14-3-2022

Casus Kramse

Project Business Intelligence

Jeremy Smits

2141535

Contents

[Inleiding 2](#_Toc99465183)

[Opdracht 1: Extraheer de gegevens 2](#_Toc99465184)

[Opdracht 2: Ontwerp een ster- of sneeuwvlok- of galaxy schema 3](#_Toc99465185)

[Opdracht 3: Implementeer het schema 3](#_Toc99465186)

[Opdracht 4: Vul het datawarehouse vanuit de staging area 4](#_Toc99465187)

[Opdracht 5: Ontwerp dashboards 4](#_Toc99465188)

[Bijlagen 5](#_Toc99465189)

[Bijlage 1: het sterschema 5](#_Toc99465190)

[Bijlage 2: SQL Script voor het sterschema 6](#_Toc99465191)

# Inleiding

Het bedrijf Kramse verzorgt het vervoer van zeecontainers tussen havens in Europa en zijn op zoek naar een business intelligence oplossing. Kramse heeft een aantal gegevens-bronnen die gebruikt kunnen worden voor business intelligence doeleinden. Kramse wil een betere besluitvorming kunnen maken om zo kostenbesparing te vergroten en investeringen beter te benutten.

Kramse heeft twee belangrijke Sustainable Development Goals (SDGs). Namelijk de uitstoot van NOx/SOx en de uitstoot CO2. Deze SDGs zijn gerelateerd aan bedrijfsgroei, brandstofsoort en brandstofverbruik. Het CO2 verbruik hangt af van de vaarroute, snelheid, belading en het type boot.

# Opdracht 1: Extraheer de gegevens

Een belangrijke start voor het business intelligence proces is om de verschillende gegevensbronnen makkelijk bereikbaar te maken en relaties te leggen. In dit project zijn er 4 verschillende bronnen van data namelijk: KramseTPS (Een mdb bestand), Container (Een txt bestand), Consignor (Een xls bestand) en MRV Publication of information (Een xls bestand).

Als eerst heb ik een database aangemaakt genaamd “Kramse\_Raw” om daar alle bestanden in te laden met behulp van de import data functie van het programma SQL Server managementstudio (SSMS). Zo heeft de database een goede tabel structuur voor de verschillende data vanuit de bestanden. Daarna heb ik een SSIS-project aangemaakt in visual studio. In het SSIS-project heb ik een package aangemaakt voor het vullen van de database met de gegevens vanuit de bestanden.

Met een gevulde database kunnen er nu connecties gelegd worden tussen de verschillende tabellen. Ik heb een nieuwe database toegevoegd aan SSMS (Kramse\_RDBMS) en een nieuwe package in het SSIS-project namelijk “RawToRDMS”. De connecties die vanuit het KramseTPS bestand komen heb ik overgenomen in mijn database. Ik heb de container tabel gekoppeld met een foreign key relatie via Shipment\_detail op het attribuut “ContainertypeId” van Shipment\_detail en “Id” van container. Ook heb ik een foreign key relatie toegevoegd tussen “CosignorId” van shipment en “Id” van Consignor. Verder was het opvallend dat het ship tabel een “shipname” artibuut heeft en MRV ook namelijk “Name”. De connectie kan niet zomaar gelegd worden omdat de waardes in ship niet consistent zijn en ook in kleine letters zijn vergeleken met de hoofdletters in het MRV-bestand. Ik heb er daarom voor gekozen om in het SSIS-project de waardes te veranderen naar hoofletters en zo ship te kopellen met MRV.

Net het leggen van de connecties heb ik de database gevuld met behulp van SSIS vanuit “Kramse\_Raw” en is de “Kramse\_RDBMS” nu een gevulde relationele database.

# Opdracht 2: Ontwerp een ster- of sneeuwvlok- of galaxy schema

“Kramse\_RDBMS” is nu een genormaliseerde database en voor de volgende stap wil ik de database gedenormaliseerd hebben, en moet er een schema gemaakt worden. Daarom heb ik de database “Kramse\_PSA” toegevoegd en ben ik goed gaan kijken naar de verschillende tabellen. Ik heb ervoor gekozen om een sterschema toe te passen op de database. De database is niet erg groot en dit is een overzichtelijke oplossing met weinig connecties tussen de verschillende tabellen. Een sneeuwvlok schema is een andere keuze die bij dit project had kunnen passen maar ik ben van mening dat een ster beter past omdat je dan de onnodige connecties weghaalt.

