

# Optimaliseren van gegevensverwerking in Microsoft Azure: Een vergelijkende analyse van implementatiemethoden voor Extractie, Transformatie en Laden (ETL) en Extractie, Laden en Transformatie (ELT).

Optionele ondertitel.

---

**Lievens Loeka.**

Scriptie voorgedragen tot het bekomen van de graad van  
Professionele bachelor in de toegepaste informatica

**Promotor:** Dhr. Bosteels Gertjan

**Co-promotor:** Dhr. Van Damme Koen

**Academiejaar:** 2023–2024

**Eerste examenperiode**

**Departement IT en Digitale Innovatie .**

**HO  
GENT**



# Woord vooraf

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

# Samenvatting

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

# Inhoudsopgave

<b>Lijst van figuren</b>	<b>vii</b>
<b>1 Inleiding</b>	<b>1</b>
1.1 Probleemstelling	1
1.2 Onderzoeksvraag	2
1.3 Onderzoeksdoelstelling	2
1.4 Opzet van deze bachelorproef	2
<b>2 Stand van zaken</b>	<b>3</b>
2.1 Wat zijn ETL's of ELT's?	3
2.2 Welke soorten ETL tools bestaan er?	3
2.3 Populairste cloud-based ETL tools	4
2.4 Azure Data Factory (ADF)	4
2.4.1 Onderdelen	4
2.4.2 Mapping Data Flows	5
2.5 Azure Databricks	6
2.5.1 Delta Lake	6
2.5.2 Delta Live Tables	6
<b>3 Methodologie</b>	<b>7</b>
3.1 Literatuuronderzoek	7
3.2 Long list	7
3.3 Short list	7
3.4 Vergelijkingscriteria	8
3.4.1 Kostprijs	8
3.4.2 Performantie	8
3.4.3 Mogelijkheid tot debuggen	8
3.4.4 Verschil in implementatietijd	8
3.4.5 Moeilijkheidsgraad in opzet	8
3.4.6 Mogelijkheden van de tool	8
3.4.7 Onderhoudbaarheid	8
3.4.8 Testbaarheid	8
3.5 Proof-of-concepts	8
3.5.1 Azure Data Factory (ADF)	8
3.5.2 Azure Databricks	30

<b>4 Conclusie</b>	<b>52</b>
<b>A Onderzoeksvoorstel</b>	<b>54</b>
A.1 Inleiding . . . . .	54
A.2 Literatuurstudie . . . . .	55
A.3 Methodologie . . . . .	56
A.4 Verwacht resultaat, conclusie . . . . .	57
<b>Bibliografie</b>	<b>58</b>

# Lijst van figuren

3.1	Aanmaken van Azure Data Factory . . . . .	9
3.2	Configuratie van Azure Data Factory . . . . .	9
3.3	Deployment complete van Azure Data Factory . . . . .	10
3.4	Toewijzen van Data Factory Contributor Role . . . . .	10
3.5	Configuratie van Git in Azure Data Factory . . . . .	11
3.6	Configuratie van Azure DevOps in Azure Data Factory . . . . .	11
3.7	Configuratie van source transformation . . . . .	12
3.8	Configuratie van linked service . . . . .	13
3.9	Configuratie van source options . . . . .	14
3.10	Configuratie van source options . . . . .	15
3.11	Importen van schema voor data flow source . . . . .	15
3.12	Voorbeeld geïmporteerd schema van data flow source . . . . .	16
3.13	Data preview in data flow . . . . .	16
3.14	Join van de tabel "new_year" op de tabel "new_syndicalpremiumrequest" . . . . .	17
3.15	Derive "EntryYear", "CalendarYear" en "RefYear" op de tabel "new_syndicalpremiumrequest" . . . . .	17
3.16	Hernoemen van kolommen op "new_syndicalpremiumrequest" en splitting in twee apparte branches . . . . .	18
3.17	Inner join van "new_organizationyear" op de tabel "new_syndicalpremiumrequest" . . . . .	19
3.18	Selecteren en hernoemen van kolommen op de tabel "new_syndicalpremiumrequest" . . . . .	19
3.19	Custom (cross) join van "new_membership" op de tabel "new_syndicalpremiumrequest" . . . . .	20
3.20	Selecteren en hernoemen van kolommen op de tabel "new_syndicalpremiumrequest" . . . . .	21
3.21	Union van twee branches . . . . .	21
3.22	Group by "spr_Id", "Group" en "RefYear" in "new_syndicalpremiumrequest" . . . . .	22
3.23	Group by "spr_Id", "Group" en "RefYear" in "new_syndicalpremiumrequest" . . . . .	23
3.24	Joinen van de tabel "new_membership" op de tabel "new_syndicalpremiumrequest" . . . . .	24
3.25	Selecteren van de juiste kolommen in "new_syndicalpremiumrequest" . . . . .	24
3.26	Herhalen van transformaties . . . . .	24
3.27	Herhalen van transformaties . . . . .	25
3.28	Bepalen van "Gr" in "new_syndicalpremiumrequest" . . . . .	25
3.29	Conditional split op "new_syndicalpremiumrequest" . . . . .	26
3.30	Derive "AntheaNumber" en "MembershipNumber" op de tabel "new_syndicalpremiumrequest" . . . . .	26
3.31	Join van de tabel "new_membership" op de tabel "new_syndicalpremiumrequest" aan de hand van "new_groupid", "new_personid" en "CalendarYear" . . . . .	27
3.32	Select op de tabel "new_syndicalpremiumrequest" . . . . .	27

3.33 Join van de tabel “new_membership” op de tabel “new_syndicalpremiumrequest” aan de hand van “new_groupid”, “new_personid” en “EntryYear” . . . . .	28
3.34 Join van de tabel “new_membership” op de tabel “new_syndicalpremiumrequest” aan de hand van “new_groupid”, “new_personid” en “RefYear” . . . . .	28
3.35 Derive van kolommen “AntheaNumber” en “MembershipNumber” op de tabel “new_syndicalpremiumrequest” . . . . .	28
3.36 Verwijderen van onnodige kolommen op de tabel “new_syndicalpremiumrequest”	29
3.37 Union van twee streams . . . . .	30
3.38 Aanmaken van Azure Databricks . . . . .	30
3.39 Configuratie van Azure Databricks . . . . .	31
3.40 Deployment complete van Azure Databricks . . . . .	31
3.41 Aanmaken van compute resource . . . . .	32
3.42 Configuratie van compute resource . . . . .	32
3.43 Installatie van “spark-cdm-connector” . . . . .	33
3.44 Gebruikers toevoegen/verwijderen in Azure Databricks . . . . .	33
3.45 Rechten wijzigen van gebruikers in Azure Databricks . . . . .	34
3.46 Groepen toevoegen of verwijderen in Azure Databricks . . . . .	34
3.47 Clone Git repository in Databricks folders . . . . .	35
3.48 Aanmaken van een notebook in databricks . . . . .	35
3.49 Commit en push in Databricks . . . . .	35
3.50 Commit en push in Databricks . . . . .	36
3.51 App registration aanmaken . . . . .	36
3.52 Configuratie app registration . . . . .	37
3.53 Aanmaken client secret . . . . .	37
3.54 Role assignment toevoegen aan storage account . . . . .	38
3.55 Job function role kiezen voor role assignment . . . . .	38
3.56 Members kiezen voor role assignment . . . . .	39
3.57 Key vault aanmaken . . . . .	39
3.58 Configuratie van key vault . . . . .	40
3.59 Permission model van key vault . . . . .	40
3.60 Networking configuratie van key vault . . . . .	41
3.61 Deployment complete van key vault . . . . .	42
3.62 Toevoegen van een secret in key vault . . . . .	42
3.63 Opzoeken van properties van key vault . . . . .	42
3.64 Aanmaken van secret scope in databricks . . . . .	43



# 1

## Inleiding

De inleiding moet de lezer net genoeg informatie verschaffen om het onderwerp te begrijpen en in te zien waarom de onderzoeksvraag de moeite waard is om te onderzoeken. In de inleiding ga je literatuurverwijzingen beperken, zodat de tekst vlot leesbaar blijft. Je kan de inleiding verder onderverdelen in secties als dit de tekst verduidelijkt. Zaken die aan bod kunnen komen in de inleiding (Pollefliet, 2011):

- context, achtergrond
- afbakenen van het onderwerp
- verantwoording van het onderwerp, methodologie
- probleemstelling
- onderzoeksdoelstelling
- onderzoeksvraag
- ...

### 1.1. Probleemstelling

Uit je probleemstelling moet duidelijk zijn dat je onderzoek een meerwaarde heeft voor een concrete doelgroep. De doelgroep moet goed gedefinieerd en afgelijnd zijn. Doelgroepen als “bedrijven,” “KMO’s”, systeembeheerders, enz. zijn nog te vaag. Als je een lijstje kan maken van de personen/organisaties die een meerwaarde zullen vinden in deze bachelorproef (dit is eigenlijk je steekproefkader), dan is dat een indicatie dat de doelgroep goed gedefinieerd is. Dit kan een enkel bedrijf zijn of zelfs één persoon (je co-promotor/opdrachtgever).

## 1.2. Onderzoeksvraag

Wees zo concreet mogelijk bij het formuleren van je onderzoeksvraag. Een onderzoeksvraag is trouwens iets waar nog niemand op dit moment een antwoord heeft (voor zover je kan nagaan). Het opzoeken van bestaande informatie (bv. “welke tools bestaan er voor deze toepassing?”) is dus geen onderzoeksvraag. Je kan de onderzoeksvraag verder specificeren in deelvragen. Bv. als je onderzoek gaat over performantiemetingen, dan

## 1.3. Onderzoeksdoelstelling

Wat is het beoogde resultaat van je bachelorproef? Wat zijn de criteria voor succes? Beschrijf die zo concreet mogelijk. Gaat het bv. om een proof-of-concept, een prototype, een verslag met aanbevelingen, een vergelijkende studie, enz.

## 1.4. Opzet van deze bachelorproef

De rest van deze bachelorproef is als volgt opgebouwd:

In Hoofdstuk 2 wordt een overzicht gegeven van de stand van zaken binnen het onderzoeksdomein, op basis van een literatuurstudie.

In Hoofdstuk 3 wordt de methodologie toegelicht en worden de gebruikte onderzoekstechnieken besproken om een antwoord te kunnen formuleren op de onderzoeksvragen.

In Hoofdstuk 4, tenslotte, wordt de conclusie gegeven en een antwoord geformuleerd op de onderzoeksvragen. Daarbij wordt ook een aanzet gegeven voor toekomstig onderzoek binnen dit domein.

# 2

## Stand van zaken

Bedrijven slaan veel data op. Doordat deze data nuttig kan zijn voor het identificeren van nieuwe kansen is het belangrijk om deze data klaar te maken voor business analytics. Dit is het proces van verzamelen, organiseren, analyseren en interpreteren van gegevens om inzichten te krijgen. Er kan bijvoorbeeld gekeken worden naar klantgegevens om zo patronen en trends te vinden in het gedrag van de klant. (J., [2023](#))

Doordat bedrijven vaak werken met veel verschillende soorten data dat op verschillende plekken opgeslaan wordt, is het vaak belangrijk dat deze data eerst opgekuist, getransformeerd en georganiseerd moet worden. Dit is waarbij het implementeren van ETL's en ELT's van pas komt. (Inmon, [2023](#))

### 2.1. Wat zijn ETL's of ELT's?

ETL's en ELT's zijn processen die organisaties gebruiken voor het verzamelen en samenvoegen van data uit meerdere bronnen. Bij ETL's wordt de data getransformeerd voor het naar de doelopslagplaats geladen wordt, terwijl dit bij ELT's pas achteraf gebeurt. Daardoor staat ETL voor Extract, Transform and Load en ELT voor Extract, Load and Transform. (Bartley, [2023](#))

### 2.2. Welke soorten ETL tools bestaan er?

Er bestaan verschillende soorten ETL tools. Zo zijn er de cloud-based ETL tools. Deze worden gehost op cloud infrastructure, zijn zeer schaalbaar en bieden pay-as-you-go prijs modellen aan. (Ethan, [2024](#))

Daarnaast zijn er ook on-premises ETL tools. Deze worden gehost op de infrastructuur van het bedrijf waardoor het bedrijf er de volledige controle over heeft. (Ethan, [2024](#))

Afhankelijk van wat men nodig heeft kan er ook gekozen worden voor hybrid ETL

tools. Dit is een combinatie van het gebruik van cloud-based tools met het gebruik van on-premises tools. (Ethan, 2024)

Ten slotte zijn er ook open source ETL tools. Dit zijn gratis ETL tools. Voorbeelden hiervan zijn Portable, Apache NiFi, AWS Glue, Airbyte en Informatica. (Ethan, 2024)

### 2.3. Populairste cloud-based ETL tools

Zoals te zien in de enquête van Vines en Tanasescu (2023) is Microsoft Azure, gevolgd door Amazon Web Services (AWS) en Google Cloud Services, de populairste cloud provider. Deze cloud-providers bieden dan ook de meest populaire cloud-based ETL tools aan.

Microsoft biedt bijvoorbeeld Azure Data Factory en Azure Databricks aan. Binnen Azure Data Factory kan er gebruik gemaakt worden van Mapping Data Flows, dit is een code-vrije manier waarmee ETL's opgebouwd kunnen worden. De logica achter de ETL kan hierna makkelijk getest worden op live data en samples. (Kromer, 2022b)

Daarnaast biedt Azure ook Azure Databricks aan. Het verschil hierbij is dat de ETL's worden geïmplementeerd via code terwijl dat bij Azure Data Factory via de UI tools gebeurt. Azure Databricks is gebaseerd op het Apache Spark opensource project. Het grote voordeel is dat het platform het toelaat om makkelijker te kunnen samen werken. Daarnaast is Apache Spark niet enkel gelimiteerd tot het maken van ETL's maar kan het ook gebruikt worden voor real-time analytics, machine learning, graph processing, etc. (Etaati, 2019)

Ook Amazon Web Services (AWS) en Google Cloud Services bieden ETL tools aan. Zo heeft AWS bijvoorbeeld AWS Glue (Khan e.a., 2024) en Google Cloud heeft Google Data Fusion. (Jaiswal, 2022)

### 2.4. Azure Data Factory (ADF)

Azure Data Factory is een platform-as-a-service (PAAS) voor het implementeren van ETL's en ELT's. Zowel on-premises als cloudgegevensbronnen worden hierbij ge ondersteund voor het verplaatsen van gegevens. (Rawat & Narain, 2018)

#### 2.4.1. Onderdelen

Azure Data Factory is opgebouwd uit verschillende onderdelen. Als eerste hebben we een pipeline. Dit is een groep van activiteiten die een reeks processen uitvoert zoals bijvoorbeeld het extraheren of transformeren van gegevens. Een voorbeeld van een activity is een Mapping Data Flow. Hiermee kan men logica voor datatransformaties ontwikkelen zonder code te schrijven. Daarnaast zijn er ook datasets. Dit is een representatie of verwijzing naar de daadwerkelijke gegevens in gegevens-opslag. Een dataset is steeds gekoppeld aan een linked service. Deze slaan de informatie op die Azure Data Factory nodig heeft voor het connecteren naar een

externe dataopslag. Een pipeline wordt uitgevoerd door een trigger. Er zijn veel verschillende soorten triggers voor veel verschillende soorten events. Daarnaast kan er voor een pipeline parameters gedefinieerd worden. Dit zijn read-only key-value pairs die een configuratie vormen. Ook kunnen er variables gebruikt worden om tijdelijk waarden op te slaan. In combinatie met parameters kunnen dan waarden tussen pipelines, data flows, en andere activiteiten doorgegeven worden. Wanneer een pipeline wordt uitgevoerd zal er een pipeline run aangemaakt worden. (Microsoft, 2024d)

### **2.4.2. Mapping Data Flows**

Met behulp van Mapping Data Flows kunnen ETL's geïmplementeerd worden zonder hiervoor gebruik te moeten maken van code. De resulterende data flows worden uitgevoerd als activiteiten binnen een Azure Data Factory pipeline dat gebruik maakt van Apache Spark Clusters. Deze Mapping Data Flows maken gebruik van data flow scripts. Dit zijn artifacten die gegenereerd worden door de UI. Het is een taal die de data transformatie beschrijft dat de Spark Cluster zal moeten uitvoeren. De UI van Azure Data Factory beheert het data flow script maar het script kan ook bekeken en handmatig bewerkt worden. (Kromer, 2022a)

In Mapping Data Flows kunnen verschillende soorten transformaties gedaan worden:

- Multiple inputs/outputs
  - New Branch
  - Join
  - Conditional Split
  - Exists
  - Union
  - Lookup
- Schema modifier
  - Derived Column
  - Select
  - Aggregate
  - Surrogate Key
  - Pivot
  - Unpivot
  - Window
  - Rank

- External Call
- Formatters
  - Flatten
  - Parse
  - Stringify
- Row modifier
  - Filter
  - Sort
  - Alter Row
  - Assert
- Flowlets
  - Flowlet
- Destination
  - Sink

## 2.5. Azure Databricks

Azure Databricks is een geavanceerd platform voor data-analyse dat zich integreert met Azure services. Het biedt een complete omgeving voor het ontwikkelen, implementeren en delen van krachtige data-analyses en AI-toepassingen op grote schaal. Het integreert zich ook met de opensource-community zoals bijvoorbeeld Delta Lake, Delta Sharing, MLflow, Apache Spark en Redash. Veel voorkomende use cases van Azure Databricks zijn het bouwen van een data lakehouse voor ondernemingen, het implementeren van ETL's, gebruik van machine learning en dergelijke. (Microsoft, [2024b](#))

### 2.5.1. Delta Lake

Delta Lake is het standaard opslagformaat voor alle operaties binnen Azure Databricks. Het maakt gebruik van Parquet data bestanden met een file-based transaction log voor ACID transactions. (Microsoft, [2024c](#))

### 2.5.2. Delta Live Tables

Delta Live Tables is een framework voor het bouwen van processing pipelines. Hierbij wordt er gebruik gemaakt van streaming tables en materialized views. Streaming tables zijn Delta tables waarbij er extra support is voor streaming of incremental data processing en materialized views zijn views waarbij de resultaten pre-computed zijn. (Microsoft, [2024a](#))

# 3

## Methodologie

### 3.1. Literatuuronderzoek

In het literatuuronderzoek zijn we in detail op zoek gegaan naar de mogelijkheden die er zijn om ETL's en ELT's te gaan implementeren. Hierbij houden we rekening dat Net IT met Microsoft producten werkt waardoor er vooral naar Azure gekeken wordt. Er is dus ook verder in detail gegaan op deze Azure producten doordat deze meer aan bod komen in de bachelorproef.

### 3.2. Long list

Hier alle mogelijkheden voor het implementeren van ETL's/ELT's opnoemen? Dit kan een lange lijst worden met mogelijkheden die niet relevant zijn voor Net IT aangezien er van Azure gebruik gemaakt wordt. Is dit een stap die dus overgeslaan kan worden?

### 3.3. Short list

Op basis van de long list houden we twee mogelijkheden over voor het implementeren van ELT's en ETL's. Dit zijn Azure Data Factory en Azure Databricks. Er is gekozen voor deze twee technologieën doordat deze beide van Azure zijn.

