Testjes OData V4 beta implementatie CBS

Han Oostdijk

file created : 25Dec2018

Load the libraries that will be used

We use the following packages but these are not loaded/attached by a `library` or `require` statement: `dplyr`, `tidyr`, `purrr`, `jsonlite`, `httr`, `glue`, `xlm2`, `fs` and `HOQCutil`. Functions of these packages are prefixed with the package name and `::` as e.g. in `HOQCutil::ref_tab`. All packages are on CRAN with the exception of my package `HOQCutil` that can be installed with the following code:

```
devtools::install_github("HanOostdijk/HOQCutil")
```

Introduction

Statistics Netherlands (CBS) has announced the beta version of version 4 for the Open Data Protocol (OData) on the following webpage. In december 2018 a LinkedIn Post announced some changes in the Release Notes .

In this document I try to redo the tests done in October 2018 with the new environment.

Conclusion

An important change is that the names in the new beta version are changed to the names in alpha version.
Also MeasureGroups and DimensionGroups and DimensionCodes are no longer present when they are empty.

In general the tests work as I expected. Because for the only table available there are no hierarchies, I could not check these. NB this is probably no longer true: see todo.

I expected that the `$select=` command would work, but apparently it does not. And indeed it is not mentioned in the implementation page. For the December data I thought at one time to have a case for which the `$select=` clause works but I can not reproduce that case. In any case it does not work for a `select` on Catalog.

In LinkedIn group Centraal Bureau voor de Statistiek Open Data in answer to my question Dennis Ramondt tells that the `$select=` will only be fully implemented in a later stage.

Todo

things to do:

- check hierarchies

Retrieve global information

I fix the `odata_root` and `odata_cat` arguments for the `HOQCutil::get_table_cbs_odata4` function by setting the first to the root of the beta data structures and the second to an empty catalog (assuming that I don't know the names of the currently available catalogs in the beta version).

```
beta_root <- "https://acc-ccb.cbs.nl/OData4/"
get_beta_nc <-
  purrr::partial(HOQCutil::get_table_cbs_odata4,
    odata_root = beta_root, odata_cat = ""
  )

conts1 <- get_beta_nc(verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4/
```

```
(dim(conts1))
```

```
[1] 2 3
```

The global data structure can be found in Table 1 on page 2.

name	kind	url
Catalogs	EntitySet	Catalogs
Datasets	EntitySet	Datasets

Table 1: global data structure for CBS OData V4 beta version

The currently available `Catalogs` and `Datasets` can be retrieved in the following way:

```
cats1 <- get_beta_nc(subtable = "Catalogs", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//Catalogs
```

```
dsets1 <- get_beta_nc(subtable = "Datasets", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//Datasets
```

They have the following dimensions and column names:

```
(dim(cats1))
```

```
[1] 2 9
```

```
(names(cats1))
```

```
[1] "Identifier"   "Title"        "Description"  "Publisher"    "Language"     "License"      "Homepage"     "Authority"
[9] "ContactPoint"
```

```
(dim(dsets1))
```

```
[1] 16 13
```

```
(names(dsets1))
```

```
 [1] "Description"          "Identifier"           "Language"             "Title"                "Modified"
 [6] "Catalog"              "Version"              "VersionNotes"         "VersionReason"        "Status"
[11] "ObservationsModified" "ObservationCount"     "DatasetType"
```

I show some fields of the Catalogs in Table 2 on page 4 and the Datasets in Table 3 on page 5. So I see that the beta version currently has 2 Catalogs and 16 Datasets.

Identifier	Title	Description
CBS-Maatwerk	CBS-Maatwerk	Catalogus van het CBS met datasets die niet op StatLine staan
CBS	CBS	Catalogus van het CBS

Table 2: Catalogs for CBS OData V4 beta version

Retrieving table information

I will use the first Dataset (**900001NED**) from Catalog (**CBS-Maatwerk**) to see what information is available for tables. Again I will fix the `odata_root` and `odata_cat` arguments for the `HOQCutil::get_table_cbs_odata4` function by setting the first to the root of the beta data structures and the second to this catalog. And again I can request the available catalogs and datasets for this catalog (`CBS-Maatwerk` is then the only catalog of course).

```
beta_root <- "https://acc-ccb.cbs.nl/OData4/"
get_beta <-
  purrr::partial(HOQCutil::get_table_cbs_odata4,
    odata_root = beta_root, odata_cat = "CBS-Maatwerk"
  )
cats2a <- get_beta(subtable = "Catalogs", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/Catalogs
```

```
(dim(cats2a))
```

```
[1] 1 9
```

```
dsets2a <- get_beta(subtable = "Datasets", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/Datasets
```

```
(dim(dsets2a))
```

Identifier	Title	Catalog	ObservationCount
84120NED	Overheid; ontvangen belastingen en wettelijke premies	CBS	5772
83765NED	Kerncijfers wijken en buurten 2017	CBS	1355087
83487NED	Kerncijfers wijken en buurten 2016	CBS	1380546
83433NED	Consumentenprijzen; werknemers laag, alle basisjaren 1969-1995	CBS	2580
70072ned	Regionale kerncijfers Nederland	CBS	4663946
37230ned	Bevolkingsontwikkeling; regio per maand	CBS	2350408
900001NED	Energielevering aan woningen en bedrijven; postcode 6, 2017	CBS-Maatwerk	1219456
81589NED	Bedrijven; bedrijfstak	CBS	899150
900002NED	Omzetting identifiers van onderwerpen van Odata 3 naar Odata 4	CBS-Maatwerk	1172
81575NED	Vestigingen van bedrijven; bedrijfstak, gemeente	CBS	115668
83220NED	Kerncijfers wijken en buurten 2015	CBS	1401105
82931NED	Kerncijfers wijken en buurten 2014	CBS	1348572
83749NED	Nabijheid voorzieningen; afstand locatie, wijk- en buurtcijfers 2016	CBS	1415016
82245NED	Bevolking en huishoudens; viercijferige postcode, 1 januari 2013	CBS	326174
83136NED	Consumentenprijzen; basisjaren vanaf 1969	CBS	5832
83131NED	Consumentenprijzen; prijsindex 2015=100	CBS	844074

