25-3-2020

Productverantwoordingsverslag

Kramse

Vincent Hendriks
STUDENTNUMBER 2139897, 23INVT2B

Inhoud

1. Inleiding

Dit document dient als ondersteuning bij de Business Intelligence uitwerking die opgesteld is voor Kramse.
In het kader van het vak “Project Business Intelligence” is een casus beschreven over het hypothetische bedrijf “Kramse”. Kramse is een zeecontainer vervoersbedrijf tussen havens in Europa. Om dit verslag beter te belichten zijn een aantal kernpunten binnen de casus samengevat.
De missie en visie van Kramse luidt: “Best in class to connect and simplify supply chains across the globe”.
De strategie van Kramse is samengevat onder de volgende drie punten:

- Managed and operated as an integrated company.
- A one company structure with multiple brands.
- Growing topline, earnings for our owners, and opportunities for our people.

Dit document betreft voornamelijk informatie over de gezette stappen om de data te kunnen analyseren. Er wordt beschreven wat er met de data is gedaan, hoe de data is verwerkt en waarvoor de data gebruikt kan worden.

Kramse heeft voor 2017 de volgende (sub)doelen geformuleerd:
· CO2 reductie/NOx/SOx: verlagen van het brandstofverbruik met 10% (brandstofverbruik/per ton vracht per zeemijl)
· Utilisatiegraad verbeteren: Reduceren van de wachttijden in de havens, de zogenaamde idle time, met 5%.

Kostenverlaging en utilisatie
· Beladingsgraad per schip per haven.
· Wachttijden (‘idle time’) per haven en per schip inclusief trends (afgelopen jaren).
· Brandstofverbruik (in relatie tot optimale snelheid) in tonnen brandstof per ton vracht per zeemijl; een en ander per type boot, vaarroute en seizoen.
· Kosten per ton vracht per schip per route.

Besluitvorming
Op basis van bovenstaande informatievragen moeten de directie in staat zijn zodanig te sturen dat de eerder geformuleerde doelen kunnen worden gerealiseerd.

2. Extractie van brondata

De data van Kramse is verdeeld over vier verschillende bestanden. Dit betreft een Microsoft Access database, een tekstbestand en twee Microsoft Excel bestanden. Er is een SQL database aangemaakt en gevuld met de informatie uit de verschillende bronbestanden. Voor de Excel bestanden en het tekstbestand zijn drie nieuwe tabellen aangemaakt in de SQL database. De bestanden zijn opgeslagen in een bronfolder, wanneer er wijzigingen zijn kunnen deze via een proces in Visual Studio doorgevoerd worden naar de SQL server. In Visual Studio is gebruik gemaakt van een SSIS project. Het doel van de extractie is om een centrale plek te maken voor de brondata. Zo is er maar 1 verbinding nodig met de "RAW database" om de data op te halen.

2.1. Extractie van de Microsoft Access database

Voor de extractie van de Microsoft Access database is in SQL server onder tasks gebruik gemaakt van "Import Data". Vervolgens is als source een Microsoft Access database bestand gekozen. Op deze wijze zijn de kolomnamen en tabellen uit de Access database geïmporteerd in SQL Server. De data in de database is met behulp van een Visual Studio proces overgezet. Dit is gedaan door middel van een package. Deze package bevat tasks. Voor elke tabel is een task aangemaakt. Elke task haalt de data binnen een specifieke table (een bestand in het geval van de excel sheets of een tekstbestand) op en verplaatst deze naar een toegewezen table in de SQL database. In het geval van de Microsoft Access database is er een OLE DB Source node die een referentie bevat naar het Microsoft Access bronbestand. Er is tevens een OLE DB Destination node die de opgehaalde data verplaatst naar de SQL Server Database.

2.2. Extractie van de Microsoft Excel bestanden

Voor de Excelbestanden is grofweg hetzelfde proces uitgevoerd. Net als voor de Access database zijn er tasks aangemaakt voor de Excel bestanden.
In het MRV bestand is wel een probleem ontdekt: waarden uit 1 kolom waren truncated omdat de waarden in de kolom te groot waren. Als oplossing hiervoor is ingesteld dat de waardes uit de betreffende kolom geen error genereren wanneer deze te groot zijn.

2.3. Extractie van het tekstbestand

Voor het tekstbestand is gebruik gemaakt van een task voor het importeren van een flat file. Er zijn hier geen fouten gedetecteerd en het was mogelijk om het tekstbestand zonder problemen als table in de database te importeren. Het tekstbestand wordt ingelezen als nvarchar(50). Later in het proces worden de correcte gegevenstypen toegewezen aan de kolommen.

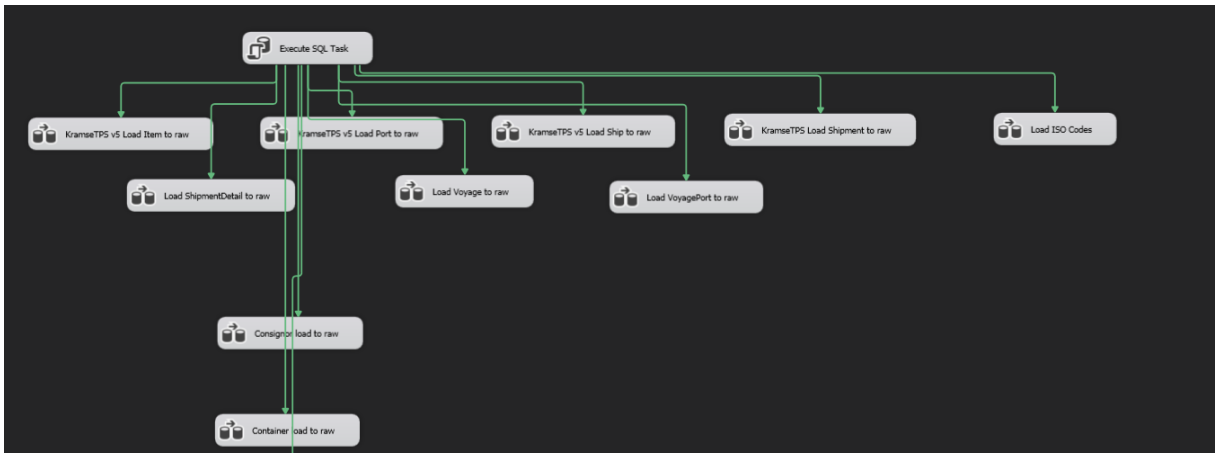
2.4. Naamgeving binnen de database

Naast de tabellen uit de Microsoft Access Database is er één table aangemaakt per ander bronbestand.
Dit zijn:

- dbo.EU_MRV
- dbo.Consignor
- dbo.Container

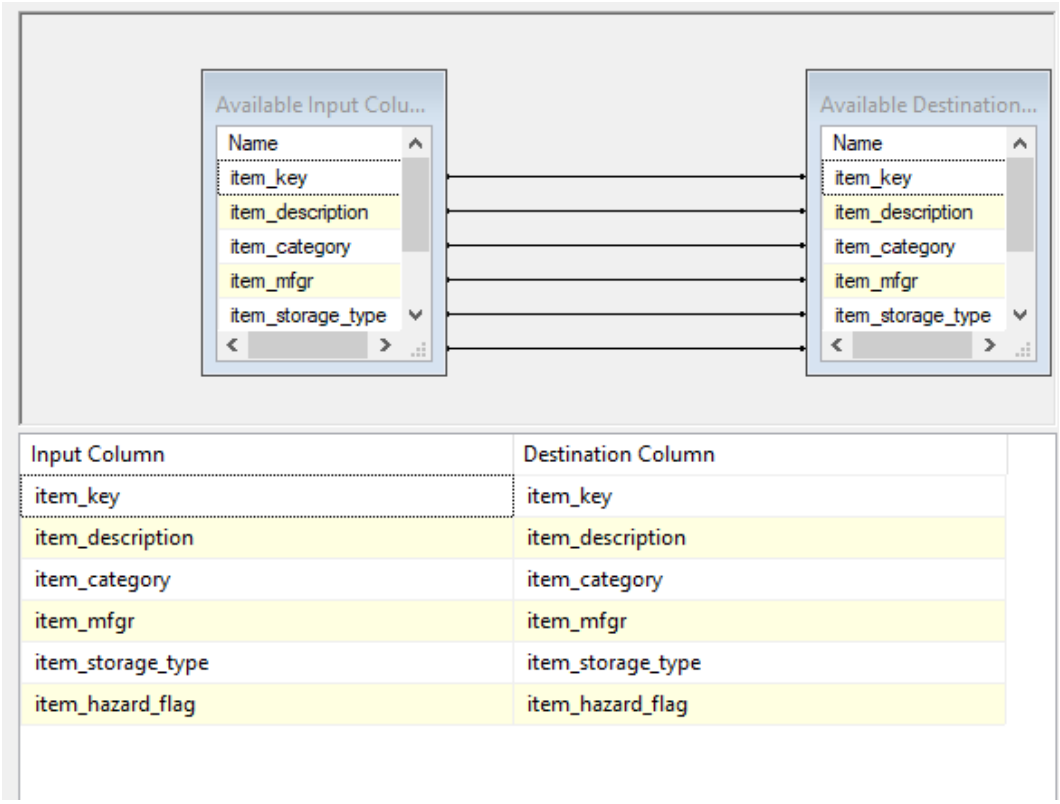
Er is niet gekozen voor het toevoegen van speciale tekens omdat er verwacht werd dat dit complicaties op zou kunnen leveren. Om deze reden zijn de tabelnamen afwijkend van de bestandsnamen bij het importeren.

2.5. Brondata in Visual Studio



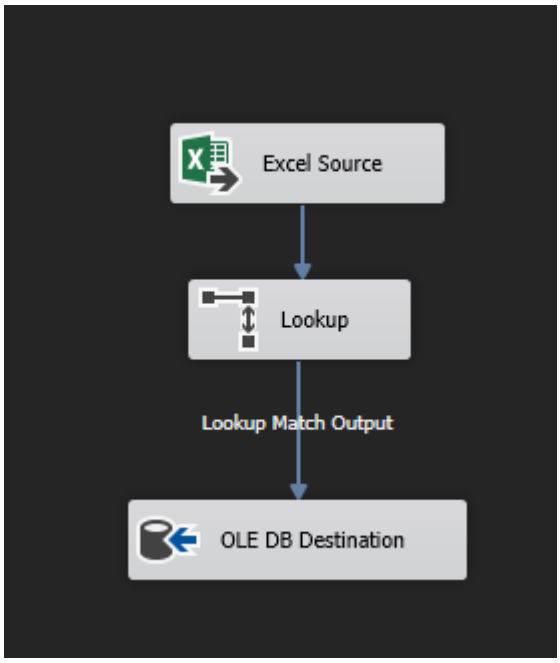
Figuur 2.5.1: Een afbeelding van de brondata in Visual Studio

De brondata wordt ingeladen in Visual Studio, hierin wordt deze overgezet uit source bestanden naar de RAW database. Er zitten in de RAW database nog geen relaties.
Voor het uitvoeren van dit proces in Visual Studio wordt eerst alle bestaande data verwijderd om duplicatie van gegevens te vermijden.

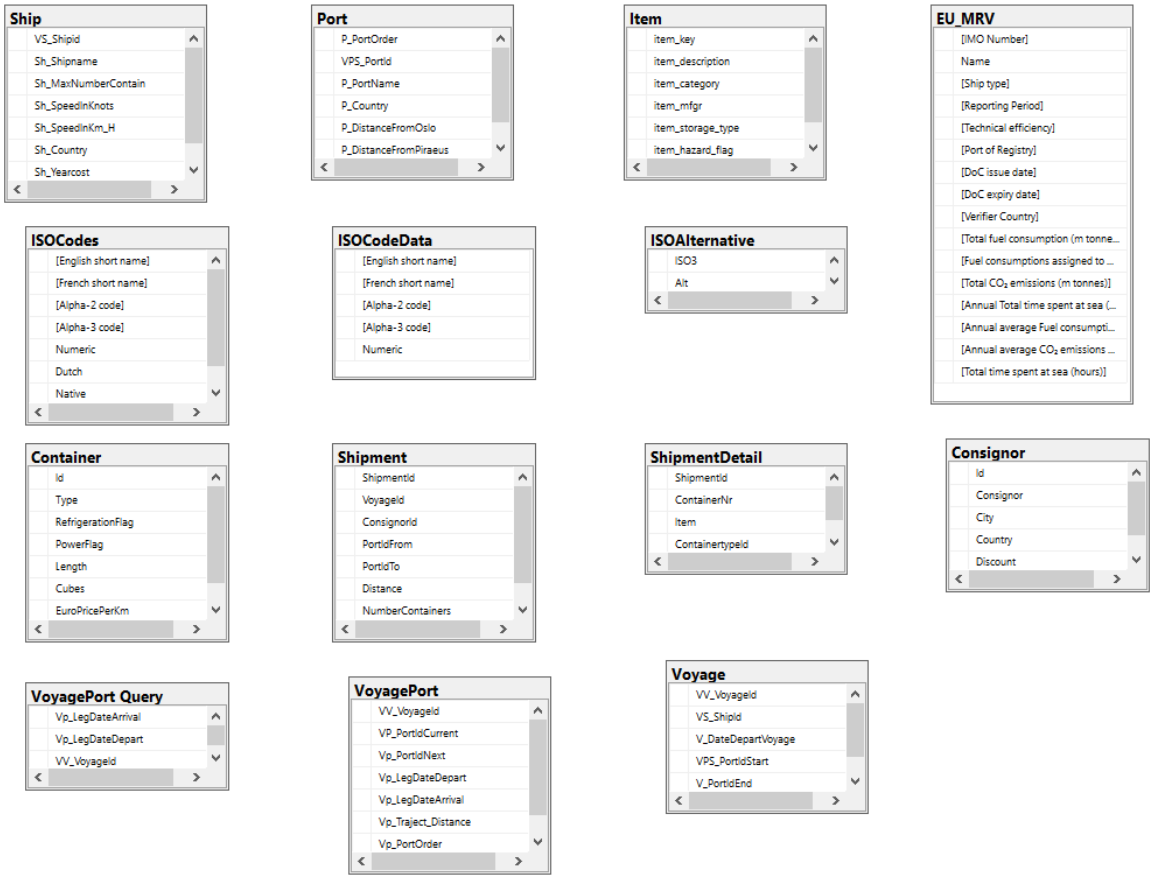


Figuur 2.5.2: Een afbeelding van het “item destination” task panel. De kolommen van de source data worden gekoppeld aan de kolommen van de raw database.