Om tot een sterschema te komen heb ik de tabellen “Voyage”, ”Voyage\_Port”, “Shipment\_Detail” en “MRV” samengevoegd in het feiten tabel shipment. Ik heb ervoor gekozen om de tabellen: “Item”, “Container”, “Ship”, “Port” en “Consignor” te gebruiken als dimensies voor mijn Ster schema. Dit heb ik gedaan in de database door gebruik te maken van de designer ingebouwd in SSMS om zo de tabellen te veranderen.

De opgeleverde bestanden van Kramse staan vol met onbruikbare attributen voor dit project. Een hoop data mist er in de verschillende tabellen en er zijn tabellen die zonder context niet duidelijk zijn voor de gebruikers van de data. Ik heb er daarom voor gekozen om alle tabellen die te veel lege velden bevatten niet op te slaan in mijn stermodel. Daarna ben ik gaan kijken naar dubbele data of data die hetzelfde beschrijft op verschillende plaatsen, Bijvoorbeeld de tabel “Voyage” en “Voyage\_Port” bevatten dezelfde data en de dubbele data heb ik ook uit mijn model gehaald. Daarna ben ik nog gaan kijken naar data die niet te begrijpen is zonder context en data die niks toe kan voegen aan een business intelligence dashboard en heb ik die er ook uitgehaald. In bijlagen 1 zie je het sterschema met de verwijderde tabellen.

# Opdracht 3: Implementeer het schema

Voor het implementeren van het stermodel in een database heb ik een SQL-bestand gemaakt die je kan uitvoeren. Zo krijg je de actuele versie van mijn sterschema die ik gebruikt heb bij mijn Persistent staging area en Operationele Datastore. Het SQL-bestand vind je in bijlage 2.

# Opdracht 4: Vul het datawarehouse vanuit de staging area

In het SSIS project heb ik de packages “RDBMSToPSA” en “PSAToODS” toegevoegd. Tijdens het proces waarbij ik mijn PSA vul vanuit mijn Relationele database heb ik een lookup gebruikt om shipment te vullen met data vanuit het MRV bestand. Verder heb ik een aantal lookups gebruikt die bij shipment detail bijkomen. Wat mij opviel is dat een shipment meerdere shipment details kan hebben. Er kunnen namelijk meerdere items vervoerd worden bij een shipment. Daarom heb ik gekozen om shipment Id te veranderen naar een identity, en heb ik de lookups gedaan op shipment detail. Ik heb “Shipment”, “Voyage” en “Voyage\_Port” toegevoegd aan shipment detail met lookups.

In het “PSAToODS” package heb ik ervoor gekozen om de tabellen “Item” en “Port” te vullen als een fixed slowly changing dimension. Dit komt omdat ik er van uit ga dat de attributen van de twee tabellen niet veranderen in de toekomst. De tabellen “Consignor”, “Container” en “ship” te vullen als een historical slowly changing dimension. Dit komt omdat ik denk dat de tabellen attributen hebben die veranderd kunnen worden, en dat het belangrijk is om deze veranderingen vast te leggen in een database. Bij de tabellen heb ik een nieuwe Id toegevoegd die de rij van de tabel representeert. Ook heb ik een start datum en eind datum toegevoegd aan de tabellen om zo te zien wanneer de rij actief was. In het shipment feitentabel heb ik een aantal lookups gebruikt die zoeken naar het Id actieve record rij. Dit komt natuurlijk omdat de feit origineel een andere Id gebruikte die niet verijsde naar een rij.