Table 3: Datasets for CBS OData V4 beta version

```
[1]  2 13
```

I also set `table_id=900001NED` and use `get_beta` to retrieve the various subtables of this table. These are shown in Table 4 on page 6. NB. subtable **Properties** is not a 'real' table but a list. The difference with the October 2018 version is that the empty subtables are left out. For this table the subtables `MeasureGroups`, `PostCode6Groups`, `PostCode6Groups`, `PostCode6Codes` and `TypegebruikGroups` no longer exist.

```
table_id <- "900001NED"
subtables1 <- get_beta(table_id = table_id, verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED
```

From the CBS info page I understand that the following subtables exist for each table: **MeasureCodes**, **MeasureGroups**, **Dimensions**, **Observations** and **Properties** (again with the latter being a list). The table **Dimensions** specifies the extra dimensions (next to the 'MeasureCodes') and for each of these extra dimensions two additional tables exist: one with

name	kind	url
MeasureCodes	EntitySet	MeasureCodes
Dimensions	EntitySet	Dimensions
TypegebruikCodes	EntitySet	TypegebruikCodes
Observations	EntitySet	Observations
Properties	Singleton	Properties

Table 4: Subtables of 900001NED for CBS OData V4 beta version

codes and one with groups in line with **MeasureCodes**, **MeasureGroups**. It is possible that one or more of these subtables are empty and from December 2018 onwards those empty tables no longer exist.

Table properties

Properties is a list that describes the characteristics of the table. See the following `str` output for its contents.

```
# retrieve the names and types of the subtables and the property table
props1 <- get_beta(table_id = table_id, subtable = "Properties", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Properties
```

```
HOQCutil::cap.out(str(props1, nchar.max = 5000))
```

```
List of 28
 $ @odata.context      : chr "https://acc-ccb.cbs.nl/OData4/CBS-Maatwerk/900001NED/$metadata#Properties"
 $ Identifier          : chr "900001NED"
 $ Title               : chr "Energielevering aan woningen en bedrijven; postcode 6, 2017"
 $ Description         : chr "Deze tabel bevat per postcodegebied de gemiddelde energielevering (aardgas en elektriciteit) vanu
it  het openbaar net aan particuliere woningen en bedrijven. \nGegevens beschikbaar: over 2017.\nStatus van de cijfers: Definit
ief.\n\nWijzigingen per 5 september 2018:\nGeen. Dit is een nieuwe tabel.\n\nWanneer komen er nieuwe cijfers?\nDefinitieve cijf
ers: 3e kwartaal van het jaar volgend op het verslagjaar.\n\nDefinities:\nParticuliere woning:\nEen verblijfsobject in de Basis
 Registraties Adressen en Gebouwen (BAG) met als gebruiksfunctie “woonfunctie” en géén andere functie.\n\nDe cijfers van de gem
iddelde energieleveringen zijn afgerond op tientallen en worden vermeld vanaf  vijf geldige leveringen."
 $ Modified            : chr "2018-10-18T11:00:00+02:00"
 $ Language            : chr "nl"
```

```
 $ TemporalCoverage    : chr "2017"
 $ Authority           : chr "http://standaarden.overheid.nl/owms/terms/Centraal_Bureau_voor_de_Statistiek"
 $ Catalog             : chr "CBS-Maatwerk"
 $ Publisher           : chr "http://standaarden.overheid.nl/owms/terms/Centraal_Bureau_voor_de_Statistiek"
 $ ContactPoint        : chr "infoservice@cbs.nl"
 $ Version             : chr "201810180900"
 $ VersionNotes        : chr "Dit is de eerste levering van een nieuwe dataset."
 $ VersionReason       : chr "Nieuw"
 $ Frequency           : chr "Eenmalig"
 $ Status              : chr "Regulier"
 $ ObservationCount    : int 1219456
 $ ObservationsModified: chr "2018-09-12T11:00:00+02:00"
 $ DatasetType         : chr "Numeric"
 $ DefaultPresentation : chr "graphType=table"
 $ GraphTypes          : chr ""
 $ SearchPriority      : chr "2"
 $ License             : chr "https://creativecommons.org/licenses/by/4.0/"
 $ Source              : chr "CBS."
 $ Summary             : chr "Energieverbruik Gas en Electra, Postcode 6 naar bedrijf/woning"
 $ LongDescription     : chr "Deze tabel bevat per postcodegebied de gemiddelde energielevering (aardgas en elektriciteit) vanu
it  het openbaar net aan particuliere woningen en bedrijven. \nGegevens beschikbaar: over 2017.\nStatus van de cijfers: Definit
ief.\n\nWijzigingen per 5 september 2018:\nGeen. Dit is een nieuwe tabel.\n\nWanneer komen er nieuwe cijfers?\nDefinitieve cijf
ers: 3e kwartaal van het jaar volgend op het verslagjaar.\n\nDefinities:\nParticuliere woning:\nEen verblijfsobject in de Basis
 Registraties Adressen en Gebouwen (BAG) met als gebruiksfunctie “woonfunctie” en géén andere functie.\n\nDe cijfers van de gem
iddelde energieleveringen zijn afgerond op tientallen en worden vermeld vanaf  vijf geldige leveringen."
 $ Provenance          :'data.frame':   1 obs. of  2 variables:
  ..$ Title: chr "Leveringen van elektriciteit en aardgas via het openbare net"
  ..$ Url  : chr "https://www.cbs.nl/nl-nl/onze-diensten/methoden/onderzoeksomschrijvingen/korte-onderzoeksbeschrijvingen/lever
ingen-van-elektriciteit-en-aardgas-via-het-openbare-net"
 $ RelatedSources      :'data.frame':   8 obs. of  2 variables:
  ..$ Title: chr [1:8] "Energieverbruik particuliere woningen; woningtype, wijken en buurten, 2010" "Energieverbruik particulie
re woningen; woningtype, wijken en buurten, 2011" "Energieverbruik particuliere woningen; woningtype, wijken en buurten, 2012"
"Energieverbruik particuliere woningen; woningtype, wijken en buurten, 2013" ...
  ..$ Url  : chr [1:8] "https://opendata.cbs.nl/statline/#/CBS/nl/dataset/83026NED/table" "https://opendata.cbs.nl/statline/#/C
BS/nl/dataset/83025NED/table" "https://opendata.cbs.nl/statline/#/CBS/nl/dataset/83023NED/table" "https://opendata.cbs.nl/statl
ine/#/CBS/nl/dataset/83022NED/table" ...
```

