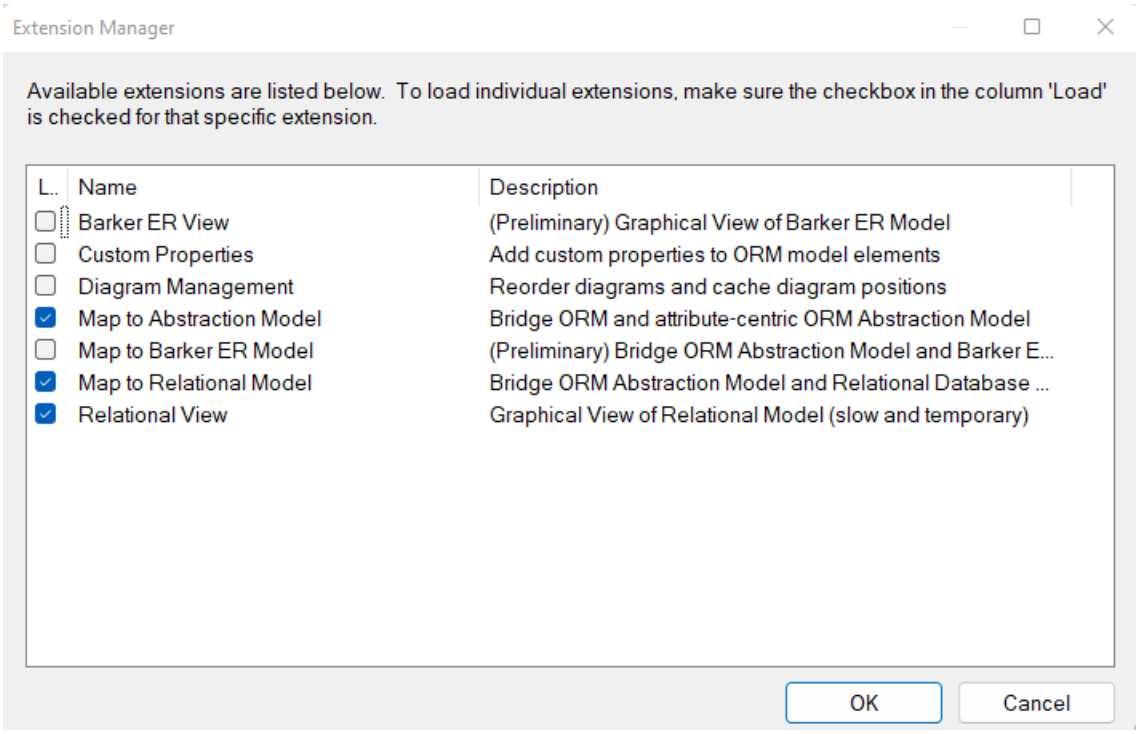
Het genereren van een ERD

Nu er geen foutmeldingen staan in de ERD kun je een ERD laten generen van het schema uit de **Document Window**.

Klik met rechts in de Document Window en klik op **Extension Manager**.

