# **Werking (technisch)**

Het uitvoeren van het project bestaat uit 2 stappen:

1. Initiële run: De gegevens worden voor de eerste keer in de tabel geplaatst
2. Niet-initiële run: De gegevens worden vergeleken en daarna in de tabel geplaatst.

Groen: Functies binnen python

*Cursief*: Parameters in functies

Blauw: Functies binnen SQL (stored procedures)

## Functies

1. Functie: locateReports(*file\_extension, file\_root, standard\_path*)

Deze python-functie lokaliseert de rapporten binnen de hoofdmap en de domeinen.

* + De *file\_extension* is de extensie ‘.lgx’ waar in dit geval naar gezocht moet worden
  + De *file\_root* is de beginmap waar vanuit gezocht moet worden
  + Het *standard\_path* is de mapopbouw ‘\_Definitions/\_Reports’, de plek waar altijd de ‘lgx’-bestanden te vinden zijn.
  + *Return*: De functie geeft als *return* een lijst met de bestandsnaam en locatie (path) terug: *ReportLocations*

1. Functie: returnReportData(*ReportLocations*) (Zie:

Deze python-functie doorloopt de lijst *ReportLocations* en haalt de volgende gegevens op uit de rapporten:

Binnen het bestand:

Algemeen

* Rapportnaam (*ReportName*)
* Laatst gewijzigd door (*ModifiedBy*)

Uit de datalayer (De gegevens over query’s bevinden zich in datalayer nodes)

* ID van de datalayer (*DatalayerID*)
* De server waar de query op uitgevoerd wordt (*ConnectionID*)
* Query (*Query*)
  + Het object, de database waaruit informatie opgehaald wordt (*Object*)

De volgende gegevens worden extra opgehaald:

* De locatie van het rapport (*path*)
* De server waar het rapport zich bevindt (*ServerName*)
* Het domein waarin het rapport zich bevindt (bijvoorbeeld BeheerAll/Wmr) (*Domain*)
* De OTAP omgeving (*Environment*)
* De datum waarop het bestand is gemaakt (*DateCreated*)
* De datum waarop het bestand voor het laatst is gewijzigd (*DateModified*)
* De datum waarop het bestand is geanalyseerd (*DateExtracted*)

Eventueel:

* *ReportCaption, Securityreportrightsid, reportID, sharedElementFile, sharedElementID, ElementID*
* Er wordt een hash (*DataHash*) gecreëerd aan de hand van de volgende gegevens:
* Query, Object, Datalayer ID, Connection ID
* *Return:* De functie geeft als *return* een lijst met alle gegevens

### Initieel

1. Functie: fillSQL(ReportData, init\_run = True)

Deze python-functie roept de volgende stored procedures aan:

* + TRUNCATE TABLE logi.xml\_data
    - Maakt ‘logi.xml\_data’ tabel leeg
  + EXEC CreateTempTable:
    - Maakt tijdelijke tabel ‘##all\_data’
  + EXEC InsertData:
    - Loopt door de gegevens en voegt alle gegevens vanuit python in ‘##all\_data’ tabel
  + EXEC ExtractDataInit:
    - Maakt ‘logi.temp\_xml\_data’ tabel leeg
    - Vergelijkt gegevens uit ##all\_data met schema’s uit CDWH
      * Voegt deze in ‘logi.xml\_data’ tabel met als ‘ValidFrom’-flag de huidige datum

Na het beëindigen van de sessie wordt de tijdelijke tabel ‘##all\_data’ automatisch verwijderd

### Niet-Initieel

1. Functie: fillSQL(ReportData, init\_run = False)

Deze python-functie roept de volgende stored procedures aan:

* 1. EXEC CreateTempTable:
     + Maakt tijdelijke tabel ‘##all\_data’

1. EXEC InsertData:
   * + Loopt door alle gegevens en voegt alle gegevens vanuit python in ‘##all\_data’ tabel
2. EXEC ExtractData:
   * + Maakt ‘logi.temp\_xml\_data’ tabel leeg
     + Vergelijkt gegevens uit ##all\_data met schema’s uit CDWH
       - Insert deze in ‘logi.temp\_xml\_data’
     + Vergelijkt temp\_xml\_data met xml\_data voor verwijderde waarden
       - Geeft verwijderde rijen uit xml\_data flag: ValidTo (huidige datum)
     + Vergelijkt temp\_xml\_data met xml\_data voor nieuwe waarden
       - Geef nieuwe waarden in xml\_data flag: ValidFrom (huidige datum)
     + Vergelijkt temp\_xml\_data met xml\_data voor aangepaste waarden
       - Geef aangepaste waarden in xml\_data flag: ValidTo
       - Geef nieuwe waarden in xml\_data flag: ValidFrom