data warehouse en business intelligence

The Cloud Consultants

**PROCESS LOG**

# Versiebeheer

|  |  |  |  |
| --- | --- | --- | --- |
| Versie | Authors | Datum | Opmerkingen |
| 0.1 | A. van Dalen  S.A. Twardowski | 09-11-2020 | Eerste versie |
| 0.3 |  |  |  |

Inhoudsopgave

[Versiebeheer 1](#_Toc56598878)

[Introductie 3](#_Toc56598879)

[Voorbereiding 4](#_Toc56598880)

[Software keuze 4](#_Toc56598881)

[Schema 4](#_Toc56598882)

[ETL 4](#_Toc56598883)

[Data Warehousing 5](#_Toc56598884)

[Visualisatie 5](#_Toc56598885)

[Data set keuze 6](#_Toc56598886)

[Business vragen 7](#_Toc56598887)

[Uitwerking 8](#_Toc56598888)

[Data warehouse schema’s 8](#_Toc56598889)

[Data warehouse import 9](#_Toc56598890)

# Introductie

Na het afronden van het onderzoek naar data warehouses en business intelligence is het tijd om de theorie in de praktijk toe te passen. Om te kijken of data warehouses en business intelligence goed bij het bedrijf passen is van belang om een voorbeeld systeem op te zetten, een zogenaamd prototype.

Het prototype zal inzicht geven in wat voor soorten informatie uit een data warehouse gehaald kan worden en wat voor toegevoegde waarde dit voor het bedrijf heeft. Dit document beschrijft het gehele prototype wat gemaakt.

# Voorbereiding

## Software keuze

### Schema

Voor schema diagram tekeningen blijven we bij *DRAW.IO.* Goede versie beheer integratie en eenvoudig interface zorgen hier voor vloeibaar iteratie proces.

