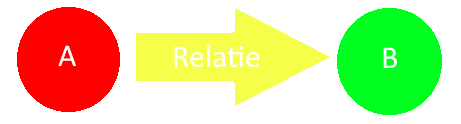
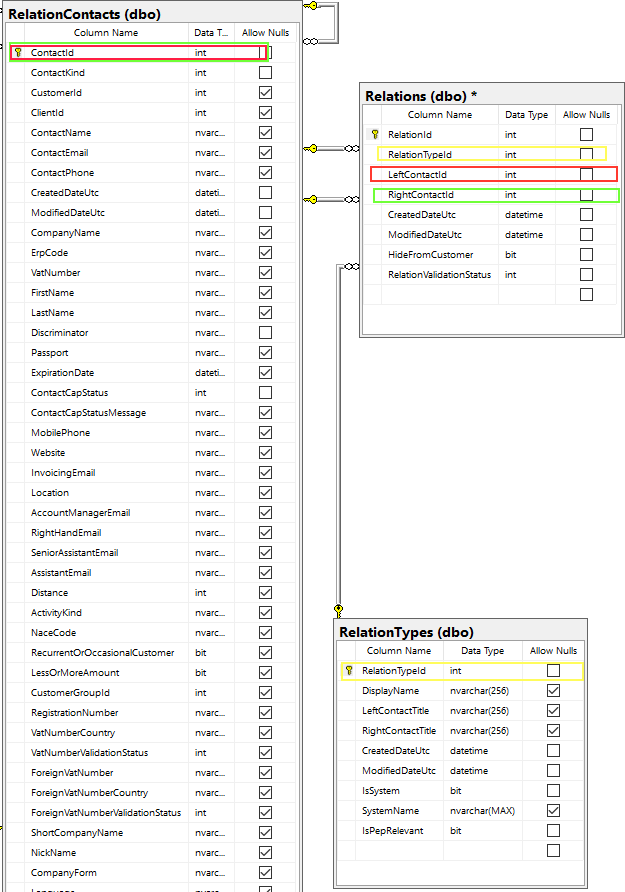
# Technische analyse:

## 1. In een SQL database (in de cloud) zijn al gedefinieerde relaties bekend. Reverse engineer deze SQL database om de relaties te begrijpen en visualiseer deze relaties.

select a.ContactName, t.LeftContactTitle, b.ContactName

from RelationContacts a inner join relations y ON (a.ContactId = y.LeftContactId) inner join RelationContacts b ON (b.ContactId = y.RightContactId) inner join RelationTypes t ON (t.RelationTypeId = y.RelationTypeId)





De daadwerkelijke visualisatie op het platform zou werken door relevante entiteiten van de DB af te halen en om te zetten in een JSON bestand, die gelezen kan worden door D3JS code die een visuele weergave zou genereren.

### Minimum vereisten:

* Het visualiseren van de resultaten uit de relatie structuur

### Overige vereisten:

* Het gebruiken van een key-woord dat niet in een tekst voorkomt als zoekterm.   
  (topicmodelling)

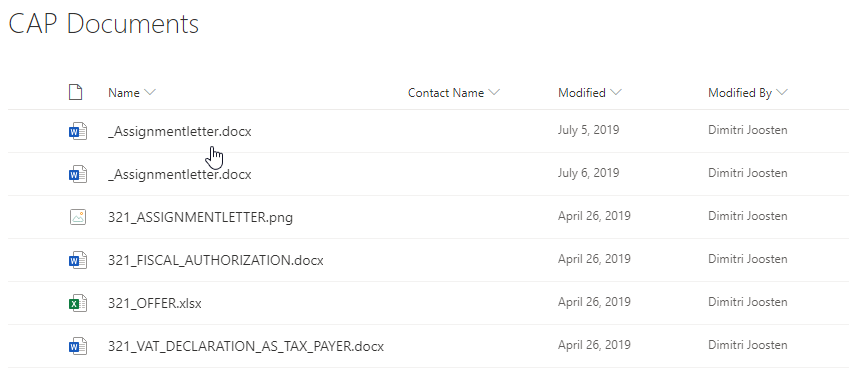
• Bij het zoeken rekening houden met de afgelegde weg en op basis hiervan documenten

voorstellen.