### 3.4. Vergelijkingscriteria

#### 3.4.1. Kostprijs

#### 3.4.2. Performantie

#### 3.4.3. Mogelijkheid tot debuggen

#### 3.4.4. Verschil in implementatietijd

#### 3.4.5. Moeilijkheidsgraad in opzet

#### 3.4.6. Mogelijkheden van de tool

#### 3.4.7. Onderhoudbaarheid

#### 3.4.8. Testbaarheid

### 3.5. Proof-of-concepts

Binnen Net IT wordt data van Microsoft 365 Customer Engagement geëxporteerd naar CSV bestanden en in Azure Data Lake geplaatst. Deze bestanden moeten minstens één keer per dag opgesplitst worden per groep per jaar en zullen moeten doorgestuurd worden naar de klant. Hiervoor moet er dus een ETL of ELT geïmplementeerd worden. Doordat er onderzocht wordt naar wat de beste mogelijkheid is voor het implementeren van deze ETL of ELT zal er dus een proof-of-concept uitgewerkt worden voor zowel Azure Data Factory en Azure Databricks. Voor het implementeren van deze proof-of-concepts zal er een pipeline gemaakt worden die ook bij de klant gebruikt wordt. Belangrijk hierbij is dat er voor deze bachelorproef gebruik gemaakt wordt van dummy data.

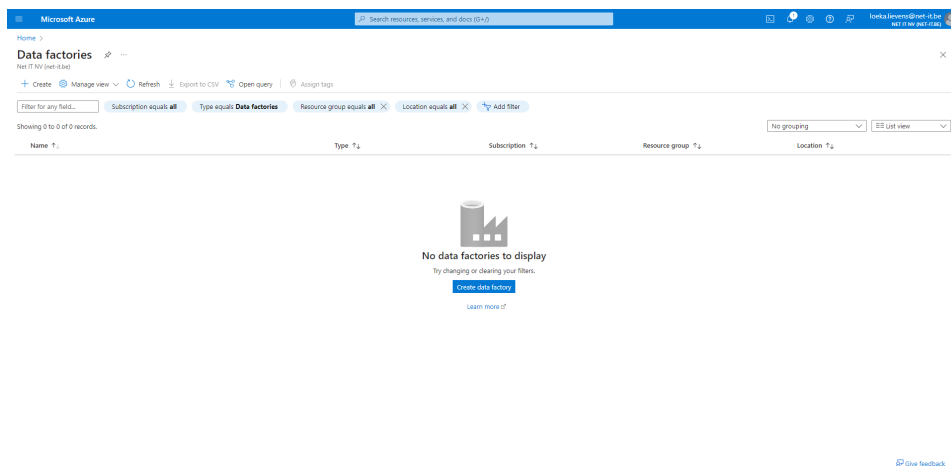
De tabellen uit data lake die gebruikt worden zijn “new\_syndicalpremiumrequest”, “new\_person”, “new\_bankaccount”, “new\_year”, “new\_membership”, “new\_group” en “new\_organizationyear”. Voor zowel Data Factory als Databricks wordt er eerst gekeken naar hoe deze opgezet kunnen worden. Vervolgens wordt er gekeken hoe er samen gewerkt kan worden en hoe source control geïmplementeerd kan worden. Daarnaast wordt er ook gekeken hoe men data kan ophalen uit data lake met behulp van het Common Data Model en worden de belangrijkste transformaties overlopen. Op deze manier kan Data Factory en Databricks makkelijk vergeleken worden.

Er wordt steeds gewerkt vanuit de tabel “new\_syndicalpremiumrequest”. Dit zijn de premies die geëxporteerd worden vanuit Microsoft 365 Customer Engagement. Deze premies worden opgesplitst per groep, per jaar. Voor de huidige proof-of-concepts worden deze premies naar slechts één CSV bestand geëxporteerd. Dit zodat het geëxporteerde CSV bestand van Data Factory dan makkelijk vergeleken kan worden met het geëxporteerde CSV bestand van Databricks. Het resultaat van de ETL of ELT is dus een CSV bestand waarbij elke rij een premie is, waarbij elke premie een groep en jaartal heeft.

#### 3.5.1. Azure Data Factory (ADF)



## Opzet van resources



**Figuur (3.1)**

Aanmaken van Azure Data Factory

Door in Microsoft Azure naar Data Factories te navigeren kunnen we een nieuwe data factory gaan aanmaken.

**Basics**   Git configuration   Networking   Advanced   Tags   Review + create

One-click to create data factory with sample pipeline and datasets. [Try it](#)

**Project details**

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

Resource group \* ⓘ  [Create new](#)

**Instance details**

Name \* ⓘ  ✓

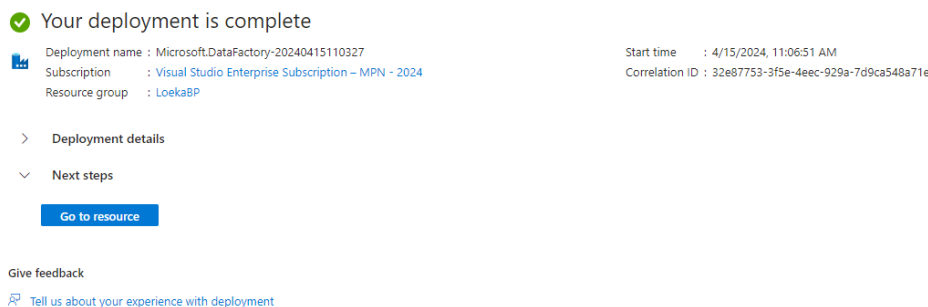
Region \* ⓘ

Version \* ⓘ

**Figuur (3.2)**

Configuratie van Azure Data Factory

Bij het aanmaken van een data factory moet er een subscription en resource group gekozen worden. Er kan een nieuwe resource group aangemaakt worden of een reeds bestaande gekozen worden. Daarnaast moet er een naam, gewenste regio en versie voor Data Factory gekozen worden. Git configuratie zal later aan bod komen.

**Figuur (3.3)**

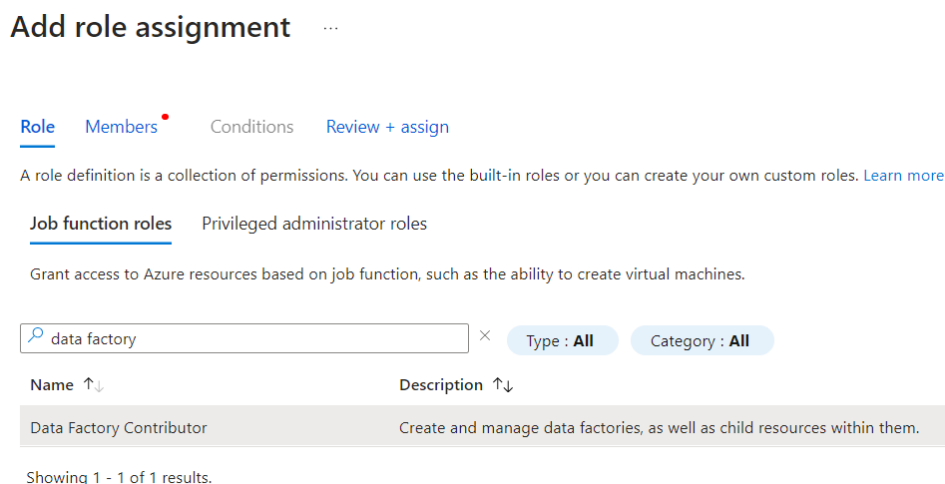
Deployment complete van Azure Data Factory

Wanneer de resource is aangemaakt kan Azure Data Factory opgestart worden.

### Collaboration en source control

Binnen Azure Data Factory kan er op 2 manieren samen gewerkt worden.

### Roles en permissions

**Figuur (3.4)**

Toewijzen van Data Factory Contributor Role

Door de bij de resource group van de data factory de Data Factory Contributor role toe te wijzen kan men toegang geven tot volgende zaken:

- Het aanmaken, wijzigen en verwijderen van data factories en child resources
- Deployment van Resource Manager templates
- Het managen van App Insight alerts voor Data Factory
- Het aanmaken van support tickets

### Source control

Azure Data Factory laat het toe om een Git repository te configureren via Azure Repos of GitHub.

### Configure a repository

Specify the settings that you want to use when connecting to your repository.

Repository type \* ⓘ

 Azure DevOps Git

☒ Cloud ⓘ ☐ Cloud (cross-tenant sign-in) ⓘ ☐ Server (on-premise) ⓘ

Microsoft Entra ID ⓘ


Net IT NV (118054c8-f9ce-4058-a30c-77ee43ef2918)

#### Figuur (3.5)

Configuratie van Git in Azure Data Factory

We kiezen voor Azure DevOps doordat er binnen Net IT hiermee gewerkt wordt.

### Configure a repository

 Net IT NV (118054c8-f9ce-4058-a30c-77ee43ef2918)

Specify the settings that you want to use when connecting to your repository.

☒ Select repository ☐ Use repository link

Azure DevOps organization name \* ⓘ

loekalievans

Project name \* ⓘ

Example Data Factory

Repository name \* ⓘ

Example Data Factory

Collaboration branch \* ⓘ

main

Publish branch \* ⓘ

adf\_publish

Root folder \* ⓘ

/

Custom comment

☒ Use custom comment

Import existing resources

☒ Import existing resources to repository

Import resource into this branch ⓘ

#### Figuur (3.6)

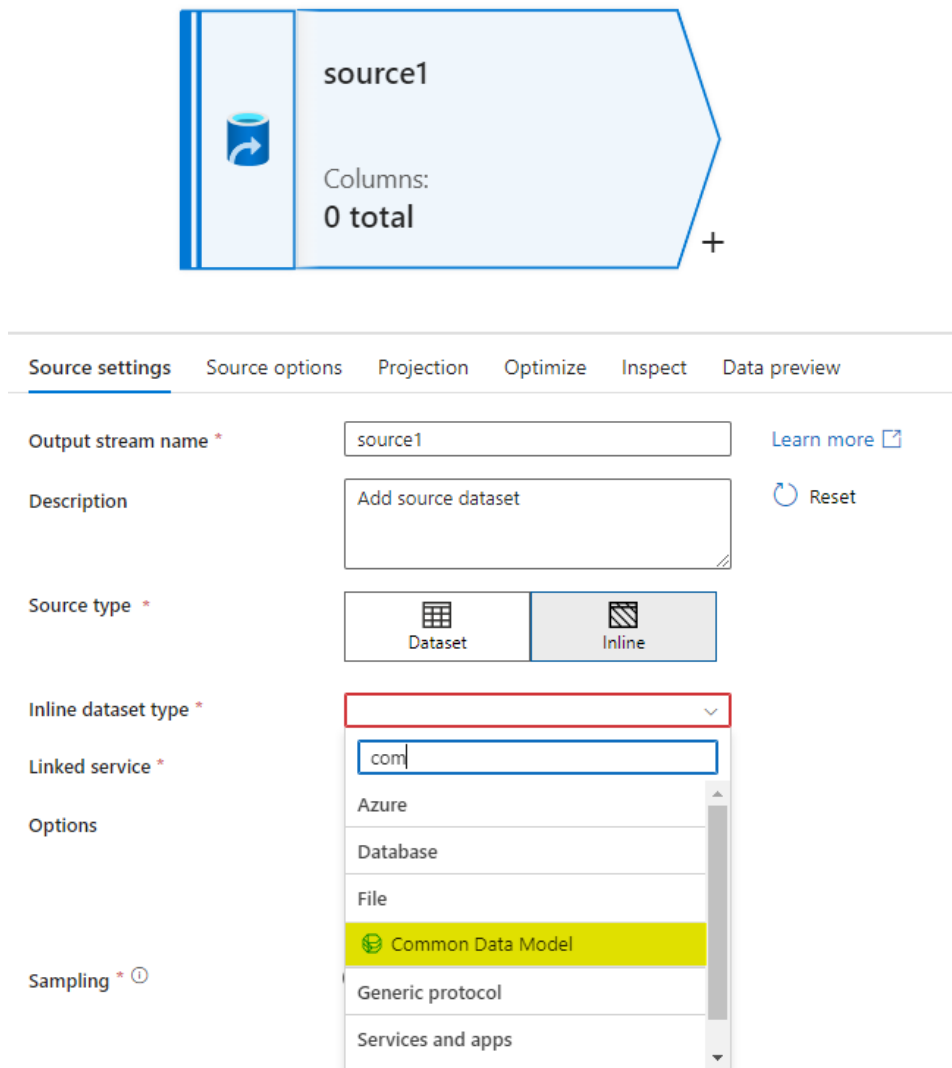
Configuratie van Azure DevOps in Azure Data Factory

De collaboration branch is de enigste branch waarbij de publish knop zichtbaar zal

zijn. Door te werken met feature branches en hiermee pull requests te maken op de collaboration branch kan er dus samen gewerkt worden. De publish branch is de branch waar alle ARM templates van de gepubliceerde factory opgeslaan worden.

### Ophalen van data uit Azure Data Lake

Het ophalen van data uit Data Lake in Data Flow gebeurt steeds op dezelfde manier.



source1

Columns:  
0 total

Source settings Source options Projection Optimize Inspect Data preview

Output stream name \* source1 [Learn more](#)

Description Add source dataset [Reset](#)

Source type \* ☐ Dataset ☒ Inline

Inline dataset type \*

Linked service \*

Options

Sampling \* ⓘ

- Azure
- Database
- File
- Common Data Model**
- Generic protocol
- Services and apps

**Figuur (3.7)**

Configuratie van source transformation

Als source type wordt er steeds gekozen voor “inline”. Dit doordat we slechts werken met één enkele dataflow en geen gedeelde datasets nodig hebben. Als inline data set type kiezen we voor Common Data Model.

**New linked service**

Search

All Azure File

Amazon S3

Azure Data Lake Storage Gen2

Google Cloud Storage

Microsoft Fabric Lakehouse

**New linked service**

Azure Data Lake Storage Gen2 [Learn more](#)

**ⓘ** To avoid publishing immediately to Data Factory, please use Azure Key Vault to retrieve secrets securely. [Learn more here](#)

Name \*

AzureDataLakeStorage3

Description

Connect via integration runtime ⓘ

AutoResolveIntegrationRuntime

Authentication type

Account key

Account selection method ⓘ

☒ From Azure subscription ☐ Enter manually

Azure subscription ⓘ

Visual Studio Enterprise Subscription - MPN - 2024 (46ea3764-ea2f-42ee-9b84-2d47...

Storage account name \*

stdtestbploka

Test connection ⓘ

☒ To linked service ☐ To file path

Annotations

+ New

> Parameters

> Advanced ⓘ

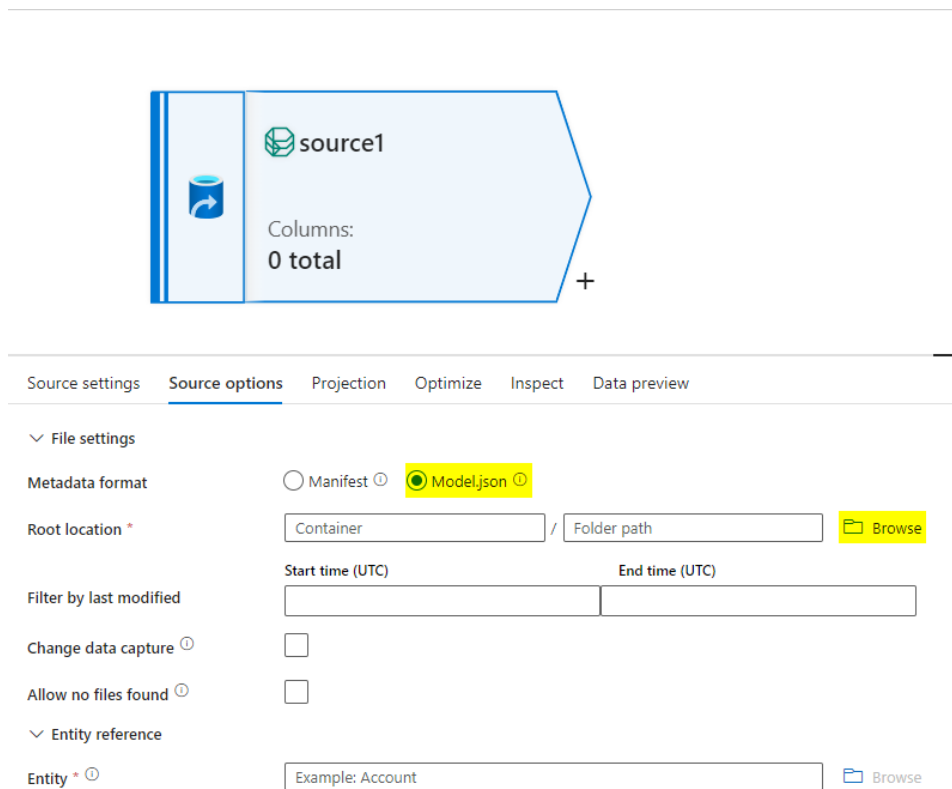
Continue Cancel Create Back Test connection Cancel

**Figuur (3.8)**

Configuratie van linked service

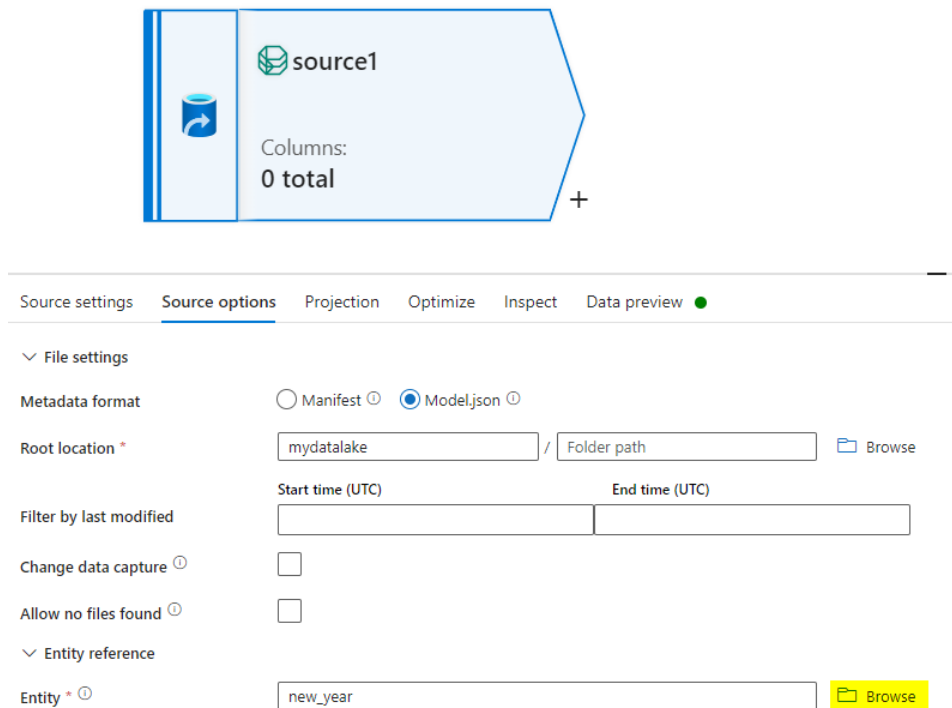
Er zal éénmalig een Linked Service aangemaakt moeten worden. Hierbij kiezen we voor Azure Data Lake Storage Gen2. We kunnen makkelijk gaan koppelen met de juiste data lake door een Azure Subscription en Storage account name aan te duiden. Door op “Test connection” te klikken kunnen we kijken of de connectie met data lake is gelukt. Door op “Create” te klikken hebben we nu een Linked Service die steeds bij elke Source gebruikt kan worden.

