LinkedPipes FDPtoRDF pipeline

developer documentation

Table of contents:

Overview of known issues to be solved/discussed

FDP Fiscal period property

- Not yet implemented
- Clear mapping to obeu-dimension:fiscalPeriod, if we can rely on the FDP values being xs:date formatted

FDP date type dimensions

- Not yet implemented
- Cannot support arbitrary formatted dates
- Can we agree on supporting only xs:date or xs:dateTime values?

FDP entity type dimensions

- Currently mapped to subproperties of obeu-dimension:organization which is fine by me
- Just pointing out that OBEU has more fine grained resolution capabilities: differentiating entities to obeu-dimension:budgetaryUnit and obeu-dimension:partner according to their relation to the payment/revenue

Hierarchical classifications

- The pipeline tries to transform it into skos hierarchy
- The qb:Observation is linked to the most specific classification (which should be linked to more general ones)
  - Currently does not take into account a situation where the most specific classification is not defined for some observation (is missing in some source CSV row)
  - Example:

Amount	Level1_title	Level2_title
1000	Education	Higher Education
3000	Education	-

  -

Normalized tables and referencing through multi-field foreign keys

- One-field foreign keys are alright
- Multi-field foreign keys are also supported through a kind of hack that does not take order of the fields into account (leading to possibly linking "John Jack" key to "Jack John", which is probably a rare situation)
- Could be made 100% accurate if JSON Table Schema contained indexes along with foreign key field names

Example: instead of

```
{
"foreignKeys": [
        {
          "fields": ["name", "surname"],
          "reference": {
            "datapackage": "my-openspending-datapackage",
            "resource": "entities",
            "fields": ["person_name", "person_surname"]
          }
        }
      ]
}
```

We would like it to be something like

```
{
"foreignKeys": [
        {
          "fields": [
                {
                  "name": "name",
                  "Index": "1"
                },
                {
                  "Name": "surname",
                  "Index": "2",
                }
           ],
          "reference": {
            "datapackage": "my-openspending-datapackage",
            "resource": "entities",
            "fields": [
                {
                  "name": "person_name",
                  "Index": "1"
                },
                {
                  "Name": "person_surname",
                  "Index": "2"
                }
        ]
      }
            }
        ]
}
```

Namespaces and URI generation

- I tried to follow OBEU style of namespaces used in existing examples
- The base URI for newly created entities is however temporarily "http://test.openbudgets.eu/"
    - That should be changed to something better
    - Would be better if it is also dereferencable

Loading the results into Virtuoso

- The pipeline loads the results always into the same graph
- For debugging purposes, the graph is overwritten during each new run of the pipeline
- Would be probably nicer if the DPU was configurable and we would name the graph according the datapackage name

There are also some minor parts of FDP that are not implemented yet, but there should be no problem with it, it is just a matter of time (mentioned in the next section)

High-level mapping overview as comments in FDP specification

To make more clear what is mapped and what not, I took the original FDP specification from http://fiscal.dataprotocols.org/spec/ and added comments to parts that are mapped by the pipeline. More clarification can be brought by the examples further below.

Legend:

Implemented mapping (blue bold font)
Mapping to be implemented (no problems expected) (red bold font)
To be implemented, discussion might be needed (red with yellow background)
Everything else is the original FDP specification

```
// REQUIRED (DataPackage): a url-compatible short name ("slug")
  // for the package
  "name": "Australia2014",
```

-> used in URI of qb:DataSet

```
  // REQUIRED (DataPackage): a human readable title for the package
  "title": "Australian annual budget 2013-14",
```

TBD -> qb:DataSet dc:title (no problem)

```
  // RECOMMENDED (DataPackage): the license for the data in this
  // package.
  "license": "cc-by 3.0",
```

-> TBD map to dc-terms license

```
  // RECOMMENDED: other properties such as description, homepage,
  // version, sources, author, contributors, keywords, as specified
  // in dataprotocols.org/data-packages/

  // RECOMMENDED: a valid 2-digit ISO country code (ISO 3166-1
  // alpha-2), or, an array of valid ISO codes (if this relates to
  // multiple countries). This field is for listing the country of
  // countries associated to this data.  For example, if this the
  // budget for country then you would put that country's ISO code.
  "countryCode": "au", // or [ "au", "nz" ]
- don't know where to map yet - do we need it?
```