# Opdracht 5: Ontwerp dashboards

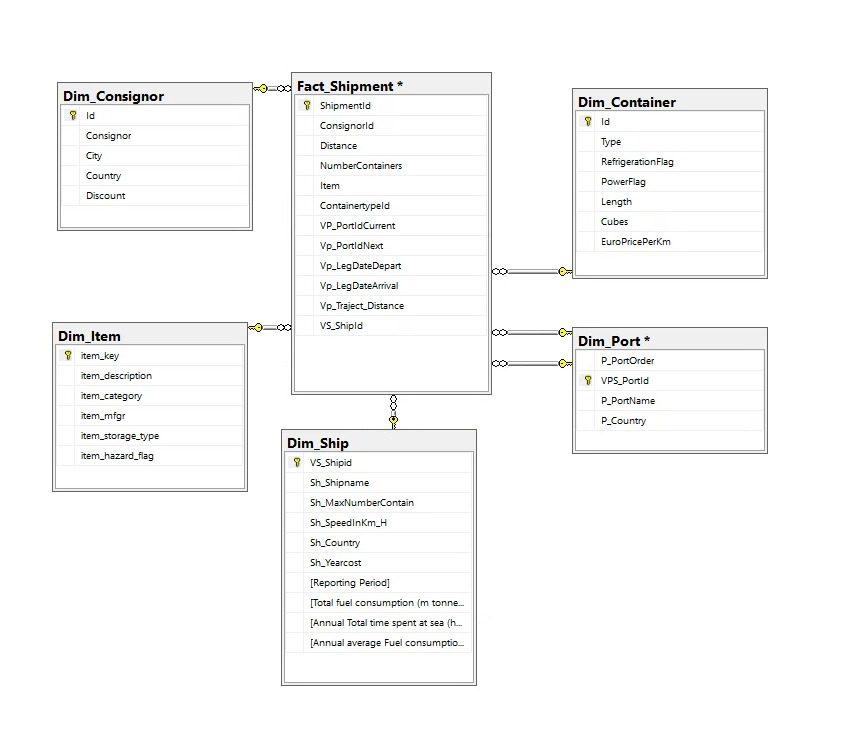
* Ontwerp één of meerdere dashboards waarmee in de informatiebehoeften van Kramse voorzien kan worden.

*Neem in je verslag in ieder geval antwoord op de volgende vragen op:*

* *Waarom heb je gekozen voor (de vormgeving van) de door jou ontworpen dashboards?*
* *Welke conclusies kan Kramse trekken op basis van de dashboards?*
* *Welke acties zou Kramse (kunnen) ondernemen op basis van de voorgaande conclusies?*

# Bijlagen

### Bijlage 1: het sterschema



### Bijlage 2: SQL Script voor het sterschema

USE [Kramse\_PSA]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Dim\_Consignor] Script Date: 29/03/2022 16:39:17 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Dim\_Consignor](

[Id] [int] NOT NULL,

[Consignor] [nvarchar](255) NULL,

[City] [nvarchar](255) NULL,

[Country] [nvarchar](255) NULL,

[Discount] [float] NULL,

CONSTRAINT [PK\_RAW\_CONSIGNOR] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Dim\_Container] Script Date: 29/03/2022 16:39:17 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Dim\_Container](

[Id] [int] NOT NULL,

[Type] [nvarchar](50) NULL,

[RefrigerationFlag] [nvarchar](50) NULL,

[PowerFlag] [nvarchar](50) NULL,

[Length] [decimal](28, 2) NULL,

[Cubes] [decimal](28, 2) NULL,

[EuroPricePerKm] [decimal](28, 2) NULL,

CONSTRAINT [PK\_RAW\_CONTAINER] PRIMARY KEY CLUSTERED

(

[Id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Dim\_Item] Script Date: 29/03/2022 16:39:17 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Dim\_Item](

[item\_key] [int] NOT NULL,

[item\_description] [nvarchar](50) NULL,

[item\_category] [nvarchar](50) NULL,

[item\_mfgr] [nvarchar](50) NULL,

[item\_storage\_type] [nvarchar](50) NULL,

[item\_hazard\_flag] [nvarchar](50) NULL,

CONSTRAINT [PK\_TPS\_Item] PRIMARY KEY CLUSTERED

(

[item\_key] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Dim\_Port] Script Date: 29/03/2022 16:39:17 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Dim\_Port](