**Let op:** Doordat Git geen secrets opslaat is het aanbevolen om gebruik te maken van Azure Key Vault voor het opslaan van connection strings of passwords.

**Figuur (3.9)**

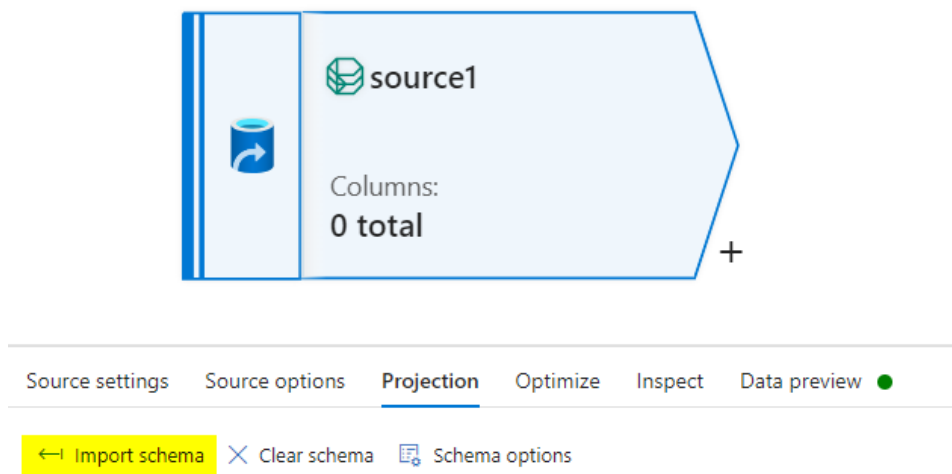
Configuratie van source options

Door naar “Source options” te navigeren kan “Model.json” aangeduid worden. Door hierna op “Browse” te klikken kan er aangeduid worden waar het Model.json bestand te vinden is in data lake. Dit JSON bestand beschrijft hoe de data in Data Lake er uit ziet.

**Figuur (3.10)**

Configuratie van source options

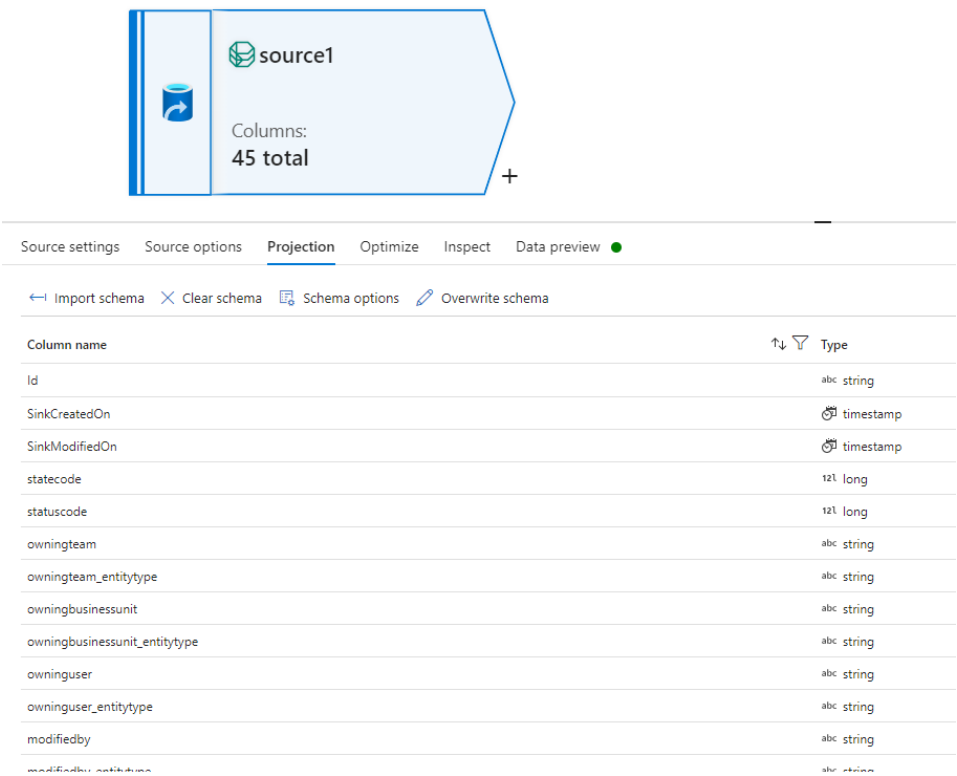
De gewenste entiteit kan nu geselecteerd worden door op “Browse” naast “Entity” te klikken. Let op: hier voor zal Data flow debug aan moeten staan.

**Figuur (3.11)**

Importen van schema voor data flow source

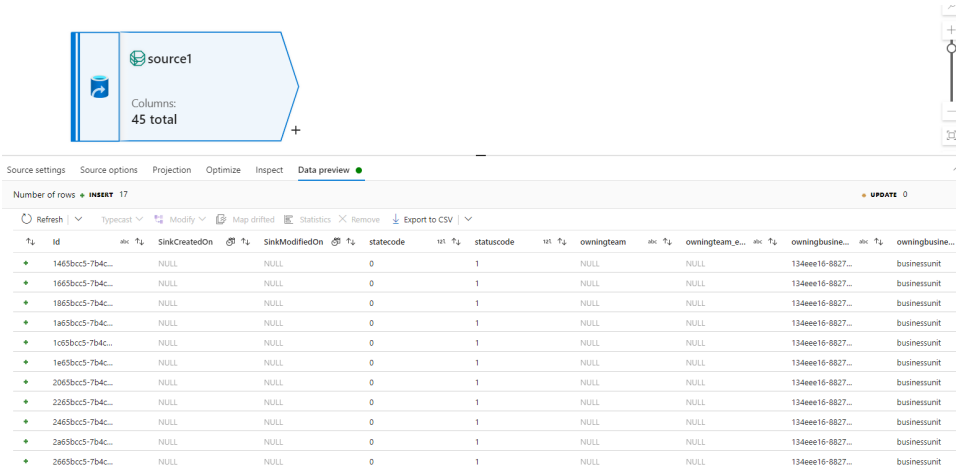
Onder “Projection” kan er nu op “Import schema” geklikt worden om de verschil-

lende kolommen met bijhorende types op te halen.



**Figuur (3.12)**  
Voorbeeld geïmporteerd schema van data flow source

De foto hierboven toont een voorbeeld van een geïmporteerd schema.



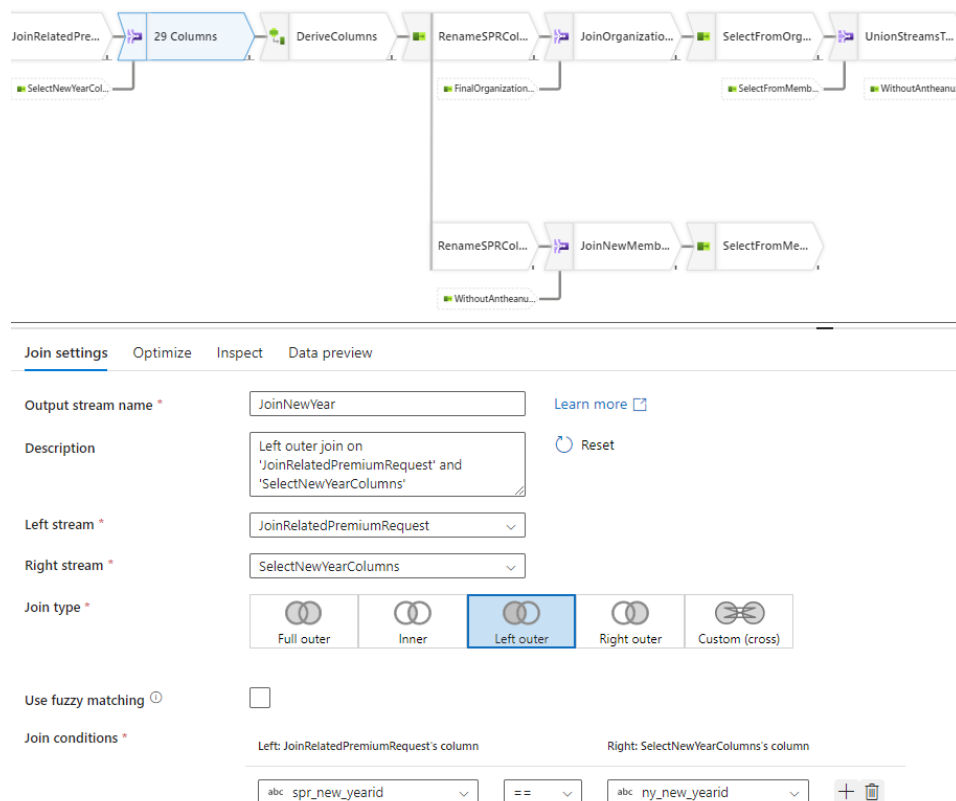
**Figuur (3.13)**  
Data preview in data flow

Door naar “Data preview” te navigeren kan men een preview zien van de data uit de gekozen tabel. Bij elke transformatie die er in data flow gebeurt kan dit preview tab gebruikt worden.



## Belangrijkste transformaties

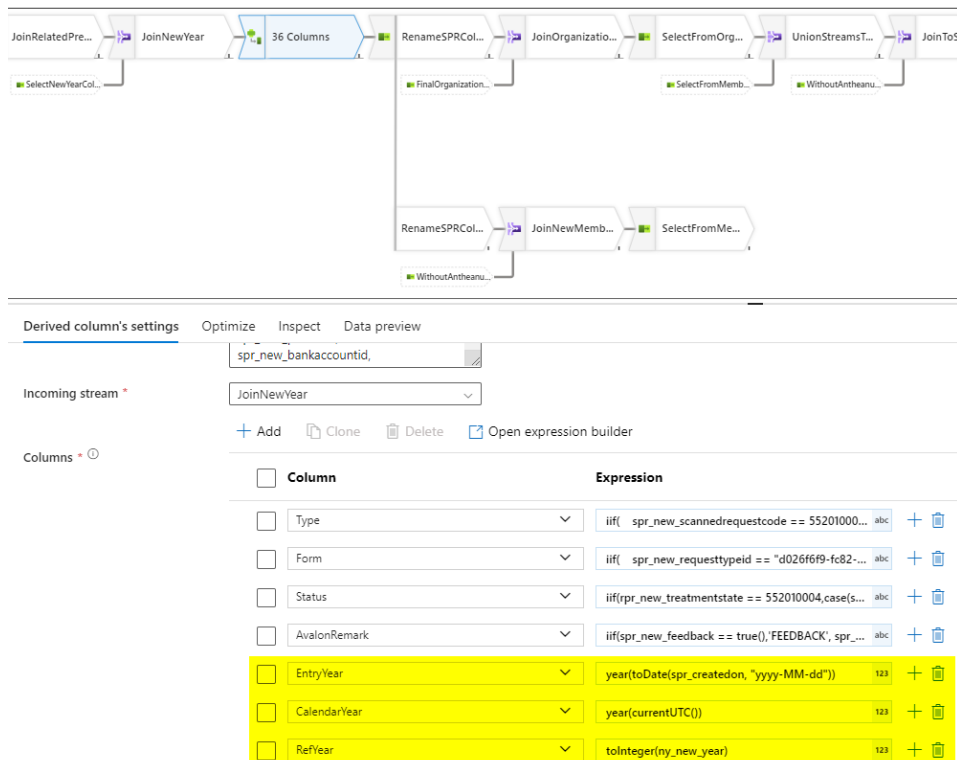
### Determinatie van welke groepen de premie in hun bestand krijgen



**Figuur (3.14)**

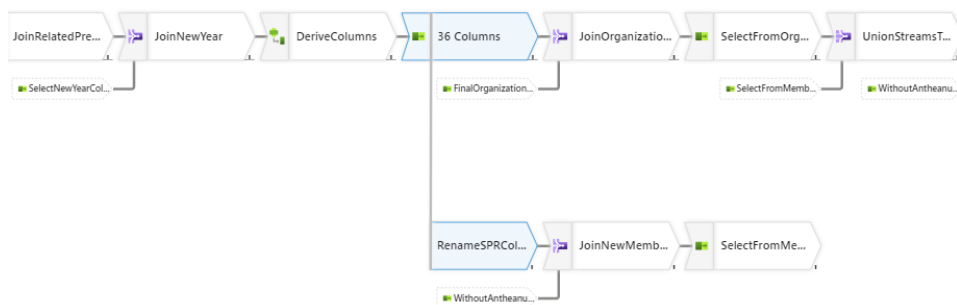
Join van de tabel "new\_year" op de tabel "new\_syndicalpremiumrequest"

De tabel "new\_syndicalpremiumrequest" heeft een kolom "spr\_new\_yearid". Om te gaan bepalen wat het referentiejaar van deze premie is zal dus de tabel "new\_year" op de tabel "new\_syndicalpremiumrequest" gejoind moeten worden aan de hand van dit id.

**Figuur (3.15)**

Derive "EntryYear", "CalendarYear" en "RefYear" op de tabel "new\_syndicalpremiumrequest"

Voor het bepalen van de groepen hebben we 3 nieuwe kolommen nodig. Als eerste hebben we het EntryYear nodig, dit is het jaartal van "spr\_createdon", de datum wanneer de record is aangemaakt. Daarnaast hebben we CalendarYear nodig, dit is het jaartal van de huidige datum. En ten slotte hebben we RefYear nodig, dit is het jaartal van de tabel "new\_year" die net gejoind is geweest.

**Figuur (3.16)**

Hernoemen van kolommen op "new\_syndicalpremiumrequest" en splitsing in twee aparte branches

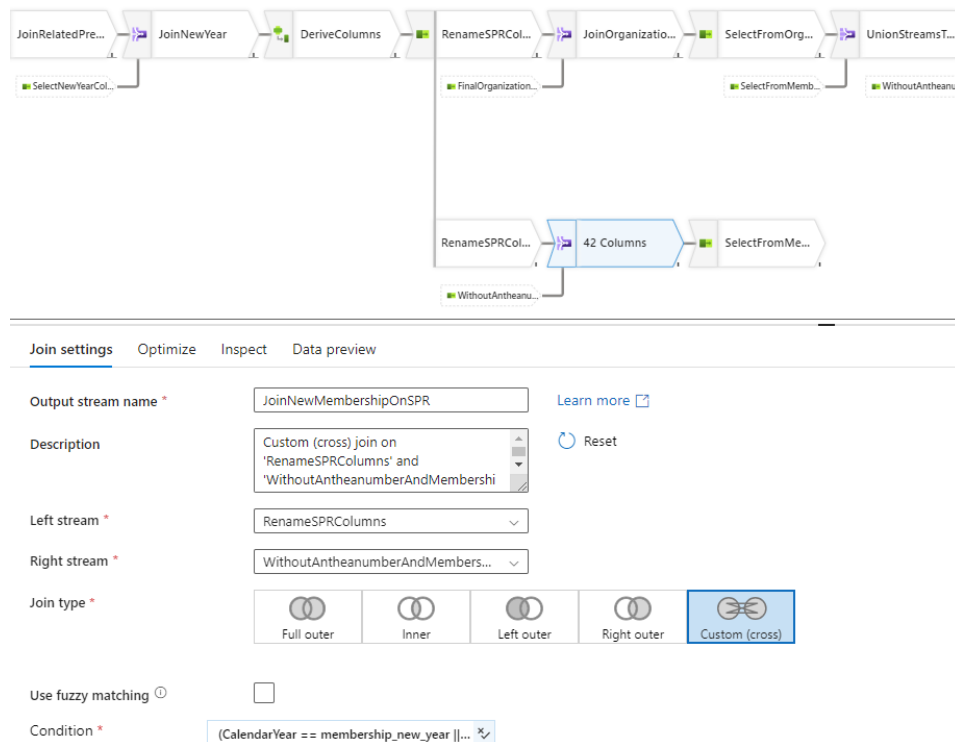
Vervolgens worden bepaalde kolommen van naam hernoemt. Welke kolommen dit zijn is onbelangrijk voor deze transformatie. Wat wel belangrijk is dat de pipeline zich nu opsplijst in twee aparte branches. Dit doordat er 2 inner joins zullen gebeuren.



Er gebeurt nu een inner join van de tabel “new\_organizationyear” op de tabel “new\_syndicalpremium”. Hierbij wordt er aan de hand van IDADM en het id van het referentiejaar gejoind.



Vervolgens worden alle kolommen, die er voor de join waren, geselecteerd. Daarnaast wordt er één kolom “noy\_new\_groupid” geselecteerd en hernoemd naar “Group”.



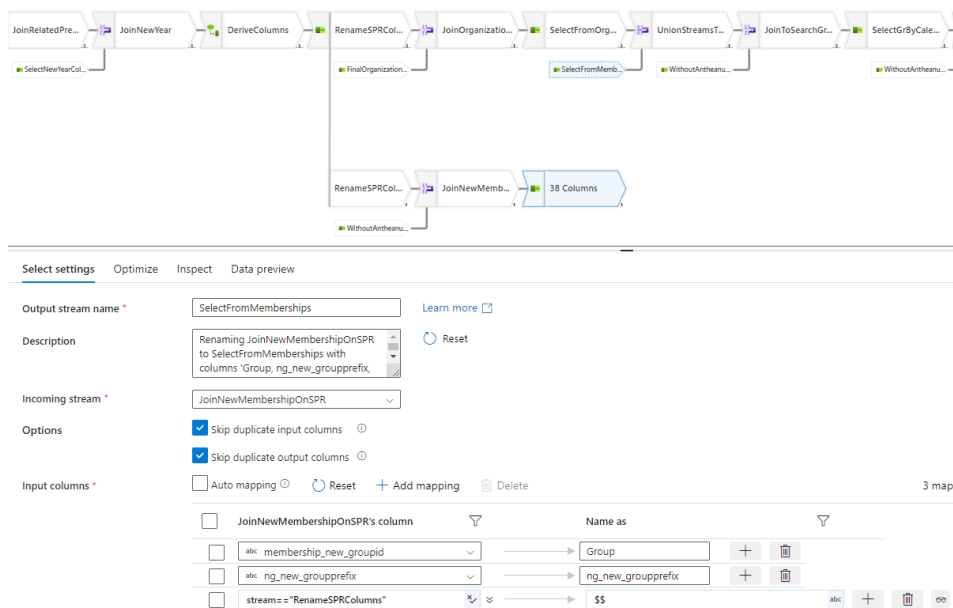
**Figuur (3.19)**

Custom (cross) join van “new\_membership” op de tabel “new\_syndicalpremiumrequest”

Bij de tweede branch wordt de tabel “new\_membership” gejoind op de tabel “new\_syndicalpremiumrequest”. Er wordt gebruik gemaakt van een custom (cross) join doordat er OR condities gebruikt worden. Doordat dit een custom (cross) join is met condities, zal deze join werken net zoals een inner join.