Dit proces wordt ook uitgevoerd voor de andere tables op dezelfde wijze. De kolommen worden direct aan elkaar gelinkt omdat er geen wijzigingen in data plaatsvinden. De uitzondering hiervoor is de MRV publication to RAW. Hier is eerst een lookup naar de scheepsnamen die voorkomen in de Ship tabel.



Figuur 2.5.3: Een afbeelding van het MRV publicatie naar RAW extractie proces. De lookup bevat een referentie naar de scheepsnaam. Als de scheepsnaam gevonden kan worden wordt het bijbehorende record van de EU MRV publicatie overgezet naar de RAW database.

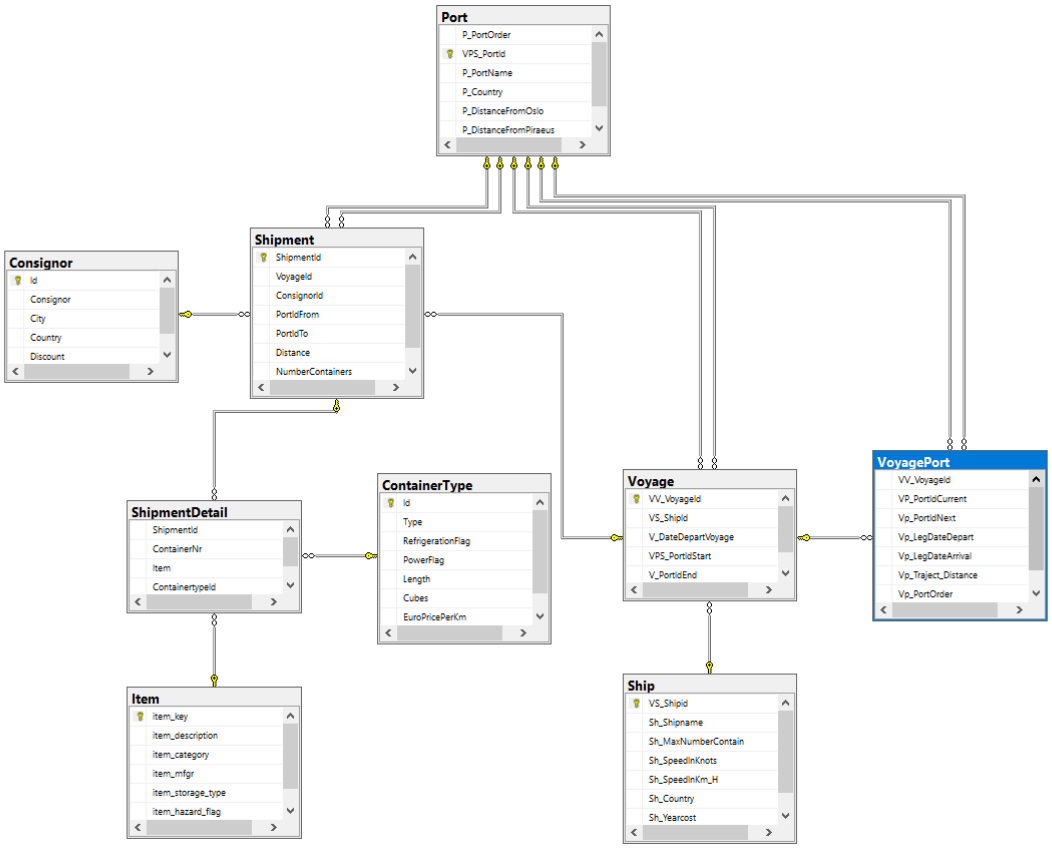


Figuur 2.5.4: Een afbeelding van de RAW database.

Zoals zichtbaar in de bovenstaande afbeelding zijn er nog geen relaties zichtbaar in de RAW database. De data is nu enkel verplaatst naar een centrale bron via een geautomatiseerd proces.

3. Brondata overzetten van Raw naar Relationeel

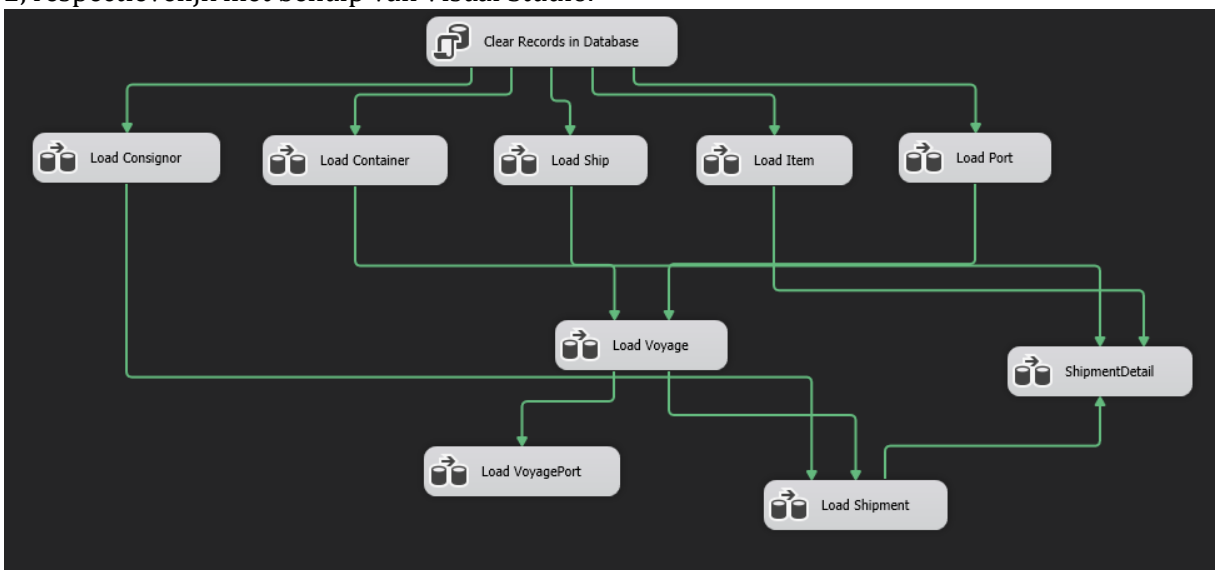
Voor het overzetten van de data van RAW naar relationeel wordt gebruik gemaakt van een geautomatiseerde omgeving binnen Visual Studio.



Figuur 3.0.1: Een afbeelding van het relationele model binnen de SQL database.
Zoals zichtbaar in figuur 3.0.1 zijn alle tabellen met behulp van keys aan elkaar gelinkt. De table VoyagePort heeft referenties met bijna alle tables, direct dan wel indirect. De tables Consignor en Container zijn tevens opgenomen in het diagram. Consignor heeft een relatie naar Shipment omdat Shipment een ConsignorId bevat. Container(type) heeft een relatie naar ShipmentDetail omdat deze een ContainerTypeId bevat. Ik heb het besluit genomen om de EU_MRV niet mee te nemen in deze database omdat ik niet verwacht iets met de data te kunnen omdat deze incompleet is. Meer uitleg hierover volgt later in dit verantwoordingsverslag.

3.1. Data overdracht binnen Visual Studio

De data wordt overgezet van de RAW database naar de relationele database, project 1 en project 2, respectievelijk met behulp van Visual Studio.



Figuur 3.1.1: Een afbeelding van de data flows binnen Visual Studio.

Ten eerste worden de bestaande records verwijderd om consistente data te garanderen ten opzichte van de data in de RAW database. Dit zodat de data bruikbaar is. Dit wordt gedaan doormiddel van het uitvoeren van een query, bovenin te zien als “Execute SQL Task”.

Vervolgens worden de verschillende databronnen ingeladen, elke data flow task[1] bevat een source en een destination om de data over te zetten van de RAW database naar de relationele database.

In de tabel container is een transformatie toegepast om de RefrigerationFlag aan te passen van datatype naar een boolean, waarbij de waarde in plaats van “Yes” of “No”, 0 of 1 is. Hiervoor is gekozen omdat andere programma’s dit kunnen omzetten naar een boolean, wat met Yes of No niet mogelijk is.

```
DELETE FROM VoyagePort;
DELETE FROM Shipment;
DELETE FROM ShipmentDetail;
DELETE FROM Voyage;
DELETE FROM Port;
DELETE FROM Ship;
DELETE FROM ContainerType;
DELETE FROM Consignor;
DELETE FROM Item;
```

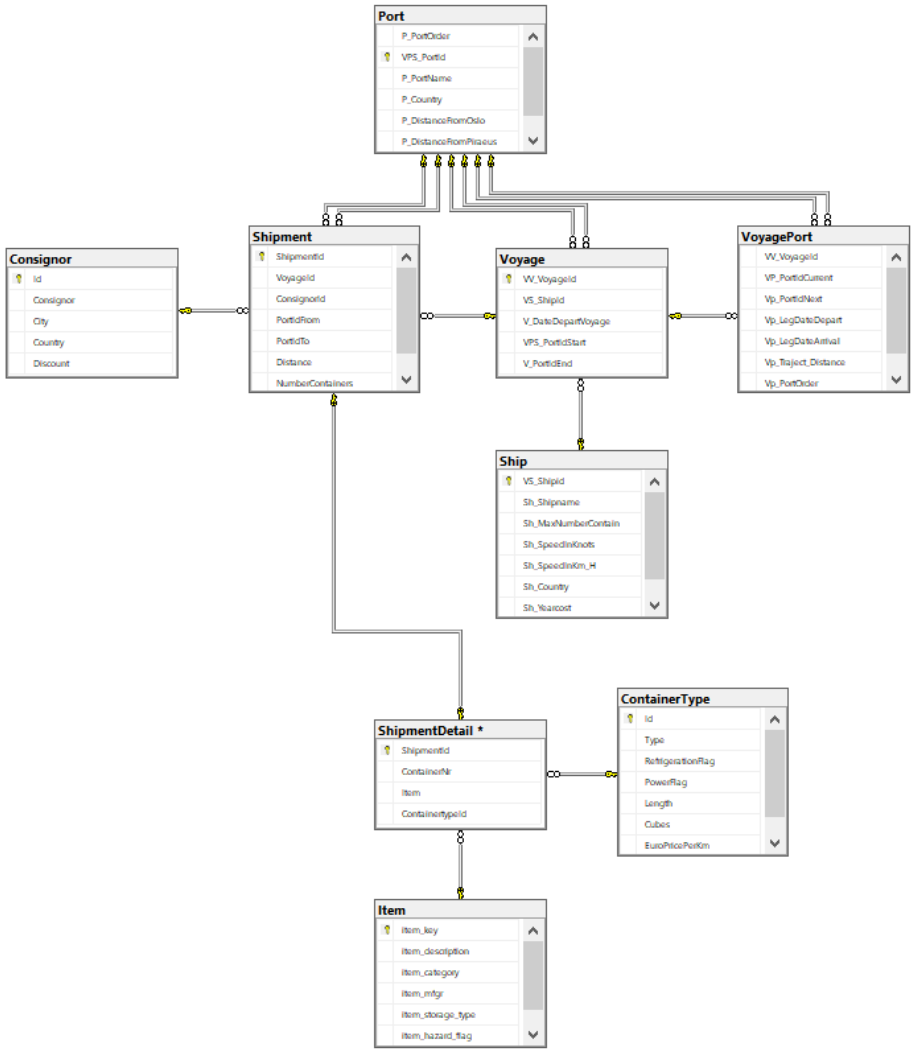
Figuur 3.1.2: Een afbeelding van het SQL statement die boven de data flow tasks staat.

Eerst moeten de tables met foreign key references ingeladen worden. Dit zijn: “Consignor”, “Container”, “Ship”, “Item” en “Port”. Hierna worden “Voyage” en “ShipmentDetail” ingeladen. Hierna worden VoyagePort en Shipment ingeladen. Op deze manier worden key violations en constraints niet geschonden en worden errors voorkomen.

[1] Een data flow task wordt gebruikt om data tussen een bron en een bestemming te transformeren en/of te verplaatsen.

3.2. Informatie over de relationele database

De relationele database bevat een aantal verschillende foreign keys[2].



Figuur 3.2.1: Een afbeelding van het schema van de relationele database.

De relationele database bevat referenties vanuit Shipment naar Consignor, Port, ShipmentDetail en Voyage. In deze stap is het zichtbaar dat voor de PSA wellicht iets gedaan kan worden met Shipment, Voyage, VoyagePort en ShipmentDetail. Shipment bevat namelijk referenties naar het merendeel van de tables binnen de database. ShipmentDetail, VoyagePort en Voyage zorgen voor de andere relaties. Als je deze tabellen samen zou voegen zou je in theorie een ster schema kunnen realiseren omdat VoyageShipmentShipmentDetail, een combinatie van de bovengenoemde tables dan een feiten table zou kunnen worden. Elke table bevat een unique id dat gebruikt wordt als primaire sleutel Als een id ook voorkomt in een andere tabel kan hiernaar gelinkt worden met behulp van een foreign key.

[2] Foreign keys worden gebruikt om referenties te leggen naar andere tabellen binnen dezelfde database.

Voor een groot deel van de kolommen in de verschillende tables zijn al de correcte data typen toegekend. Hiervoor is gekozen omdat de stap van relationeel naar PSA dan kleiner is, als er dan een fout gemaakt wordt is dat minder werk om aan te passen. Als het data type nog veranderd zou moeten worden met behulp van een proces in Visual Studio is er voor gekozen om dit niet in de relationele database te doen maar in de PSA. Enkele kolommen worden echter wel direct omgezet om de stappen op te delen. Achteraf zou het misschien verstandiger zijn geweest om dit in één stap te doen in verband met consistentie en het vermijden van verwarring.