Measure subtables

Subtable **MeasureCodes** lists the code of the measures that are included in the **Observations** subtable with a description of the codes. Subtable **MeasureGroups** can define a hierarchy on these codes but is, as indicated above, not available for table 900001NED

```
mc1 <- get_beta(table_id = table_id, subtable = "MeasureCodes", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes
```

```
(dim(mc1))
```

```
[1]  3 10
```

```
mg1 <- get_beta(table_id = table_id, subtable = "MeasureGroups", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureGroups
```

```
(mg1)
```

```
[1] "Client error: (404) Not Found"
```

From this I see that the **MeasureGroups** subtable does not exist and that there are 3 measures in **MeasureCodes**. I list these with their description in Table 5 on page 9,

Dimensions

Subtable **Dimensions** lists the extra dimensions of the data besides the measure. These are listed in Table 6 on page 10.

```
dims1 <- get_beta(table_id = table_id, subtable = "Dimensions", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Dimensions
```

Identifier	Index	Title	Description	MeasureGroupID	DataType	Unit	Format	Decimals	PresentationType
M00000001	1	Gemiddelde aardgaslevering	De gemiddelde aardgaslevering in het betreffende jaar aan particuliere woningen of zakelijke aansluitingen, zoals berekend uit de aansluitingenregisters van de netbeheerders. De cijfers zijn afgerond op tientallen en vermeld bij vijf of meer aansluitingsadressen met een geldige levering.		Long	m3		0	
M00000002	2	Gem. aardgaslevering klimaatgecorrigeerd	De gemiddelde aardgaslevering in het betreffende jaar aan particuliere woningen, gecorrigeerd voor temperatuurverschillen tussen jaren. Deze gecorrigeerde waarde is berekend volgens de zogenaamde "Profielenmethodiek", en komt daarmee overeen met het Standaardjaarverbruik (SJV) uit het aansluitingenregister. De cijfers zijn afgerond op tientallen en vermeld bij vijf of meer aansluitingsadressen met een geldige levering.		Long	m3		0	
M00000003	3	Gemiddelde elektriciteitslevering	De gemiddelde elektriciteitslevering in het betreffende jaar aan particuliere woningen of zakelijke aansluitingen, zoals berekend vanuit de aansluitingenregisters van de netbeheerders. De eigen opwekking van elektriciteit, bijvoorbeeld met zonnepanelen, is niet bekend en dus ook niet inbegrepen in de gemiddelde jaarlevering. Ook collectieve verbruiken van bijvoorbeeld liftinstallaties of hal-/galerijverlichting zijn niet meegeteld bij de berekening. De cijfers zijn afgerond op tientallen en vermeld bij vijf of meer aansluitingsadressen met een geldige levering.		Long	kWh		0	

Table 5: Subtable MeasureCodes of 900001NED for CBS OData V4 beta version

```
(dim(dims1))
```

```
[1] 2 6
```

The extra dimensions are PostCode6 and Typegebruik and therefore I have as possible additional dimension tables PostCode6Codes, TypegebruikCodes, PostCode6Groups and TypegebruikGroups. .

Dimension subtables

Identifier	Title	Description	Kind	MapYear	ReleasePolicy
PostCode6	Postcode 6	De beschrijving volgt	Dimension		
Typegebruik	Type Gebruik	De beschrijving volgt	Dimension		

Table 6: Subtable Dimensions of 900001NED for CBS OData V4 beta version

```
(dimtables <- setdiff(
  dplyr::pull(subtables1, "name"),
  c("MeasureCodes", "Dimensions", "Observations", "Properties")
))
```

```
[1] "TypegebruikCodes"
```

For this table (900001NED) we see that only TypegebruikCodes exists.

```
tc <- purrr::map(
  dimtables,
  function(dt) {
    get_beta(table_id = table_id, subtable = dt)
  }
)
names(tc) <- dimtables
tc <- purrr::keep(tc, function(x) length(x) > 0)
names(tc)
```

```
[1] "TypegebruikCodes"
```

So we have 1 additional dimension table: **TypegebruikCodes**. See Table 7 on page 11.

Identifier	Index	Title	Description	DimensionGroupID
A045558	0	Energielevering; particuliere woning	Gemiddelde energielevering aan verblijfsobjecten in de Basis Registraties Adressen en Gebouwen (BAG) met als gebruiksfunctie "woonfunctie" en geen zakelijke gebruiksfunctie.	
A045559	0	Energielevering; zakelijke aansluiting	Gemiddelde energielevering aan verblijfsobjecten in de Basis Registraties Adressen en Gebouwen (BAG) met een zakelijke gebruiksfunctie en levering aan inrichtingselementen met een zakelijke eigenaar die niet in de BAG voorkomen zoals bijvoorbeeld GSM masten en (riool)pompen.	

Table 7: Subtable TypegebruikCodes of 900001NED for CBS OData V4 beta version

Observations subtable

The **Observations** subtable contains the actual data of the table. It makes use of the codes in **MeasureCodes** and the dimension subtables. Here I show how to retrieve the first 15 rows of table **900001NED**. These rows are listed in Table 8 on page 12.

```
nrrows <- 15
```

```
obs1 <- get_beta(
  table_id = table_id, subtable = "Observations", verbose = T,
  query = glue::glue("$top={nrrows}")
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$top=15
unencoded query: $top=15
```

```
(dim(obs1))
```

```
[1] 15  6
```

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
1	M00000001	None	2060	1011AC	A045558
2	M00000001	None	850	1011AH	A045558
3	M00000001	None	1400	1011AJ	A045558
4	M00000001	None	930	1011AN	A045558
5	M00000001	None	1250	1011AP	A045558
6	M00000001	None	1170	1011AR	A045558
7	M00000001	None	1340	1011AT	A045558
8	M00000001	None	1200	1011AW	A045558
9	M00000001	None	2100	1011AX	A045558
10	M00000001	None	760	1011AZ	A045558
11	M00000001	None	1030	1011BA	A045558
12	M00000001	None	1190	1011BC	A045558
13	M00000001	None	1040	1011BD	A045558
14	M00000001	None	1380	1011BH	A045558
15	M00000001	None	1310	1011BJ	A045558

Table 8: First 15 rows of subtable Observations of 900001NED for CBS Odata V4 beta version

Table data

In the document OData protocol v4 the various query possibilities are described. The subset supported by CBS can be found in Implementatie OData v4 functies. With these queries one can extract the table data.