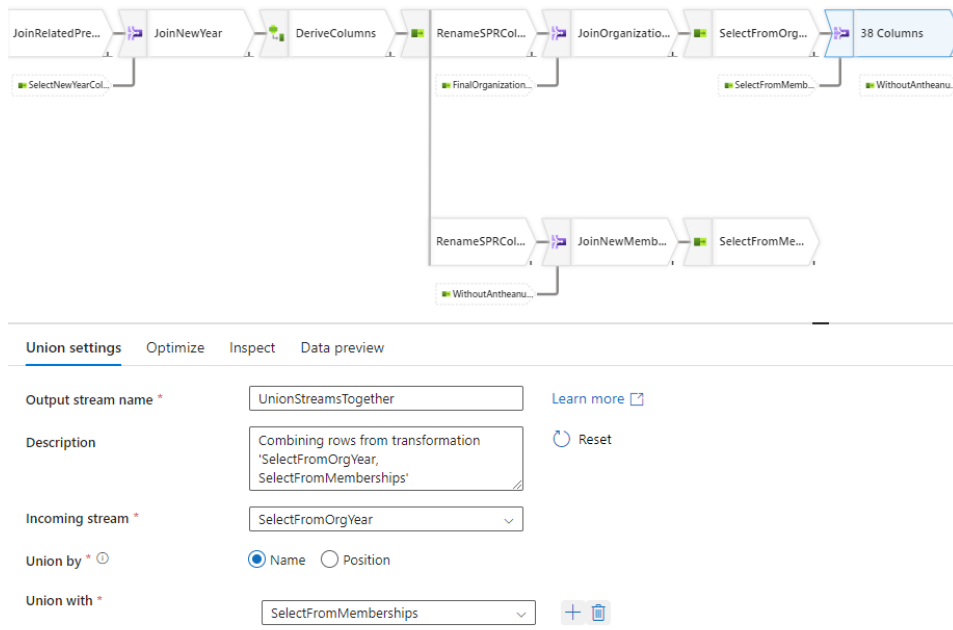
```
1 (CalendarYear = membership_new_year || EntryYear =
    membership_new_year || RefYear = membership_new_year) &&
    spr_new_personid = membership_new_personid
```

In de conditie van de custom (cross) join wordt er vergeleken of CalendarYear, EntryYear of RefYear overeenkomt met het jaartal van de membership. Daarnaast wordt er ook gekeken of personid overeen komt.

**Figuur (3.20)**

Selecteren en hernoemen van kolommen op de tabel “new\_syndicalpremiumrequest”

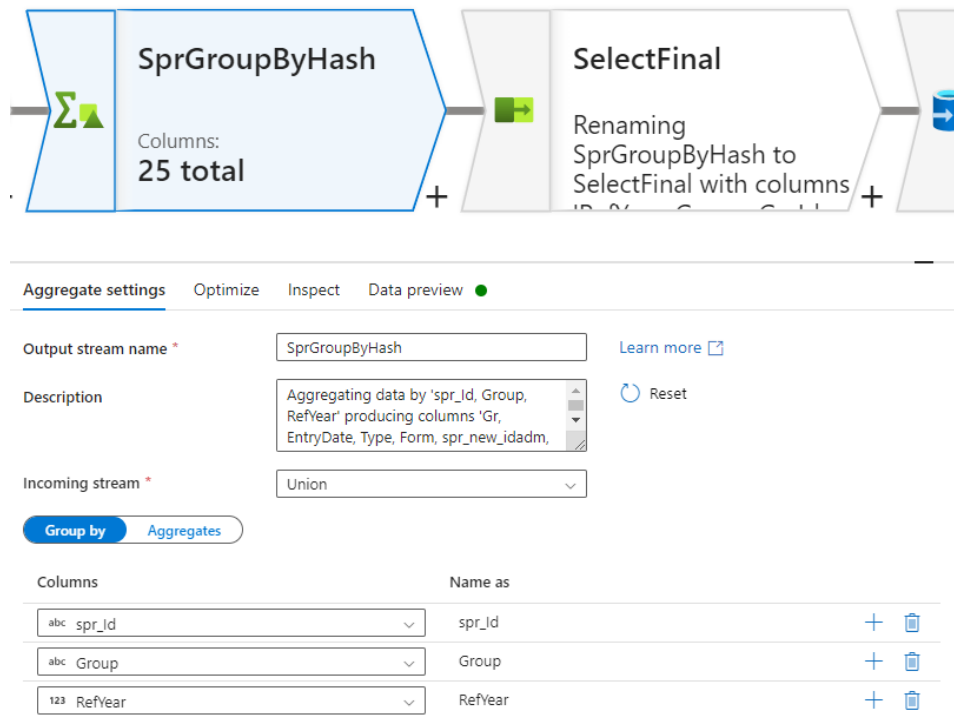
Ook na deze join worden alle kolommen die er voor de join waren geselecteerd. Daarnaast wordt er 1 kolom “membership\_new\_groupid” geselecteerd en hernoemd naar “Group”. Ten slotte wordt de kolom “ng\_new\_groupprefix” geselecteerd maar dit heeft te maken met een andere transformatie.

**Figuur (3.21)**

Union van twee branches

Beide branches hebben nu dezelfde kolommen met een extra kolom “Group”. Daarnaast heeft de onderste branch nog één extra kolom “ng\_new\_groupprefix”. De

beide branches worden nu samen gevoegd met behulp van een union. De bovenste branch die de kolom “ng\_new\_groupprefix” niet heeft zal voor deze kolom de waarde “NULL” krijgen in de records komende van deze branch.



**Figuur (3.22)**

Group by “spr\_Id”, “Group” en “RefYear” in “new\_syndicalpremiumrequest”

Om te voorkomen dat een premie twee keer in het zelfde bestand terecht komt voor een bepaalde groep en referentiejaar zal er op het einde van de pipeline ge-aggregeerd worden op basis van id van de premie, groep en referentiejaar.

Aggregate settings   Optimize   Inspect   Data preview ●

Output stream name \*  [Learn more](#)

Description  [Reset](#)

Incoming stream \*

[Group by](#) [Aggregates](#)

Grouped by: spr\_Id, Group, RefYear

[+ Add](#) [Clone](#) [Delete](#) [Open expression builder](#)

Column	Expression
<input type="checkbox"/> <b>Each column that matches</b>	name != "spr_Id" && name != "Group" && n... creates 1 column(s)
<input type="text" value="\$"/>	<input type="text" value="first(\$\$, true())"/> ANY

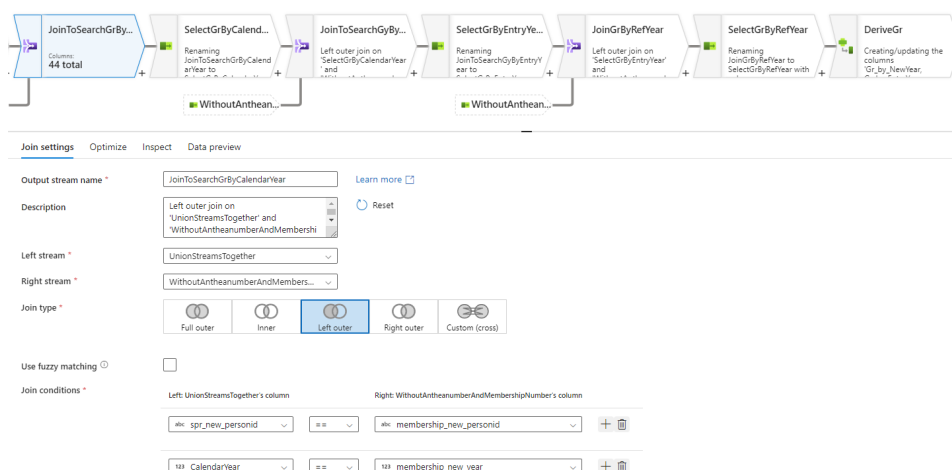
**Figuur (3.23)**

Group by "spr\_Id", "Group" en "RefYear" in "new\_syndicalpremiumrequest"

```
1 name ≠ "spr_Id" && name ≠ "Group" && name ≠ "RefYear"
```

Voor alle kolommen behalve de kolommen die in de "Group by" gebruikt worden zal de aggregatiefunctie "first" gebruikt worden. De tweede parameter "true()" wordt gebruikt om aan te geven dat "NULL" waardes genegeerd moeten worden. Dit zal dus resulteren in de eerste waarde die niet "NULL" is van de kolom groep. Indien alle waardes "NULL" zijn zal dit wel in "NULL" resulteren.

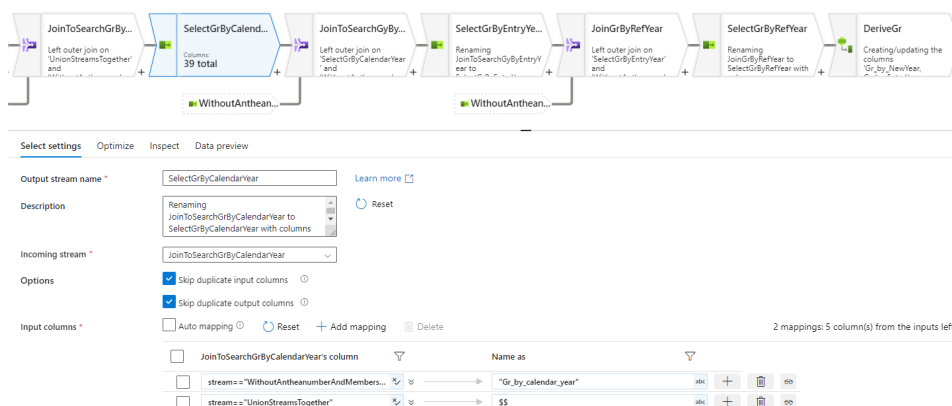
## Bepalen van de kolom “Gr” voor een premie



**Figuur (3.24)**

Joinen van de tabel “new\_membership” op de tabel “new\_syndicalpremiumrequest”

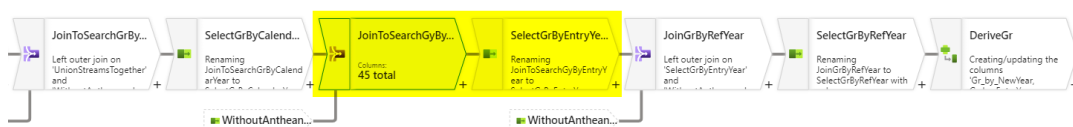
De tabel “new\_membership” wordt gejoind op de tabel “new\_syndicalpremiumrequest”. Dit gebeurt aan de hand van “new\_personid” en het kalenderjaar van de premie.



**Figuur (3.25)**

Selecteren van de juiste kolommen in “new\_syndicalpremiumrequest”

De kolommen die reeds bestonden voor de join worden geselecteerd. Daarnaast wordt de kolom “ng\_new\_groupprefix” geselecteerd en hernoemd naar “Gr\_by\_calendar\_year”.



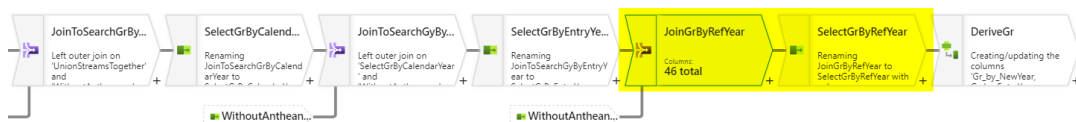
**Figuur (3.26)**

Herhalen van transformaties

Dezelfde transformaties gebeuren opnieuw maar deze keer aan de hand van “EntryYear”. Ook hier worden de reeds bestaande kolommen geselecteerd en wordt



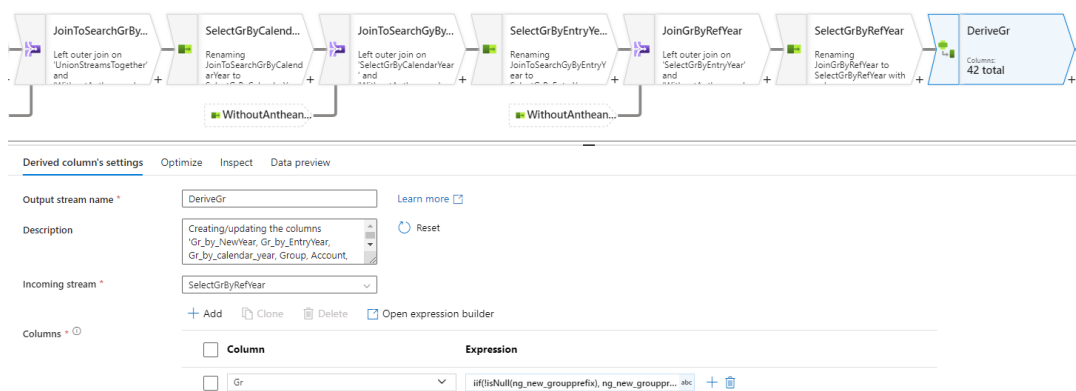
de nieuwe kolom “ng\_new\_groupprefix” hernoemd naar “Gr\_by\_EntryYear”.



**Figuur (3.27)**

Herhalen van transformaties

Ook voor “RefYear” zullen deze transformaties gebeuren, resulterend in een nieuwe kolom “Gr\_by\_RefYear”.



**Figuur (3.28)**

Bepalen van “Gr” in “new\_syndicalpremiumrequest”

We hebben nu een group prefix reeds komende uit een andere transformatie 3.20 en group prefixes komende uit de joins die we hebben uitgevoerd.

```

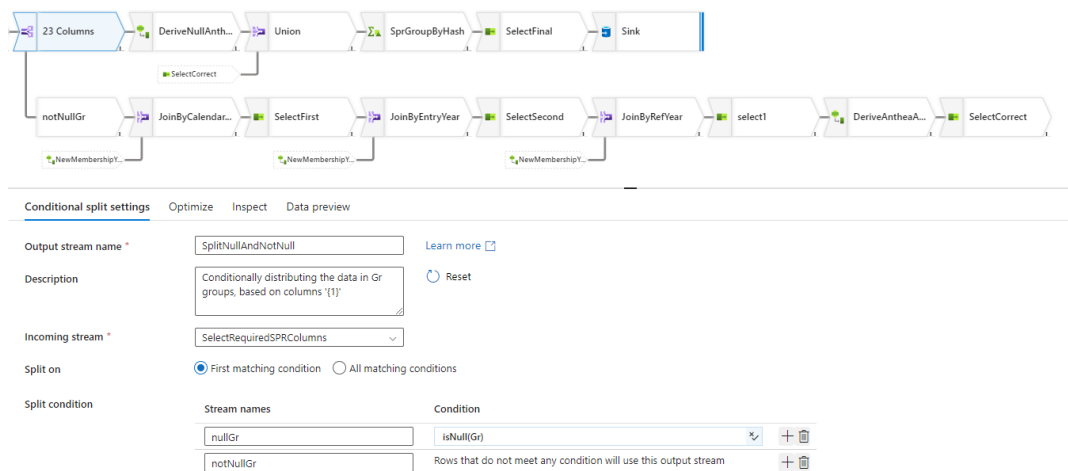
1 iif(!isNull(ng_new_groupprefix), ng_new_groupprefix,
2   iif(!isNull(Gr_by_calendar_year), Gr_by_calendar_year,
3     iif(!isNull(Gr_by_EntryYear), Gr_by_EntryYear,
4       iif(!isNull(Gr_by_NewYear), Gr_by_RefYear,
         toString(null())))))

```

Om de kolom “Gr” te bepalen zal er gezocht worden naar de eerste kolom die niet “NULL” is startende van uit “ng\_new\_groupprefix” waarbij er vervolgens gekeken wordt naar “Gr\_by\_calendar\_year”, “Gr\_by\_EntryYear” en “Gr\_by\_NewYear”.

Ook voor deze transformatie is het belangrijk dat er geaggregeerd wordt zoals bij Figuur 3.22 om te voorkomen dat dezelfde premie twee keer in een groep voor een bepaald jaartal terecht komt.

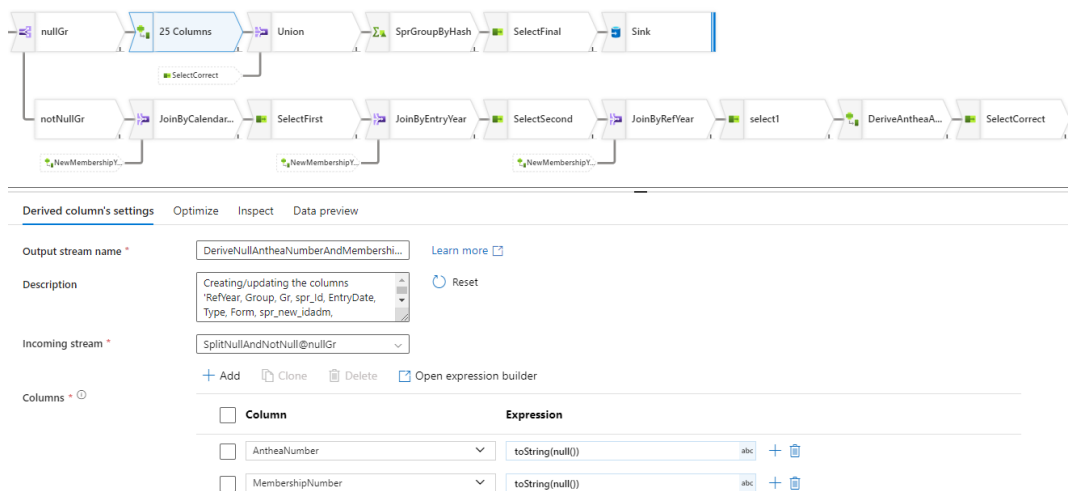
## Bepalen van de kolommen “AntheaNumber” en “MembershipNumber” voor een premie



**Figuur (3.29)**

Conditional split op “new\_syndicalpremiumrequest”

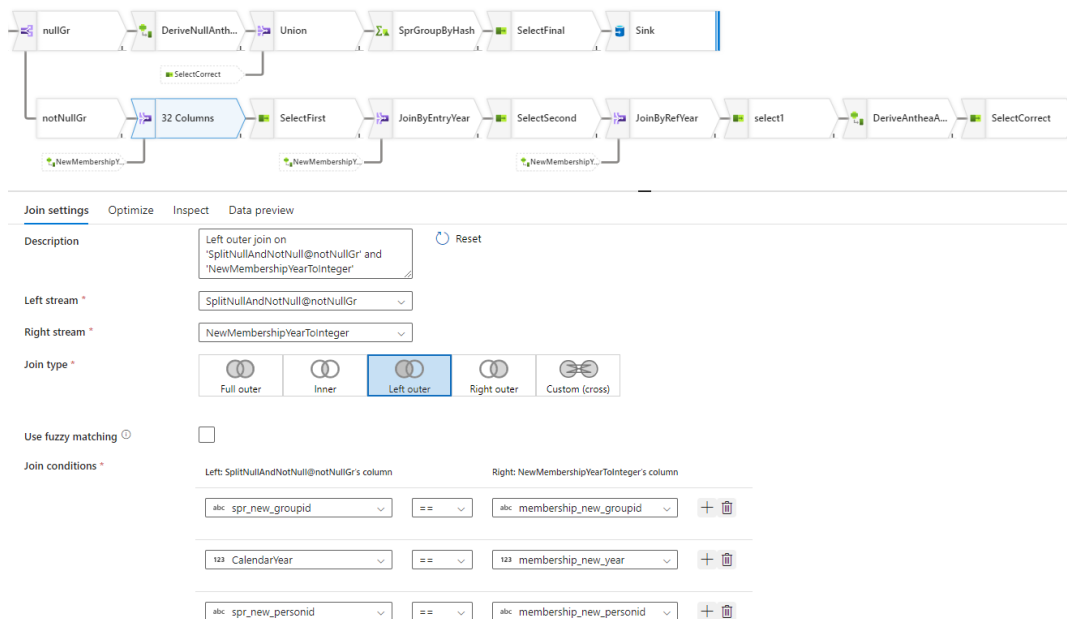
De pipeline zal splitten in twee branches met behulp van een conditional split. Alle rijen waarbij de kolom “Gr”, “NULL” is, zullen boven verder gaan. Alle rijen waarbij de kolom “Gr”, niet “NULL”, is zullen onderaan verder gaan.