-> TBD map to cl-geo like http://data.openbudgets.eu/resource/codelist/cl-geo/CZ

```
  // RECOMMENDED: the "profile set" for this package. If the
// `profiles` key is present, it MUST be set to the following
// hash:
"profiles": {
  "fiscal": "*",
  "tabular": "*"
},

// OPTIONAL: a keyword that represents the type of spend data:
//   * "transaction": rows have dates and correspond to
//     individual transactions
//   * "aggregated": rows are summaries of expenditure across a
//     fiscal period
"granularity": "aggregated",

// OPTIONAL: the fiscal period of the dataset
"fiscalPeriod": {
  "start": "1982-04-22",
  "end": "1983-04-21"
```

```
  },
```
TBD -> obeu-dimension:fiscalPeriod
```
  // OPTIONAL: ...other properties...

  // REQUIRED: array of CSV files contained in the package. Defined
  // in http://dataprotocols.org/data-packages/ and
  // http://dataprotocols.org/tabular-data-package/ . Note:
  //   * Each data file `MUST` have an entry in the `resources`
  //     array
  //   * That entry in the `resources` array `MUST` have a JSON
  //     table schema describing the data file.
  //     (see http://dataprotocols.org/json-table-schema/)
  //    * Each entry must have a `name` attribute in order to be referenced
  //      in the `model` section.

  "resources": [ /* ... */ ],

  // REQUIRED, see "Model"
  "model": {
    // REQUIRED: the measures object in logical model
    "measures": {
      /* ... */ // REQUIRED at least 1: see "Measures"
    },
```
-> subProperty of obeu-measure:amount; when multiple measures, a separate DSD (separate "cube") is created for each; each amount value in separate qb:Observation
```

    // REQUIRED: the dimensions object in logical model
    "dimensions": {
      /* ... */ // REQUIRED at least 1: see "Dimensions"
    }
  }

"measures": {
  "measure-name": {
    // REQUIRED: Name of source field
    "source": "amount",

    // REQUIRED: Any valid ISO 4217 currency code.
    "currency": "USD",
```
-> [qb:DataSet] obeu-attribute:currency http://dbpedia.org/resource/[currency code]

- will be changed to OKFGR code list (https://github.com/openbudgets/Code-lists/tree/master/UnifiedViews/skosified/currencies)
```

    // OPTIONAL: A factor by which to multiple the raw monetary
    // values to get the real monetary amount, eg `1000`. Defaults
    // to `1`.
    "factor": 1,
```
Used to multiply measure values when transformed
```

    // OPTIONAL: Resource (referenced by `name` attribute) containing the source field. Defaults to
    // the first resource in the `resources` array.
    "resource": "budget-2014-au",

    // OPTIONAL: A keyword that represents the *direction* of the
    // spend, being one of "expenditure" or "revenue".
    "direction": "expenditure",
```

-> [qb:DataSet] obeu-dimension:operationCharacter obeu-operation:expenditure or obeu-operation:revenue

```
    // OPTIONAL: The phase of the budget that the values in this
    // measure relate to. It `MUST` be one of the following strings:
    // proposed, approved, adjusted, executed.
    "phase": "proposed",
```

TBD -> obeu-budgetphase:draft, approved, revised, executed

```
    // OPTIONAL: Other properties allowed.
  }
  //...
}

"dimensions": {
  "project-class": {
    // REQUIRED: An attributes object that defines the attributes of the
    // dimension. Think of each attribute as a column on that dimension in
    // a database. An attribute MUST have `source` information -
    // i.e. where the data comes from for that property
    "attributes": {
      "project": {
        // REQUIRED:
        // EITHER: the field name where the value comes from for
        // this property (see "Describing Sources" above);
        "source": "proj",
        // OR: a single value that applies for all rows of the
        // dataset.
        "constant": "Some Project",

        // OPTIONAL: the resource (referenced by `name` attribute) in which the field is located.
        // Defaults to the first resource in the `resources` array.
        "resource": "budget-2014-au"

        // OPTIONAL: the key referencing an attribute within this
        // dimension (if it exists) for which this attribute
        // provides a label.  For instance, given two dimension
        // attributes named "project_code" and "project_label",
        // the attribute "project_label" will provide a "labelfor"
        // pointing to "project_code"
        "labelfor": "..."
      },
      "code": {
        "source": "class_code"
      }
    },

    // REQUIRED: Either an array of strings corresponding to the
    // attribute keys in the `attributes` object or a single string
    // corresponding to one of these. The value of `primaryKey`
    // indicates the primary key or primary keys for the dimension.

    "primaryKey": ["project", "code"],

    // OPTIONAL: Describes what kind of a dimension it is.
    // `dimensionType` is a string that `MUST` be one of the
    // following:
    //
    // * "datetime": the date of a transaction
```