Retrieve data in long form

As an example I show the use of the `startswith` filter option for the **900001NED** table applied on the **Observations** subtable. I also retrieve the **MeasureCodes** and **TypegebruikCodes** subtable. These are used to translate the codes to readable information. This is done outside the OData protocol with the help of the R package **dplyr**. The result table can be found in Table 9 on page 15.

I need only the fields `Identifier` and `Title` from the two code tables, so I would have liked to use the *$select=* construct, but apparently this does not work. Therefore I used `dplyr::select`.

```
obs2 <- get_beta(
  table_id = "900001NED", subtable = "Observations",
  query = "$filter=startswith(PostCode6,'1181N')", verbose = T
) %>%
  dplyr::select(., Measure, Value, PostCode6, Typegebruik) %>%
  dplyr::mutate(., Value = as.numeric(Value))
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=startswith(PostCode6,'1181N')
unencoded query: $filter=startswith(PostCode6,'1181N')
```

```
mc <- get_beta(table_id = "900001NED", subtable = "MeasureCodes", verbose = T) %>%
  dplyr::select(., Identifier, Title)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes
```

```
tc <- get_beta(table_id = "900001NED", subtable = "TypegebruikCodes", verbose = T) %>%
  dplyr::select(., Identifier, Title)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/TypegebruikCodes
```

```
obs3 <- obs2 %>%
  dplyr::inner_join(., mc, by = c("Measure" = "Identifier")) %>%
  dplyr::rename(., Product = Title) %>%
  dplyr::inner_join(., tc, by = c("Typegebruik" = "Identifier")) %>%
  dplyr::rename(., Type = Title) %>%
  dplyr::select(., PostCode6, Product, Type, Value) %>%
  dplyr::arrange(., PostCode6, Product, Type)
```

PostCode6	Product	Type		Value
1181NA	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		3030
1181NA	Gemiddelde aardgaslevering	Energielevering; particuliere woning		2870
1181NA	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		4650
1181NC	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		2420
1181NC	Gemiddelde aardgaslevering	Energielevering; particuliere woning		2290
1181NC	Gemiddelde aardgaslevering	Energielevering; zakelijke aansluiting		9400
1181NC	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		4480
1181NC	Gemiddelde elektriciteitslevering	Energielevering; zakelijke aansluiting		88600
1181ND	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		900
1181ND	Gemiddelde aardgaslevering	Energielevering; particuliere woning		850
1181ND	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		2210
1181NE	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		1700
1181NE	Gemiddelde aardgaslevering	Energielevering; particuliere woning		1610
1181NE	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		2800
1181NG	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		1720
1181NG	Gemiddelde aardgaslevering	Energielevering; particuliere woning		1630
1181NG	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		3040
1181NH	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		1230
1181NH	Gemiddelde aardgaslevering	Energielevering; particuliere woning		1160
1181NH	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		2590
1181NJ	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		1270
1181NJ	Gemiddelde aardgaslevering	Energielevering; particuliere woning		1200
1181NJ	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		2530
1181NK	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		2280
1181NK	Gemiddelde aardgaslevering	Energielevering; particuliere woning		2160
1181NK	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		4120
1181NL	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		1620
1181NL	Gemiddelde aardgaslevering	Energielevering; particuliere woning		1530
1181NL	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		3200
1181NM	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning		1750
1181NM	Gemiddelde aardgaslevering	Energielevering; particuliere woning		1650
1181NM	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning		3160

Table 9 – continued on next page

Table 9 – continued from previous page

PostCode6	Product	Type	Value
1181NN	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	2740
1181NN	Gemiddelde aardgaslevering	Energielevering; particuliere woning	2600
1181NN	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	5860
1181NP	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	2660
1181NP	Gemiddelde aardgaslevering	Energielevering; particuliere woning	2520
1181NP	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	4960
1181NR	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	1680
1181NR	Gemiddelde aardgaslevering	Energielevering; particuliere woning	1590
1181NR	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	3010
1181NS	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	1690
1181NS	Gemiddelde aardgaslevering	Energielevering; particuliere woning	1600
1181NS	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	2860
1181NT	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	2470
1181NT	Gemiddelde aardgaslevering	Energielevering; particuliere woning	2340
1181NT	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	2150
1181NV	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	1570
1181NV	Gemiddelde aardgaslevering	Energielevering; particuliere woning	1480
1181NV	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	3390
1181NW	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	1620
1181NW	Gemiddelde aardgaslevering	Energielevering; particuliere woning	1540
1181NW	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	3110
1181NX	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	1600
1181NX	Gemiddelde aardgaslevering	Energielevering; particuliere woning	1520
1181NX	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	2990
1181NZ	Gem. aardgaslevering klimaatgecorrigeerd	Energielevering; particuliere woning	1180
1181NZ	Gemiddelde aardgaslevering	Energielevering; particuliere woning	1120
1181NZ	Gemiddelde elektriciteitslevering	Energielevering; particuliere woning	2530
1181NZ	Gemiddelde elektriciteitslevering	Energielevering; zakelijke aansluiting	5210

Table 9: Energy consumption in 2017 for postcode 1181N*

Retrieve data in wide form

Because I know both the `MeasureCodes` and `TypegebruikCodes` tables are very small (see Table 10 on page 16 and Table 11 on page 16) I can combine the six combinations and do a `spread` for the variables. I use the data that was already gathered in `obs2`. See result in Table 12 on page 17.