RefrigerationFlag wordt van raw naar relationeel omgezet naar een bit, omdat ik niet verwacht dat er iets nuttigs gedaan kan worden met de data “Yes” of “No”. In het geval dat de database wordt gebruikt bij een zelfgemaakte tool kan makkelijk de refrigerationflag als boolean beschouwd worden wat filteren van data aanzienlijk makkelijker maakt. In plaats van “Yes” of “No” staat er nu dus 0 of 1.
Data uit andere tables wordt wel rechtstreeks ingeladen, eventuele transformaties vinden plaats in de stap van relationeel naar PSA. Hier is ook voor gekozen omdat de relationele database nu voor meerdere oplossingen gebruikt kan worden doordat de data zo min mogelijk aangepast is. De data kan gebruikt worden voor verschillende doeleinden waarbij verschillende tools gebruikt kunnen worden, waar de PSA vooral gemaakt is voor een bepaalde oplossing (het huidige vraagstuk van Kramse) kun je met de relationele database nog alle kanten op.

3.3. Exclusie van de EU MRV publicatie table in de relationele database

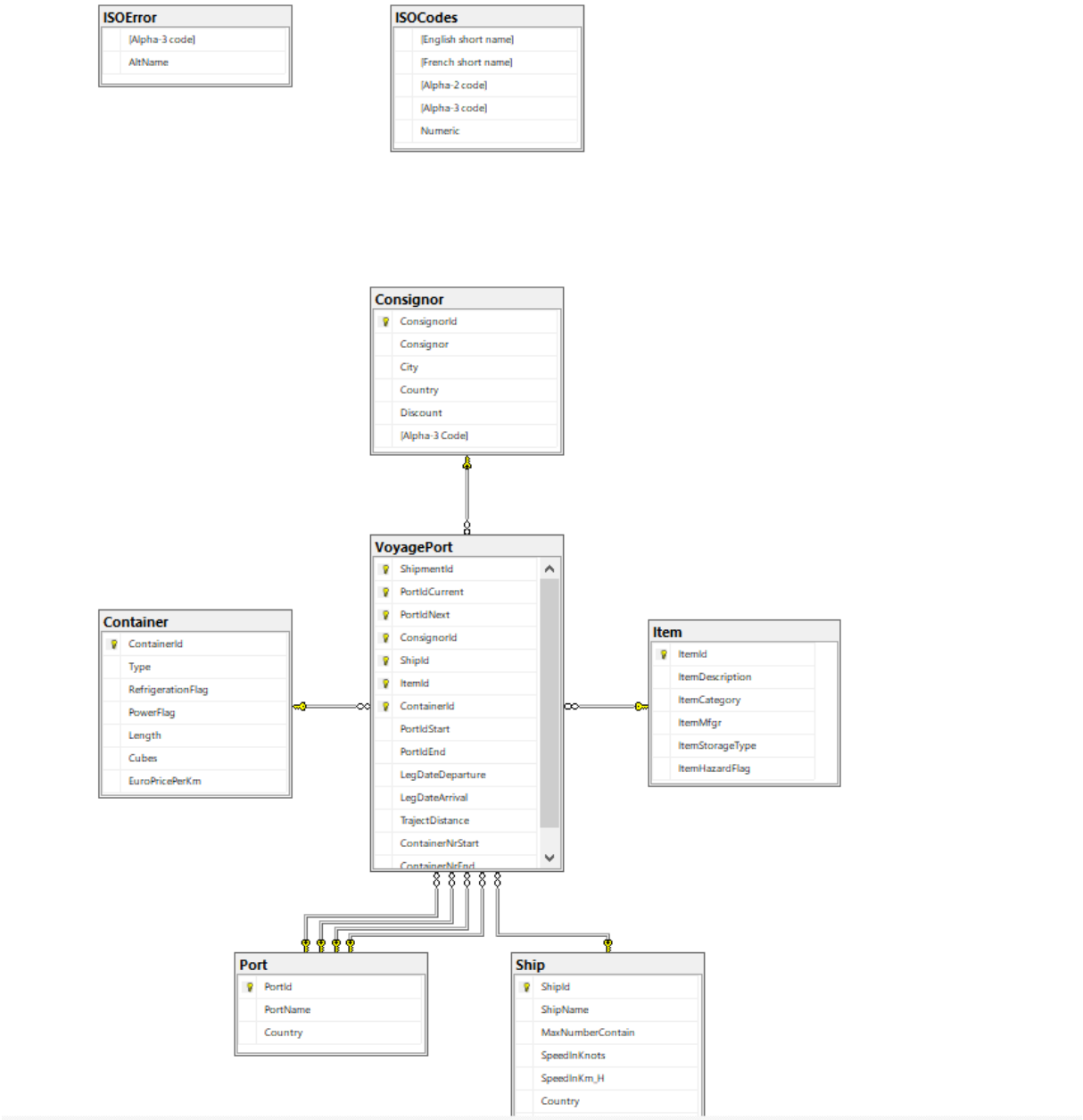
In deze stap is tevens besloten dat de EU MRV publicatie table niet beschikbaar zal zijn in de relationele database. Hiervoor is gekozen omdat de data dusdanig incompleet en onbetrouwbaar is dat het niet verstandig wordt geacht deze mee te nemen in een database die voor doeleinden buiten deze casus kan worden gebruikt. In het geval dat iemand toch deze data wenst te gebruiken zal deze toegevoegd moeten worden. In het geval van de casus wordt deze tabel dus ook niet meegenomen omdat er slechts data is van drie schepen van Kramse. Bij analyses mist er te veel data en zouden er te veel aannames gedaan moeten worden waardoor het als onverantwoord wordt geacht om deze data mee te nemen en te gebruiken.

4. Brondata overzetten van relationeel naar PSA

Het grootste verschil tussen de relationele database en de PSA is dat de PSA structuur voor een sterschema bevat. Dit is gerealiseerd door tabellen samen te voegen, namelijk Voyage, VoyagePort, Shipment en ShipmentDetail. Tevens zijn er een aantal transformaties toegevoegd. Dit is gedaan door het wijzigen en toevoegen van kolommen. Data typen zijn aangepast naar de verwachte benodigdheden binnen deze opdracht. Daarnaast is er rekening gehouden met eventuele extra benodigdheden die in de toekomst naar voren zouden kunnen komen.

4.1. Informatie over de PSA database

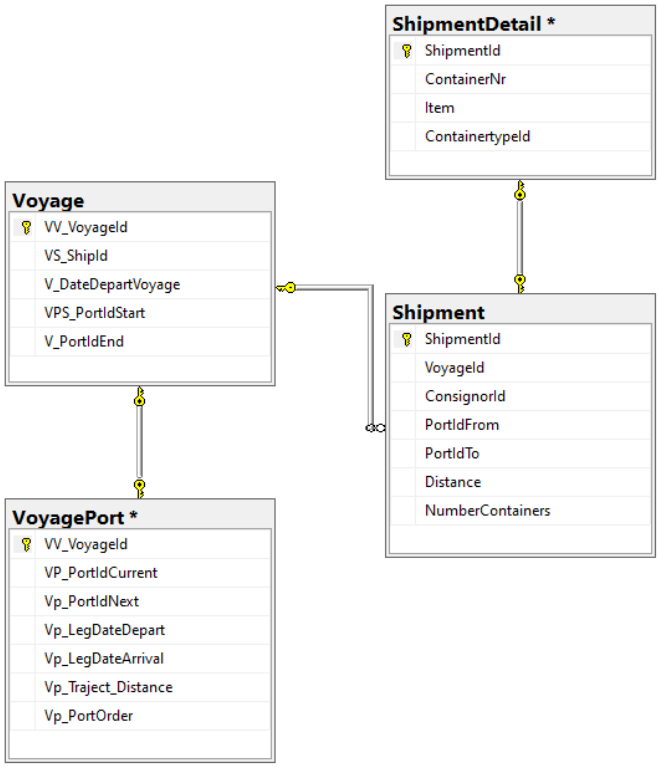
Er zijn een aantal dingen veranderd in deze database ten opzichte van de database beschreven in sectie 3.2.



Figuur 4.1.1: Een weergave van het schema van de PSA database.

4.2. Onderbouwing VoyagePort table

Er zijn veel wijzigingen ten opzichte van de relationele database. De tables Shipment, ShipmentDetail, Voyage en VoyagePort zijn samengevoegd.



Figuur 4.2.1: Een afbeelding van de tables Voyage, VoyagePort, Shipment en ShipmentDetail.

VoyageId bestaat niet meer in VoyagePort. De reden hiervoor is dat de gekozen grain berust op het ShipmentId, VoyageId biedt hiervoor geen toegevoegde waarde omdat de voyage tabel niet meer bestaat en samengevoegd is met Shipment. De tables PortIdFrom en PortIdTo uit Shipment zijn verwijderd, deze kolommen zijn identiek aan PortIdCurrent en PortIdNext. NumberContainers uit Shipment is weggehaald omdat die data redundant is. NumberContainers is namelijk al te bepalen met de data over containers die bekend is. In de table Ship staat het maximum aantal containers dat een schip mee kan dragen. De table ShipmentDetail bevat informatie over het aantal containers die op een schip staan. NumberContainers in Shipment bevat het aantal containers en ContainerNr in ShipmentDetail bevat dit dus ook. ContainerNr bevat echter een opgeslagen reeks met containers, je weet daarbij dus ook welke containers dit zijn, deze informatie kan later nuttig zijn, daarom heb ik ervoor gekozen om deze te gebruiken en niet NumberContainers.

De “Distance” in Shipment is hetzelfde als de “Traject_Distance” in VoyagePort. VoyagePort bevat de afstand gebaseerd op Voyage Id en Shipment bevat de afstand gebaseerd op ShipmentId.

Er is een Unique key toegevoegd, de grain (ook wel het detailniveau genoemd) van de tables ligt op het ShipmentId. Doordat de grain verplaatst is naar een dieper niveau is er meer redundantie. Om dit op te lossen is er een Unique key gemaakt. Deze unique key bestaat uit zeven kolommen. Ik had geanticipeerd dat er zes key velden nodig zouden zijn, maar dat werden er zeven omdat records met zes velden nog niet uniek waren. De unique key bestaat dus uit ShipmentId, PortIdCurrent, PortIdNext, ConsignorId, ShipId, ItemId en ContainerId.

	VoyageId	Distance
1	1	300
2	1	300
3	1	300
4	1	300
5	2	700
6	2	700
7	2	700
8	2	700
9	3	1200
10	3	1200
11	3	1200
12	3	800

	VV_VoyageId	Vp_Traject_Distance
1	1	300
2	2	700
4	3	1200
3	3	800

Figuur 4.2.2: Wanneer de afstanden naast elkaar staan weergegeven is zichtbaar dat de Traject_Distance in VoyagePort gelijk is aan de Distance in Shipment.

De waardes in ShipmentDetail worden allemaal overgezet naar “ShipmentVoyageVoyagePort”. Voor het gemak blijft de naam van deze table echter VoyagePort, omdat deze table de meeste referenties bevat en de andere tables hierin worden samengevoegd.

4.3. Onderbouwing Consignor table

	ConsignorId	Consignor	City	Country	Discount	Alpha-3 Code
1	1	P. Abramson	Stockholm	Sweden	8	SWE
2	2	C. Lassenius	Helsinki	Finland	10	FIN
3	3	D. Damian	Vancouver	Canada	9	CAN
4	4	S. Marzak	Rio de Janiero	Brazilië	7	BRA
5	5	R. Helms	Utrecht	Nederland	0	NLD
6	6	H. Sharpe	Edinborough	Scotland	4	GBR
7	7	D. Freer	Newcastle	NL	3	NLD
8	8	F. Delpaz	Roma	Italia	5	ITA
9	9	K. Lindemeijer	Hamburg	Germany	10	DEU
10	10	M. di Sassi	Barcelona	Spain	1	ESP
11	11	O. Freire	Lisboa	Portugal	3	PRT

Figuur 4.3.1: Een afbeelding van records binnen Consignor.

De Consignor is de afzender. Deze table is veelal hetzelfde gebleven, er is echter wel een kolom toegevoegd. De Alpha-3 Code is toegevoegd om te zorgen dat er een consistente benaming van de benoemde landen beschikbaar is[3].

4.4. Onderbouwing Item table

Er zijn geen wijzigingen in de Item table in de PSA ten opzichte van de Item table in de relationele database.

	ItemId	ItemDescription	ItemCategory	ItemMfgr	ItemStorageType	ItemHazardFlag
1	1	Mountains bikes	Bikes	Gazelle	standard	low
2	2	Netbooks	Computers	Asus	safe	high
3	3	Computer books	Books	AW	standard	low
4	4	Flatscreens	domestic appliance	Philips	protected	medium
5	5	Cigarettes	Smoking goods	Marlboro	safe	medium

Figuur 4.4.1: Een afbeelding van de Item table in de PSA, die identiek is aan de table met dezelfde naam in de relationele database.

[3] Meer uitleg over Consignor volgt in “Informatie over het omzetten van landcodes naar Alpha-3”

4.5. Onderbouwing Container table

Er is een wijziging doorgevoerd in de Container table waarbij de Length en EuroPricePerKm omgezet worden naar decimale getallen. Daarnaast is het Id hernoemt naar ContainerId. Hiervoor is gekozen zodat het consistent is en de table die bij het id hoort extra duidelijk is. Dit maakt processen in Visual Studio duidelijker omdat bij een lookup kolom niet alleen 'Id' staat, maar bijvoorbeeld 'consignorId', zo weet je ook waar de kolom vandaan komt en waar de table aan gekoppeld moet worden.

4.6. Onderbouwing Ship table

De Ship table is niet veranderd. Er is gekozen om alles hetzelfde te laten omdat ik geen reden had gezien om dingen aan te passen of te verwijderen. Ik had eventueel ook de Alpha-3 codes toe kunnen passen om consistent te zijn met de Consignor table.

4.7. Onderbouwing Port table

De port table is veranderd. De DistanceFromOslo en DistanceFromPiraeus worden weggelaten omdat deze waardes gebaseerd zijn op nutteloze gegevens. De DistanceFromOslo en DistanceFromPiraeus is voor een port niet relevant. PortOrder wordt ook niet gebruikt omdat deze waarde redundant is. Er is al data over de volgorde van het afgaan van ports door PortIdCurrent en PortIdNext. PortOrder heeft hierbij geen extra mogelijkheden te bieden.