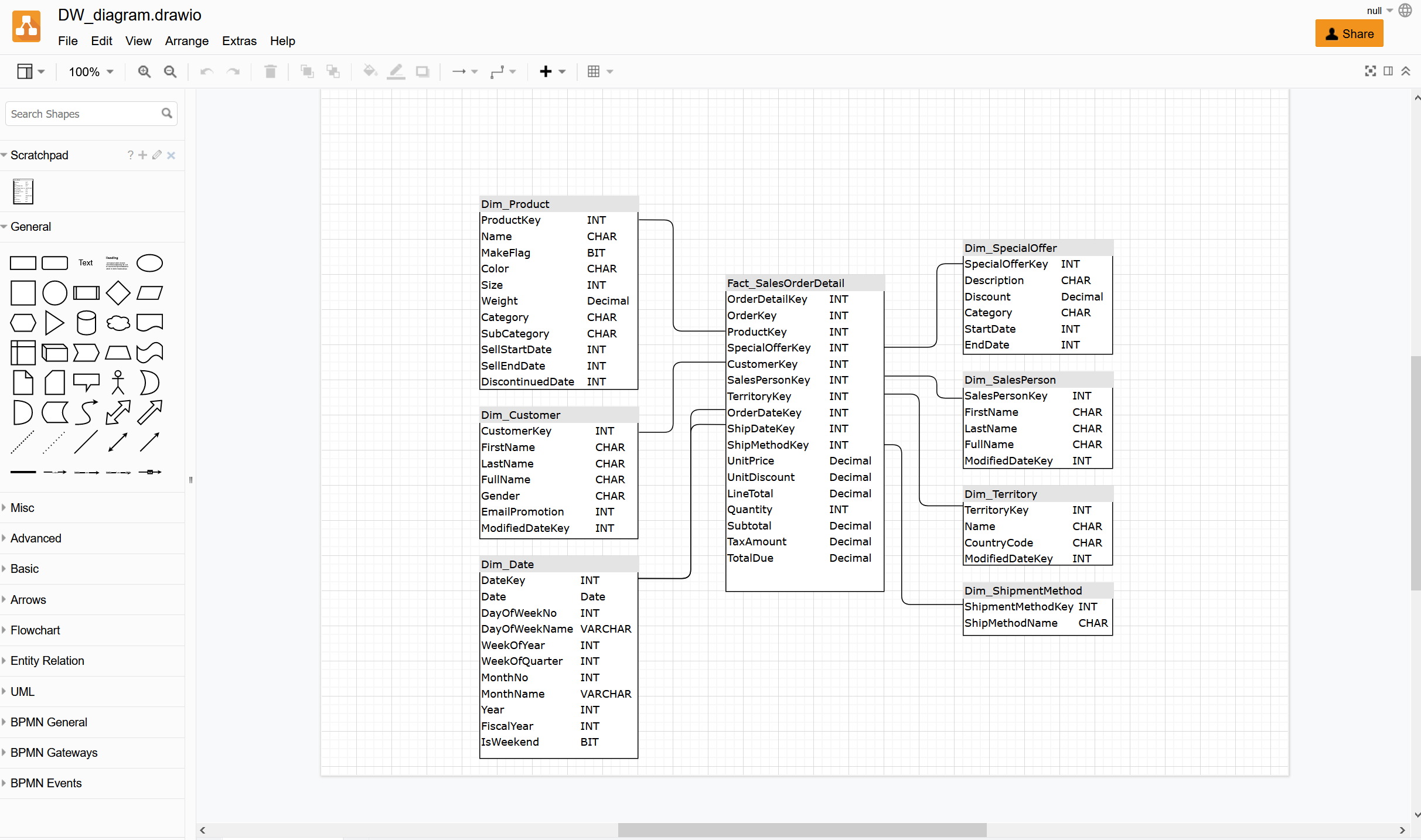


Fig. 1 DRAW.IO

### ETL

We hebben gedurende onderzoek fase aantal verschillende programma’s/platformen onderzocht, zoals *Informatica, Sprinkle, Xplenty, Skyvia* en andere. Iedere platform heeft natuurlijk zijn voordelen en nadelen maar we moeten ook rekening houden met de kosten. Ons keus was dus gemotiveerd niet alleen door de functionaliteit die we nodig hebben om de opdracht te realiseren maar ook de toegankelijkheid van de software voor ons als studenten. Daarom ging ons voorkeur naar *Microsoft SQl Server Integrated Services.*