Identifier	Title
M00000001	Gemiddelde aardgaslevering
M00000002	Gem. aardgaslevering klimaatgecorrigeerd
M00000003	Gemiddelde elektriciteitslevering

Table 10: MeasureCodes subtable (only Identifier and Title fields)

Identifier	Title
A045558	Energielevering; particuliere woning
A045559	Energielevering; zakelijke aansluiting

Table 11: TypegebruikCodes subtable (only Identifier and Title fields)

```
obs3 <- obs2 %>%
  dplyr::mutate(.,
    Code = dplyr::case_when(
      Measure == "M00000001" & Typegebruik == "A045558" ~ "Gas   P",
      Measure == "M00000002" & Typegebruik == "A045558" ~ "GasKC P",
      Measure == "M00000003" & Typegebruik == "A045558" ~ "Elec  P",
      Measure == "M00000001" & Typegebruik == "A045559" ~ "Gas   Z",
      Measure == "M00000002" & Typegebruik == "A045559" ~ "GasKC Z",
      Measure == "M00000003" & Typegebruik == "A045559" ~ "Elec  Z"
    )
  ) %>%
  dplyr::select(., PostCode6, Code, Value) %>%
  dplyr::group_by(., PostCode6) %>%
  tidyr::spread(., Code, Value) %>%
```

```
  dplyr::ungroup(.)
```

PostCode6	Elec P	Elec Z	Gas P	Gas Z	GasKC P
1181NA	4650		2870		3030
1181NC	4480	88600	2290	9400	2420
1181ND	2210		850		900
1181NE	2800		1610		1700
1181NG	3040		1630		1720
1181NH	2590		1160		1230
1181NJ	2530		1200		1270
1181NK	4120		2160		2280
1181NL	3200		1530		1620
1181NM	3160		1650		1750
1181NN	5860		2600		2740
1181NP	4960		2520		2660
1181NR	3010		1590		1680
1181NS	2860		1600		1690
1181NT	2150		2340		2470
1181NV	3390		1480		1570
1181NW	3110		1540		1620
1181NX	2990		1520		1600
1181NZ	2530	5210	1120		1180

Table 12: Energy consumption in 2017 for postcode 1181N* v2

Tests for implementation

Metadata Document Request

The indicated url now uses correctly the catalog `CBS-maatwerk`. The document that is returned for this url is not a `json` document but an `xml` document. I can't tell if the document is correct but it is readable.

```
meta1 <- get_beta(table_id = table_id, subtable = "$Metadata", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/$Metadata
```

```
class(meta1)
```

```
[1] "xml_document" "xml_node"
```

```
tmp_file <- fs::file_temp()
xml2::write_xml(meta1, tmp_file)
HOQCutil::cap.out(readLines(tmp_file))
```

```
  [1] "<?xml version=\"1.0\" encoding=\"UTF-8\"?>"
  [2] "<edmx:Edmx xmlns:edmx=\"http://docs.oasis-open.org/odata/ns/edmx\" Version=\"4.0\">"
  [3] "  <edmx:DataServices>"
  [4] "    <Schema xmlns=\"http://docs.oasis-open.org/odata/ns/edm\" Namespace=\"Cbs.Ccb.Models\">"
  [5] "      <EntityType Name=\"Dataset\">"
  [6] "        <Key>"
  [7] "          <PropertyRef Name=\"Identifier\"/>"
  [8] "        </Key>"
  [9] "        <Property Name=\"Identifier\" Type=\"Edm.String\" Nullable=\"false\"/>"
 [10] "        <Property Name=\"Title\" Type=\"Edm.String\" Nullable=\"false\"/>"
 [11] "        <Property Name=\"Description\" Type=\"Edm.String\" Nullable=\"false\"/>"
 [12] "        <Property Name=\"Modified\" Type=\"Edm.DateTimeOffset\" Nullable=\"false\"/>"
 [13] "        <Property Name=\"Language\" Type=\"Edm.String\" Nullable=\"false\"/>"
 [14] "        <Property Name=\"TemporalCoverage\" Type=\"Edm.String\"/>"
 [15] "        <Property Name=\"Authority\" Type=\"Edm.String\" Nullable=\"false\"/>"
 [16] "        <Property Name=\"Catalog\" Type=\"Edm.String\"/>"
 [17] "        <Property Name=\"Publisher\" Type=\"Edm.String\"/>"
 [18] "        <Property Name=\"ContactPoint\" Type=\"Edm.String\"/>"
 [19] "        <Property Name=\"Version\" Type=\"Edm.String\"/>"
 [20] "        <Property Name=\"VersionNotes\" Type=\"Edm.String\"/>"
 [21] "        <Property Name=\"VersionReason\" Type=\"Edm.String\"/>"
 [22] "        <Property Name=\"Frequency\" Type=\"Edm.String\"/>"
 [23] "        <Property Name=\"Provenance\" Type=\"Collection(Cbs.Ccb.Models.Link)\"/>"
```

```
[24] "        <Property Name=\"Status\" Type=\"Edm.String\"/>"
[25] "        <Property Name=\"ObservationCount\" Type=\"Edm.Int64\" Nullable=\"false\"/>"
[26] "        <Property Name=\"ObservationsModified\" Type=\"Edm.DateTimeOffset\" Nullable=\"false\"/>"
[27] "        <Property Name=\"DatasetType\" Type=\"Edm.String\"/>"
[28] "        <Property Name=\"RelatedSources\" Type=\"Collection(Cbs.Ccb.Models.Link)\"/>"
[29] "        <Property Name=\"DefaultPresentation\" Type=\"Edm.String\"/>"
[30] "        <Property Name=\"GraphTypes\" Type=\"Edm.String\"/>"
[31] "        <Property Name=\"SearchPriority\" Type=\"Edm.String\"/>"
[32] "        <Property Name=\"License\" Type=\"Edm.String\"/>"
[33] "        <Property Name=\"Source\" Type=\"Edm.String\"/>"
[34] "        <Property Name=\"Summary\" Type=\"Edm.String\"/>"
[35] "        <Property Name=\"LongDescription\" Type=\"Edm.String\"/>"
[36] "      </EntityType>"
[37] "      <EntityType Name=\"MeasureCode\">"
[38] "        <Key>"
[39] "          <PropertyRef Name=\"Identifier\"/>"
[40] "        </Key>"
[41] "        <Property Name=\"Identifier\" Type=\"Edm.String\" Nullable=\"false\"/>"
[42] "        <Property Name=\"Index\" Type=\"Edm.Int32\" Nullable=\"false\"/>"
[43] "        <Property Name=\"Title\" Type=\"Edm.String\"/>"
[44] "        <Property Name=\"Description\" Type=\"Edm.String\"/>"
[45] "        <Property Name=\"MeasureGroupID\" Type=\"Edm.String\"/>"
[46] "        <Property Name=\"DataType\" Type=\"Edm.String\"/>"
[47] "        <Property Name=\"Unit\" Type=\"Edm.String\"/>"
[48] "        <Property Name=\"Format\" Type=\"Edm.String\"/>"
[49] "        <Property Name=\"Decimals\" Type=\"Edm.Int16\" Nullable=\"false\"/>"
[50] "        <Property Name=\"PresentationType\" Type=\"Edm.String\"/>"
[51] "      </EntityType>"
[52] "      <EntityType Name=\"Dimension\">"
[53] "        <Key>"
[54] "          <PropertyRef Name=\"Identifier\"/>"
[55] "        </Key>"
[56] "        <Property Name=\"Identifier\" Type=\"Edm.String\" Nullable=\"false\"/>"
[57] "        <Property Name=\"Title\" Type=\"Edm.String\"/>"
[58] "        <Property Name=\"Description\" Type=\"Edm.String\"/>"
[59] "        <Property Name=\"Kind\" Type=\"Edm.String\"/>"
[60] "        <Property Name=\"MapYear\" Type=\"Edm.String\"/>"
```

```
[61] "        <Property Name=\"ReleasePolicy\" Type=\"Edm.String\"/>"
[62] "      </EntityType>"
[63] "      <EntityType Name=\"DimensionCode\">"
[64] "        <Key>"
[65] "          <PropertyRef Name=\"Identifier\"/>"
[66] "        </Key>"
[67] "        <Property Name=\"Identifier\" Type=\"Edm.String\" Nullable=\"false\"/>"
[68] "        <Property Name=\"Index\" Type=\"Edm.Int32\" Nullable=\"false\"/>"
[69] "        <Property Name=\"Title\" Type=\"Edm.String\"/>"
[70] "        <Property Name=\"Description\" Type=\"Edm.String\"/>"
[71] "        <Property Name=\"DimensionGroupID\" Type=\"Edm.String\"/>"
[72] "      </EntityType>"
[73] "      <ComplexType Name=\"Link\">"
[74] "        <Property Name=\"Title\" Type=\"Edm.String\"/>"
[75] "        <Property Name=\"Url\" Type=\"Edm.String\"/>"
[76] "      </ComplexType>"
[77] "    </Schema>"
[78] "    <Schema xmlns=\"http://docs.oasis-open.org/odata/ns/edm\" Namespace=\"Cbs.Ccb.MemModels\">"
[79] "      <EntityType Name=\"O900001NED201810180900\">"
[80] "        <Key>"
[81] "          <PropertyRef Name=\"Id\"/>"
[82] "        </Key>"
[83] "        <Property Name=\"Id\" Type=\"Edm.Int64\" Nullable=\"false\"/>"
[84] "        <Property Name=\"Measure\" Type=\"Edm.String\"/>"
[85] "        <Property Name=\"ValueAttribute\" Type=\"Edm.String\"/>"
[86] "        <Property Name=\"Value\" Type=\"Edm.Double\"/>"
[87] "        <Property Name=\"PostCode6\" Type=\"Edm.String\"/>"
[88] "        <Property Name=\"Typegebruik\" Type=\"Edm.String\"/>"
[89] "      </EntityType>"
[90] "    </Schema>"
[91] "    <Schema xmlns=\"http://docs.oasis-open.org/odata/ns/edm\" Namespace=\"Default\">"
[92] "      <EntityContainer Name=\"Container\">"
[93] "        <EntitySet Name=\"MeasureCodes\" EntityType=\"Cbs.Ccb.Models.MeasureCode\"/>"
[94] "        <EntitySet Name=\"Dimensions\" EntityType=\"Cbs.Ccb.Models.Dimension\"/>"
[95] "        <EntitySet Name=\"TypegebruikCodes\" EntityType=\"Cbs.Ccb.Models.DimensionCode\"/>"
[96] "        <EntitySet Name=\"Observations\" EntityType=\"Cbs.Ccb.MemModels.O900001NED201810180900\"/>"
[97] "        <Singleton Name=\"Properties\" Type=\"Cbs.Ccb.Models.Dataset\"/>"
```

```
 [98] "      </EntityContainer>"
 [99] "    </Schema>"
[100] "  </edmx:DataServices>"
[101] "</edmx:Edmx>"
```

```
d <- fs::file_delete(tmp_file)
```