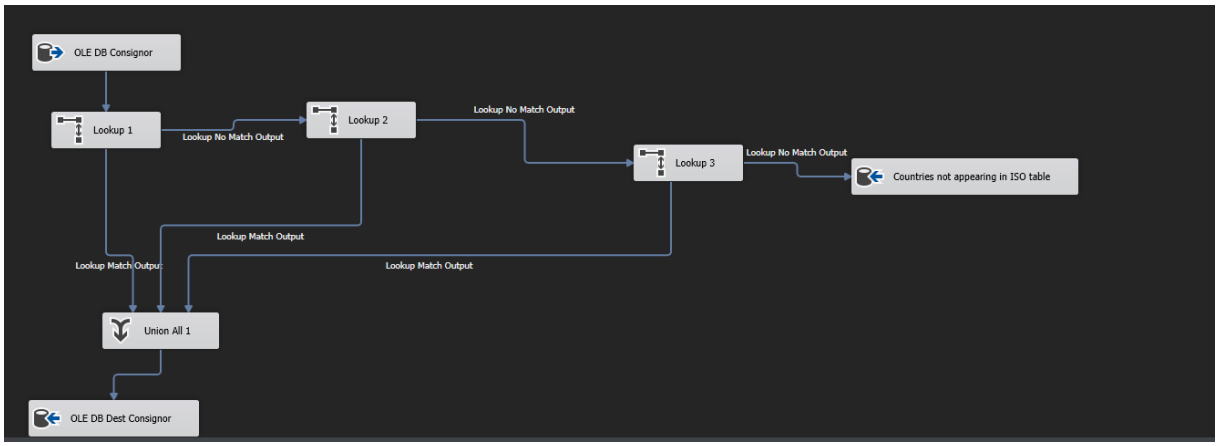
PortId	PortName	Country
1	Rotterdam	Netherlands
2	Amsterdam	Netherlands
3	Alborg	Denmark
4	Helsinki	Finland
5	Le Havre	France

Figuur 4.7.1: Een afbeelding van gegevens in de port table volgens de PSA.

De PortName wordt bewaard omdat er in één land meerdere ports kunnen zijn. Hiernaast wordt verwacht dat dit zinvol inzicht kan bieden in analyses.

4.8. Informatie over het omzetten van landcodes naar Alpha-3

Ten eerste zijn er twee tabellen toegevoegd, dit zijn ISO en ISOError. Deze tabellen of tables worden gebruikt om landcodes (Alpha-3) te koppelen aan landnamen. Deze staan namelijk in verschillende formaten en talen weergegeven in Consignor. Een persoon kan achterhalen dan NLD, NL en Nederland allemaal Nederland is. Voor het inlezen zijn dit echter andere waarden. Hierom wordt er aan elke alias een code van drie letters gekoppeld. Dit is de "Alpha-3". Er is gekozen voor een structuur van 2 tables, één table die functioneert voor de Engelse term en één table waar de bijbehorende ISO in wordt opgezocht. Nadat de Alpha-3 code is opgezocht wordt deze in de Consignor tabel gestopt in de kolom "Alpha-3-Code". Als een alias van een landnaam onbekend is wordt deze toegevoegd aan de ISOError table. Er wordt dan verwacht dat iemand de ISOAlternative table in Project 1 update met de bijbehorende Alpha-3 code van het betreffende land. De ISOAlternative[4] table staat in Project 1 omdat het gezien wordt als bronbestand, daarom heb ik gekozen om het bij de andere bronbestanden te zetten. Als ooit de brondata overgezet moet worden kan dat in één keer en hoeven de handmatig ingevoerde ISO codes niet nogmaals overgezet te worden.

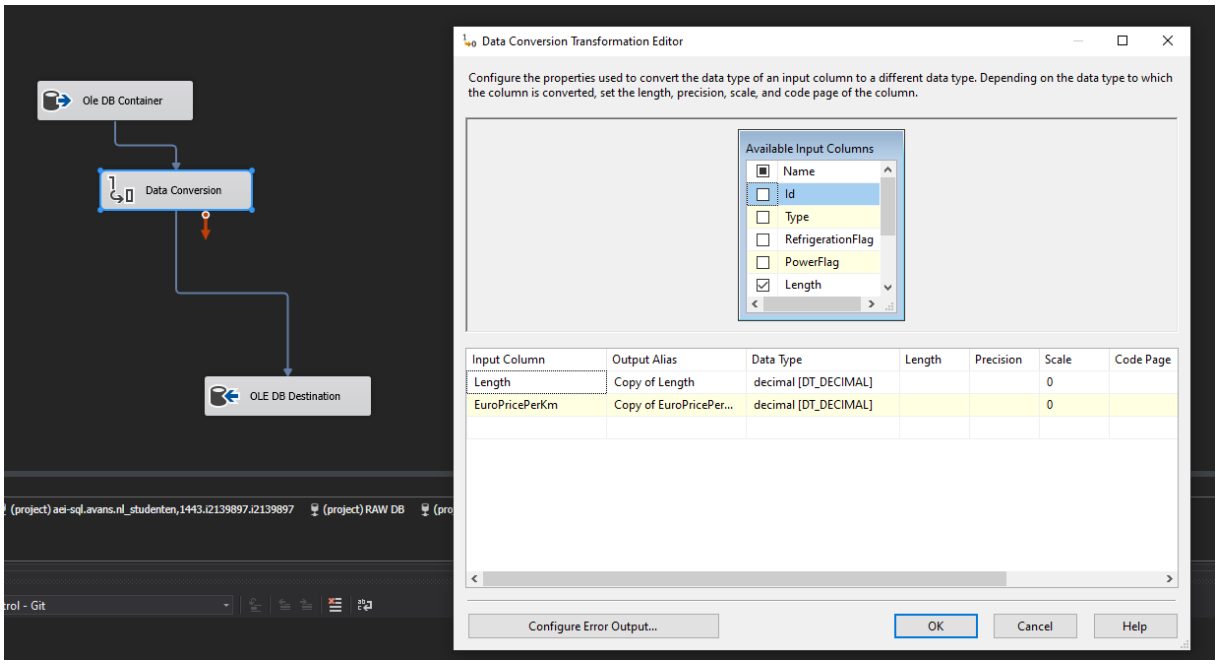


Figuur 4.1.2: Een afbeelding van het lookup proces voor de alpha-3 codes binnen Visual Studio.

[4] Een afbeelding van de ISOAlternative table is toegevoegd in de bijlage.

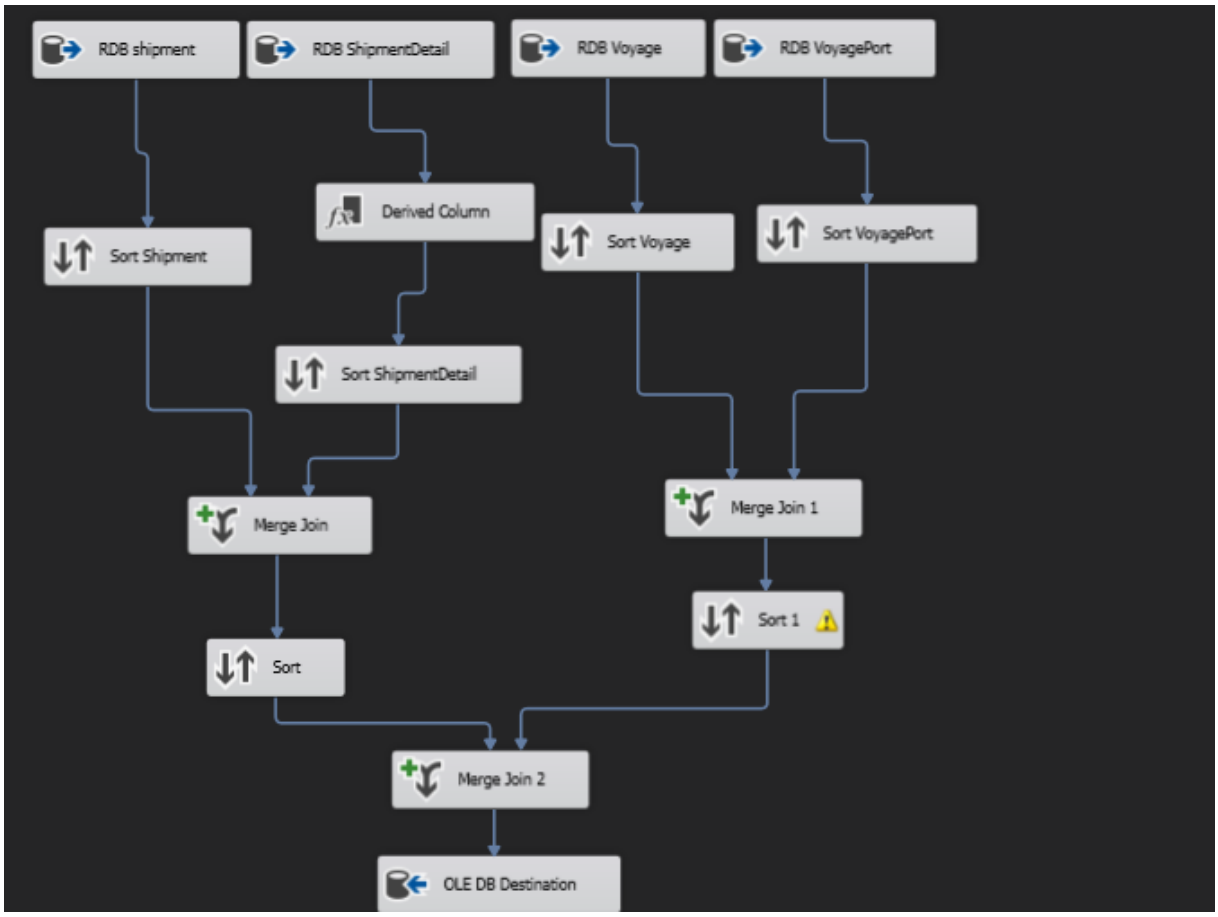
4.9. Informatie over het omzetten van waardes binnen Visual Studio

In container worden de kolommen Length en EuroPricePerKm omgezet van het type string naar type decimal zodat hier eventueel mee gerekend kan worden. Dit wordt gedaan met behulp van een data conversion flow task.



Figuur 4.1.3: Een afbeelding van de data conversion voor de length en EuroPricePerKm.

Naast het converten van de onderdelen benoemd op voorgaande bladzijden is er binnen Visual Studio een proces voor het samenvoegen (mergen) van de tables Shipment, ShipmentDetail, Voyage en VoyagePort.



Figuur 4.1.4: Een afbeelding van de data flow tasks voor het combineren van de bovengenoemde tables.

Naast het samenvoegen van de tables wordt er nog een andere stap uitgevoerd. Het aantal containers wat een shipment bevat wordt gesplitst in twee kolommen met behulp van een “derived column” task.

Derived Column Name	Derived Column	Expression
ContainerNrStart	<add as new column>	(DT_I4)LEFT(ContainerNr,FINDSTRING(ContainerNr,...
ContainerNrEnd	<add as new column>	(DT_I4)RIGHT(ContainerNr,LEN(ContainerNr) - FIN...

Figuur 4.1.5: Een afbeelding van de derived column task voor het splitsen van de container nummers.

Voor ContainerNrStart wordt de volgende expressie gebruikt:
(DT_I4)LEFT(ContainerNr,FINDSTRING(ContainerNr,"-",1) - 1)

Voor ContainerNrEnd wordt de volgende expressie gebruikt:

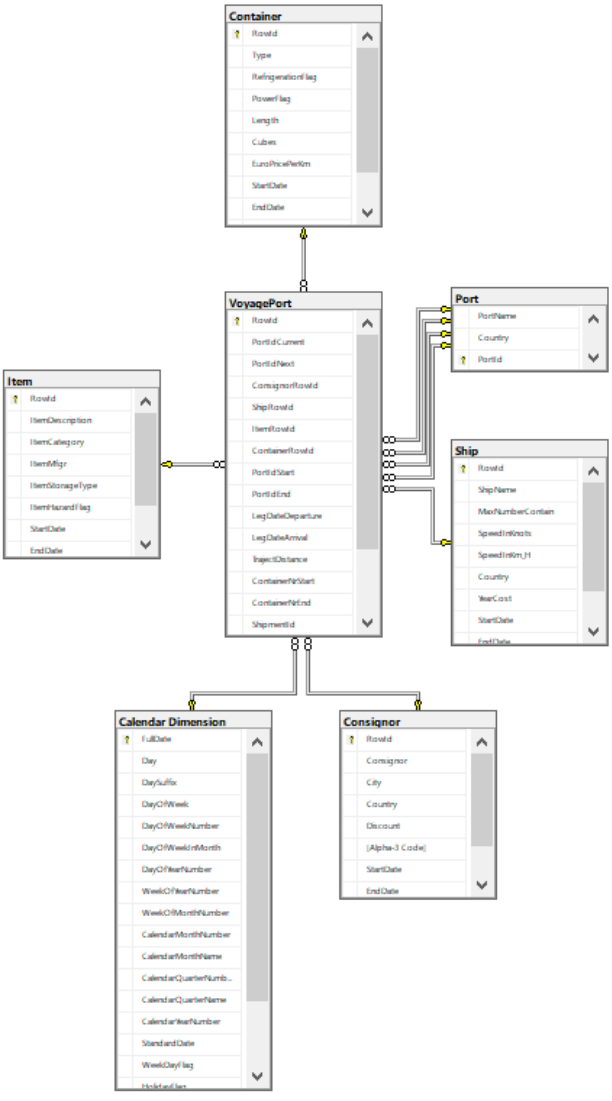
(DT_I4)RIGHT(ContainerNr,LEN(ContainerNr) - FINDSTRING(ContainerNr,"-",1))

5. data overzetten van PSA naar ODS

Voor het overzetten van de data van de PSA naar de ODS wordt gebruik gemaakt van “Slowly Changing Dimensions”. Een SCD wordt gebruikt voor het bewaren en beheren van data. Er zijn verschillende typen, zo is er bijvoorbeeld “fixed”, maar ook “changing” en “historical”. Dit zijn de typen die ondersteund worden door Visual Studio.

5.1. Informatie over de ODS database

De ODS database lijkt op de PSA database, er is echter historie toegevoegd en er is een tijd dimensie.



Figuur 5.1.1: Een afbeelding van de ODS database.

De ODS database bevat ook een tijd dimensie. Deze kan worden gebruikt voor analyses in PowerBI.