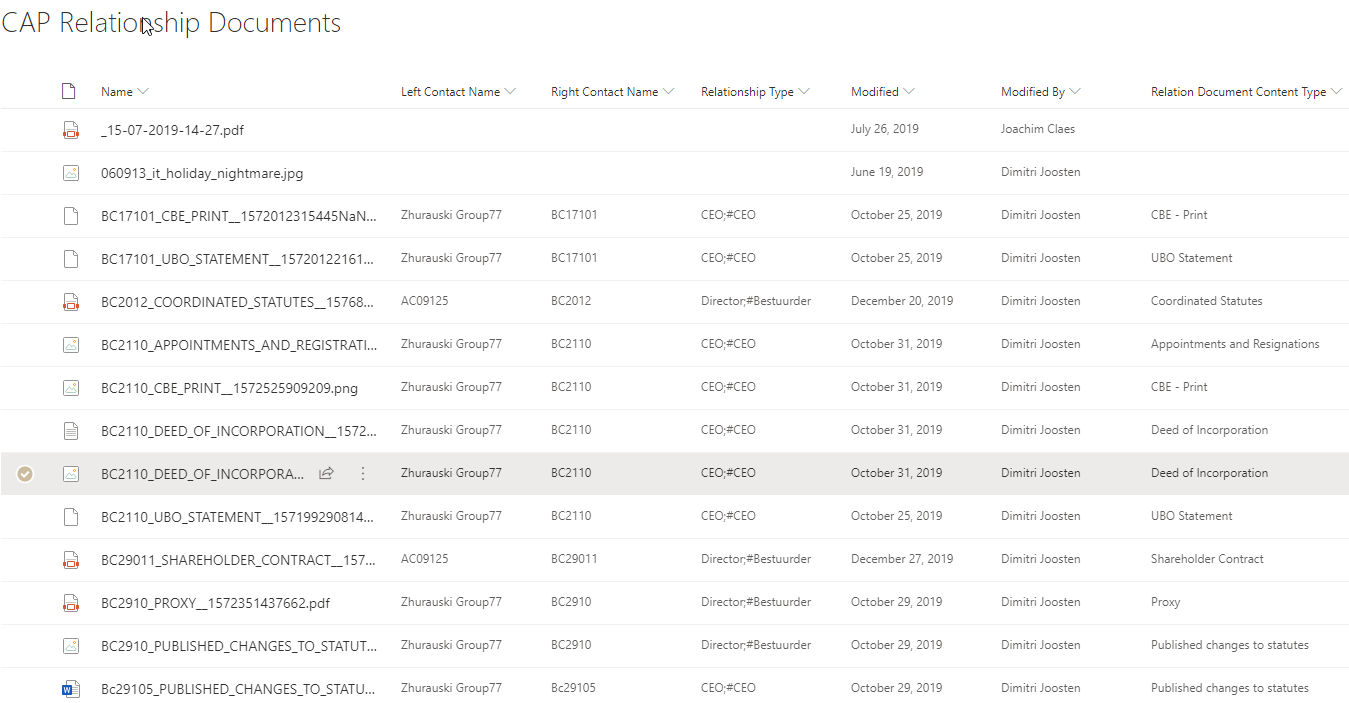
## 2. Door het combineren van een azure active directory & de SQL database is het mogelijk om te achterhalen wie welke toegangsrechten heeft voor bepaalde documenten. Achterhaal wie er toegang heeft tot welke documenten en visualiseer dit.

A) Documenten:

1. CAP Documents:



2. CAP Relationship Documents:



3. REST API

Door gebruik te maken van de sharepoint rest api zou het mogelijk moeten zijn om de CAP Documents en de CAP Relationship documenten in de database op te slaan met zijn gerelateerde informatie. <https://docs.microsoft.com/en-us/sharepoint/dev/sp-add-ins/complete-basic-operations-using-sharepoint-rest-endpoints>

### Minimum vereisten:

* Het visualiseren van de toegangsrechten uit de azure active directory en SQL.
* Het ontwikkelen van een tussentaal die gebruikt kan worden voor de visualisatie en die op een efficiënte manier kan gaan zoeken tussen de documenten. Vergelijking tussen Neo4J en SQL

## 3. De SQL database is slecht opgebouwd en bevat 1 entry met een enorme Json string. Parse deze Json string zodat hier entiteiten uit gehaald kunnen worden. Sla deze entiteiten opnieuw op, zodat dit een volgend vertrekpunt kan zijn voor de visualisatie.