Service Document Request

The indicated url now uses correctly the catalog `CBS-maatwerk`. The following code is the same as used to produce Table 4 on page 6.

```
subtables1 <- get_beta(table_id = "900001NED", verbose = T)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED
```

Query options `$filter(startswith)`

We used this filter above on subtable `Observations` to create e.g. in Table 9 on page 15. It also works on e.g. `MeasureCodes` (originally 3 rows) where with the following code 1 row is selected:

```
mc2 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$filter=startswith(Title,'Gem.')"
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$filter=startswith(Title,'Gem.')
unencoded query: $filter=startswith(Title,'Gem.')
```

```
nrow(mc2)
```

```
[1] 1
```

Query options `$count`

Apparently `$count` can only be used on a full (sub) table: it can't be used to find the number of rows after a query.

```
mc2 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$filter=startswith(Title,'Gem.')&$count", error_msg = T
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$filter=startswith(Title,'Gem.')&$count
unencoded query: $filter=startswith(Title,'Gem.')&$count

error message (get_table_cbs_odata4_GET) :
The query specified in the URI is not valid. The value for OData query '$count' cannot be empty.
```

```
(mc2)
```

```
[1] "Client error: (400) Bad Request"
```

On a full table it seems to work fine. Remark: the generated url should not have the question mark!

```
mc3 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$count"
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes/$count
unencoded query: $count
```

```
print(mc3)
```

```
[1] "3"
```

Query options `(id)`

Apparently `(id)` can only be used on the `Observations` subtable: this is also the only table with an `id` field.

```
obs_id <- 23
obs4 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = glue::glue("({obs_id})"), response = TRUE
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations(23)
unencoded query: (23)
```

```
jsonlite::prettify(httr::content(obs4, as = "text"))
```


```
{
    "@odata.context": "https://acc-ccb.cbs.nl/OData4/CBS-Maatwerk/900001NED/$metadata#Observations/$entity",
    "Id": 23,
    "Measure": "M00000001",
    "ValueAttribute": "None",
    "Value": 1040.0,
    "PostCode6": "1011BW",
    "Typegebruik": "A045558"
}
```