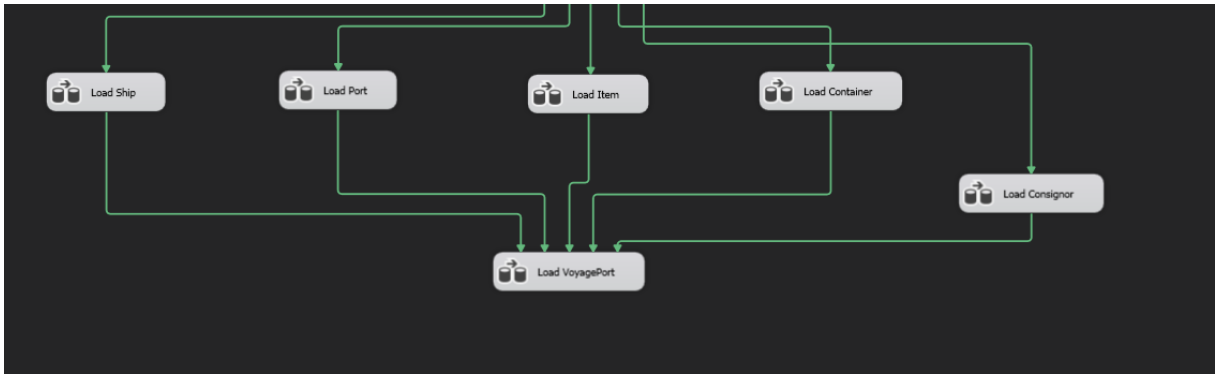
Er zijn RowId's toegevoegd aan de tables Consignor, Container en Item. Dit zijn Slowly Changing Dimensions en bevatten attributen met het SCD type historical. Hiervoor is gekozen omdat elk van deze dimensies veranderd moet kunnen worden.

Bij container moet ooit misschien een flag, type of gewicht aangepast moet kunnen worden. Consignor bevat ook attributen van het SCD type historical omdat een Consignor naar een ander land of stad zou kunnen verplaatsen. Hetzelfde gaat op voor Item en Ship, het kan namelijk zijn dat de categorie van een item aangepast moet kunnen worden, in het geval van ship kan het zijn dat het maximum aantal containers aangepast moet worden in verband met onderhoud, hetzelfde geldt voor de snelheid en het gekoppelde land.

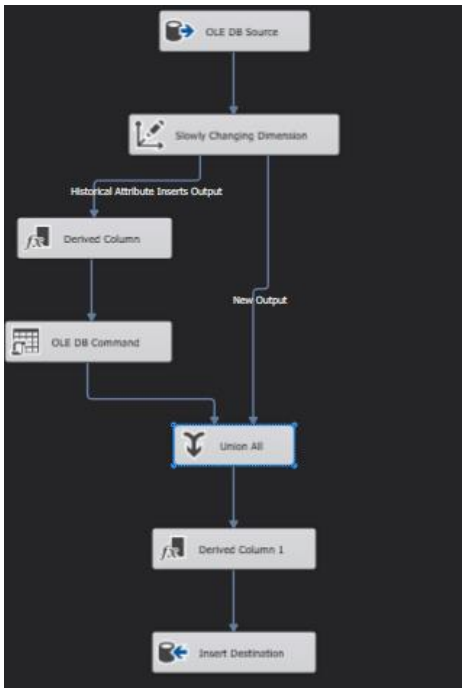
Deze kolommen bevatten een StartDate en een EndDate zodat er meerdere versies van een record bijgehouden kunnen worden. Zo kun je bekijken wat verschillen zijn tussen een actief record en een verlopen record, daarnaast kun je ook een oud record terugzetten als iets niet correct is aangepast.

Er is gekozen om geen EXTRACTION_DATE te utiliseren omdat er geen scenario is waarin historie van dimensies bijgehouden zou moeten worden die aansluit op de gegeven casus.

De feitentabel bevat tevens ook geen historie omdat dit geen dimensie betreft, en deze informatie bevat over dimensies.

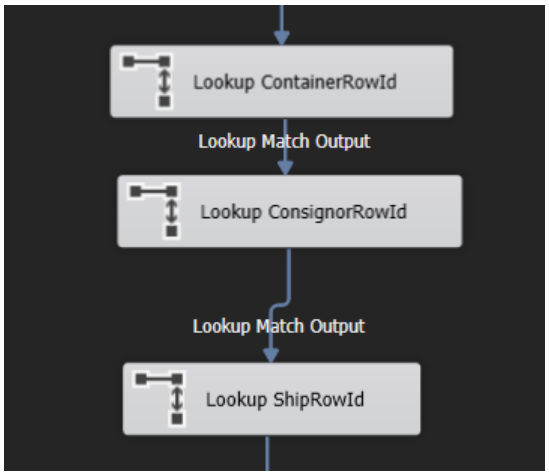


Figuur 5.1.2: Een afbeelding van het Visual Studio proces.



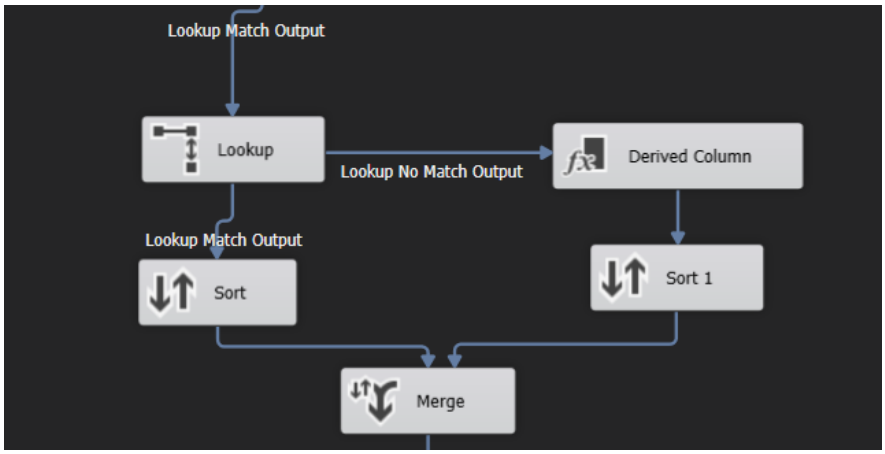
Figuur 5.1.3: Een afbeelding van een Slowly Changing Dimension binnen Visual Studio.

Bovenstaande afbeelding betreft een weergave van de SCD in “Ship”. Er wordt gebruik gemaakt van een “Derived column” om een StartDate in te stellen. De EndDate staat bij het toevoegen op NULL omdat het record dan nog geldig is. Wanneer een record wordt overschreven en er bestaat al een record met hetzelfde id, wordt het oude record ongeldig.



Figuur 5.1.4: Een afbeelding van de VoyagePort data task in Visual Studio.

In de VoyagePort data task worden lookups gedaan naar elk van de dimensies die historische attributen bevatten. Na de lookups wordt er een lookup met een query uitgevoerd.



Figuur 5.1.5: Een afbeelding van een lookup task gevolgd door een aantal tasks.

Deze lookup voert een query uit.

```
SELECT DISTINCT TOP(100) PERCENT VoyagePort.ShipmentId,
VoyagePort.ConsignorId,VoyagePort.ContainerId,VoyagePort.It
emId, VoyagePort.ShipId,VoyagePort.PortIdNext AS PortID,
VoyagePort.LegDateArrival AS Arrival,
VoyagePort.LegDateDeparture AS Depart, DATEDIFF
(day,VoyagePort.LegDateArrival,t2.LegDateDeparture) AS
StayDays FROM VoyagePort inner join (SELECT * FROM
VoyagePort) AS t2 ON VoyagePort.PortIdNext =
t2.PortIdCurrent AND VoyagePort.ShipId = t2.ShipId ORDER BY
Arrival
```

De query zorgt ervoor dat het verschil tussen de departure en de arrival van een schip wordt berekend. Deze wordt vervolgens toegevoegd als de kolom “StayDays”. Dit is het aantal dagen dat een schip in een haven verblijft en is de Idle Time. Wanneer er geen record gevonden wordt in de query wordt als StayDays 0 ingevoerd. Wanneer er wel een resultaat is wordt deze oplopend gesorteerd op alle velden van VoyagePort. De resultaten van Match output en No Match output worden vervolgens samengevoegd. Het PortIdNext wordt gelijkgesteld aan PortIdCurrent omdat deze data niet over 1 record loopt. De arrival is namelijk pas in het volgende record terwijl een schip al vertrekt in de departure.

Na het uitrekenen van deze waarden is er een derived column task. Deze task is verantwoordelijk voor het creëren van de kolom “TotalContainerCount”. Dit is een optelling binnen een VoyagePort record en wordt berekend door ContainerNrStart af te trekken van ContainerNrEnd. Er blijft een reeks containers over. Je weet daardoor per VoyagePort record het aantal containers dat op een schip is.

De weight factor kan hierover berekend worden door het getelde aantal containers te delen door het maximum aantal containers dat op een schip mag staan. Hierna worden de bestaande feiten opgehaald, als er no match output is worden de feiten vervangen door de nieuwe data. Het zoeken naar de bestaande feiten wordt gedaan aan de hand van de unique key.

5.2. Overige informatie

De naamgeving is tevens veranderd in deze stap. Er is gezorgd dat er consistente naamgeving is met identieke casing om de dimensies overzichtelijker te maken.

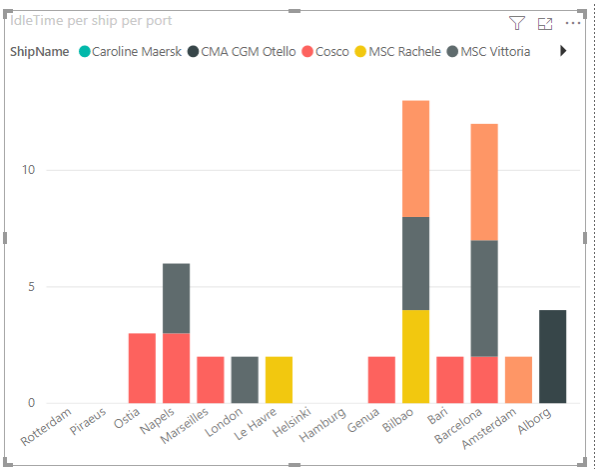
Er is gekozen om geen gebruik te maken van een cube[5]. Measures die in een cube gemaakt worden kunnen bij Power BI niet op de assen worden gezet wat voor mij een groot probleem opleverde. Ik kon hier geen oplossing voor kunnen vinden. De cube bestaat nog als mogelijkheid in de toekomst voor andere doeleinden maar in PowerBI wordt rechtstreeks verbinding gemaakt met de ODS, hierdoor kunnen er direct metingen worden toegevoegd en kunnen deze op de assen worden geplaatst. Het doorvoeren van wijzigingen wordt hierdoor ook makkelijker omdat wijzigingen makkelijk en deels geautomatiseerd geïmporteerd kunnen worden zonder dat het nodig is om handmatig wijzigingen door te voeren van de ODS naar de cube.

[5] Een data cube of gegevenskubus is een reeks multidimensionale waarden.

6. Gegevensanalyse

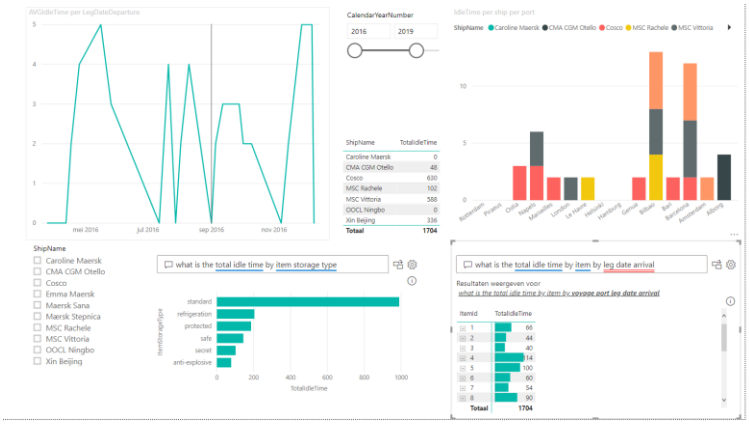
6.1. Informatie over de Idle Time

De Idle Time kan inzichtgeven in de activeiten van schepen met betrekking tot laden en lossen. Doordat mijn Idle Time wordt berekend vanuit VoyagePort kan er overlap zijn omdat het is berekend per ShipmentId record en niet per reis. Dit is dus afhankelijk van de grain. Hierdoor kan er redundante data zijn. Er is in PowerBI een dashhboard gemaakt waarin deze fout wordt opgelost door een average te nemen. Dit geeft een accurate representatie van het aantal IdleDays. Volgende keer zou ik er waarschijnlijk voor kiezen een galaxy schema te maken omdat de grain dan ligt op een Voyage en niet op een ShipmentId record. Dit zou het bepalen van de IdleDays makkelijker maken.



Figuur 6.1.1: Een afbeelding van een Power BI grafiek die de Idle Time toont. De totale idle time komt neer op 50 dagen.

Ter aanvullende informatie is er een dashboard opgesteld voor het weergeven van de idle time of stay days. Dit dashboard bevat bijvoorbeeld informatie over hoelang containers verblijven in een bepaalde haven en wat de idle times zijn over een bepaalde tijd.

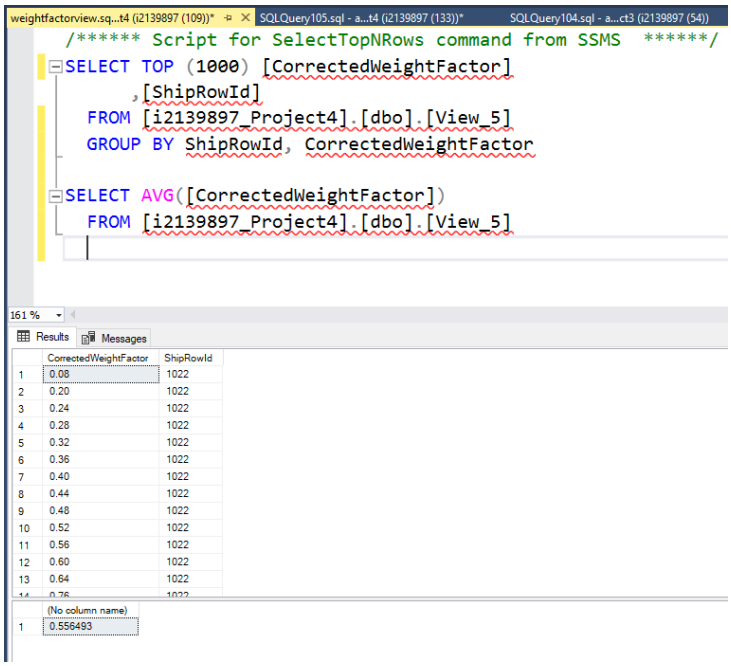


Figuur 6.1.2: Een afbeelding van een Power BI dashboard met informatie over de idle times.