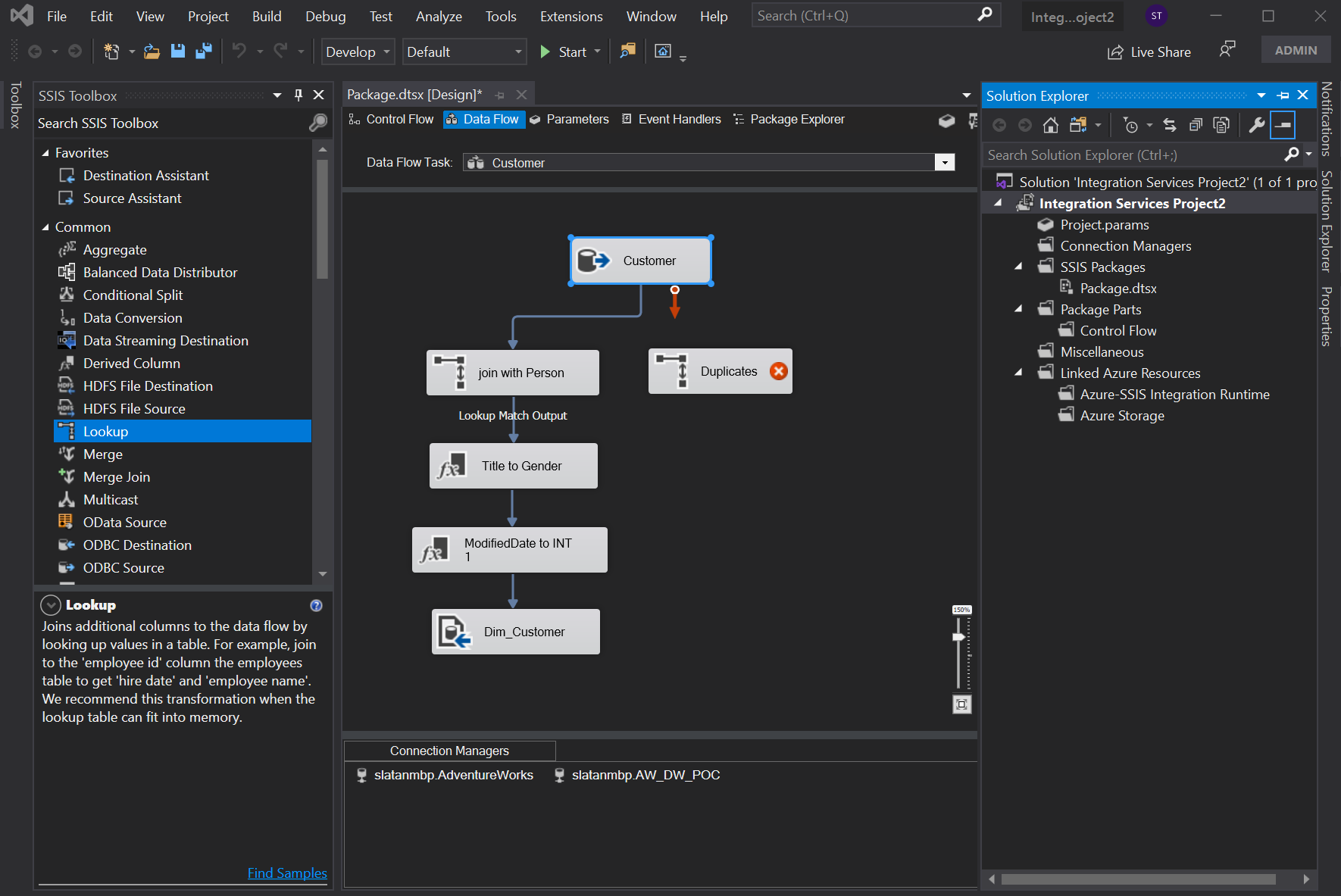
**

Fig. 2 MS SQL SSIS

### Data Warehousing

Voor data warehousing hebben we *SQL Server* gebruikt. Het sluit goed aan op onze behoeftes en we hebben al ervaring ermee. Zo ontkomen we ook potentiële compatibiliteit issues.