**Figuur (3.30)**

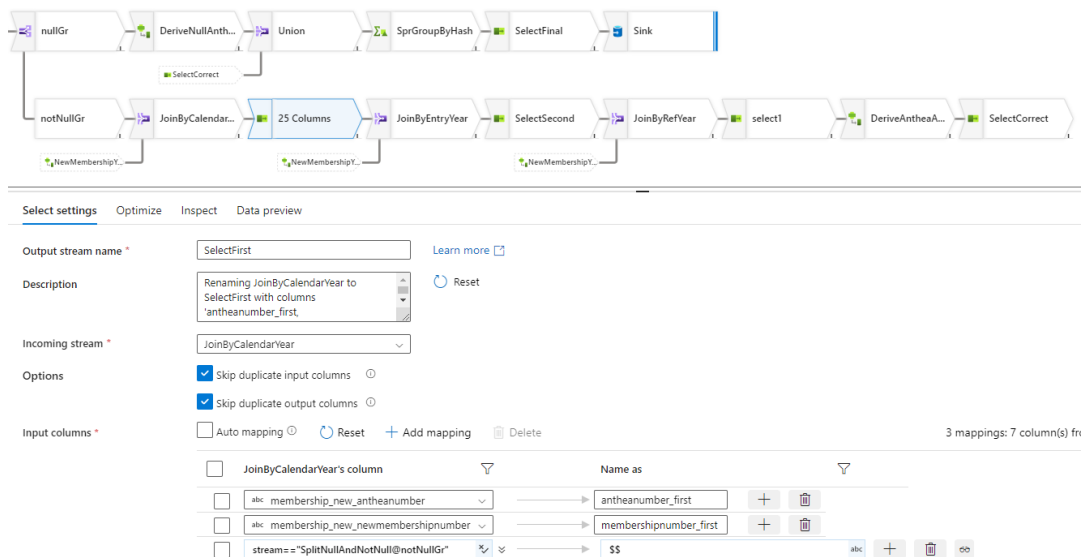
Derive “AntheaNumber” en “MembershipNumber” op de tabel “new\_syndicalpremiumrequest”

Voor de rijen waarbij de kolom “Gr”, “NULL” was, zullen de kolommen “AntheaNumber” en “MembershipNumber” ook NULL zijn.

**Figuur (3.31)**

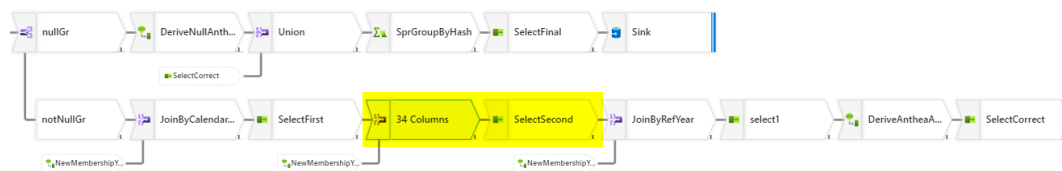
Join van de tabel “new\_membership” op de tabel “new\_syndicalpremiumrequest” aan de hand van “new\_groupid”, “new\_personid” en “CalendarYear”

Voor de rijen waarbij de kolom “Gr” niet “NULL” was zal de tabel “new\_membership” gejoind worden. Hierbij wordt er gebruik gemaakt van “group\_id”, “new\_personid” en “CalendarYear”.

**Figuur (3.32)**

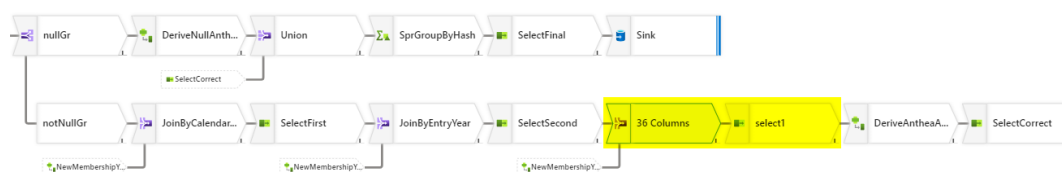
Select op de tabel “new\_syndicalpremiumrequest”

Alle kolommen die er voor de join waren zullen opnieuw geselecteerd worden. Daarnaast worden “membership\_new\_antheanumber” en “membership\_new\_newmembershipnumber” geselecteerd en hernoemd naar “antheanumber\_first” en “membershipnumber\_first”.

**Figuur (3.33)**

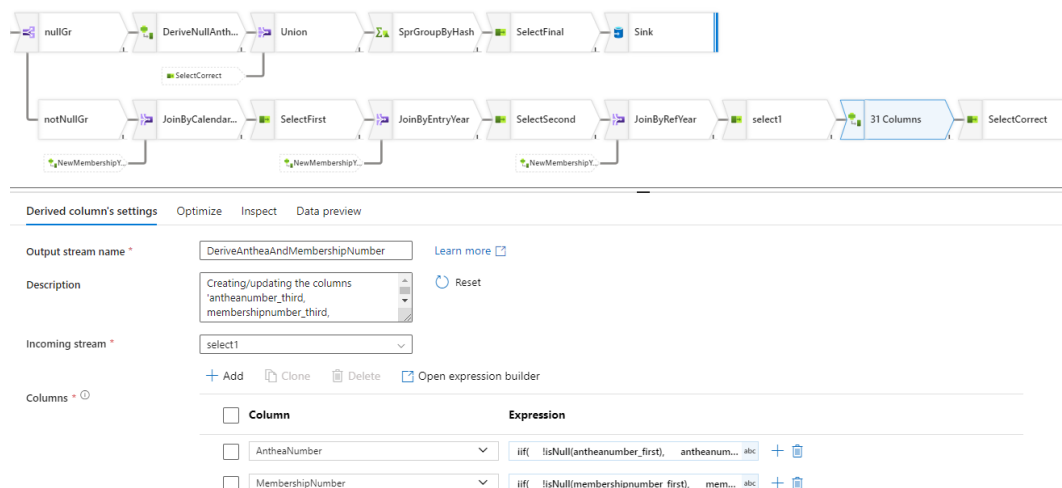
Join van de tabel “new\_membership” op de tabel “new\_syndicalpremiumrequest” aan de hand van “new\_groupid”, “new\_personid” en “EntryYear”

De vorige twee transformatie gebeuren opnieuw maar deze keer wordt er gebruik gemaakt van “EntryYear” in plaats van “CalendarYear”. Daarnaast worden de kolommen “membership\_new\_antheanumber” en “membership\_new\_newmembershipnumber” deze keer hernoemd naar “antheanumber\_second” en “membershipnumber\_second”.

**Figuur (3.34)**

Join van de tabel “new\_membership” op de tabel “new\_syndicalpremiumrequest” aan de hand van “new\_groupid”, “new\_personid” en “RefYear”

Ook zal voor “RefYear” in plaats van “CalendarYear” deze transformaties opnieuw gebeuren. Deze keer met de kolommen “antheanumber\_third” en “membershipnumber\_third” als resultaat.

**Figuur (3.35)**

Derive van kolommen “AntheaNumber” en “MembershipNumber” op de tabel “new\_syndicalpremiumrequest”

De kolommen “AntheaNumber” en “MembershipNumber” worden nu berekent.

```

1 iif(!isNull(antheanumber_first), antheanumber_first,
2   iif(!isNull(antheanumber_second), antheanumber_second,
3     iif(!isNull(antheanumber_third), antheanumber_third,
        toString(null()))))

```

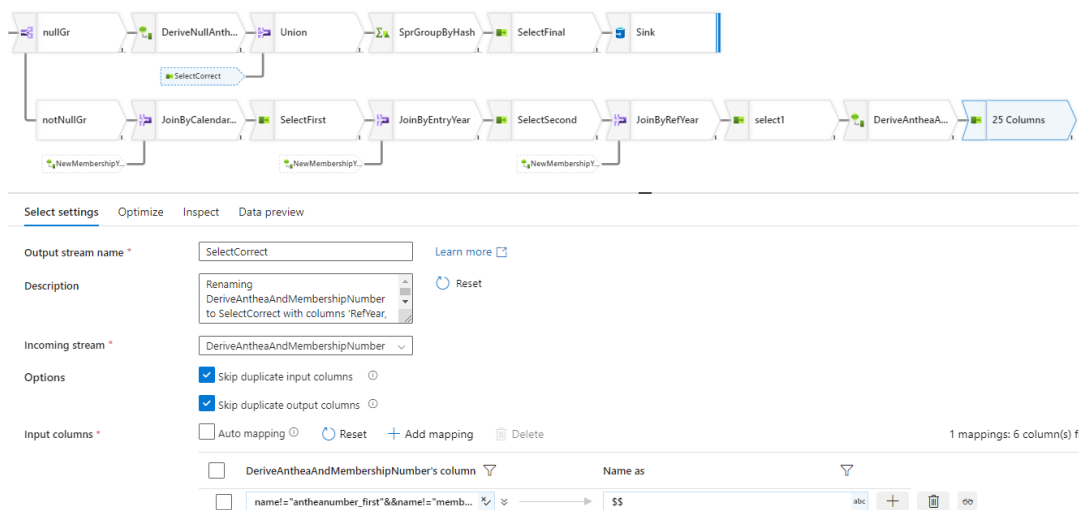
Wanneer het antheanummer van de eerste join niet “NULL” is zal deze gebruikt worden, anders zal er gekeken worden of de tweede niet “NULL” is. Als de tweede “NULL” is zal de derde gebruikt worden. Indien de derde ook “NULL” is zal dit resulteren in “NULL” als waarde voor de kolom “AntheaNumber”.

```

1 iif(!isNull(membershipnumber_first), membershipnumber_first,
2   iif(!isNull(membershipnumber_second), membershipnumber_second,
3     iif(!isNull(membershipnumber_third), membershipnumber_third,
        toString(null()))))

```

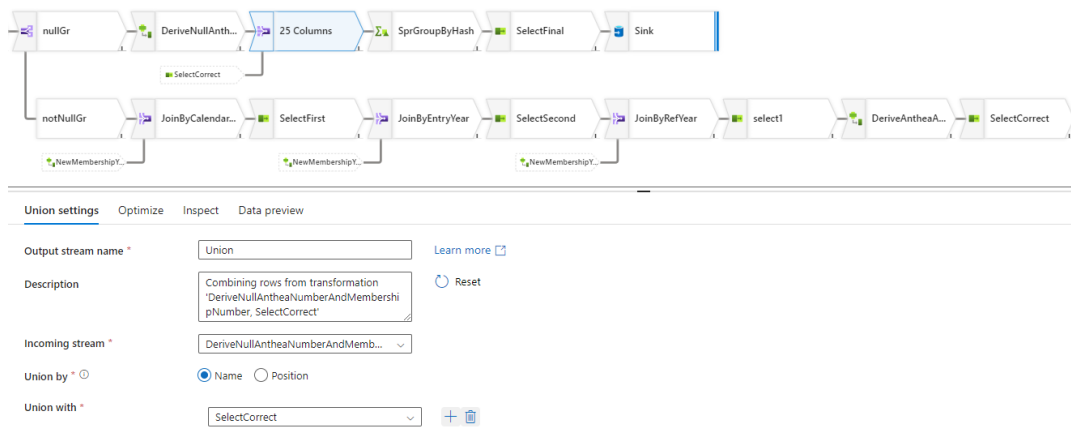
Ook de kolom “MembershipNumber” wordt op dezelfde manier berekent.



**Figuur (3.36)**

Verwijderen van onnodige kolommen op de tabel “new\_syndicalpremiumrequest”

De kolommen die niet nodig zijn worden verwijderd.

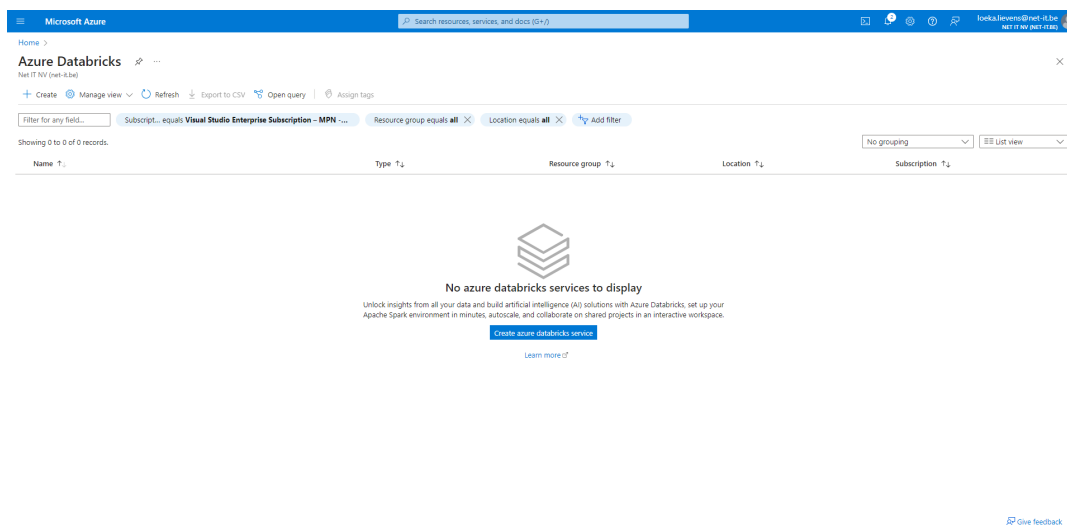
**Figuur (3.37)**

Union van twee streams

Beide streams worden nu samen gevoegd met behulp van een union. En ten slotte is het ook voor deze transformatie belangrijk dat er geaggregeerd wordt zoals bij Figuur 3.22 om te voorkomen dat dezelfde premie twee keer in een groep voor een bepaald jaartal terecht komt.

### 3.5.2. Azure Databricks

#### Opzet van resources

**Figuur (3.38)**

Aanmaken van Azure Databricks

Door in Microsoft Azure naar Databricks te navigeren kan er een nieuwe Azure Databricks workspace aangemaakt worden.

[Basics](#) [Networking](#) [Encryption](#) [Security & compliance](#) [Tags](#) [Review + create](#)

**Project Details**

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

Visual Studio Enterprise Subscription – MPN - Spoke 2

Resource group \* ⓘ

LoekaBP

[Create new](#)

**Instance Details**

Workspace name \*

databricks-loeka-standard

Region \*

West Europe

Pricing Tier \* ⓘ

Standard (Apache Spark, Secure with Microsoft Entra ID)


Managed Resource Group name


databricks-loeka-standard-mrg

**Figuur (3.39)**

Configuratie van Azure Databricks

Bij het aanmaken van databricks moet er een subscription en resource group gekozen worden. Er kan een nieuwe resource group aangemaakt worden of een reeds bestaande gekozen worden. Daarnaast moet er een naam, gewenste regio en pricing tier gekozen worden. Als pricing tier kiezen we hier voor “Standard” doordat we geen gebruik gaan maken van Premium features. Ten slotte kan er ook een Managed Resource Group name gekozen worden. Deze resource group houdt alle resource bij die databricks nodig heeft, zoals bijvoorbeeld virtual machines, storage accounts en virtual networks.

 **Your deployment is complete**


 Deployment name : LoekaBP\_databricks-loeka-standard  
Subscription : [Visual Studio Enterprise Subscription – MPN - Spoke 2](#)  
Resource group : [LoekaBP](#)

> Deployment details

∨ Next steps

[Go to resource](#)

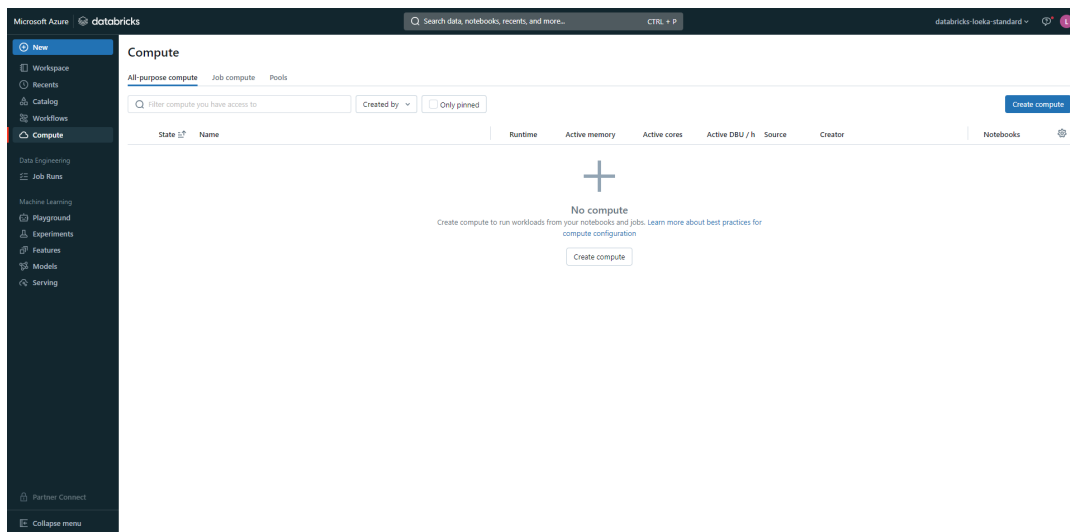
[Give feedback](#)

 [Tell us about your experience with deployment](#)

**Figuur (3.40)**

Deployment complete van Azure Databricks

Wanneer de resource is aangemaakt kan Azure Databricks opgestart worden.



**Figuur (3.41)**

Aanmaken van compute resource

Voordat notebooks en jobs uitgevoerd kunnen worden zal er eerst een compute resource aangemaakt moeten worden.