6.2. Informatie over de beladingsgraad

De beladingsgraad is uitgerekend door het aantal containers te delen door het maximum aantal containers. Dit geeft een “Weight Factor”. Wanneer deze met 100 vermenigvuldigd wordt resulteert dit in een percentage, de gemiddelde load factor.

De load factor wordt in Power BI weergegeven met behulp van een view.



```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP (1000) [CorrectedWeightFactor]
      ,[ShipRowId]
  FROM [i2139897_Project4].[dbo].[View_5]
  GROUP BY ShipRowId, CorrectedWeightFactor

SELECT AVG([CorrectedWeightFactor])
  FROM [i2139897_Project4].[dbo].[View_5]
```

Results | Messages

	CorrectedWeightFactor	ShipRowId
1	0.08	1022
2	0.20	1022
3	0.24	1022
4	0.28	1022
5	0.32	1022
6	0.36	1022
7	0.40	1022
8	0.44	1022
9	0.48	1022
10	0.52	1022
11	0.56	1022
12	0.60	1022
13	0.64	1022

	(No column name)
1	0.556493

Figuur 6.2.1: Een afbeelding van een selectie toegepast op een view. Dit resulteert in een data set met gecorrigeerde load factors en een procentuele gemiddelde load factor van 56%. Het is mogelijk dat de load factor niet helemaal accuraat is omdat er een overlap zit in containers. Naast een mogelijke overlap doordat één container meerdere item id’s en container id’s bevat. Dit is echter heel moeilijk om op te lossen. De gecorrigeerde weight factor bevat een zo veilig mogelijke waarde waaraan zo min mogelijk veranderd is om tot een realistisch resultaat te komen. Er zijn ook schepen die bijna geen lading hebben, hierdoor wordt de gemiddelde load factor aanzienlijk lager. De CorrectedWeightFactor is gebaseerd op verschillende records waarbij de grain ligt op het ShipmentId.

	RowId	PortIdCurrent	PortIdNext	ConsignorRowId	ShipRowId	ItemRowId	ContainerRowId	PortIdStart	PortIdEnd	LegDateDeparture	LegDateArrival	TrajectDistance	ContainerNrStart	ContainerNrEnd	ShipmentId	StayDays	TotalContainerCount
1	1	1	21	1048	1031	1002	1022	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	2001	5000	1	0	3000.00
2	2	1	21	1048	1031	1004	1022	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	1	2000	1	0	2000.00
3	3	1	21	1047	1031	1005	1019	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	1	1000	2	0	1000.00
4	4	1	21	1049	1031	1006	1023	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	1001	2000	3	0	1000.00
5	5	1	21	1049	1031	1010	1024	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	1	1000	3	0	1000.00
6	6	1	21	1050	1031	1005	1019	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	1	600	4	0	600.00
7	7	1	21	1050	1031	1012	1022	1	21	2016-03-25 00:00:00.000	2016-03-26 00:00:00.000	300	601	1200	4	0	600.00
8	8	7	2	1049	1028	1003	1022	7	2	2016-04-04 00:00:00.000	2016-04-06 00:00:00.000	700	2001	3000	5	0	1000.00
9	9	7	2	1049	1028	1007	1018	7	2	2016-04-04 00:00:00.000	2016-04-06 00:00:00.000	700	1	1000	5	0	1000.00
10	10	7	2	1049	1028	1009	1020	7	2	2016-04-04 00:00:00.000	2016-04-06 00:00:00.000	700	1001	2000	5	0	1000.00
11	11	7	2	1047	1028	1005	1022	7	2	2016-04-04 00:00:00.000	2016-04-06 00:00:00.000	700	1501	2000	6	0	500.00
12	12	7	2	1047	1028	1008	1018	7	2	2016-04-04 00:00:00.000	2016-04-06 00:00:00.000	700	1	1500	6	0	1500.00

Figuur 6.2.2: Een afbeelding van een aantal records met een bijbehorende TotalContainerCount.

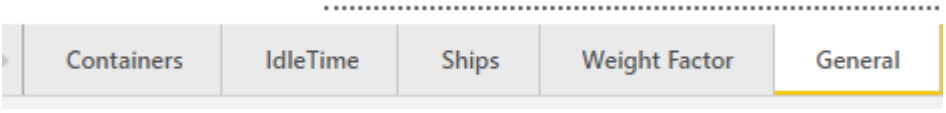
7. Conclusie

In dit hoofdstuk wordt teruggekeken op het project en wordt een advies gedaan aan Kramse.

Hoewel dit project ingeleverd wordt op 25 maart 2020 zal een BI Project nooit klaar zijn. Daarom is er rekening gehouden met mogelijke uitbreidingen in de toekomst, hierdoor kan er verder gewerkt worden met verschillende versies van databases geschikt voor een weide hoeveelheid programma's. Er is een pipeline aanwezig voor het geautomatiseerd doorvoeren van gegevens en er is met behulp van Power BI informatie beschikbaar die makkelijk gelezen kan worden.

7.1. Dashboards

Er zijn voor Kramse een tal dashboards gemaakt die inzicht dienen te geven in data, die gebruikt kunnen worden voor het sturen van het bedrijf. Er zijn meerdere dashboards gemaakt voor het kijken op verschillende niveau's.`



Figuur 7.1.1: Een afbeelding van de verschillende Dashboards.

Figuur 7.1.2: Een afbeelding van de verschillende Dashboards.

7.2. Beladingsgraad

Kramse heeft inzicht in de beladingsgraad via de Dashboards die in de vorige paragraaf staan weergegeven.

7.3. Idle Time

Kramse heeft inzicht in de Idle Time. Het is mogelijk de utilisatiegraad in te zien met het gestelde doel, de utilisatiegraad kan echter maar voor 1 jaar bekeken worden omdat er alleen data is van 2016. Het is dus niet mogelijk om de Idle Time voor de afgelopen jaren weer te geven.

- "Wachttijden ('idle time') per haven en per schip inclusief trends (afgelopen jaren)."

7.4. Brandstofverbruik

Het is niet mogelijk deze vraag te beantwoorden omdat er te veel variabelen zijn die niet inzichtelijk zijn. Hoewel er een aanname gedaan kan worden over het gewicht van een container. Het is niet mogelijk te zien wat het gewicht is van vracht, er zou eventueel een aanname mogelijk zijn over het gewicht van een lege container, het probleem hierbij is echter dat een container verschillende beladingen kan hebben, daarnaast zal een schip verder in het water zakken bij een zwaardere beladingsgraad. Hierdoor ontstaat wrijving en zal het schip langzamer varen. Daarnaast is niet zichtbaar wat de totale lengte van de schepen is en kan er geen aanname worden gedaan over het type containerschip, het is dus niet bekend hoe hard het containerschip kan varen. Het brandstofverbruik kan berekend worden wanneer bekend is hoeveel liter brandstof het betreffende schip per uur gebruikt. Het is namelijk bekend wat de afstand is die een schip vaart en in welke tijd het schip deze afstand vaart. Er zou dan echter informatie moeten zijn over de verschillende boten, deze informatie kon niet gehaald worden uit EU_MRV omdat er slechts data beschikbaar was van drie schepen. Doordat deze data niet compleet is achtte ik het onverantwoord om met deze data te rekenen omdat het zou kunnen leiden tot grote afwijkingen van de realiteit en dus zou kunnen leiden tot verkeerde business keuzes.

7.5. Kosten per ton vracht per route

Deze vraag is niet te beantwoorden om de volgende redenen:

- Het gewicht van containers is niet bekend, hoewel het type container bekend is kan er volgens de data lading inzitten van verschillende afzenders.
- Het is niet bekend wat in een specifieke container zit omdat laden en lossen gebeurt in grote hoeveelheden.
- Hierdoor is het nooit bekend wat er exact is veranderd na het laden of lossen van een schip.
- Er kunnen aanvullende kosten zijn omdat er verschillende typen containers zijn. Het vervoeren van een container met wapens of een gekoelde container zou aanzienlijk duurder kunnen zijn dan het vervoeren van een 'standaard' container.

Om bovengenoemde redenen zijn de kosten per ton vracht per route niet uitgerekend omdat deze naar verwachting te groot zullen afwijken van de realiteit. Doordat er doorgerekend moet worden met de beladingsgraad ontstaan er ook problemen omdat het niet accuraat is door te rekenen met niet exacte waarden. Dit zou kunnen leiden tot zeer grote afwijkingen wat grote gevolgen zou kunnen hebben voor de besluitvorming van Kramse door incorrecte voorspellingen.

Het advies aan Kramse is om schepen zoveel mogelijk in te zetten en beschikbaar te stellen in Italië en Spanje. Schepen zijn hier relatief volgeladen en er worden veel containers vervoerd.

8. Bijlagen

8.1. Een weergave van de ISOAlternative table

	ISO3	Alt
1	Bel	België
2	BEL	BE
3	BEL	Belgique
4	BEL	B
5	BRA	Brazilië
6	CHN	China
7	DEU	D
8	DEU	DEUTSCHLAND
9	FRA	Francia
10	FRA	FR
11	FRA	Frankrijk
12	GBR	Great Britain
13	GBR	UK
14	GBR	GB
15	GBR	England
16	GRC	Griekenland
17	HRV	Yougoslavia

Figuur 6.1.1: Een afbeelding van een aantal records in de ISOAlternative table. Deze table staat in Project 1 bij de rest van de data sources. Hiervoor is gekozen omdat alle brondata dan op een consistente plek staat en deze data ook in andere projecten gebruikt kan worden.

8.2. Een weergave van de ISOCodes table

	English short name	French short name	Alpha-2 code	Alpha-3 code	Numeric
1	Afghanistan	Afghanistan (l')	AF	AFG	4
2	Albania	Albanie (l')	AL	ALB	8
3	Algeria	Algérie (l')	DZ	DZA	12
4	American Samoa	Samoa américaines (les)	AS	ASM	16
5	Andorra	Andorre (l')	AD	AND	20
6	Angola	Angola (l')	AO	AGO	24
7	Anguilla	Anguilla	AI	AIA	660
8	Antarctica	Antarctique (l')	AQ	ATA	10
9	Antigua and Barbuda	Antigua-et-Barbuda	AG	ATG	28
10	Argentina	Argentine (l')	AR	ARG	32
11	Armenia	Arménie (l')	AM	ARM	51
12	Aruba	Aruba	AW	ABW	533
13	Australia	Australie (l')	AU	AUS	36
14	Austria	Autriche (l')	AT	AUT	40
15	Azerbaijan	Azerbaïdjan (l')	AZ	AZE	31
16	Bahamas (the)	Bahamas (les)	BS	BHS	44
17	Bahrain	Bahreïn	BH	BHR	48
18	Bangladesh	Bangladesh (le)	BD	BGD	50
19	Barbados	Barbade (la)	BB	BRB	52
20	Belarus	Bélarus (le)	BY	BLR	112
21	Belgium	Belgique (la)	BE	BEL	56
22	Belize	Belize (le)	BZ	BLZ	84
23	Benin	Bénin (le)	BJ	BEN	204
24	Bermuda	Bermudes (les)	BM	BMU	60
25	Bhutan	Bhoutan (le)	BT	BTN	64
26	Bolivia (Plurinationa...	Bolivie (État plurinatio...	BO	BOL	68
27	Bonaire, Sint Eustat...	Bonaire, Saint-Eustac...	BQ	BES	535
28	Bosnia and Herzeg...	Bosnie-Herzégovine (la)	BA	BIH	70
29	Botswana	Botswana (le)	BW	BWA	72

Figuur 6.2.1: Een afbeelding van een aantal records in de ISOCodes table. Deze table staat in de PSA database en wordt gebruikt voor lookups om te achterhalen wat een ISO code is bij landnamen. Deze worden vervolgens toegevoegd aan de Consignor table. Als een bijbehorende Alpha-3 code niet gevonden kan worden wordt er een nieuw record gemaakt in de ISOError table.

8.3. Een weergave van de ISOError table

	Alpha-3 code	AltName
	NULL	Deutschland
	NULL	Belgie
	NULL	Netherlands
	NULL	Deutschland
	NULL	Deutschland
	NULL	United Kingdom
	NULL	Deutschland
	NULL	Belgie
	NULL	Netherlands
0	NULL	Deutschland
1	NULL	Deutschland
2	NULL	United Kingdom
3	NULL	Deutschland
4	NULL	Belgie
5	NULL	Netherlands
6	NULL	Deutschland
7	NULL	Deutschland
8	NULL	United Kingdom
9	NULL	Deutschland
0	NULL	Belgie
1	NULL	Netherlands

Figuur 6.3.1: Een afbeelding met een aantal records van de ISOError table. Wanneer een alternatieve naam nog niet bekend is (er wordt een naam gevonden die geen bijbehorende Alpha-3 code heeft) wordt een nieuw record aangemaakt in de ISOError table. Het is de bedoeling dat iemand deze records elke X hoeveelheid tijd doorneemt en de alternatieve naam met de bijbehorende Alpha-3 code noteert en toevoegt in de ISOAlternative table.