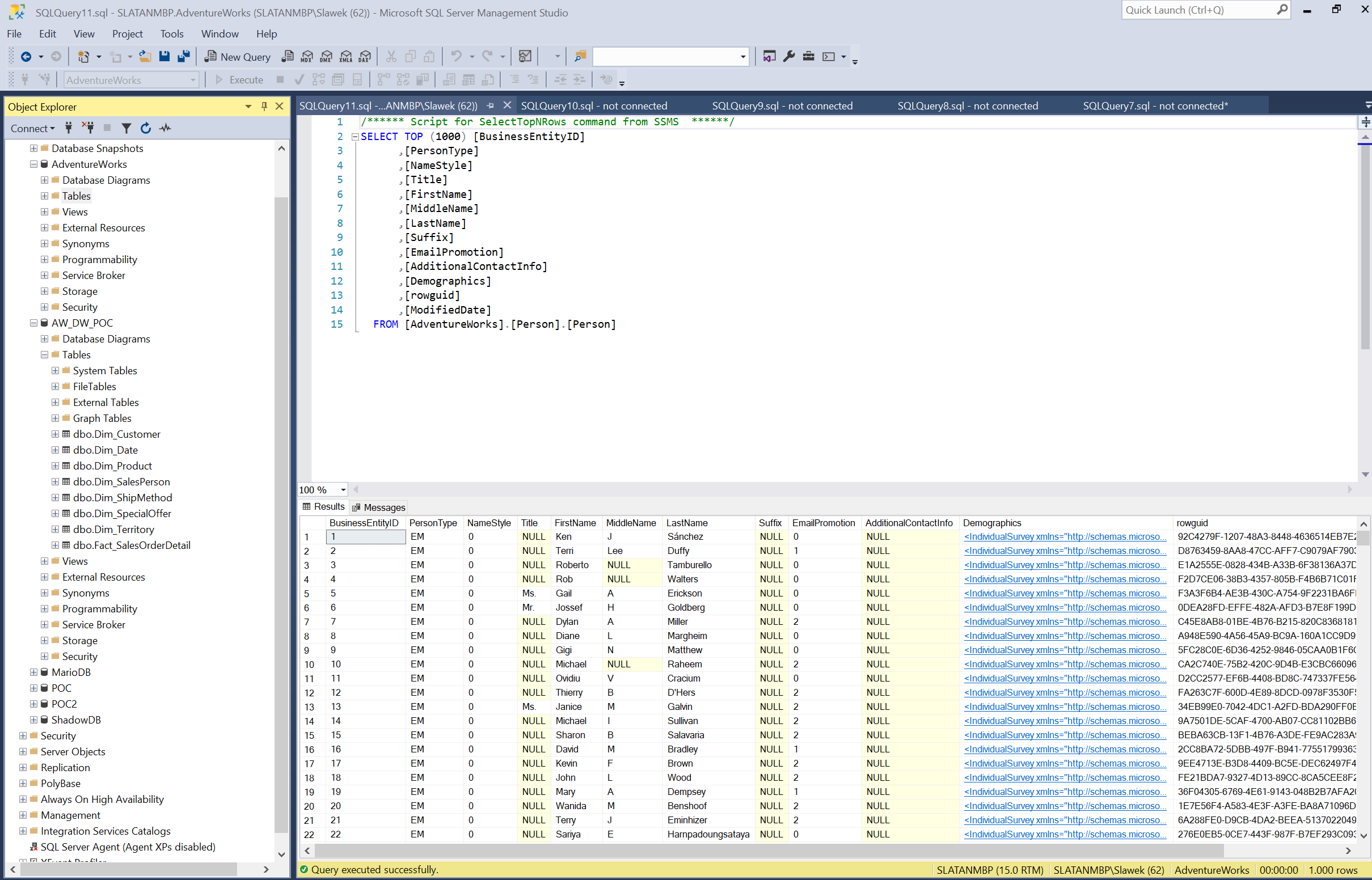


Fig. 3 MS SQL Server

### Visualisatie

Blijvend in *Microsoft* omgeving gaan we gebruik maken van *PowerBI Desktop*. Het blijft steeds veel gebruikt binnen het domain en heeft brede visualisatie functionaliteiten en is gratis voor non-commercial gebruik.