In the example above (notice I used `response=TRUE`) I request the observation with number '23'. The result is not a table but a json object. Wondering why, I did post a question about this in the LinkedIn group. Here I also stated the following alternative query and list the object in json format. With `response=FALSE` (the default) a table object would be returned where `response=TRUE` returns the `httr` response object.

```
obs5 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = glue::glue("$skip={obs_id-1}&$top=1"), response = TRUE
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$skip=22&$top=1
unencoded query: $skip=22&$top=1
```

```
jsonlite::prettify(httr::content(obs5, as = "text"))
```


```
{
    "@odata.context": "https://acc-ccb.cbs.nl/OData4/CBS-Maatwerk/900001NED/$metadata#Observations",
    "value": [
        {
            "Id": 23,
            "Measure": "M00000001",
            "ValueAttribute": "None",
            "Value": 1040.0,
            "PostCode6": "1011BW",
            "Typegebruik": "A045558"
        }
    ]
}
```


The answer on my question is that this works as designed. Because I prefer to get the same answer in both cases I adapted the `HOQCutil::get_table_cbs_odata4` function to account for this. Table 13 on page 25 shows that this works now.

```
obs5 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = glue::glue("({obs_id})")
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations(23)
unencoded query: (23)
```

`$format` option

Apparently this option has no influence on the result of queries. In the info document it is said that only json documents are returned. Apparently this is not the case described in Metadata Document Request where a json document is requested and an xml-document is returned.

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
23	M00000001	None	1040	1011BW	A045558

Table 13: Result of id is table after adaption of get_table_cbs_odata4

$top and $skip option

These options work for the **Observations** subtable as shown in Query options (id). They apparently also work for the **MeasureCodes** subtable as show in Table 14 on page 25.

```
mc4 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes",
  query = "$skip=1&$top=1", verbose = T
) %>%
  dplyr::select(., Identifier, Title)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$skip=1&$top=1
unencoded query: $skip=1&$top=1
```

Identifier	Title
M00000002	Gem. aardgaslevering klimaatgecorrigeerd

Table 14: $skip and $top done on MeasureCodes subtable

$filter (And), (Or) , (Less Than) , (Greater Than), (Equals), (Less Than or Equal), (Greater Than or Equal)

Table 15 on page 26 shows the results of a query with (most of) the logical and comparison operators. It is remarkable (?) that the ordering appears to be alphabetically. For a numerical ordering the `$orderby` option helps in this case: See Table 16 on page 27.

```r
obs6 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = "$filter=(Id lt 6 and Id gt 3) or (Id eq 8) or (Id le 15 and Id ge 12)"
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=(Id%20lt%206%20and%20Id%20gt%203)%20or
%20(Id%20eq%208)%20or%20(Id%20le%2015%20and%20Id%20ge%2012)
unencoded query: $filter=(Id lt 6 and Id gt 3) or (Id eq 8) or (Id le 15 and Id ge 12)
```

```r
obs7 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = "$filter=(Id lt 6 and Id gt 3) or (Id eq 8) or (Id le 15 and Id ge 12)&$orderby=Id asc"
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=(Id%20lt%206%20and%20Id%20gt%203)%20or
%20(Id%20eq%208)%20or%20(Id%20le%2015%20and%20Id%20ge%2012)&$orderby=Id%20asc
unencoded query: $filter=(Id lt 6 and Id gt 3) or (Id eq 8) or (Id le 15 and Id ge 12)&$orderby=Id asc
```

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
12	M00000001	None	1190	1011BC	A045558
13	M00000001	None	1040	1011BD	A045558
14	M00000001	None	1380	1011BH	A045558
15	M00000001	None	1310	1011BJ	A045558
4	M00000001	None	930	1011AN	A045558
5	M00000001	None	1250	1011AP	A045558
8	M00000001	None	1200	1011AW	A045558

Table 15: Test for and, or, lt, gt, eq, le, ge

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
4	M00000001	None	930	1011AN	A045558
5	M00000001	None	1250	1011AP	A045558
8	M00000001	None	1200	1011AW	A045558
12	M00000001	None	1190	1011BC	A045558
13	M00000001	None	1040	1011BD	A045558
14	M00000001	None	1380	1011BH	A045558
15	M00000001	None	1310	1011BJ	A045558

Table 16: Test for and, or, lt, gt, eq, le, ge but now sorted

$filter (Not Equals), mod

Table 17 on page 28 shows the results of a query with the `ne` operator and the `mod` function.

```
obs8 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = "$filter=(Id mod 5000) eq 1 and Id ne 10001"
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=(Id%20mod%205000)%20eq%201%20and%20Id%
20ne%2010001
unencoded query: $filter=(Id mod 5000) eq 1 and Id ne 10001
```

```
dim(obs8)
```

```
[1] 243   6
```

$filter (Not), (endswith) and (startswith)

Table 18 on page 28 shows the results of a query using `not` in combination with `endswith`. The use of `startswith` was shown in section Retrieve data in long form.

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
1	M00000001	None	2060	1011AC	A045558
5001	M00000001	None	890	1053NP	A045558
15001	M00000001	None	1360	1106PZ	A045558
20001	M00000001	None	1970	1191EG	A045558
25001	M00000001	None	0	1314WG	A045558
30001	M00000001	None	1670	1394LG	A045558

Table 17: Test for ne and mod (first 6 rows only)

```
obs9 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$filter=not endswith(Identifier,'2')"
) %>%
  dplyr::select(Identifier, Title)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$filter=not%20endswith(Identifier,'2')
unencoded query: $filter=not endswith(Identifier,'2')
```

Identifier	Title
M00000001	Gemiddelde aardgaslevering
M00000003	Gemiddelde elektriciteitslevering

Table 18: Test for not and endswith

The following query only works (Table 19 on page 29) when parentheses are used.

```
obs10 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = "$filter=not (Id gt 25) and Id mod 5 eq 1"
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=not%20(Id%20gt%2025)%20and%20Id%20mod%
205%20eq%201
unencoded query: $filter=not (Id gt 25) and Id mod 5 eq 1
```