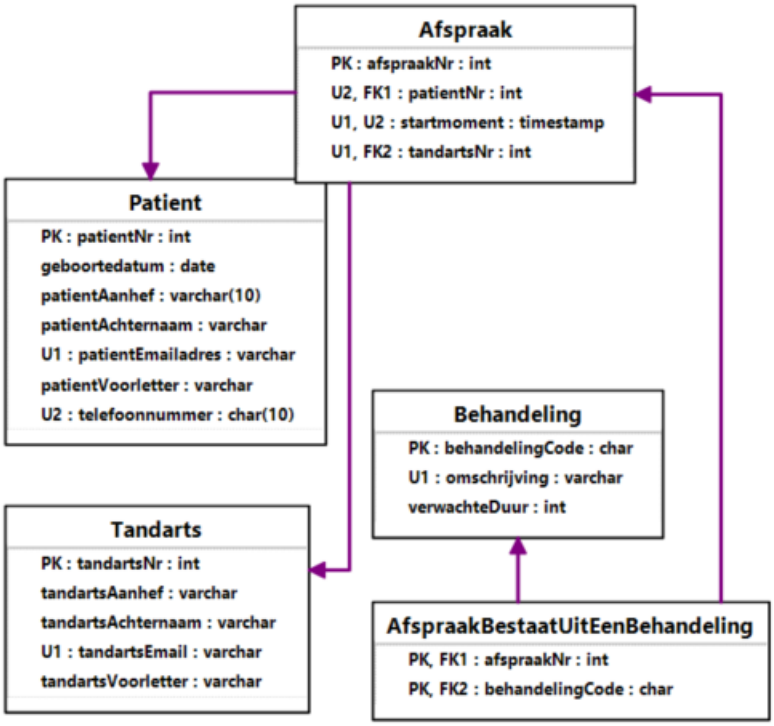


Kies vervolgens voor de opties:



Er is nu een tabblad bijgekomen onder het **Document Window**.

Klik op **Relational View** om de ERD te bekijken.



De weergave ziet er iets anders uit dan je gewend bent, maar de ERD is zo goed als gelijk aan de ERD die je eerder bij PO gezien hebt.

De pijl is hier echter de één kant en de lijn zelf is de véél kant.

Daarnaast zie je hier ook de alternatieve sleutels en de datatypen terugkomen.

Alle dikgedrukte velden zijn ook hier verplichte velden.

Model aanpassen vanuit de Relational View

Omdat de ERD wordt gegenereerd zal niet alles altijd netjes zijn. Als laatste ga je vanuit je ERD de boel netjes maken en controleren. Mocht dat nodig zijn kun je ook nog het een en ander aanpassen aan je model. Dit wordt dan ook direct doorgevoerd.

Tabellen aanpassen

Controleer sowieso of alle tabelnamen netjes zijn. De tabel AfspraakBestaatUitEenBehandeling kun je bijvoorbeeld beter AfspraakBehandeling noemen.

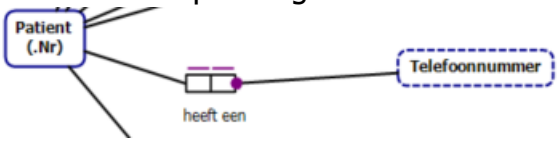
Klik daarvoor op de tabelnaam en pas de propertie **Name** aan.

Verder kun je de Kolomvolgorde aanpassen door ze te verslepen.

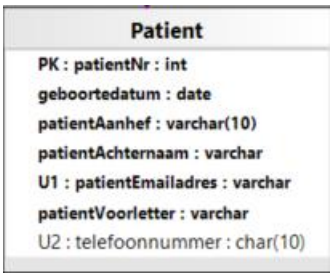
Kolommen aanpassen

Naast tabellen kun je ook van kolommen het een en ander aanpassen. Het telefoonnummer van een patiënt is nu dikgedrukt. Dat betekent dat het een verplicht veld is. Dit kun je nu alsnog aanpassen door op de kolom te klikken en de property **IsNullable** op **true** te zetten.

Een nettere oplossing is echter om de mandatory vanuit het model aan te passen:



Want ook dan wordt de ERD direct aangepast.



Je kunt ook vanuit de **Relational View** de datatypen aanpassen mocht dat nodig zijn. Dit kan echter wel weer leiden tot foutmeldingen in je model, omdat bijvoorbeeld je populatie niet meer klopt. Kijk hier dus mee uit 😊.

Als laatste kun je natuurlijk ook de veldnamen aanpassen. Zorg er voor dat de veldnamen logisch zijn. Dit zijn ook de namen die je straks daadwerkelijk in je database gaat gebruiken. Voorkom dus dat namen van velden die geen primary key of foreign key zijn geen naam van de entiteit in zich hebben.