Compute > New compute >

### Loeka Lievens's Cluster [✎](#)

☒ Multi node ☐ Single node

Access mode ⓘ ☐ Single user access ⓘ

Single user ⓘ

#### Performance

Databricks runtime version ⓘ

Runtime: 11.3 LTS (Scala 2.12, Spark 3.3.0) ⓘ

☒ Use Photon Acceleration ⓘ

Worker type ⓘ Min workers Max workers

Standard\_DS3\_v2 14 GB Memory, 4 Cores ⓘ 2 8 ☐ Spot instances ⓘ

Driver type

Same as worker 14 GB Memory, 4 Cores ⓘ

☒ Enable autoscaling ⓘ

☒ Terminate after 60 minutes of inactivity ⓘ

#### Tags ⓘ

Add tags

Key Value Add

> Automatically added tags

► Advanced options

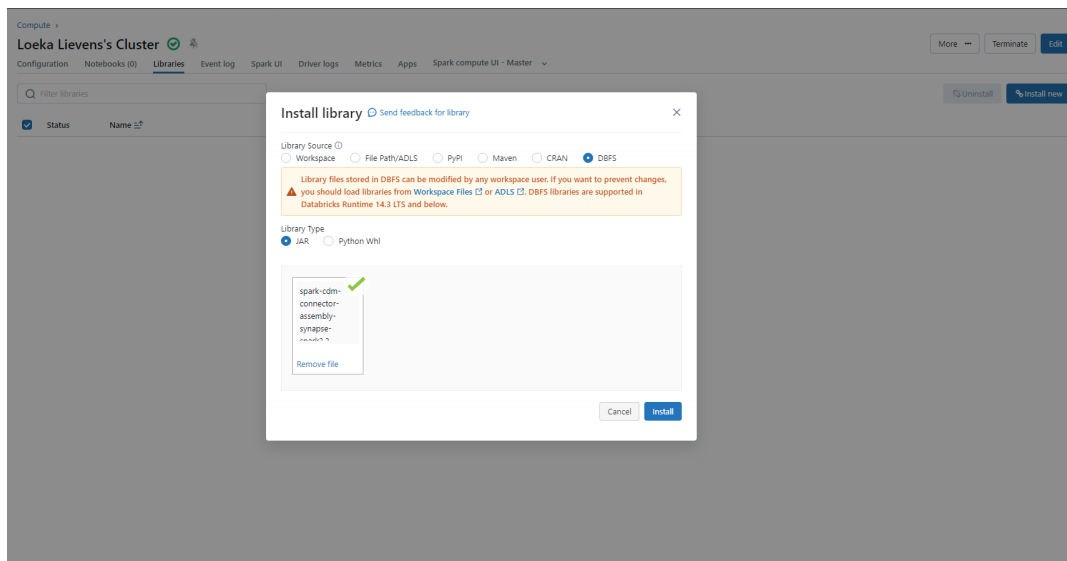
Create compute Cancel

**Figuur (3.42)**

Configuratie van compute resource



De databricks runtime version moet op 11.3 LTS gezet worden zodat we Apache Spark 3.3.0 kunnen gebruiken. Dit omdat we gebruik gaan maken van “spark-cdm-connector”. Ook hebben we ingesteld dat de cluster zichzelf zal uitschakelen na 60 minuten om onnodige kosten te voorkomen.

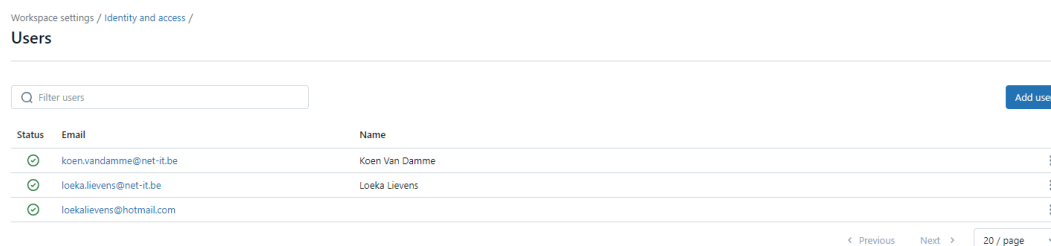


**Figuur (3.43)**

Installatie van “spark-cdm-connector”

Ten slotte moet de [jar file](#) van “spark-cdm-connector” geïnstalleerd worden in het aangemaakte compute resource om gebruik te kunnen maken van het Common Data Model in onze pipeline.

## Collaboration en source control Management en permissions



**Figuur (3.44)**

Gebruikers toevoegen/verwijderen in Azure Databricks

Gebruikers die behoren tot het AAD directory van de Azure Databricks environment kunnen makkelijk toegevoegd worden met behulp van het e-mailadres van de gebruiker.

Workspace settings / Identity and access / Users /

### Edit user

---

Email loekalievens@hotmail.com

---

[Remove user](#)

---

### Entitlements

**Admin access**  
Can manage this workspace and its users, groups, resources, and settings On ☒

---

**Workspace access**  
Can access data engineering and ML environments On ☐

---

**Databricks SQL access**  
Can access SQL environment On ☐

---

**Unrestricted cluster creation**  
Can create clusters; when disabled, the user is restricted to access granted by cluster policies On ☐

---

**Figuur (3.45)**

Rechten wijzigen van gebruikers in Azure Databricks

Rechten van gebruikers kunnen gewijzigd worden.

Workspace settings / Identity and access /

### Groups

[Add group](#)

Name	Members	Source
<a href="#">admins</a>	3	System ⓘ
<a href="#">users</a>	3	System ⓘ

< Previous   Next >   20 / page

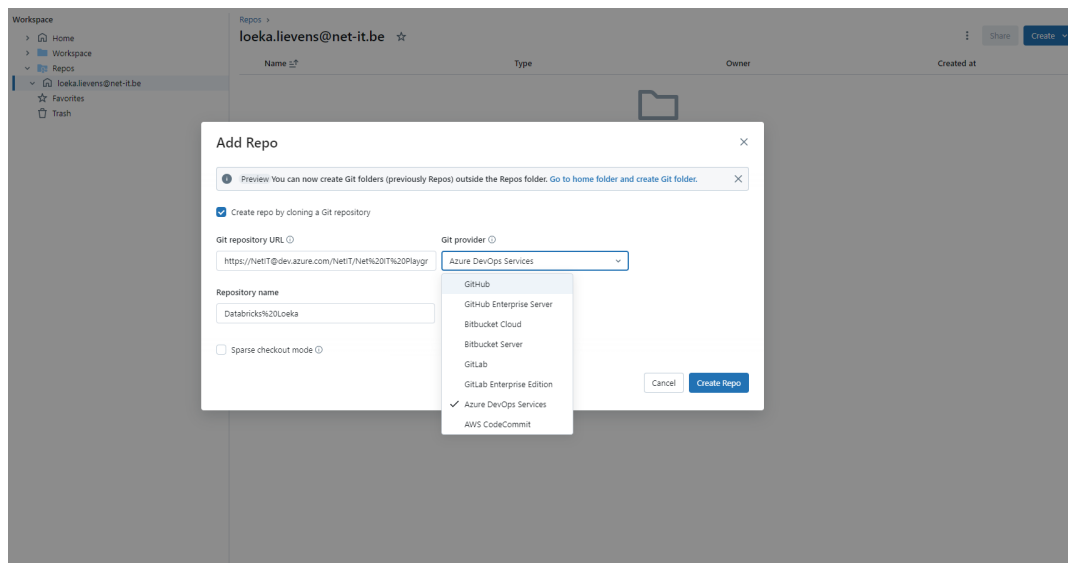
**Figuur (3.46)**

Groepen toevoegen of verwijderen in Azure Databricks

Er kunnen groepen toegevoegd of verwijderd worden aan Azure Databricks. Gebruikers kunnen dan aan deze groepen toegevoegd worden. Dit vereenvoudigt het om toegang toe te wijzen aan werkruimten, gegevens en andere objecten.

## Source control

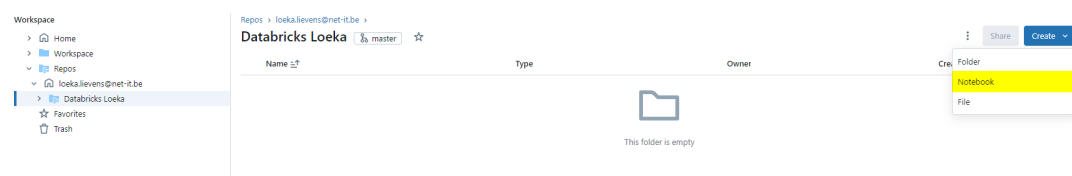
Databricks folders is een visuele Git client binnen Azure Databricks om gebruik te kunnen maken van source control.



**Figuur (3.47)**

Clone Git repository in Databricks folders

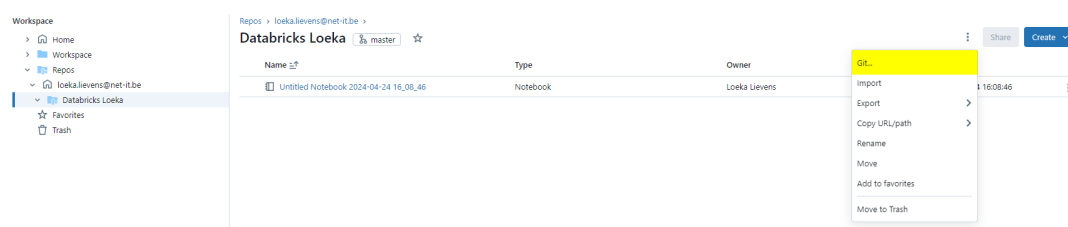
Het geeft de optie om verschillende Git providers te gaan gebruiken. Doordat er binnen Net IT gewerkt wordt met Azure DevOps Services zal hier voor gekozen worden.



**Figuur (3.48)**

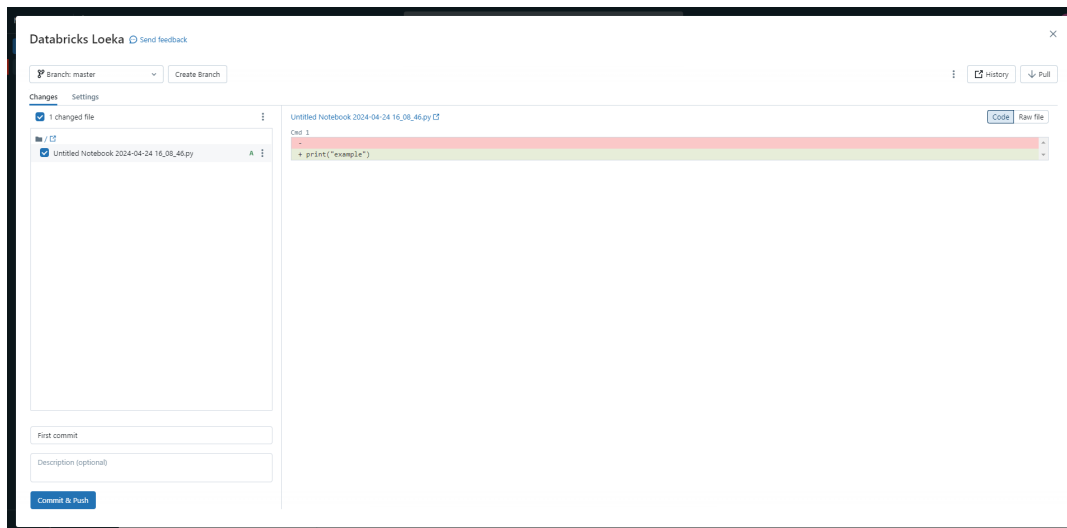
Aanmaken van een notebook in databricks

Ter illustratie zal er een voorbeeld notebook aangemaakt worden.



**Figuur (3.49)**

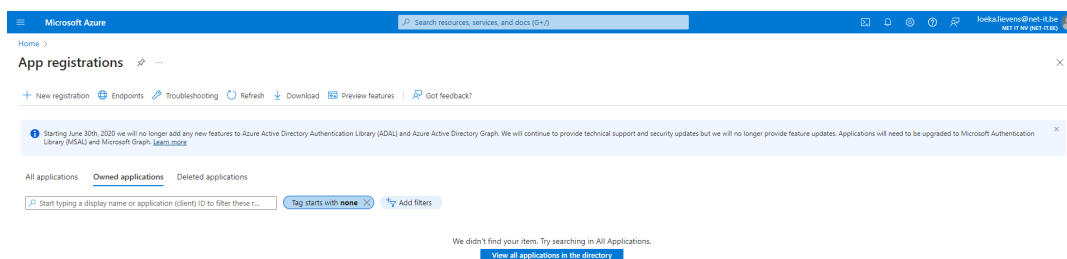
Commit en push in Databricks

**Figuur (3.50)**

Commit en push in Databricks

Deze notebook kan dan gecommited worden naar Git.

## Ophalen van data uit Azure Data Lake

**Figuur (3.51)**

App registration aanmaken

## Register an application ...

### \* Name

The user-facing display name for this application (this can be changed later).

bp-loeka-2 ✓

### Supported account types

Who can use this application or access this API?

- ☒ Accounts in this organizational directory only (Net IT NV only - Single tenant)
- ☐ Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant)
- ☐ Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)
- ☐ Personal Microsoft accounts only

[Help me choose...](#)

### Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Select a platform ▼

e.g. <https://example.com/auth>

Register an app you're working on here. Integrate gallery apps and other apps from outside your organization by adding from [Enterprise applications](#).

By proceeding, you agree to the [Microsoft Platform Policies](#) ↗

Register

### Figuur (3.52)

Configuratie app registratie

Door in Microsoft Azure naar “App registrations” te navigeren kan er een nieuwe app registration aanmaken.

## Add a client secret



Description

secret

Expires

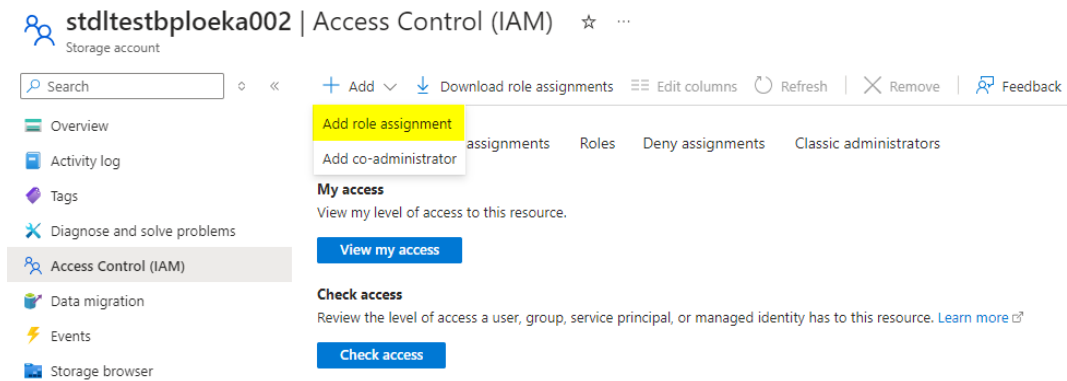
730 days (24 months) ▼

### Figuur (3.53)

Aanmaken client secret

Door naar “Certificates & secrets” in de app registration te navigeren kan er een client secret aangemaakt worden. Het is belangrijk om deze client secret op te

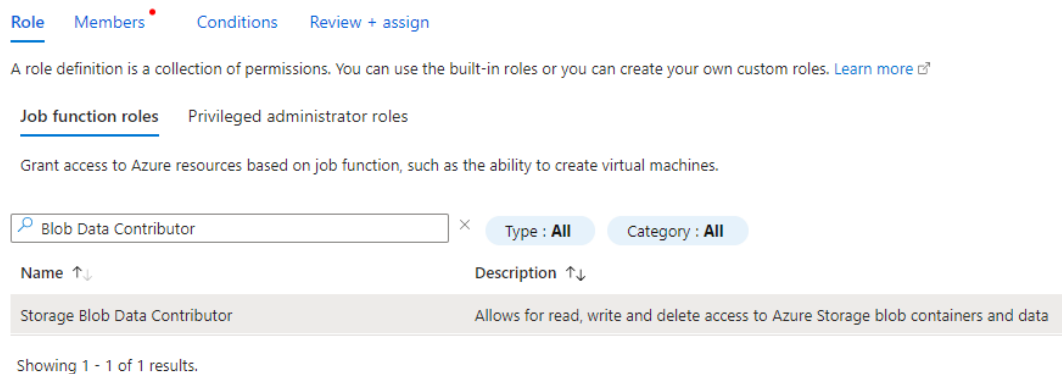
slaan.



**Figuur (3.54)**

Role assignment toevoegen aan storage account

Door naar “Access Control (IAM)” te navigeren van het storage account waarmee we een connectie willen maken, kunnen we nu een role assignment gaan toevoegen.



**Figuur (3.55)**

Job function role kiezen voor role assignment

Als job function role moet er voor “Storage Blob Data Contributor” gekozen worden.

Role **Members** Conditions Review + assign

Showing a filtered list of roles because your permissions include a condition. [Learn more](#)  
[View my access](#)

**Selected role** Storage Blob Data Contributor

**Assign access to**  
☒ User, group, or service principal  
☐ Managed identity

**Members** + Select members

Name	Object ID	Type
bp-loeka-2	6a06fcfb-1331-42d6-975b-8bec1b86928a	App

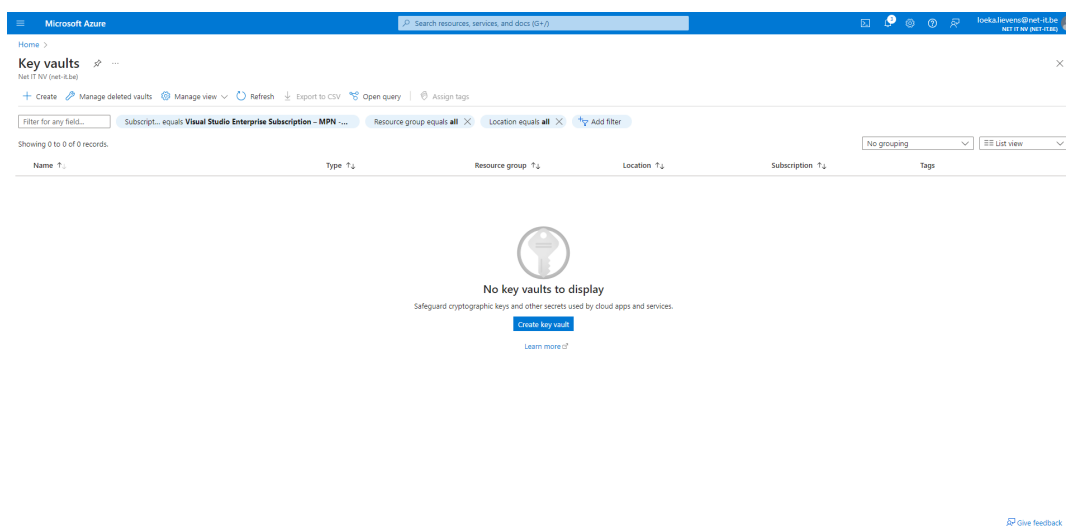
**Description**

Optional

**Figuur (3.56)**

Members kiezen voor role assignment

Bij “Members” kiezen we voor de service principal die we net hebben aangemaakt. Dit zorgt er voor dat de app registration die we net hebben aangemaakt toegang heeft tot het data lake.

**Figuur (3.57)**

Key vault aanmaken

Doordat we de client secret die we net hebben aangemaakt niet in onze databricks code willen zetten zal er een key vault aangemaakt worden. Dit kan door in Microsoft Azure naar “Key vaults” te navigeren.

[Basics](#) [Access configuration](#) [Networking](#) [Tags](#) [Review + create](#)

Azure Key Vault is a cloud service used to manage keys, secrets, and certificates. Key Vault eliminates the need for developers to store security information in their code. It allows you to centralize the storage of your application secrets which greatly reduces the chances that secrets may be leaked. Key Vault also allows you to securely store secrets and keys backed by Hardware Security Modules or HSMs. The HSMs used are Federal Information Processing Standards (FIPS) 140-2 Level 2 validated. In addition, key vault provides logs of all access and usage attempts of your secrets so you have a complete audit trail for compliance.

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \*

Resource group \*  [Create new](#)

Instance details

Key vault name \*  ✓

Region \*

Pricing tier \*

Recovery options

Soft delete protection will automatically be enabled on this key vault. This feature allows you to recover or permanently delete a key vault and secrets for the duration of the retention period. This protection applies to the key vault and the secrets stored within the key vault.

To enforce a mandatory retention period and prevent the permanent deletion of key vaults or secrets prior to the retention period elapsing, you can turn on purge protection. When purge protection is enabled, secrets cannot be purged by users or by Microsoft.