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
1	M00000001	None	2060	1011AC	A045558
6	M00000001	None	1170	1011AR	A045558
11	M00000001	None	1030	1011BA	A045558
16	M00000001	None	1300	1011BK	A045558
21	M00000001	None	1090	1011BT	A045558

Table 19: Test for not and mod

Without the parentheses the expression is apparently interpreted as `(not Id) gt 25` . Coding the query as

```
obs11 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = "$filter=not Id gt 25 and Id mod 5 eq 1", error_msg = T
)
```

gives the following error message:

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=not%20Id%20gt%2025%20and%20Id%20mod%20
5%20eq%201
unencoded query: $filter=not Id gt 25 and Id mod 5 eq 1
```

```
error message (get_table_cbs_odata4_GET) :
The query specified in the URI is not valid. A unary operator with an incompatible type was detected. Found operand type
'Edm.Int64' for operator kind 'Not'.
```

$filter (indexof)

Table 20 on page 30 shows the rows of subtable `MeasureCodes` that have the characters `gas` in the `Title` field starting in index position 9 (i.e. the 10th letter)

```
mc5 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$filter=indexof(Title,'gas') eq 9", error_msg = T
) %>%
  dplyr::select(Identifier, Title)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$filter=indexof(Title,'gas')%20eq%209
unencoded query: $filter=indexof(Title,'gas') eq 9
```

Identifier	Title
M00000002	Gem. aardgaslevering klimaatgecorrigeerd

Table 20: Test for indexof

$filter (tolower), (toupper)

Table 21 on page 31 shows the result of a query done on the upper and lower case version of postcodes.

```
obs12 <- get_beta(
  table_id = "900001NED", subtable = "Observations", verbose = T,
  query = "$filter=tolower(Postcode6) eq '1181nz' or toupper(Postcode6) eq '1011BA' ", error_msg = T
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/Observations?$filter=tolower(Postcode6)%20eq%20'1181nz'%20o
r%20toupper(Postcode6)%20eq%20'1011BA'%20
unencoded query: $filter=tolower(Postcode6) eq '1181nz' or toupper(Postcode6) eq '1011BA'
```

Id	Measure	ValueAttribute	Value	PostCode6	Typegebruik
11	M00000001	None	1030	1011BA	A045558
408831	M00000002	None	1080	1011BA	A045558
782584	M00000003	None	2980	1011BA	A045558
18393	M00000001	None	1120	1181NZ	A045558
427213	M00000002	None	1180	1181NZ	A045558
801428	M00000003	None	2530	1181NZ	A045558
1162761	M00000003	None	5210	1181NZ	A045559

Table 21: Test for toupper and tolower

However it looks like nesting of functions is not allowed because the following query results in an incomplete json string:

```
mc6 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$filter=indexof(tolower(Title),'gas') eq 9", error_msg = T
)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$filter=indexof(tolower(Title),'gas')%20eq%209
unencoded query: $filter=indexof(tolower(Title),'gas') eq 9

error in json ? (get_table_cbs_odata4_GET) :
{"@odata.context":"https://acc-ccb.cbs.nl/OData4/CBS-Maatwerk/900001NED/$metadata#MeasureCodes","value":[
```

$filter (length)

Table 22 on page 32 shows the result of a query done with a `length` selection.

```
mc7 <- get_beta(
  table_id = "900001NED", subtable = "MeasureCodes", verbose = T,
  query = "$filter=length(Title) gt 30", error_msg = T
) %>%
  dplyr::select(Identifier, Index, Title)
```

```
generated url  : https://acc-ccb.cbs.nl/OData4//CBS-Maatwerk/900001NED/MeasureCodes?$filter=length(Title)%20gt%2030
unencoded query: $filter=length(Title) gt 30
```

Identifier	Index	Title
M00000002	2	Gem. aardgaslevering klimaatgecorrigeerd
M00000003	3	Gemiddelde elektriciteitslevering

Table 22: Test with length filter

SessionInfo

```
sessionInfo()
```

```
R version 3.5.2 (2018-12-20)
Platform: x86_64-w64-mingw32/x64 (64-bit)
Running under: Windows 10 x64 (build 17134)

Matrix products: default

locale:
[1] LC_COLLATE=English_United States.1252  LC_CTYPE=English_United States.1252    LC_MONETARY=English_United States.1252
[4] LC_NUMERIC=C                           LC_TIME=English_United States.1252

attached base packages:
```

```
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] bindrcpp_0.2.2 HOQCutil_0.1.9

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.0       rstudioapi_0.8   xml2_1.2.0.9000  bindr_0.1.1      knitr_1.21       rematch2_2.0.1   magrittr_1.5
 [8] tidyselect_0.2.5 xtable_1.8-3     R6_2.3.0         rlang_0.3.0.1    dplyr_0.7.8      stringr_1.3.1    styler_1.1.0
[15] httr_1.4.0       tools_3.5.2      xfun_0.3         htmltools_0.3.6  assertthat_0.2.0 yaml_2.2.0       digest_0.6.17
[22] tibble_1.4.2     crayon_1.3.4     tidyr_0.8.2      purrr_0.2.5      fs_1.2.6         curl_3.2         mime_0.5
[29] glue_1.3.0       evaluate_0.11    rmarkdown_1.11   stringi_1.2.4    compiler_3.5.2   pillar_1.3.0     backports_1.1.2
[36] jsonlite_1.6     pkgconfig_2.0.2
```

Links in this document

webpage :
https://beta.opendata.cbs.nl/OData4/index.html
LinkedIn Post :
https://www.linkedin.com/feed/update/urn:li:activity:6476445191598276608
Release Notes :
https://acc-ccb.cbs.nl/OData4/releasenotes.html
Centraal Bureau voor de Statistiek Open Data :
https://www.linkedin.com/groups/5190698/
my question :
https://www.linkedin.com/feed/update/urn:li:activity:6449917998194008064/
info page :
https://beta.opendata.cbs.nl/OData4/info.html
OData protocol v4 :
http://docs.oasis-open.org/odata/odata/v4.0/odata-v4.0-part1-protocol.html
Implementatie OData v4 functies :
https://beta.opendata.cbs.nl/OData4/implement.html
LinkedIn group :
https://www.linkedin.com/feed/update/urn:li:activity:6451478262030835713