8.4. Relationele database SQL script

```
USE [i2139897_Project2]
GO
/****** Object:  Table [dbo].[Consignor]    Script Date: 21-3-2020 19:49:23 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Consignor](
	[Id] [int] NOT NULL,
	[Consignor] [nvarchar](255) NULL,
	[City] [nvarchar](255) NULL,
	[Country] [nvarchar](255) NULL,
	[Discount] [int] NULL,
 CONSTRAINT [PK_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) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ContainerType]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ContainerType](
	[Id] [int] NOT NULL,
	[Type] [varchar](50) NULL,
	[RefrigerationFlag] [bit] NULL,
	[PowerFlag] [varchar](50) NULL,
	[Length] [varchar](50) NULL,
	[Cubes] [varchar](50) NULL,
	[EuroPricePerKm] [varchar](50) NULL,
 CONSTRAINT [PK_Container_Type] PRIMARY KEY CLUSTERED
(
	[Id] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Item]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[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_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) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Port]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Port](
	[P_PortOrder] [nvarchar](50) NULL,
	[VPS_PortId] [int] NOT NULL,
```

```
	[P_PortName] [nvarchar](50) NULL,
	[P_Country] [nvarchar](50) NULL,
	[P_DistanceFromOslo] [int] NULL,
	[P_DistanceFromPiraeus] [int] NULL,
 CONSTRAINT [PK_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) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Ship]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Ship](
	[VS_Shipid] [int] NOT NULL,
	[Sh_Shipname] [nvarchar](50) NULL,
	[Sh_MaxNumberContain] [int] NULL,
	[Sh_SpeedInKnots] [int] NULL,
	[Sh_SpeedInKm_H] [int] NULL,
	[Sh_Country] [nvarchar](50) NULL,
	[Sh_Yearcost] [money] NULL,
 CONSTRAINT [PK_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) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Shipment]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Shipment](
	[ShipmentId] [int] NOT NULL,
	[VoyageId] [int] NULL,
	[ConsignorId] [int] NULL,
	[PortIdFrom] [int] NULL,
	[PortIdTo] [int] NULL,
	[Distance] [int] NULL,
	[NumberContainers] [int] NULL,
 CONSTRAINT [PK_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) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ShipmentDetail]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ShipmentDetail](
	[ShipmentId] [int] NOT NULL,
	[ContainerNr] [nvarchar](50) NULL,
	[Item] [int] NULL,
	[ContainertypeId] [int] NULL
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Voyage]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Voyage](
```

```
	[VV_VoyageId] [int] NOT NULL,
	[VS_ShipId] [int] NULL,
	[V_DateDepartVoyage] [datetime] NULL,
	[VPS_PortIdStart] [int] NULL,
	[V_PortIdEnd] [int] NULL,
 CONSTRAINT [PK_Voyage] PRIMARY KEY CLUSTERED
(
	[VV_VoyageId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[VoyagePort]    Script Date: 21-3-2020 19:49:24 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[VoyagePort](
	[VV_VoyageId] [int] NOT NULL,
	[VP_PortIdCurrent] [int] NOT NULL,
	[Vp_PortIdNext] [int] NOT NULL,
	[Vp_LegDateDepart] [datetime] NULL,
	[Vp_LegDateArrival] [datetime] NULL,
	[Vp_Traject_Distance] [int] NULL,
	[Vp_PortOrder] [nvarchar](50) NULL
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Shipment]  WITH CHECK ADD  CONSTRAINT [FK_Shipment_Consignor] FOREIGN KEY([ConsignorId])
REFERENCES [dbo].[Consignor] ([Id])
GO
ALTER TABLE [dbo].[Shipment] CHECK CONSTRAINT [FK_Shipment_Consignor]
GO
ALTER TABLE [dbo].[Shipment]  WITH CHECK ADD  CONSTRAINT [FK_Shipment_Port] FOREIGN KEY([PortIdTo])
REFERENCES [dbo].[Port] ([VPS_PortId])
GO
ALTER TABLE [dbo].[Shipment] CHECK CONSTRAINT [FK_Shipment_Port]
GO
ALTER TABLE [dbo].[Shipment]  WITH CHECK ADD  CONSTRAINT [FK_Shipment_Port1] FOREIGN KEY([PortIdFrom])
REFERENCES [dbo].[Port] ([VPS_PortId])
GO
ALTER TABLE [dbo].[Shipment] CHECK CONSTRAINT [FK_Shipment_Port1]
GO
ALTER TABLE [dbo].[Shipment]  WITH CHECK ADD  CONSTRAINT [FK_Shipment_Voyage] FOREIGN KEY([VoyageId])
REFERENCES [dbo].[Voyage] ([VV_VoyageId])
GO
ALTER TABLE [dbo].[Shipment] CHECK CONSTRAINT [FK_Shipment_Voyage]
GO
ALTER TABLE [dbo].[ShipmentDetail]  WITH CHECK ADD  CONSTRAINT [FK_ShipmentDetail_Container_Type] FOREIGN
KEY([ContainertypeId])
REFERENCES [dbo].[ContainerType] ([Id])
GO
ALTER TABLE [dbo].[ShipmentDetail] CHECK CONSTRAINT [FK_ShipmentDetail_Container_Type]
GO
ALTER TABLE [dbo].[ShipmentDetail]  WITH CHECK ADD  CONSTRAINT [FK_ShipmentDetail_Item] FOREIGN KEY([Item])
REFERENCES [dbo].[Item] ([item_key])
GO
ALTER TABLE [dbo].[ShipmentDetail] CHECK CONSTRAINT [FK_ShipmentDetail_Item]
GO
ALTER TABLE [dbo].[ShipmentDetail]  WITH CHECK ADD  CONSTRAINT [FK_ShipmentDetail_Shipment] FOREIGN
KEY([ShipmentId])
REFERENCES [dbo].[Shipment] ([ShipmentId])
GO
ALTER TABLE [dbo].[ShipmentDetail] CHECK CONSTRAINT [FK_ShipmentDetail_Shipment]
GO
ALTER TABLE [dbo].[Voyage]  WITH CHECK ADD  CONSTRAINT [FK_Voyage_Port] FOREIGN KEY([VPS_PortIdStart])
REFERENCES [dbo].[Port] ([VPS_PortId])
GO
ALTER TABLE [dbo].[Voyage] CHECK CONSTRAINT [FK_Voyage_Port]
GO
```

```
ALTER TABLE [dbo].[Voyage]  WITH CHECK ADD  CONSTRAINT [FK_Voyage_Port1] FOREIGN KEY([V_PortIdEnd])
REFERENCES [dbo].[Port] ([VPS_PortId])
GO
ALTER TABLE [dbo].[Voyage] CHECK CONSTRAINT [FK_Voyage_Port1]
GO
ALTER TABLE [dbo].[Voyage]  WITH CHECK ADD  CONSTRAINT [FK_Voyage_Ship] FOREIGN KEY([VS_ShipId])
REFERENCES [dbo].[Ship] ([VS_Shipid])
GO
ALTER TABLE [dbo].[Voyage] CHECK CONSTRAINT [FK_Voyage_Ship]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Port] FOREIGN KEY([VP_PortIdCurrent])
REFERENCES [dbo].[Port] ([VPS_PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Port]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Port1] FOREIGN KEY([Vp_PortIdNext])
REFERENCES [dbo].[Port] ([VPS_PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Port1]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Voyage] FOREIGN KEY([VV_VoyageId])
REFERENCES [dbo].[Voyage] ([VV_VoyageId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Voyage]
GO
```

8.5. PSA SQL script

```
USE [i2139897_Project3]
GO
/****** Object:  Table [dbo].[Consignor]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Consignor](
	[ConsignorId] [int] NOT NULL,
	[Consignor] [nvarchar](255) NULL,
	[City] [nvarchar](255) NULL,
	[Country] [nvarchar](255) NULL,
	[Discount] [int] NULL,
	[Alpha-3 Code] [nvarchar](255) NULL,
PRIMARY KEY CLUSTERED
(
	[ConsignorId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Container]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Container](
	[ContainerId] [int] NOT NULL,
	[Type] [varchar](50) NULL,
	[RefrigerationFlag] [bit] NULL,
	[PowerFlag] [varchar](50) NULL,
	[Length] [decimal](18, 2) NULL,
	[Cubes] [int] NULL,
	[EuroPricePerKm] [decimal](18, 2) NULL,
 CONSTRAINT [PK__Containe__037960BB6BD9212D] PRIMARY KEY CLUSTERED
(
	[ContainerId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ISOCodes]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ISOCodes](
	[English short name] [nvarchar](255) NULL,
	[French short name] [nvarchar](255) NULL,
	[Alpha-2 code] [nvarchar](255) NULL,
	[Alpha-3 code] [nvarchar](255) NULL,
	[Numeric] [float] NULL
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ISOError]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ISOError](
	[Alpha-3 code] [nvarchar](255) NULL,
	[AltName] [nvarchar](255) NULL
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Item]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
```

```
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Item](
	[ItemId] [int] NOT NULL,
	[ItemDescription] [nvarchar](50) NULL,
	[ItemCategory] [nvarchar](50) NULL,
	[ItemMfgr] [nvarchar](50) NULL,
	[ItemStorageType] [nvarchar](50) NULL,
	[ItemHazardFlag] [nvarchar](50) NULL,
PRIMARY KEY CLUSTERED
(
	[ItemId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Port]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Port](
	[PortId] [int] NOT NULL,
	[PortName] [nvarchar](50) NULL,
	[Country] [nvarchar](50) NULL,
PRIMARY KEY CLUSTERED
(
	[PortId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Ship]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Ship](
	[ShipId] [int] NOT NULL,
	[ShipName] [nvarchar](50) NULL,
	[MaxNumberContain] [int] NULL,
	[SpeedInKnots] [int] NULL,
	[SpeedInKm_H] [int] NULL,
	[Country] [nvarchar](50) NULL,
	[YearCost] [money] NULL,
PRIMARY KEY CLUSTERED
(
	[ShipId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[VoyagePort]    Script Date: 21-3-2020 19:52:40 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[VoyagePort](
	[ShipmentId] [int] NOT NULL,
	[PortIdCurrent] [int] NOT NULL,
	[PortIdNext] [int] NOT NULL,
	[ConsignorId] [int] NOT NULL,
	[ShipId] [int] NOT NULL,
	[ItemId] [int] NOT NULL,
	[ContainerId] [int] NOT NULL,
	[PortIdStart] [int] NOT NULL,
	[PortIdEnd] [int] NOT NULL,
	[LegDateDeparture] [datetime] NOT NULL,
	[LegDateArrival] [datetime] NOT NULL,
```

```
	[TrajectDistance] [int] NULL,
	[ContainerNrStart] [int] NULL,
	[ContainerNrEnd] [int] NULL,
 CONSTRAINT [PK_VoyagePort] PRIMARY KEY CLUSTERED
(
	[ShipmentId] ASC,
	[PortIdCurrent] ASC,
	[PortIdNext] ASC,
	[ConsignorId] ASC,
	[ShipId] ASC,
	[ItemId] ASC,
	[ContainerId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Consignor] FOREIGN KEY([ConsignorId])
REFERENCES [dbo].[Consignor] ([ConsignorId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Consignor]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Container] FOREIGN KEY([ContainerId])
REFERENCES [dbo].[Container] ([ContainerId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Container]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Item] FOREIGN KEY([ItemId])
REFERENCES [dbo].[Item] ([ItemId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Item]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Ship] FOREIGN KEY([ShipId])
REFERENCES [dbo].[Ship] ([ShipId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Ship]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePortCurrent] FOREIGN KEY([PortIdCurrent])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePortCurrent]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePortEnd] FOREIGN KEY([PortIdEnd])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePortEnd]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePortNext] FOREIGN KEY([PortIdNext])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePortNext]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePortStart] FOREIGN KEY([PortIdStart])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePortStart]
GO
```