**Figuur (3.58)**

Configuratie van key vault

Bij het aanmaken van een key vault moet er een subscription en resource group gekozen worden. Er kan een nieuwe resource group aangemaakt worden of een reeds bestaande gekozen worden. Daarnaast moet er een naam, gewenste regio en pricing tier gekozen worden.

### Permission model

Grant data plane access by using a [Azure RBAC](#) or [Key Vault access policy](#)

- ☐ Azure role-based access control (recommended) ⓘ
- ☒ Vault access policy ⓘ

**Figuur (3.59)**

Permission model van key vault

Onder “Access configuration” wordt er gekozen voor “Vault access policy” als per-



mission model.

[Basics](#) [Access configuration](#) [Networking](#) [Tags](#) [Review + create](#)

You can connect to this key vault either publicly, via public IP addresses or service endpoints, or privately, using a private endpoint.


Enable public access ☒

### Public Access

Allow access from:


☐ All networks

☒ Selected networks

 Only networks you choose can access this key vault. [Learn more](#)

### Virtual networks

Allow selected virtual networks to connect to your resource securely and directly using service endpoints [Learn more](#)


[+](#) Add a virtual network 

Virtual network	Subnet	Resource group	Subscription
No virtual networks are selected.			

### Exception

Enabling access to resources requires you allow trusted Microsoft services to bypass firewall.

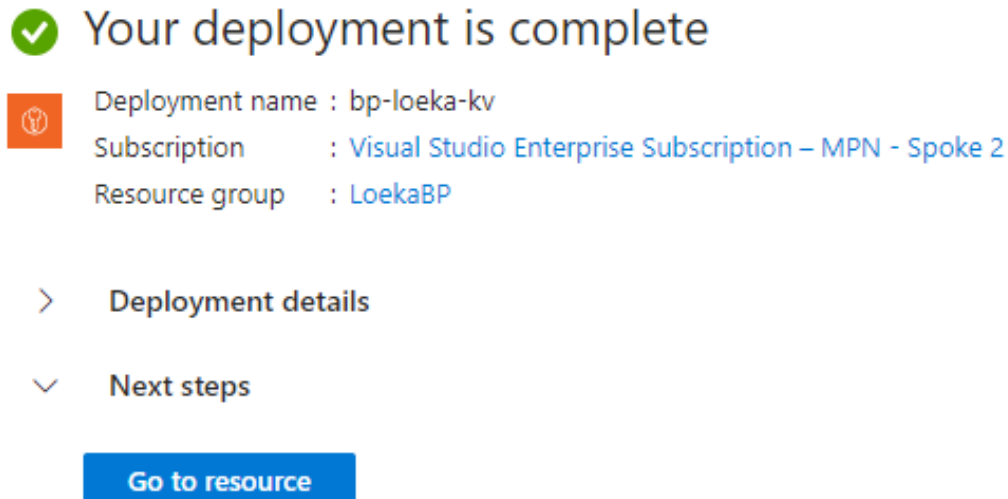
☒ Allow trusted Microsoft services to bypass this firewall

 This setting is related to firewall only. In order to access this key vault, the trusted service must also be given explicit permissions the Access Policies section. [Learn more](#)

### Figuur (3.60)

Networking configuratie van key vault

Er kan gekozen worden om enkel geselecteerde networks public access te geven of om public access uit te schakelen. Belangrijk is dat “Allow trusted Microsoft services to bypass this firewall” aangevinkt staat.

**Figuur (3.61)**

Deployment complete van key vault

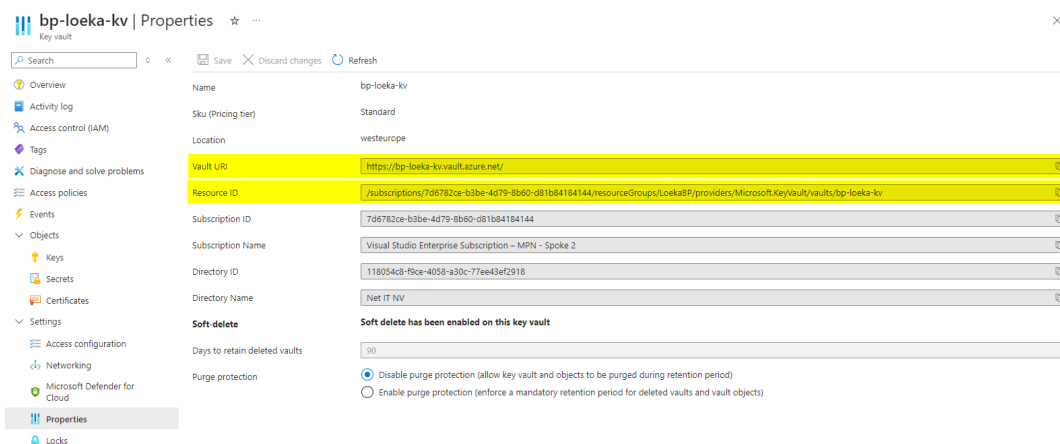
Wanneer onze resource aangemaakt is kunnen we naar de key vault navigeren.

Upload options	Manual
Name *	AppKey
Secret value *	*****
Content type (optional)	
Set activation date	<input type="checkbox"/>
Set expiration date	<input type="checkbox"/>
Enabled	<input checked="" type="radio"/> Yes <input type="radio"/> No
Tags	0 tags

**Figuur (3.62)**

Toevoegen van een secret in key vault

Door naar “Secrets” te navigeren in de key vault kan het client secret toegevoegd worden.

**Figuur (3.63)**

Opzoeken van properties van key vault

Door naar “Properties” te navigeren kunnen we nu “Vault URI” en “Resource ID” terugvinden. Deze zijn belangrijk voor de volgende stap.

[HomePage](#) / [Create Secret Scope](#)

## Create Secret Scope

Cancel

Create

A store for secrets that is identified by a name and backed by a specific store type. [Learn more](#)

Scope Name ?

KeyVault

Manage Principal ?

All Users

Azure Key Vault ?

DNS Name

<https://bp-loeka-kv.vault.azure.net/>

Resource ID

/subscriptions/7d6782ce-b3be-4d79-8b60-d81b84184144/resourceGroups/LoekaBF

**Figuur (3.64)**

Aanmaken van secret scope in databricks

Door naar “<https://<databricks-instance>#secrets/createScope>” te navigeren kunnen we nu een secret scope aanmaken in databricks. Hiervoor hebben we “Vault URI” en “Resource ID” nodig uit de vorige stap.

```
1 app_key = dbutils.secrets.get(scope = "KeyVault", key = "AppKey")
```

We kunnen nu in onze notebooks met bovenstaande code secrets zoals bijvoorbeeld “AppKey” ophalen uit key vault.

```
1 storage_account = "<storage-account>"
2 app_id = "<application-id>"
3 app_key = dbutils.secrets.get(scope = "KeyVault", key = "AppKey")
4 tenant_id = "<directory-id>"
```

Om een connectie te kunnen maken met data lake hebben we “storage\_account”,

“app\_id”, “app\_key” en “tenant\_id” nodig. Dit kunnen we doen door het volgende te vervangen:

- “<storage-account>” met de naam van het Azure storage account
- “<application-id>” met het Application (client) ID voor de Microsoft Entra ID application
- “<directory-id>” met het Directory (tenant) ID voor de Microsoft Entra ID application

```
1 df = spark.read.format("com.microsoft.cdm") \
2     .option("appId", app_id) \
3     .option("appKey", app_key) \
4     .option("tenantId", tenant_id) \
5     .option("storage", f"{storage_account}.dfs.core.windows.net") \
6     .option("manifestPath", "sourcedata/model.json") \
7     .option("mode", "permissive")
```

We kunnen nu gebruik maken van “spark-cdm-connector” door “com.microsoft.cdm” mee te geven. Het is belangrijk dat “mode” op “permissive” staat, dit zorgt er voor dat er “NULL” values assigned worden wanneer een CSV rij minder aantal kolommen heeft dan het entity schema.

```
1 spr = df.option("entity", "new_syndicalpremiumrequest") \
2     .load()
```

Met bovenstaande code kan een entiteit ingeladen worden in een variabele. Dit zal dus ook gebeuren met alle entiteiten die we hebben.

```
1 spr \
2     .select("Id", "new_personid", "new_bankaccountid",
3           "new_scannedrequestcode", "new_isdeclarationonhonour",
4           "new_requesttypeid", "new_formnumber",
5           "new_treatmentstate", "new_feedback",
6           "new_reasonforcontrol", "new_yearid", "createdon",
7           "new_idadm", "new_contributionamount", "new_premiumamount",
8           "new_paymentdate", "new_remarkgroupfinal", "new_groupid") \
9     .createOrReplaceTempView("new_syndicalpremiumrequest")
```

Voor elke entiteit die we hebben ingeladen in een variabele zullen we nu de nodige kolommen gaan selecteren. Hierna kan er een temporary view aangemaakt worden voor de entiteiten. Daardoor kunnen we later queries gaan uitvoeren in SQL om zo transformaties uit te voeren.

**Belangrijkste transformaties****Determinatie van welke groepen de premie in hun bestand krijgen**

```

1 CREATE
2 OR REPLACE TEMPORARY VIEW spr_stage_1 AS
3 SELECT
4     YEAR(spr.createdon) EntryYear,
5     YEAR(GETDATE()) CalendarYear,
6     CAST(ny.new_year AS INTEGER) RefYear,
7     IF(spr.new_scannedrequestcode = 552010000,
8         IF(spr.new_isdeclarationonhonour, "DOH", "FORM"),
9         IF(spr.new_scannedrequestcode = 552010001, "QR",
10            "UNKNOWN")) Type,
11
12 IF(spr.new_requesttypeid =
13     "d026f6f9-fc82-ea11-a811-000d3a2d0171",
14     CONCAT(SUBSTRING(new_formnumber, 1, 4), "-",
15     SUBSTRING(new_formnumber, 5, 6), "-",
16     SUBSTRING(new_formnumber, 11, 20)), spr.new_formnumber)
17 Form,
18
19 IF(rpr.new_treatmentstate = 552010004,
20     CASE
21         WHEN spr.new_treatmentstate = 552010000 THEN "1"
22         WHEN spr.new_treatmentstate = 552010001 THEN "10"
23         WHEN spr.new_treatmentstate = 552010002 THEN "1"
24         WHEN spr.new_treatmentstate = 552010003 THEN "10"
25         WHEN spr.new_treatmentstate = 552010004 THEN "5"
26         WHEN spr.new_treatmentstate = 552010005 THEN "11"
27         WHEN spr.new_treatmentstate = 552010006 THEN "1"
28         WHEN spr.new_treatmentstate = 552010007 THEN "10"
29         WHEN spr.new_treatmentstate = 552010008 THEN "8"
30         ELSE ""
31     END,
32     CASE
33         WHEN spr.new_treatmentstate = 552010000 THEN "1"
34         WHEN spr.new_treatmentstate = 552010001 THEN "2"
35         WHEN spr.new_treatmentstate = 552010002 THEN "1"
36         WHEN spr.new_treatmentstate = 552010003 THEN "2"
37         WHEN spr.new_treatmentstate = 552010004 THEN "5"
38         WHEN spr.new_treatmentstate = 552010005 THEN "11"

```

```

31         WHEN spr.new_treatmentstate = 552010006 THEN "1"
32         WHEN spr.new_treatmentstate = 552010007 THEN "2"
33         WHEN spr.new_treatmentstate = 552010008 THEN "8"
34         ELSE ""
35     END
36 ) Status,
37 IF(spr.new_feedback, "FEEDBACK", spr.new_reasonforcontrol)
    AvalonRemark,
38
39     spr.new_personid,
40     spr.new_idadm,
41     spr.new_yearid,
42     spr.id,
43     spr.createdon,
44     spr.new_contributionamount,
45     spr.new_premiumamount,
46     spr.new_paymentdate,
47     ba.new_bankaccount,
48     spr.new_remarkgroupfinal,
49     p.new_firstname,
50     p.new_lastname,
51     p.new_dateofbirth
52 FROM new_syndicalpremiumrequest spr
53 LEFT JOIN new_person p ON spr.new_personid = p.new_personid
54 LEFT JOIN new_bankaccount ba ON spr.new_bankaccountid =
    ba.new_bankaccountid
55 LEFT JOIN related_premium_requests rpr ON spr.Id =
    rpr.new_syndicalpremiumrequestid
56 LEFT JOIN new_year ny ON spr.new_yearid = ny.new_yearid;

```

De tabel "new\_year" wordt opnieuw gejoind op de tabel "new\_syndicalpremiumrequest" met behulp van de kolom "new\_yearid". Daarnaast worden ook hier de drie kolommen "EntryYear", "CalendarYear" en "RefYear" berekent. Deze worden op dezelfde manier berekent als bij de transformatie in Azure Data Factory.

```

1 CREATE
2 OR REPLACE TEMPORARY VIEW spr_stage_2 AS
3 SELECT
4     explode(array(nm.new_groupid, noy.new_groupid)) new_groupid,
5     spr.*,
6     nm.new_groupprefix new_groupprefix

```

```

7 FROM spr_stage_1 spr
8 LEFT JOIN new_membership_stage_1 nm
9     ON (spr.CalendarYear = nm.new_year OR spr.EntryYear =
        nm.new_year OR RefYear = nm.new_year) AND spr.new_personid
        = nm.new_personid
10 LEFT JOIN new_organizationyear_stage_1 noy
11     ON spr.new_idadm = noy.new_idadmaux AND spr.new_yearid =
        noy.new_yearid;

```

De tabellen “new\_membership” en “new\_organizationyear” worden gejoind op de tabel “new\_syndicalpremiumrequest” om de groepen te gaan bepalen. Zoals te zien in de code snippet hierboven noemen deze tabellen “new\_membership\_stage\_1” en “new\_organizationyear\_stage\_1”. Dit komt doordat er reeds andere transformaties zijn geweest.

Bij de tabel “new\_membership” wordt er, net zoals in Azure Data Factory, gekeken of “CalendarYear”, “EntryYear” of “RefYear” overeenkomt met het jaartal van de membership. Daarnaast wordt er ook gekeken of personid overeen komt.

Bij de tabel “new\_organizationyear” wordt er gejoind met behulp van IDADM en het id van het referentiejaar.

spr.Id	nm.new_groupid	noy.new_groupid
101e4523-7c60-45ff-a928-087ca139d8f5	2ad31fb8-5117-4627-a2a3-a5fdbee5b9ae	null
f5ac28c0-6b80-412a-b0e6-f9a82feff9a1	f1015dd0-3096-478d-ae12-4226c101dd54	a2974e74-be93-4326-9d18-f4bb2abc0842

↓

spr.Id	new_groupid
101e4523-7c60-45ff-a928-087ca139d8f5	2ad31fb8-5117-4627-a2a3-a5fdbee5b9ae
101e4523-7c60-45ff-a928-087ca139d8f5	null
f5ac28c0-6b80-412a-b0e6-f9a82feff9a1	f1015dd0-3096-478d-ae12-4226c101dd54
f5ac28c0-6b80-412a-b0e6-f9a82feff9a1	a2974e74-be93-4326-9d18-f4bb2abc0842

Deze twee joins resulteren in twee nieuwe kolommen “nm.new\_groupid” en “noy.new\_groupid”. Met behulp van de “explode” functie zorgen we er voor dat deze twee kolommen

één kolom zullen worden. Één rij zal dan bijvoorbeeld twee rijen worden. Deze twee rijen zijn dan volledig hetzelfde met het verschil dat de kolom “new\_groupid” voor de eerste rij de waarde van “nm.new\_groupid” heeft en voor de tweede rij de waarde van “noy.new\_groupid”.

```

1 CREATE
2 OR REPLACE TEMPORARY VIEW spr_stage_3 AS
3 SELECT
4     FIRST(id) id,
5     FIRST(new_groupid) new_groupid,
6     FIRST(new_personid) new_personid,
7     FIRST(CalendarYear) CalendarYear,
8     FIRST(EntryYear) EntryYear,
9     FIRST(RefYear) RefYear,
10    FIRST(new_groupprefix) new_groupprefix,
11    FIRST(createdon) createdon,
12    FIRST(new_contributionamount) new_contributionamount,
13    FIRST(new_premiumamount) new_premiumamount,
14    FIRST(new_paymentdate) new_paymentdate,
15    FIRST(new_bankaccount) new_bankaccount,
16    FIRST(new_remarkgroupfinal) new_remarkgroupfinal,
17    FIRST(AvalonRemark) AvalonRemark,
18    FIRST(new_firstname) new_firstname,
19    FIRST(new_lastname) new_lastname,
20    FIRST(new_dateofbirth) new_dateofbirth,
21    FIRST(Type) Type,
22    FIRST(Form) Form,
23    FIRST(Status) Status,
24    FIRST(new_idadm) new_idadm
25 FROM spr_stage_2
26 WHERE id IS NOT NULL AND new_groupid IS NOT NULL AND RefYear IS NOT
      NULL
27 GROUP BY (id, new_groupid, RefYear);

```

Ten slotte wordt er een “GROUP BY” clause gebruikt om er voor te zorgen dat een premie niet twee keer naar dezelfde groep voor dat referentiejaar wordt gestuurd. Daarnaast wordt er ook gefilterd zodat er geen premies zijn zonder id, groupid of referentiejaar. Met behulp van de “FIRST” aggregate function selecteren we steeds de eerste waarde voor de nodige kolommen. We hebben nu een tabel met alle premies waarvan we weten naar welke groep voor welk referentiejaar ze gestuurd moeten worden.



**Bepalen van de kolom “Gr” voor een premie**

```

1 CREATE
2 OR REPLACE TEMPORARY VIEW nm_grouped_by AS
3 SELECT
4     FIRST(new_groupprefix, TRUE) new_groupprefix,
5     FIRST(new_personid) new_personid,
6     FIRST(new_year) new_year
7 FROM new_membership_stage_1
8 WHERE new_groupprefix IS NOT NULL
9 GROUP BY (new_personid, new_year);
10
11 CREATE
12 OR REPLACE TEMPORARY VIEW spr_stage_4 AS
13 SELECT
14     spr.*,
15     CASE
16         WHEN spr.new_groupprefix IS NOT NULL THEN
17             spr.new_groupprefix
18         WHEN nmCalendar.new_groupprefix IS NOT NULL THEN
19             nmCalendar.new_groupprefix
20         WHEN nmEntry.new_groupprefix IS NOT NULL THEN
21             nmEntry.new_groupprefix
22         WHEN nmRef.new_groupprefix IS NOT NULL THEN
23             nmRef.new_groupprefix
24         ELSE NULL
25     END Gr
26 FROM spr_stage_3 spr
27 LEFT JOIN nm_grouped_by nmCalendar ON spr.new_personid =
28     nmCalendar.new_personid AND spr.CalendarYear =
29     nmCalendar.new_year
30 LEFT JOIN nm_grouped_by nmEntry ON spr.new_personid =
31     nmEntry.new_personid AND spr.EntryYear = nmEntry.new_year
32 LEFT JOIN nm_grouped_by nmRef ON spr.new_personid =
33     nmRef.new_personid AND spr.RefYear = nmRef.new_year;

```

Voor het bepalen van de kolom “Gr” in databricks wordt er eerst een temporary view “nm\_grouped\_by” aangemaakt waarbij er gegroepeerd wordt op “new\_personid” en “new\_year”. Dit zal er voor zorgen dat de joins die we uitvoeren zullen resulteren in één enkele match (of geen). De tweede parameter in de functie “FIRST” (aangeduid in code snippet) zorgt er voor dat “NULL” values genegeerd worden. De eerste waarde die niet “NULL” is voor “new\_groupprefix” zal dus geselecteerd worden.