TBD -> obeu-dimension:fiscalPeriod or obeu-dimension:date - decide by FDP granularity property; will require xs:dateTime or other standard format(s)

```
    // * "entity": names the organisation doing the spending or
    //   receiving
```

-> subProperty of obeu-dimension:organization

```
    // * "classification": one or more fields that create a
    //   categorical hierarchy of the type of spending (eg:
    //   Health > Hospital services > Nursing). Combine with
    //   `classificationType` for greater expressiveness.
```

-> subProperty of obeu-dimension:classification, skos values (if classificationType not provided)

```
    // * "activity": names a specific programme or project under
    //   which the money is spent
```

-> subProperty of obeu-dimension:programmeClassification, skos values

```
    // * "fact": an attribute such as an ID or reference number
    //   attached to a transaction
```

-> subProperty of general qb:AttributeProperty, literal values

-> will be changed to subProperty of obeu:OptionalProperty

```
    // * "location": the geographical location where money is spent
```

-> subProperty of obeu-attribute:location, literal values

-> TBD schema:Place values linked to literals with schema:name

```
    // * "other": not one of the above
    "dimensionType": "classification",
```

-> subProperty of obeu-dimension:classification

```
    // RECOMMENDED (if using dimensionType="classification"). The
    // basis on which transactions are being classified, one of
    // these values:
    //
    // * "administrative": an organisational structure, such as
    //   Portfolio > Department > Branch
```

-> subProperty of obeu-dimension:administrativeClassification, skos values

```
    // * "functional": the purpose of the spending, such as
    //   Health > Hospital services > Nursing
```

-> subProperty of obeu-dimension:functionalClassification, skos values

```
    // * "economic": focused on the nature of the accounting, such
    //   as Compensation > Wages and salaries > Wages and salaries
    //   in cash
```

-> subProperty of obeu-dimension:economicClassification, skos values

```
    "classificationType": "administrative"
```

-> subProperty of obeu-dimension:administrativeClassification, skos values

```
    // OPTIONAL: Other properties allowed.
```

-> subProperty of general qb:DimensionProperty, literal values

```
  }
  //...
```

Example mappings of examples from FDP specification

http://fiscal.dataprotocols.org/examples/minimal/

```
pk,budget,budget_date
```

```
1,10000,01/01/2015
```

```
{
 "name": "my-openspending-datapackage",
 "title": "My OpenSpending Data Package",
 "resources": [
  {
   "name": "budget",
   "title": "Budget",
   "path": "budget.csv",
   "schema": {
    "fields": [
     {
      "name": "expenditure",
      "type": "number",
      "format": "currency"
     },
     {
      "name": "year",
      "type": "date"
     }
    ]
   }
  }
 ],
 "model": {
  "measures": {
   "amount": {
    "source": "expenditure",
    "currency": "USD",
    "factor": 1
   }
  },
  "dimensions": {
   "date": {
    "attributes": {
     "year": {
      "source": "year"
     }
    },
    "primaryKey": "year"
   }
  }
 }
}
```