[P\_PortOrder] [nvarchar](50) NULL,

[VPS\_PortId] [int] NOT NULL,

[P\_PortName] [nvarchar](50) NULL,

[P\_Country] [nvarchar](50) NULL,

CONSTRAINT [PK\_TPS\_Port] PRIMARY KEY CLUSTERED

(

[VPS\_PortId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Dim\_Ship] Script Date: 29/03/2022 16:39:17 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Dim\_Ship](

[VS\_Shipid] [int] NOT NULL,

[Sh\_Shipname] [nvarchar](255) NULL,

[Sh\_MaxNumberContain] [int] NULL,

[Sh\_SpeedInKm\_H] [int] NULL,

[Sh\_Country] [nvarchar](50) NULL,

[Sh\_Yearcost] [money] NULL,

[Reporting Period] [float] NULL,

[Total fuel consumption (m tonnes)] [float] NULL,

[Annual Total time spent at sea (hours)] [float] NULL,

[Annual average Fuel consumption per distance (kg / n mile)] [float] NULL,

CONSTRAINT [PK\_TPS\_Ship] PRIMARY KEY CLUSTERED

(

[VS\_Shipid] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Fact\_Shipment] Script Date: 29/03/2022 16:39:17 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Fact\_Shipment](

[ShipmentId] [int] IDENTITY(1,1) NOT NULL,

[ConsignorId] [int] NULL,

[Distance] [int] NULL,

[NumberContainers] [int] NULL,

[Item] [int] NULL,

[ContainertypeId] [int] NULL,

[VP\_PortIdCurrent] [int] NULL,

[Vp\_PortIdNext] [int] NULL,

[Vp\_LegDateDepart] [date] NULL,

[Vp\_LegDateArrival] [date] NULL,

[Vp\_Traject\_Distance] [int] NULL,

[VS\_ShipId] [int] NULL,

CONSTRAINT [PK\_TPS\_Shipment] PRIMARY KEY CLUSTERED

(

[ShipmentId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Fact\_Shipment] WITH CHECK ADD CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Container] FOREIGN KEY([ContainertypeId])

REFERENCES [dbo].[Dim\_Container] ([Id])

GO

ALTER TABLE [dbo].[Fact\_Shipment] CHECK CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Container]

GO

ALTER TABLE [dbo].[Fact\_Shipment] WITH CHECK ADD CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Item] FOREIGN KEY([Item])

REFERENCES [dbo].[Dim\_Item] ([item\_key])

GO

ALTER TABLE [dbo].[Fact\_Shipment] CHECK CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Item]

GO

ALTER TABLE [dbo].[Fact\_Shipment] WITH CHECK ADD CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Port] FOREIGN KEY([VP\_PortIdCurrent])

REFERENCES [dbo].[Dim\_Port] ([VPS\_PortId])

GO

ALTER TABLE [dbo].[Fact\_Shipment] CHECK CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Port]

GO

ALTER TABLE [dbo].[Fact\_Shipment] WITH CHECK ADD CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Port1] FOREIGN KEY([Vp\_PortIdNext])

REFERENCES [dbo].[Dim\_Port] ([VPS\_PortId])

GO

ALTER TABLE [dbo].[Fact\_Shipment] CHECK CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Port1]

GO

ALTER TABLE [dbo].[Fact\_Shipment] WITH CHECK ADD CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Ship] FOREIGN KEY([VS\_ShipId])

REFERENCES [dbo].[Dim\_Ship] ([VS\_Shipid])

GO

ALTER TABLE [dbo].[Fact\_Shipment] CHECK CONSTRAINT [FK\_Fact\_Shipment\_Dim\_Ship]

GO

ALTER TABLE [dbo].[Fact\_Shipment] WITH CHECK ADD CONSTRAINT [FK\_Fact\_Shipment\_RAW\_CONSIGNOR] FOREIGN KEY([ConsignorId])

REFERENCES [dbo].[Dim\_Consignor] ([Id])

GO

ALTER TABLE [dbo].[Fact\_Shipment] CHECK CONSTRAINT [FK\_Fact\_Shipment\_RAW\_CONSIGNOR]

GO

USE [master]

GO

ALTER DATABASE [Kramse\_PSA] SET READ\_WRITE

GO