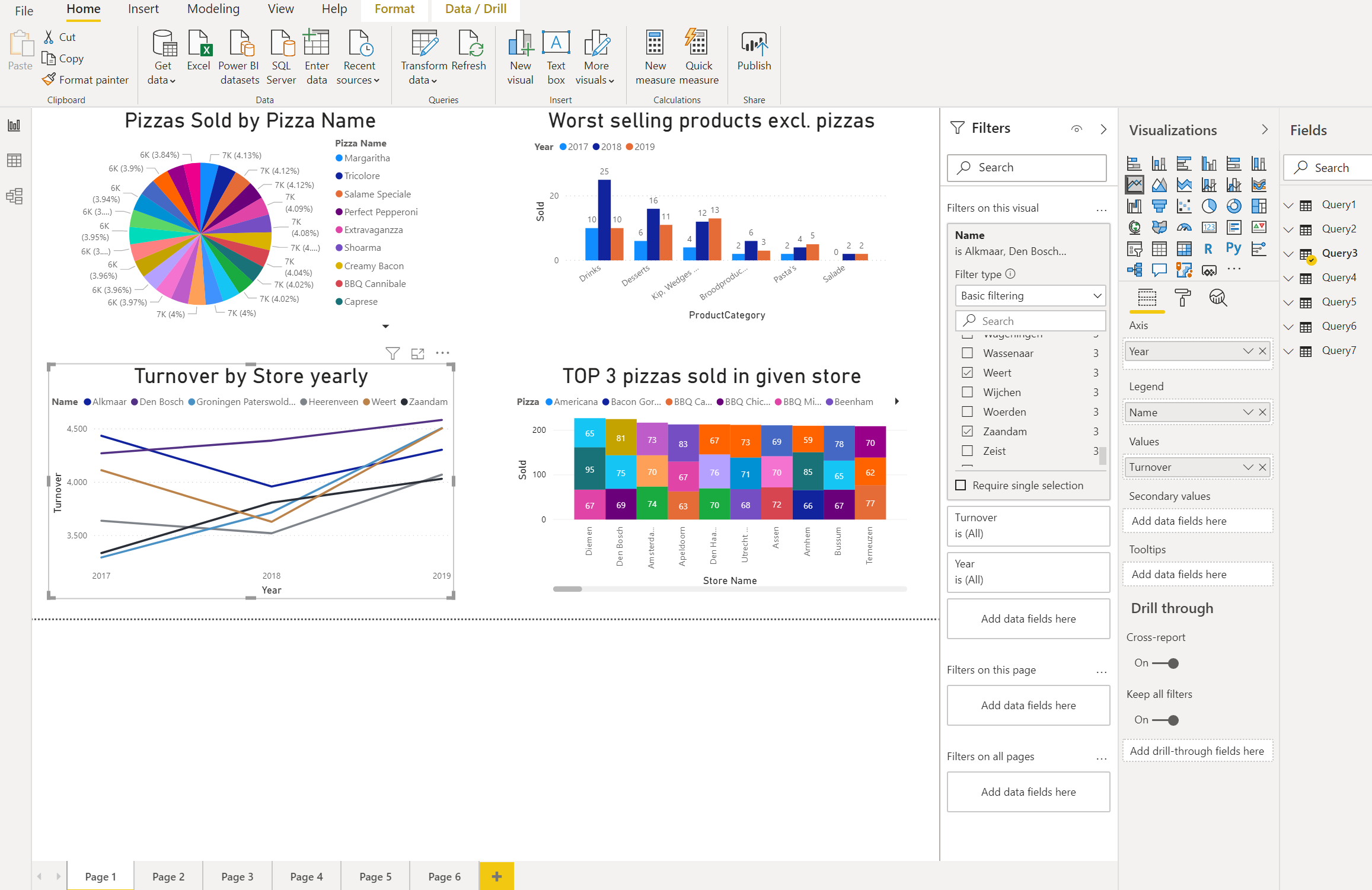


Fig. 4 Power BI

## Data set keuze

Om het proces van data warehousing en visualisatie te kunnen realiseren hebben we natuurlijk een toepasbare data set nodig. Een klassiek voorbeeld hiervan is de *AdventureWorks* data base. Het is een data set ontwikkeld door *Microsoft* en wordt als onderdeel van *SQL Server* geleverd maar is ook openbaar op *GitHub* beschikbaar. Informatie over de data connecties en de betekenis van gebruikte afkortingen etc. is gedetailleerd en veel voorkomend. Naast de officiële documentatie zijn er talloze tutorials te vinden, wat voor ons, gezien de beperkte time-frame ook van belang is.