Soorten JSON-bestanden uit de tabel ContactCAPS:

1) Json met enkel het deel RISKValidation:  
<https://puu.sh/FayCa/7d3b0bff01.json>

2) Json met het deel Contact met constumer entiteit info

2a) JSON met Contact met als type family

<https://puu.sh/FayOd/4fa80b4c16.json>

2b) JSON met Contact met als type Business

<https://puu.sh/FayMA/3b3549e79b.json>

2c) JSON met Contact met als type Individual

<https://puu.sh/FayNo/9cb22bdd31.json>

3) Json met het deel contact, productinformatie, etc… zonder customer entiteit info

3a) JSON met Contact met als type family

<https://puu.sh/FayKm/51b96c0916.json>

3b) JSON met Contact met als type Business

<https://puu.sh/FayIo/0e3aa0846a.json>

3c) JSON met Contact met als type Individual

<https://puu.sh/FayJA/df7c4300b7.json>

Deze JSON strings moeten geparsed worden en omgezet worden naar database objecten, en op de overkoepelende database gezet worden.

### Minimum vereisten:

* Een vergelijking tussen een SQL database en een Json data structuur.

### Overige vereisten:

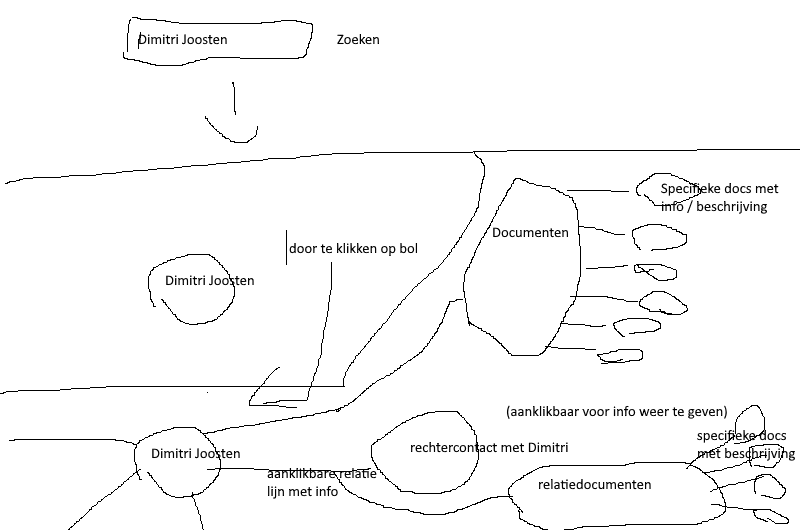
* Visueel antwoord kunnen geven op volgende vragen:

Welke klant gebruikt welke producten? (Informatie uit de Json string)

Welke klanten hebben een hoog risico? (Informatie uit de Json string)

Van welke klanten zijn de UBO/gecoördineerde statuten onvolledig?   
(Zeer gewaardeerd!) (meerdere DB)

# Vragen:

* Voor de visualisatie, van welke enteiten vertrekken we?  
  Hoe moet het er uit zien?  
  
* \* In bovenstaande tekening kan je voor Relationship CAP documenten ook nog een tak met filtering voor documentstype omdat ze op de sharepoint al gefilterd zijn op documenttype. De documenten uit CAP documents hebbe deze typering echter niet, dus moeten we wachten op andere groepjes om deze date op type te ordenen en die daar op filteren?
* Is er een makkelijkere manier om de toeganginformatie op specifieke documenten te verkrijgen en weer te kunnen geven op ons portaal ipv ze met de rest api van sharepoint af te halen en ze in de databank stoppen?
* Welke informatie moet er weergegeven worden voor specifieke contactenteiten en documenten? Tot nu toe voor contacten ga ik uit van info zoals naam, email, bedrijf, contacttype, … Maar het zou erg helpen als dit gespecifieerd zou kunnen worden.