De temporary view die we hebben aangemaakt wordt nu drie keer gejoind op de tabel “spr\_stage\_3” aan de hand van “new\_personid” en “CalendarYear”, “EntryYear” of “RefYear”. “Gr” zal bepaald worden door eerst te kijken of “spr.new\_groupprefix” niet “NULL” is, wanneer deze wel “NULL” is zal er gekeken worden naar “nmCalendar.new\_groupprefix”, “nmEntry.new\_groupprefix” en ten slotte naar “nmRef.new\_groupprefix”.

### Bepalen van de kolommen “AntheaNumber” en “MembershipNumber” voor een premie

```

1 CREATE
2 OR REPLACE TEMPORARY VIEW nm_grouped_by_2 AS
3 SELECT
4     FIRST(new_antheanumber, TRUE) new_antheanumber,
5     FIRST(new_newmembershipnumber, TRUE) new_newmembershipnumber,
6     FIRST(new_groupid) new_groupid,
7     FIRST(new_year) new_year,
8     FIRST(new_personid) new_personid
9 FROM new_membership_stage_1
10 WHERE new_antheanumber IS NOT NULL OR new_newmembershipnumber IS
    NOT NULL
11 GROUP BY (new_groupid, new_year, new_personid);
12
13 CREATE
14 OR REPLACE TEMPORARY VIEW spr_stage_5 AS
15 SELECT
16     spr.*,
17     CASE
18         WHEN Gr IS NULL THEN NULL
19         WHEN nmCalendar.new_antheanumber IS NOT NULL THEN
20             nmCalendar.new_antheanumber
21         WHEN nmEntry.new_antheanumber IS NOT NULL THEN
22             nmEntry.new_antheanumber
23         WHEN nmRef.new_antheanumber IS NOT NULL THEN
24             nmRef.new_antheanumber
25         ELSE NULL
26     END AntheaNumber,
27     CASE
28         WHEN Gr IS NULL THEN NULL
29         WHEN nmCalendar.new_newmembershipnumber IS NOT NULL THEN
30             nmCalendar.new_newmembershipnumber
31         WHEN nmEntry.new_newmembershipnumber IS NOT NULL THEN
32             nmEntry.new_newmembershipnumber

```

```
28      WHEN nmRef.new_newmembershipnumber IS NOT NULL THEN
          nmRef.new_newmembershipnumber
29      ELSE NULL
30  END MembershipNumber
31 FROM spr_stage_4 spr
32 LEFT JOIN nm_grouped_by_2 nmCalendar ON spr.new_groupid =
    nmCalendar.new_groupid AND spr.CalendarYear =
    nmCalendar.new_year AND spr.new_personid =
    nmCalendar.new_personid
33 LEFT JOIN nm_grouped_by_2 nmEntry ON spr.new_groupid =
    nmEntry.new_groupid AND spr.EntryYear = nmEntry.new_year AND
    spr.new_personid = nmEntry.new_personid
34 LEFT JOIN nm_grouped_by_2 nmRef ON spr.new_groupid =
    nmRef.new_groupid AND spr.RefYear = nmRef.new_year AND
    spr.new_personid = nmRef.new_personid;
```

Voor het bepalen van de kolommen “AntheaNumber” en “MembershipNumber” wordt er eerst een temporary view “nm\_grouped\_by\_2” aangemaakt waarbij er gegroepeerd wordt op “new\_groupid”, “new\_year” en “new\_personid”. Dit zal er voor zorgen dat de joins die uitgevoerd worden resulteren in één enkele match (of geen). De functie “FIRST” zorgt er weer voor dat “NULL” values genegeerd worden. De aangemaakte temporary view wordt nu drie keer gejoind op de tabel “spr\_stage\_4” aan de hand van “new\_groupid”, “new\_personid” en “CalendarYear”, “EntryYear” of “RefYear”. Wanneer de kolom “Gr”, “NULL” is, zal “AntheaNumber” en “MembershipNumber” ook “NULL” zijn. Wanneer dit niet zo is wordt eerst gekeken of de waarde van de join aan de hand van “CalendarYear” niet “NULL” is. Wanneer dit wel “NULL” is zal er vervolgens gekeken worden naar de join aan de hand van “EntryYear” en vervolgens naar de join aan de hand van “RefYear”.

# 4

## Conclusie

Curabitur nunc magna, posuere eget, venenatis eu, vehicula ac, velit. Aenean ornare, massa a accumsan pulvinar, quam lorem laoreet purus, eu sodales magna risus molestie lorem. Nunc erat velit, hendrerit quis, malesuada ut, aliquam vitae, wisi. Sed posuere. Suspendisse ipsum arcu, scelerisque nec, aliquam eu, molestie tincidunt, justo. Phasellus iaculis. Sed posuere lorem non ipsum. Pellentesque dapibus. Suspendisse quam libero, laoreet a, tincidunt eget, consequat at, est. Nullam ut lectus non enim consequat facilisis. Mauris leo. Quisque pede ligula, auctor vel, pellentesque vel, posuere id, turpis. Cras ipsum sem, cursus et, facilisis ut, tempus euismod, quam. Suspendisse tristique dolor eu orci. Mauris mattis. Aenean semper. Vivamus tortor magna, facilisis id, varius mattis, hendrerit in, justo. Integer purus.

Vivamus adipiscing. Curabitur imperdiet tempus turpis. Vivamus sapien dolor, congue venenatis, euismod eget, porta rhoncus, magna. Proin condimentum pretium enim. Fusce fringilla, libero et venenatis facilisis, eros enim cursus arcu, vitae facilisis odio augue vitae orci. Aliquam varius nibh ut odio. Sed condimentum condimentum nunc. Pellentesque eget massa. Pellentesque quis mauris. Donec ut ligula ac pede pulvinar lobortis. Pellentesque euismod. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent elit. Ut laoreet ornare est. Phasellus gravida vulputate nulla. Donec sit amet arcu ut sem tempor malesuada. Praesent hendrerit augue in urna. Proin enim ante, ornare vel, consequat ut, blandit in, justo. Donec felis elit, dignissim sed, sagittis ut, ullamcorper a, nulla. Aenean pharetra vulputate odio.

Quisque enim. Proin velit neque, tristique eu, eleifend eget, vestibulum nec, lacus. Vivamus odio. Duis odio urna, vehicula in, elementum aliquam, aliquet laoreet, tellus. Sed velit. Sed vel mi ac elit aliquet interdum. Etiam sapien neque, convallis et, aliquet vel, auctor non, arcu. Aliquam suscipit aliquam lectus. Proin tincidunt magna sed wisi. Integer blandit lacus ut lorem. Sed luctus justo sed enim.

Morbi malesuada hendrerit dui. Nunc mauris leo, dapibus sit amet, vestibulum et, commodo id, est. Pellentesque purus. Pellentesque tristique, nunc ac pulvinar adipiscing, justo eros consequat lectus, sit amet posuere lectus neque vel augue. Cras consectetur libero ac eros. Ut eget massa. Fusce sit amet enim eleifend sem dictum auctor. In eget risus luctus wisi convallis pulvinar. Vivamus sapien risus, tempor in, viverra in, aliquet pellentesque, eros. Aliquam euismod libero a sem. Nunc velit augue, scelerisque dignissim, lobortis et, aliquam in, risus. In eu eros. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Curabitur vulputate elit viverra augue. Mauris fringilla, tortor sit amet malesuada mollis, sapien mi dapibus odio, ac imperdiet ligula enim eget nisl. Quisque vitae pede a pede aliquet suscipit. Phasellus tellus pede, viverra vestibulum, gravida id, laoreet in, justo. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Integer commodo luctus lectus. Mauris justo. Duis varius eros. Sed quam. Cras lacus eros, rutrum eget, varius quis, convallis iaculis, velit. Mauris imperdiet, metus at tristique venenatis, purus neque pellentesque mauris, a ultrices elit lacus nec tortor. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent malesuada. Nam lacus lectus, auctor sit amet, malesuada vel, elementum eget, metus. Duis neque pede, facilisis eget, egestas elementum, nonummy id, neque.



# Onderzoeksvoorstel

Het onderwerp van deze bachelorproef is gebaseerd op een onderzoeksvoorstel dat vooraf werd beoordeeld door de promotor. Dat voorstel is opgenomen in deze bijlage.

## A.1. Inleiding

Het implementeren van ETL's en ELT's speelt een kritieke rol in de development binnen Net IT. Net IT is een bedrijf gespecialiseerd in Customer Relationship Management (CRM). Een CRM-systeem is een applicatie om voeling te houden met klanten en om processen te stroomlijnen en meer winst te genereren. Doordat Net IT een Microsoft Partner is zullen ze CRM-toepassingen implementeren met Microsoft Dynamics 365 en intelligente bedrijfstoeepassingen met Microsoft Power Platform. Binnen Microsoft 365 Dynamics Customer Engagement worden er CSV bestanden geëxporteerd naar Azure Data Lake. Deze data wordt minstens dagelijks opgekuist, aangevuld en opgesplitst om naar de klant te kunnen doorsturen via SFTP of e-mail. Momenteel wordt dit gedaan door ETL's en ELT's te implementeren in Azure Data Factory, gebruik makend van de UI tools die aangeboden worden. Doordat dit niet de enige mogelijkheid is om dit te implementeren zal er gekeken worden naar de verschillende opties die er zijn. Aangezien Net IT een Microsoft Partner is zal er vooral gekeken worden binnen Microsoft Azure. Deze verschillende mogelijkheden zullen vergeleken worden op basis van performantie, kostprijs (voor dezelfde performantie), complexiteit, moeilijkheidsgraad in opzet en configuratie van de resources, verschil in implementatietijd en mogelijkheden van de tool. De methode die gebruikt zal worden is een gemengde aanpak gebaseerd op literatuuronderzoek en het opstellen van proof-of-concepts voor de gegeven use case van Net IT. Dit zal resulteren in een gedetailleerd vergelijkingsrapport, inclusief aanbeveling voor de meest geschikte aanpak voor het implementeren van

ETL's of ELT's voor de gegeven use case.

## **A.2. Literatuurstudie**

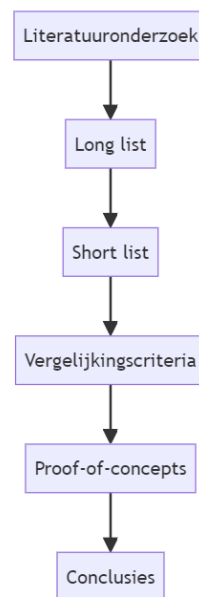
Als data engineer krijgt men data in veel verschillende vormen. Het is dus noodzakelijk om deze data klaar te maken voor business analytics. (Kromer, [2022b](#)) Hiervoor maakt men gebruik van ETL's en ELT's. Dit zijn processen die organisaties gebruiken voor het verzamelen en samenvoegen van data uit meerdere bronnen. Bij ETL's wordt de data getransformeerd voor het naar de doelopslagplaats geladen wordt, terwijl dit bij ELT's pas achteraf gebeurt. Daardoor staat ETL voor Extract, Transform and Load en ELT voor Extract, Load and Transform. (Bartley, [2023](#))

In Azure zijn er meerdere mogelijkheden voor het implementeren van ETL's en ELT's. Één van deze mogelijkheden is Azure Data Factory. Zoals te zien in de enquête van Sreemathy e.a. ([2021](#)) is dit de meest populaire data integratie service die aangeboden wordt door de cloud providers. Binnen Azure Data Factory kan er gebruik gemaakt worden van Mapping Data Flows, dit is een codevrije manier waarmee ETL's opgebouwd kunnen worden. De logica achter de ETL kan hierna makkelijk getest worden op live data en samples. (Kromer, [2022a](#))

Daarnaast biedt Azure ook Azure Databricks aan. Het verschil hierbij is dat de ETL's worden geïmplementeerd via code terwijl dat bij Azure Data Factory via de UI tools kan gebeurt. Azure Databricks is gebaseerd op het Apache Spark open source project. Het grote voordeel is dat het platform het toelaat om makkelijker te kunnen samen werken. Daarnaast is Apache Spark niet enkel gelimiteerd tot het maken van ETL's maar kan het ook gebruikt worden voor real-time analytics, machine learning, graph processing, etc. (Etaati, [2019](#))

Azure is niet de enigste cloud provider die ETL tools aanbiedt. Zo heeft AWS bijvoorbeeld AWS Glue (Khan e.a., [2024](#)) en Google Cloud heeft Google Data Fusion. (Jaiswal, [2022](#))

### A.3. Methodologie



De eerste fase is het literatuuronderzoek. Hierbij wordt er gekeken naar welke mogelijkheden er zijn voor het implementeren van ETL's en ELT's. Er zal vooral gefocust worden op de mogelijkheden binnen Azure maar ook andere opties zullen bekeken worden. Dit zal gedaan worden met behulp van academische onderzoekstools zoals Google Scholar en andere relevante databanken en bronnen. Ook de documentatie van Azure, casestudies van bedrijven die gebruik maken van Azure en blogs van experts op dit vakgebied zullen hier zeker bij van pas komen. Deze fase zal ook zeker in samenwerkingen met Net IT gebeuren zodat de huidige gebruikte technologieën voor het implementeren van ETL's en ELT's zeker niet uitgesloten worden. Dit onderdeel zal naar verwachting vier weken in beslag nemen.

In de tweede fase zullen de resultaten van het literatuuronderzoek samengevat worden in een long list. Dit onderdeel zal naar verwachting twee weken duren.

In de derde fase zullen de meest interessante opties uit de long list samengevat worden in een short list. Hierbij zal gekeken worden naar wat het interessantst is voor Net IT. Doordat dit een kleine fase is zal dit bij de tijd van de tweede fase horen.

In de vierde fase zullen er vergelijkingscriteria opgesteld worden voor de gegeven use case door Net IT. Het is belangrijk om te weten wat er moet vergeleken worden. Belangrijk om op te merken is dat niet alles even meetbaar zal zijn. Er zal dus goed gekeken worden naar hoe de vergelijkingscriteria gemeten kunnen worden. Dit onderdeel zal naar verwachting één week duren.

In de vijfde fase zal er op basis van de short list en vergelijkingscriteria een proof-of-concept uitgewerkt worden voor elke mogelijkheid dat er binnen Azure is. Er zal een situatie (gegeven door Net IT) opgezet worden met dummy data in een data lake. Het doel is dat er op basis van deze data export bestanden zullen gemaakt worden. Dit zal de langste fase zijn en zal dus vier weken in beslag nemen.



De zesde en laatste fase, die naar verwachting één week zal duren, is de evaluatie van de opties die we hebben onderzocht. Het doel is om te tonen welke optie het beste is. Daardoor zal het resultaat van deze analyse een gedetailleerd vergelijkingsrapport zijn en een aanbeveling voor welke optie het beste is.

**A.4. Verwacht resultaat, conclusie**

Het resultaat zal een gedetailleerd vergelijkingsrapport zijn, inclusief aanbevelingen voor de meest geschikte aanpak voor het implementeren van ETL's of ELT's voor de gegeven use case. Daarnaast zal er ook een resultaat per vergelijkingscriteria zijn zodat Net IT zelf ook een dieper inzicht krijgt in bijvoorbeeld operationele implicaties, kostenstructuur, etc.

# Bibliografie

- Bartley, K. (2023, januari 2). *What is the Difference Between ETL and ELT?* <https://rivery.io/blog/etl-vs-elt/>
- Etaati, L. (2019, juni 13). Azure Databricks. In *Machine Learning with Microsoft Technologies: Selecting the Right Architecture and Tools for Your Project* (pp. 159–171). Apress. [https://doi.org/10.1007/978-1-4842-3658-1\\_10](https://doi.org/10.1007/978-1-4842-3658-1_10)
- Ethan. (2024, januari 14). *100+ Best ETL Tools List Software (January 2024 Update)*. <https://portable.io/learn/best-etl-tools>
- Inmon, B. (2023, maart 31). *Understanding the Necessity of ETL in Data Integration*. <https://www.integrate.io/blog/etl-in-data-integration/>
- J., P. K. (2023, maart 6). *What is Business Analytics? Definition, Importance Examples*. <https://www.linkedin.com/pulse/what-business-analytics-definition-importance-examples-jha/>
- Jaiswal, N. (2022). Data Fusion Basics. <https://medium.com/google-cloud/data-fusion-basic-concepts-c40b09efd695>
- Khan, B., Jan, S., Khan, W., & Chughtai, M. I. (2024). An Overview of ETL Techniques, Tools, Processes and Evaluations in Data Warehousing. [https://cdn.techscience.cn/files/jbd/2024/TSP\\_JBD-6/TSP\\_JBD\\_46223/TSP\\_JBD\\_46223.pdf](https://cdn.techscience.cn/files/jbd/2024/TSP_JBD-6/TSP_JBD_46223/TSP_JBD_46223.pdf)
- Kromer, M. (2022a). Common ETL Pipeline Practices in ADF with Mapping Data Flows. In *Mapping Data Flows in Azure Data Factory: Building Scalable ETL Projects in the Microsoft Cloud*. [https://doi.org/10.1007/978-1-4842-8612-8\\_5](https://doi.org/10.1007/978-1-4842-8612-8_5)
- Kromer, M. (2022b, augustus 26). Introduction to Mapping Data Flows. In *Mapping Data Flows in Azure Data Factory: Building Scalable ETL Projects in the Microsoft Cloud* (pp. 27–50). Apress. [https://doi.org/10.1007/978-1-4842-8612-8\\_3](https://doi.org/10.1007/978-1-4842-8612-8_3)
- Microsoft. (2024a, maart 1). *What is Delta Live Tables?* (Microsoft, Red.). <https://learn.microsoft.com/en-us/azure/databricks/delta-live-tables/>
- Microsoft. (2024b, maart 7). *What is Azure Databricks?* <https://learn.microsoft.com/en-gb/azure/databricks/introduction/>
- Microsoft. (2024c, maart 7). *What is Delta Lake?* (Microsoft, Red.). <https://learn.microsoft.com/en-us/azure/databricks/delta/>
- Microsoft. (2024d, maart 19). *What is Azure Data Factory?* (Microsoft, Red.). <https://learn.microsoft.com/en-us/azure/data-factory/introduction>
- Pollefliet, L. (2011). *Schrijven van verslag tot eindwerk: do's en don'ts*. Academia Press.

- Rawat, S., & Narain, A. (2018, december 19). Introduction to Azure Data Factory. In *Understanding Azure Data Factory: Operationalizing Big Data and Advanced Analytics Solutions* (pp. 13–56). Apress. [https://doi.org/10.1007/978-1-4842-4122-6\\_2](https://doi.org/10.1007/978-1-4842-4122-6_2)
- Sreemathy, J., Brindha, R., Nagalakshmi, M. S., Suvekha, N., Ragul, N. K., & Praveenandha, M. (2021). Overview of etl tools and talend-data integration. *2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS)*. <https://intapi.sciendo.com/pdf/10.2478/picbe-2023-0182>
- Vines, A., & Tanasescu, L. (2023). An Overview of ETL Cloud Services: An Empirical Study Based on User's Experience. *Proceedings of the International Conference on Business Excellence*, 17(1), 2085–2098. <https://intapi.sciendo.com/pdf/10.2478/picbe-2023-0182>