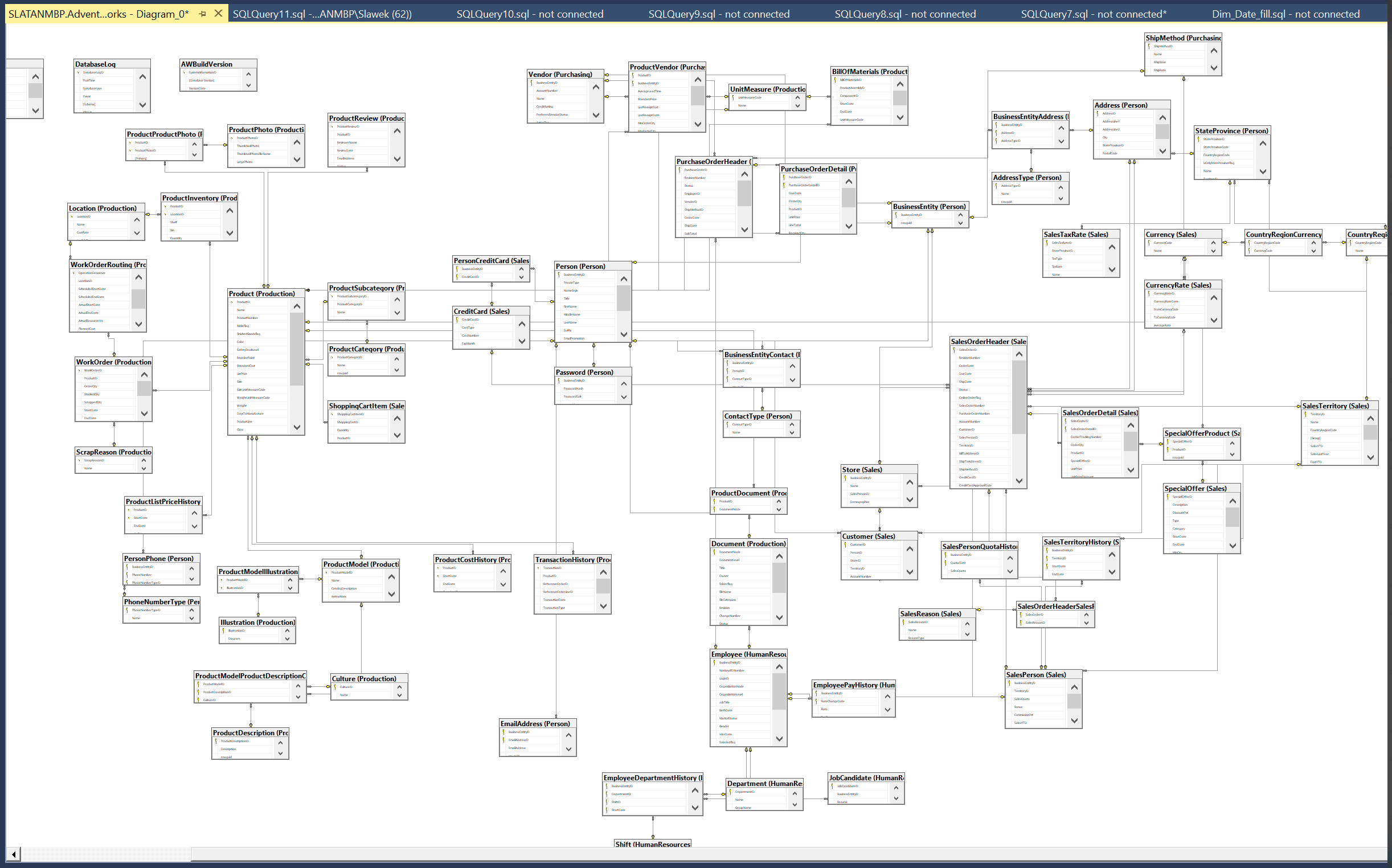


Fig. 5 AdventureWorks dataset

We hebben ons op het Sales gedeelte van de data gefocust. Het is altijd een adequate gebied bij en business scenario en het leent zichzelf goed voor dashboard visualisatie.