8.6. ODS SQL Script

```
USE [i2139897_Project4]
GO
/****** Object:  Table [dbo].[VoyagePort]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[VoyagePort](
	[RowId] [int] IDENTITY(1,1) NOT NULL,
	[PortIdCurrent] [int] NOT NULL,
	[PortIdNext] [int] NOT NULL,
	[ConsignorRowId] [int] NOT NULL,
	[ShipRowId] [int] NOT NULL,
	[ItemRowId] [int] NOT NULL,
	[ContainerRowId] [int] NOT NULL,
	[PortIdStart] [int] NOT NULL,
	[PortIdEnd] [int] NOT NULL,
	[LegDateDeparture] [datetime] NOT NULL,
	[LegDateArrival] [datetime] NOT NULL,
	[TrajectDistance] [int] NULL,
	[ContainerNrStart] [int] NULL,
	[ContainerNrEnd] [int] NULL,
	[ShipmentId] [int] NULL,
	[StayDays] [int] NULL,
	[TotalContainerCount] [decimal](18, 2) NULL,
	[WeightFactor] [decimal](18, 2) NULL,
 CONSTRAINT [PK_VoyagePort_1] PRIMARY KEY CLUSTERED
(
	[RowId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY],
 CONSTRAINT [IX_VoyagePort] UNIQUE NONCLUSTERED
(
	[ConsignorRowId] ASC,
	[ContainerRowId] ASC,
	[PortIdCurrent] ASC,
	[PortIdNext] ASC,
	[ShipRowId] ASC,
	[ItemRowId] ASC,
	[ShipmentId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Ship]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Ship](
	[RowId] [int] IDENTITY(1,1) NOT NULL,
	[ShipName] [nvarchar](50) NULL,
	[MaxNumberContain] [int] NULL,
	[SpeedInKnots] [int] NULL,
	[SpeedInKm_H] [int] NULL,
	[Country] [nvarchar](50) NULL,
	[YearCost] [money] NULL,
	[StartDate] [datetime] NULL,
	[EndDate] [datetime] NULL,
	[ShipId] [int] NOT NULL,
 CONSTRAINT [PK__Ship__2A05CAB309DFA8EF] PRIMARY KEY CLUSTERED
(
	[RowId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
```

```
/****** Object:  View [dbo].[View_3]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[View_3]
AS
SELECT       SUM(vp.WeightFactor) AS CorrectedWeightFactor, vp.ShipmentId, vp.ShipRowId, vp.LegDateArrival,
vp.LegDateDeparture
FROM         dbo.VoyagePort AS vp INNER JOIN
                  dbo.Ship AS s ON vp.ShipRowId = s.RowId
GROUP BY vp.LegDateArrival, vp.ShipmentId, vp.ShipRowId, vp.LegDateDeparture
GO
/****** Object:  View [dbo].[View_5]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[View_5]
AS
SELECT       SUM(vp.WeightFactor) AS CorrectedWeightFactor, vp.ShipmentId, vp.ShipRowId
FROM         dbo.VoyagePort AS vp INNER JOIN
                  dbo.Ship AS s ON vp.ShipRowId = s.RowId
GROUP BY vp.ShipmentId, vp.ShipRowId, vp.WeightFactor
GO
/****** Object:  View [dbo].[View_1]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[View_1]
AS
SELECT       PortIdCurrent, PortIdNext, ConsignorId, ShipId, ItemId, ContainerId
FROM         dbo.VoyagePort
GO
/****** Object:  View [dbo].[View_2]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE VIEW [dbo].[View_2]
AS
SELECT       dbo.VoyagePort.*
FROM         dbo.VoyagePort
GO
/****** Object:  Table [dbo].[Calendar Dimension]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Calendar Dimension](
            [FullDate] [datetime] NOT NULL,
            [Day] [tinyint] NOT NULL,
            [DaySuffix] [varchar](4) NOT NULL,
            [DayOfWeek] [varchar](9) NOT NULL,
            [DayOfWeekNumber] [int] NOT NULL,
            [DayOfWeekInMonth] [tinyint] NOT NULL,
            [DayOfYearNumber] [int] NOT NULL,
            [WeekOfYearNumber] [tinyint] NOT NULL,
            [WeekOfMonthNumber] [tinyint] NOT NULL,
            [CalendarMonthNumber] [tinyint] NOT NULL,
            [CalendarMonthName] [varchar](9) NOT NULL,
            [CalendarQuarterNumber] [tinyint] NOT NULL,
            [CalendarQuarterName] [varchar](6) NOT NULL,
            [CalendarYearNumber] [int] NOT NULL,
            [StandardDate] [varchar](10) NULL,
            [WeekDayFlag] [bit] NOT NULL,
            [HolidayFlag] [bit] NOT NULL,
            [OpenFlag] [bit] NOT NULL,
```

```
	[FirstDayOfCalendarMonthFlag] [bit] NOT NULL,
	[LastDayOfCalendarMonthFlag] [bit] NOT NULL,
 CONSTRAINT [PK_Calendar Dimension] PRIMARY KEY CLUSTERED
(
	[FullDate] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Consignor]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Consignor](
	[RowId] [int] IDENTITY(1,1) NOT NULL,
	[Consignor] [nvarchar](255) NULL,
	[City] [nvarchar](255) NULL,
	[Country] [nvarchar](255) NULL,
	[Discount] [int] NULL,
	[Alpha-3 Code] [nvarchar](255) NULL,
	[StartDate] [datetime] NULL,
	[EndDate] [datetime] NULL,
	[ConsignorId] [int] NOT NULL,
 CONSTRAINT [PK__Consigno__9D41263F858E2398] PRIMARY KEY CLUSTERED
(
	[RowId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Container]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Container](
	[RowId] [int] IDENTITY(1,1) NOT NULL,
	[Type] [varchar](50) NULL,
	[RefrigerationFlag] [bit] NULL,
	[PowerFlag] [varchar](50) NULL,
	[Length] [decimal](18, 2) NULL,
	[Cubes] [int] NULL,
	[EuroPricePerKm] [decimal](18, 2) NULL,
	[StartDate] [datetime] NULL,
	[EndDate] [datetime] NULL,
	[ContainerId] [int] NOT NULL,
	[Weight] [int] NULL,
 CONSTRAINT [PK__Containe__037960BB6BD9212D] PRIMARY KEY CLUSTERED
(
	[RowId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ISOCodes]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ISOCodes](
	[English short name] [nvarchar](255) NULL,
	[French short name] [nvarchar](255) NULL,
	[Alpha-2 code] [nvarchar](255) NULL,
	[Alpha-3 code] [nvarchar](255) NULL,
	[Numeric] [float] NULL
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[ISOError]    Script Date: 25-3-2020 23:40:58 ******/
```

```
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[ISOError](
	[Alpha-3 code] [nvarchar](255) NULL,
	[AltName] [nvarchar](255) NULL
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Item]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Item](
	[RowId] [int] IDENTITY(1,1) NOT NULL,
	[ItemDescription] [nvarchar](50) NULL,
	[ItemCategory] [nvarchar](50) NULL,
	[ItemMfgr] [nvarchar](50) NULL,
	[ItemStorageType] [nvarchar](50) NULL,
	[ItemHazardFlag] [nvarchar](50) NULL,
	[StartDate] [datetime] NULL,
	[EndDate] [datetime] NULL,
	[ItemId] [int] NOT NULL,
 CONSTRAINT [PK__Item__727E838B152D1045] PRIMARY KEY CLUSTERED
(
	[RowId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
/****** Object:  Table [dbo].[Port]    Script Date: 25-3-2020 23:40:58 ******/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
CREATE TABLE [dbo].[Port](
	[PortName] [nvarchar](50) NULL,
	[Country] [nvarchar](50) NULL,
	[PortId] [int] NOT NULL,
 CONSTRAINT [PK_Port] PRIMARY KEY CLUSTERED
(
	[PortId] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS = ON,
ALLOW_PAGE_LOCKS = ON) ON [PRIMARY]
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Consignor] ADD  CONSTRAINT [DF_Consignor_StartDate]  DEFAULT (getdate()) FOR [StartDate]
GO
ALTER TABLE [dbo].[Consignor] ADD  CONSTRAINT [DF_Consignor_EndDate]  DEFAULT (NULL) FOR [EndDate]
GO
ALTER TABLE [dbo].[Container] ADD  CONSTRAINT [DF_Container_StartDate]  DEFAULT (getdate()) FOR [StartDate]
GO
ALTER TABLE [dbo].[Container] ADD  CONSTRAINT [DF_Container_EndDate]  DEFAULT (NULL) FOR [EndDate]
GO
ALTER TABLE [dbo].[Item] ADD  CONSTRAINT [DF_Item_StartDate]  DEFAULT (getdate()) FOR [StartDate]
GO
ALTER TABLE [dbo].[Item] ADD  CONSTRAINT [DF_Item_EndDate]  DEFAULT (NULL) FOR [EndDate]
GO
ALTER TABLE [dbo].[Ship] ADD  CONSTRAINT [DF_Ship_StartDate]  DEFAULT (getdate()) FOR [StartDate]
GO
ALTER TABLE [dbo].[Ship] ADD  CONSTRAINT [DF_Ship_EndDate]  DEFAULT (NULL) FOR [EndDate]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Consignor] FOREIGN KEY([ConsignorRowId])
REFERENCES [dbo].[Consignor] ([RowId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Consignor]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Container] FOREIGN KEY([ContainerRowId])
```

```
REFERENCES [dbo].[Container] ([RowId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Container]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_DateDim] FOREIGN KEY([LegDateDeparture])
REFERENCES [dbo].[Calendar Dimension] ([FullDate])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_DateDim]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Item] FOREIGN KEY([ItemRowId])
REFERENCES [dbo].[Item] ([RowId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Item]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Port_current] FOREIGN KEY([PortIdCurrent])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Port_current]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Port_end] FOREIGN KEY([PortIdEnd])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Port_end]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Port_next] FOREIGN KEY([PortIdNext])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Port_next]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Port_start] FOREIGN KEY([PortIdStart])
REFERENCES [dbo].[Port] ([PortId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Port_start]
GO
ALTER TABLE [dbo].[VoyagePort]  WITH CHECK ADD  CONSTRAINT [FK_VoyagePort_Ship] FOREIGN KEY([ShipRowId])
REFERENCES [dbo].[Ship] ([RowId])
GO
ALTER TABLE [dbo].[VoyagePort] CHECK CONSTRAINT [FK_VoyagePort_Ship]
GO
EXEC sys.sp_addextendedproperty @name=N'MS_DiagramPane1', @value=N'[0E232FF0-B466-11cf-A24F-00AA00A3EFFF, 1.00]
Begin DesignProperties =
  Begin PaneConfigurations =
    Begin PaneConfiguration = 0
      NumPanes = 4
      Configuration = "(H (1[40] 4[20] 2[20] 3) )"
    End
    Begin PaneConfiguration = 1
      NumPanes = 3
      Configuration = "(H (1 [50] 4 [25] 3))"
    End
    Begin PaneConfiguration = 2
      NumPanes = 3
      Configuration = "(H (1 [50] 2 [25] 3))"
    End
    Begin PaneConfiguration = 3
      NumPanes = 3
      Configuration = "(H (4 [30] 2 [40] 3))"
    End
    Begin PaneConfiguration = 4
      NumPanes = 2
      Configuration = "(H (1 [56] 3))"
    End
    Begin PaneConfiguration = 5
      NumPanes = 2
      Configuration = "(H (2 [66] 3))"
    End
    Begin PaneConfiguration = 6
      NumPanes = 2
      Configuration = "(H (4 [50] 3))"
```

```
  End
  Begin PaneConfiguration = 7
    NumPanes = 1
    Configuration = "(V (3))"
  End
  Begin PaneConfiguration = 8
    NumPanes = 3
    Configuration = "(H (1[56] 4[18] 2) )"
  End
  Begin PaneConfiguration = 9
    NumPanes = 2
    Configuration = "(H (1 [75] 4))"
  End
  Begin PaneConfiguration = 10
    NumPanes = 2
    Configuration = "(H (1[66] 2) )"
  End
  Begin PaneConfiguration = 11
    NumPanes = 2
    Configuration = "(H (4 [60] 2))"
  End
  Begin PaneConfiguration = 12
    NumPanes = 1
    Configuration = "(H (1) )"
  End
  Begin PaneConfiguration = 13
    NumPanes = 1
    Configuration = "(V (4))"
  End
  Begin PaneConfiguration = 14
    NumPanes = 1
    Configuration = "(V (2))"
  End
  ActivePaneConfig = 0
End
Begin DiagramPane =
  Begin Origin =
    Top = 0
    Left = 0
  End
  Begin Tables =
    Begin Table = "s"
      Begin Extent =
        Top = 6
        Left = 38
        Bottom = 136
        Right = 236
      End
      DisplayFlags = 280
      TopColumn = 0
    End
    Begin Table = "vp"
      Begin Extent =
        Top = 6
        Left = 274
        Bottom = 136
        Right = 474
      End
      DisplayFlags = 280
      TopColumn = 14
    End
  End
End
Begin SQLPane =
End
Begin DataPane =
  Begin ParameterDefaults = ""
  End
End
```

```
  Begin CriteriaPane =
    Begin ColumnWidths = 12
      Column = 1440
      Alias = 900
      Table = 1170
      Output = 720
      Append = 1400
      NewValue = 1170
      SortType = 1350
      SortOrder = 1410
      GroupBy = 1350
      Filter = 1350
      Or = 1350
      Or = 1350
      Or = 1350
    End
  End
End
' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'VIEW',@level1name=N'View_3'
GO
EXEC sys.sp_addextendedproperty @name=N'MS_DiagramPaneCount', @value=1 , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'VIEW',@level1name=N'View_3'
GO
EXEC sys.sp_addextendedproperty @name=N'MS_DiagramPane1', @value=N'[0E232FF0-B466-11cf-A24F-00AA00A3EFFF, 1.00]
Begin DesignProperties =
  Begin PaneConfigurations =
    Begin PaneConfiguration = 0
      NumPanes = 4
      Configuration = "(H (1[40] 4[20] 2[20] 3) )"
    End
    Begin PaneConfiguration = 1
      NumPanes = 3
      Configuration = "(H (1 [50] 4 [25] 3))"
    End
    Begin PaneConfiguration = 2
      NumPanes = 3
      Configuration = "(H (1 [50] 2 [25] 3))"
    End
    Begin PaneConfiguration = 3
      NumPanes = 3
      Configuration = "(H (4 [30] 2 [40] 3))"
    End
    Begin PaneConfiguration = 4
      NumPanes = 2
      Configuration = "(H (1 [56] 3))"
    End
    Begin PaneConfiguration = 5
      NumPanes = 2
      Configuration = "(H (2 [66] 3))"
    End
    Begin PaneConfiguration = 6
      NumPanes = 2
      Configuration = "(H (4 [50] 3))"
    End
    Begin PaneConfiguration = 7
      NumPanes = 1
      Configuration = "(V (3))"
    End
    Begin PaneConfiguration = 8
      NumPanes = 3
      Configuration = "(H (1[56] 4[18] 2) )"
    End
    Begin PaneConfiguration = 9
      NumPanes = 2
      Configuration = "(H (1 [75] 4))"
    End
    Begin PaneConfiguration = 10
      NumPanes = 2
      Configuration = "(H (1[66] 2) )"
```

```
  End
  Begin PaneConfiguration = 11
    NumPanes = 2
    Configuration = "(H (4 [60] 2))"
  End
  Begin PaneConfiguration = 12
    NumPanes = 1
    Configuration = "(H (1) )"
  End
  Begin PaneConfiguration = 13
    NumPanes = 1
    Configuration = "(V (4))"
  End
  Begin PaneConfiguration = 14
    NumPanes = 1
    Configuration = "(V (2))"
  End
  ActivePaneConfig = 0
End
Begin DiagramPane =
  Begin Origin =
    Top = 0
    Left = 0
  End
  Begin Tables =
    Begin Table = "vp"
      Begin Extent =
        Top = 6
        Left = 38
        Bottom = 136
        Right = 238
      End
      DisplayFlags = 280
      TopColumn = 0
    End
    Begin Table = "s"
      Begin Extent =
        Top = 6
        Left = 276
        Bottom = 136
        Right = 474
      End
      DisplayFlags = 280
      TopColumn = 6
    End
  End
End
Begin SQLPane =
End
Begin DataPane =
  Begin ParameterDefaults = ""
  End
End
Begin CriteriaPane =
  Begin ColumnWidths = 12
    Column = 1440
    Alias = 900
    Table = 1170
    Output = 720
    Append = 1400
    NewValue = 1170
    SortType = 1350
    SortOrder = 1410
    GroupBy = 1350
    Filter = 1350
    Or = 1350
    Or = 1350
    Or = 1350
  End
```

```
  End
End
' , @level0type=N'SCHEMA',@level0name=N'dbo', @level1type=N'VIEW',@level1name=N'View_5'
GO
EXEC sys.sp_addextendedproperty @name=N'MS_DiagramPaneCount', @value=1 , @level0type=N'SCHEMA',@level0name=N'dbo',
@level1type=N'VIEW',@level1name=N'View_5'
GO
```