RDF result

```
_:node1akqteleix1197 a qb:Observation ;
      obeu-measure:amount 10000.0 ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/date> "01/01/2015" ;
      qb:dataSet <http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> qb:observation _:node1akqteleix1197 .

<http://test.openbudgets.eu/dsd/my-openspending-datapackage-amount> qb:component _:node1alf6vmfax3 .

_:node1alf6vmfax3 qb:dimension <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/date> ;
      fdprdf:attribute _:node1alf6vmfax4 ;
      fdprdf:valueType fdprdf:unknown .

<http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/date> a rdf:Property
,qb:DimensionProperty ;
      fdprdf:name "date" .
```

http://fiscal.dataprotocols.org/examples/labels-and-hierarchies/

```
level1_id,level1_title,level2_id,level2_title,amount,budget_date
09,Education,0901,Higher Education,10000,01/01/2015
```

```
{
  "name": "my-openspending-datapackage",
  "title": "My OpenSpending Data Package",
  "resources": [
    {
      "name": "budget",
      "title": "Budget",
      "path": "budget.csv",
      "schema": {
        "fields": [
          {
            "name": "level1_id",
            "type": "string"
          },
          {
            "name": "level1_title",
            "type": "string"
          },
          {
            "name": "level2_id",
            "type": "string"
          },
          {
            "name": "level2_title",
            "type": "string"
          },
          {
            "name": "amount",
            "type": "number"
          },
          {
            "name": "budget_date",
            "type": "date"
          }
        ]
      }
    }
  ],
  "model": {
    "measures": {
      "amount": {
        "source": "amount",
        "currency": "USD",
        "factor": 1
      }
    },
    "dimensions": {
      "date": {
        "attributes": {
          "date": {
            "source": "budget_date"
          }
        },
        "primaryKey": "date"
      },
      "classification": {
        "attributes": {
          "level1_id": {
            "source": "level1_id"
          },
          "level1_title": {
            "source": "level1_title",
            "labelfor": "level1_id"
          },
          "level2_id": {
            "source": "level2_id",
            "parent": "level1_id"
          },
          "level2_title": {
            "source": "level2_title",
            "labelfor": "level2_id"
          }
        },
        "primaryKey": ["level1_id", "level2_id"]
      }
    }
```

```
 }
}
```

RDF result

```
_:node1akqteleix1959 a qb:Observation ;
       obeu-measure:amount 10000.0 ;
       <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/date> "01/01/2015" ;
       <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/classification>
<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/level2_id/0901> ;
       qb:dataSet <http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> qb:observation _:node1akqteleix1959 .

<http://test.openbudgets.eu/dsd/my-openspending-datapackage-amount> qb:component _:node1alfc5t7bx3 , _:node1alfc5t7bx4 .

_:node1alfc5t7bx3 qb:dimension <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/date> .

<http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/date> a rdf:Property , qb:DimensionProperty ;
       fdprdf:name "date" .

_:node1alfc5t7bx4 qb:dimension <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/classification>
.

<http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/classification> a rdf:Property ,
qb:DimensionProperty ;
       fdprdf:name "classification" .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/level1_id/09> a skos:Concept ;
       skos:inScheme <http://test.openbudgets.eu/resource/my-openspending-datapackage/codelist/classification/> ;
       skos:prefLabel "Education" ;
       fdprdf:sourceRow _:node1akqteleix1959 ;
       fdprdf:valueOf _:node1alfc5t7bx5 ;
       skos:notation "09" .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/level2_id/0901> a skos:Concept ;
       skos:inScheme <http://test.openbudgets.eu/resource/my-openspending-datapackage/codelist/classification/> ;
       skos:prefLabel "Higher Education" ;
       fdprdf:sourceRow _:node1akqteleix1959 ;
       fdprdf:valueOf _:node1alfc5t7bx7 ;
       skos:notation "0901" ;
       skos:broader <http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/level1_id/09> .
```