## Business vragen

# Uitwerking

## Data warehouse schema’s

We zijn begonnen met een simpel *star schema.*

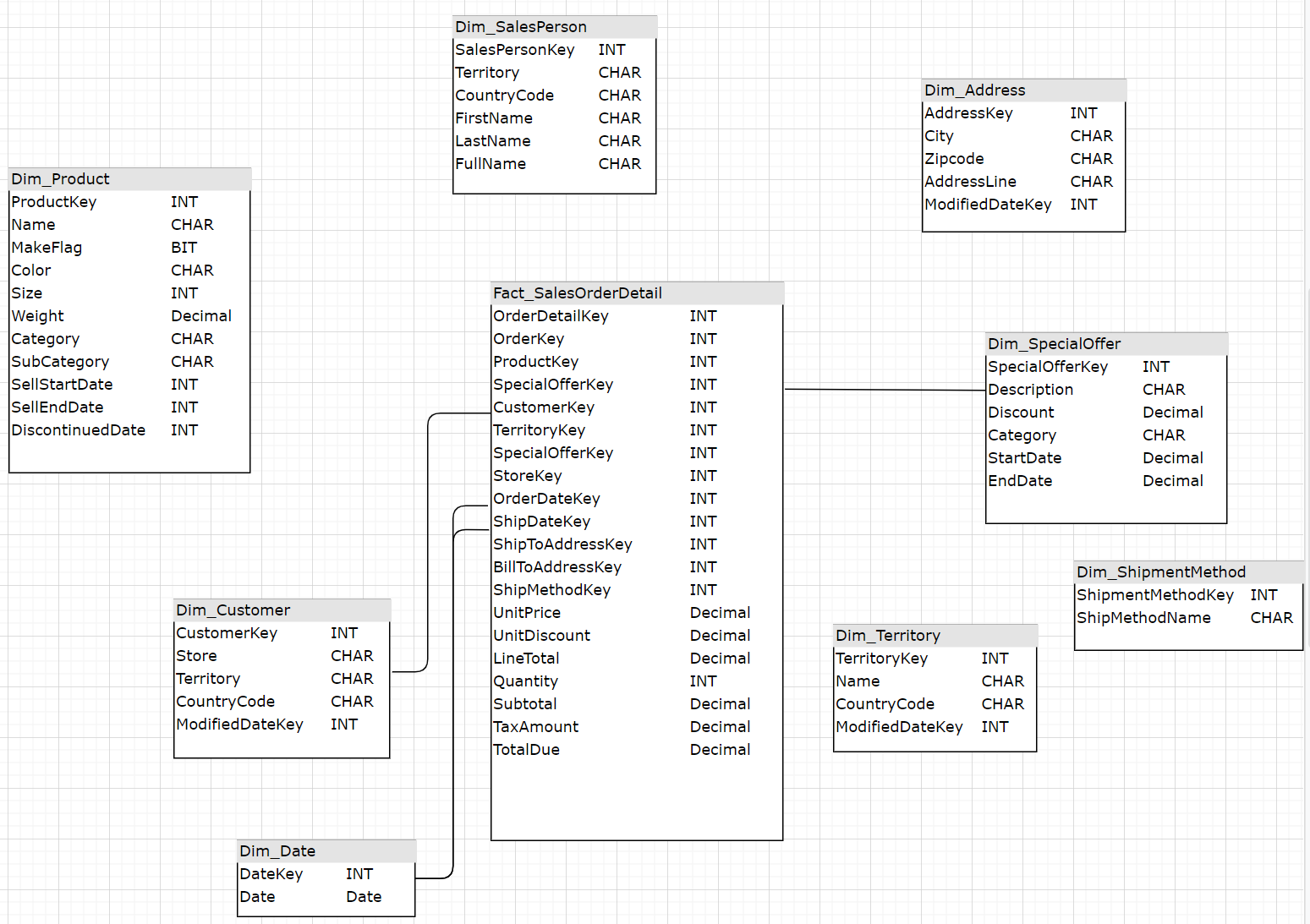
**

Fig. 6 Eerste draft start schema

Ieder tabel krijgt een Key (INT type) die abstract is. Die noemen we *Surrogate Keys*. Dat wil zeggen dat ie tijdens data warehousing gegenereerd werd en uniek blijft. Volgens onderzoek die we hebben uitgevoerd is een abstracte key van belang in een DW voor paar redenen – ten eerste als DW-beheerder heb je niet altijd het toezicht op de source data en je weet niet hoe de keys daarin gegenereerd gaan worden. Als er bv. records uit de source DB verwijderd worden en oude keys hergebruikt gaan worden dan krijg je duplicate keys in je DW. Het kan ook zijn dat voor een of andere reden keys in een DB-tabel bv. een code is, die cijfers en letters bevat. Het is dus een goede praktijk om de DW keys te ontkoppelen en voor performance redenen INT type te gaan gebruiken

Onze fact tabel hebben we gebaseerd op de *SalesOrderDetail* tabel. Daarin staan alle keys die naar de dimensie tabellen wijzen en een aantal facts die we voor visualisatie gaan benutten.

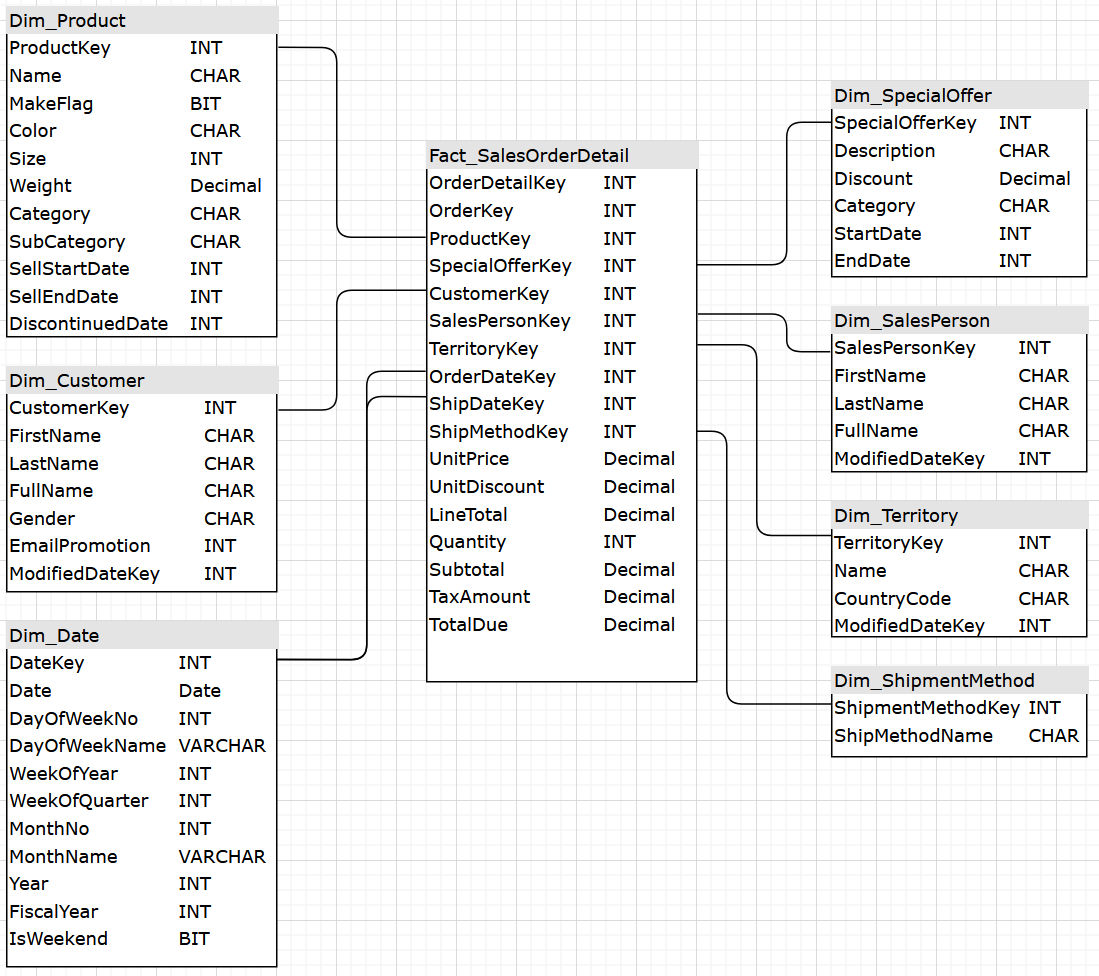


Fig. 7 2de versie van het start schema

Het bleek vrij snel dat weons schema aan moeten passen. Naar aanleiding van een online feedback sessie met de docent en de scholieren en ons eigen uitvindingen tijdens het proces van data set onderzoek, hebben we geconstateerd dat de dimensie tabellen veel informatie bevatten die privacy zorgen opwekken en niet relevant voor de business vragen zijn. Zo gaan we de namen uit de dimensie tabellen eruit halen. De *Dim\_Address* tabel is ook overbodig gezien we een *Dim\_Territory* tabel hebben die voor geografische inzichten zorgt.

## Data warehouse import

We hebben besloten om toch verder te gaan ongeacht het schema niet helemaal naar wens is. We hebben het idee dat het hele proces tot aan visualisatie toe door te lopen een goede oefening is en zou ons verder helpen met ideeën voor verbeteringen m.b.t. opbouw van de data warehouse en de specifieke data die we nodig zouden hebben voor een relevante visualisering. Nadat we het voor elkaar hebben gaan we ons schema aanpassen en de warehouse opnieuw opzetten en op basis daarvan gaan we de finale visualisatie uitwerken.

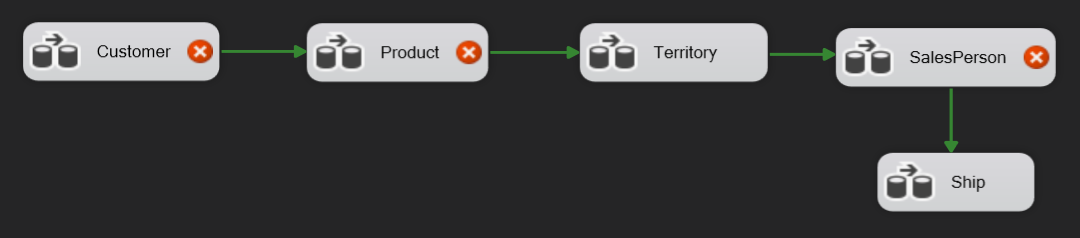


Fig. 8 Data Control Flow

Ieder tabel krijg zijn eigen Control Flow component die de Data Flow structuur opzet omvat.

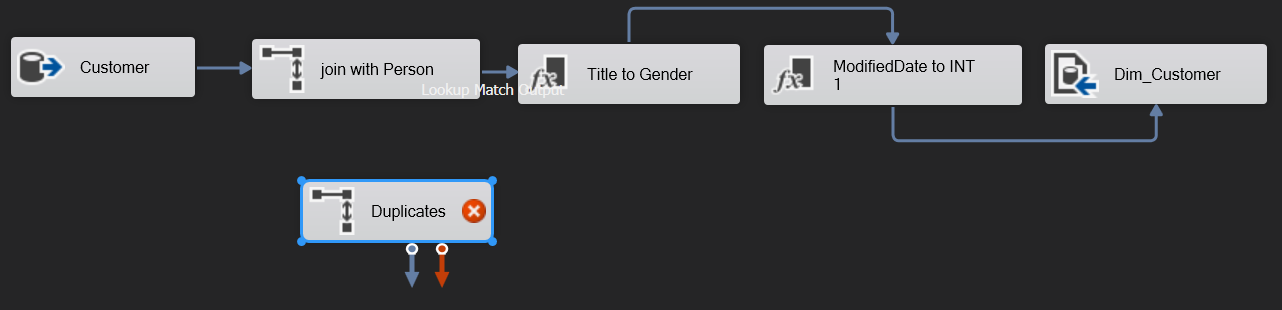


Fig. 9 Data Flow opzet voor Dim\_Customer tabel

Dit opzet zorgt ervoor dat de data import doorgaat op sequentiële manier – als een tabel gevuld wordt gaat het proces verder naar de volgende tabel. Zo kun je de connecties tussen tabellen goed beheren, het zorgt voor betere overzicht in het geval van errors en houdt de stukken gesplitst wat overzichtelijker is.  
Gezien we Module 1 veel bezig waren met SQL queries streven we deze keer naar zo veel mogelijk gebruik te maken van de in *SSIS* beschikbare modulen en de flexibiliteit of juist beperkingen daarvan te leren kennen.