http://fiscal.dataprotocols.org/examples/entities-normalized/

```
pk,amount,year,payee
1,10000,01/01/2015,1
2,20000,01/02/2015,1

id,title,description
1,Acme 1,They are the first acme company
2,Acme 2,They are the sceond acme company

{
 "name": "my-openspending-datapackage",
 "title": "My OpenSpending Data Package",
 "resources": [
  {
   "name": "budget",
   "title": "Budget",
   "path": "budget.csv",
   "schema": {
    "fields": [
     {
      "name": "id",
      "type": "string"
     },
     {
      "name": "amount",
      "type": "number",
      "format": "currency"
     },
     {
      "name": "date",
      "type": "date"
     },
     {
      "name": "payee",
      "type": "string"
     }
    ],
    "primaryKey": "id"
```

```
    },
    "foreignKeys": [
      {
        "fields": "payee",
        "reference": {
          "datapackage": "my-openspending-datapackage",
          "resource": "entities",
          "fields": "id"
        }
      }
    ]
  },
  {
    "name": "entities",
    "title": "Entities",
    "path": "entities.csv",
    "schema": {
      "fields": [
        {
          "name": "id",
          "type": "string"
        },
        {
          "name": "title",
          "type": "string"
        },
        {
          "name": "description",
          "type": "string"
        }
      ],
      "primaryKey": "id"
    }
  }
],
"model": {
  "measures": {
    "amount": {
      "source": "budget",
      "currency": "USD",
      "factor": 1
    }
  },
  "dimensions": {
    "date": {
      "dimensionType": "datetime",
      "attributes": {
        "year": {
          "source": "year"
        }
      },
      "primaryKey": "year"
    },
    "payee": {
      "dimensionType": "entity",
      "attributes": {
        "id": {
          "resource": "entities",
          "source": "id"
        },
        "title": {
          "resource": "entities",
          "source": "title"
        },
        "description": {
          "resource": "entities",
          "source": "description"
        }
      },
      "primaryKey": "id"
    }
  }
}
}
```

RDF result

```
_:node1akqteleix3103 a qb:Observation ;
      obeu-measure:amount 10000.0 ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/payee>
<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/payee/1> ;
      qb:dataSet <http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> .

_:node1akqteleix3104 a qb:Observation ;
      obeu-measure:amount 20000.0 ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/payee>
<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/payee/1> ;
      qb:dataSet <http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> .

_:node1akqteleix3106 <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/payee>
<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/payee/1> .

_:node1akqteleix3107 <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/payee>
<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/payee/2> .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount> qb:observation _:node1akqteleix3103 ,
_:node1akqteleix3104 .

<http://test.openbudgets.eu/dsd/my-openspending-datapackage-amount> qb:component _:node1alff7r7cx6 .

_:node1alff7r7cx6 qb:dimension <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/payee> ;
      fdprdf:valueType fdprdf:organization .

<http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/dimension/payee> a rdf:Property , qb:DimensionProperty ;
      rdfs:subPropertyOf obeu-dimension:organization ;
      fdprdf:name "payee" .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/payee/1>
<http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/partialproperty/id> "1" ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/partialproperty/title> "Acme 1" ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/partialproperty/description> "They are the first
acme company" ;
      fdprdf:sourceRow _:node1akqteleix3106 .

<http://test.openbudgets.eu/datasets/my-openspending-datapackage-amount/payee/2>
<http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/partialproperty/id> "2" ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/partialproperty/title> "Acme 2" ;
      <http://test.openbudgets.eu/ontology/dsd/my-openspending-datapackage/partialproperty/description> "They are the sceond
acme company" ;
      fdprdf:sourceRow _:node1akqteleix3107 .
```

Description of the transformation steps

Introduction

The whole pipeline consists of a series of SPARQL CONSTRUCT query nodes. The output of each node needs to be linked to all other nodes that use it - that is the reason why there are so many links. This could be simplified by changing the nodes to SPARQL UPDATE. The choice was driven by the fact that CONSTRUCTs are easier to debug.

To control the transformation in subsequent steps, some information is stored as annotations with properties in namespace <http://test.openbudgets.eu/fdptordf#> with prefix fdprdf. These are used only internally and although some such annotations remain in the final result, the pipeline could be edited to delete them. There is probably no need to change this namespace.

The iri with base http://test.openbudgets.eu/ is used for newly created entities. Should be changed for something we agree on - these URIs should be dereferencable. However, qb:Observations are created as blank nodes (the identifiers from CSV transformation) - that can be changed if needed.
Generally, all URIs are just temporarily chosen for development purposes and can be changed as needed.

An overview of the whole pipeline with brief comments of the transformation steps is available as a draw.io diagram. (You have to click the "open" button after following the link to see the whole image.) The comment nodes in the diagram contain links back to respective parts of this detailed documentation (you have to select the comment to view the link).

Confusion

Among other things, there are at least three different concepts called attributes:

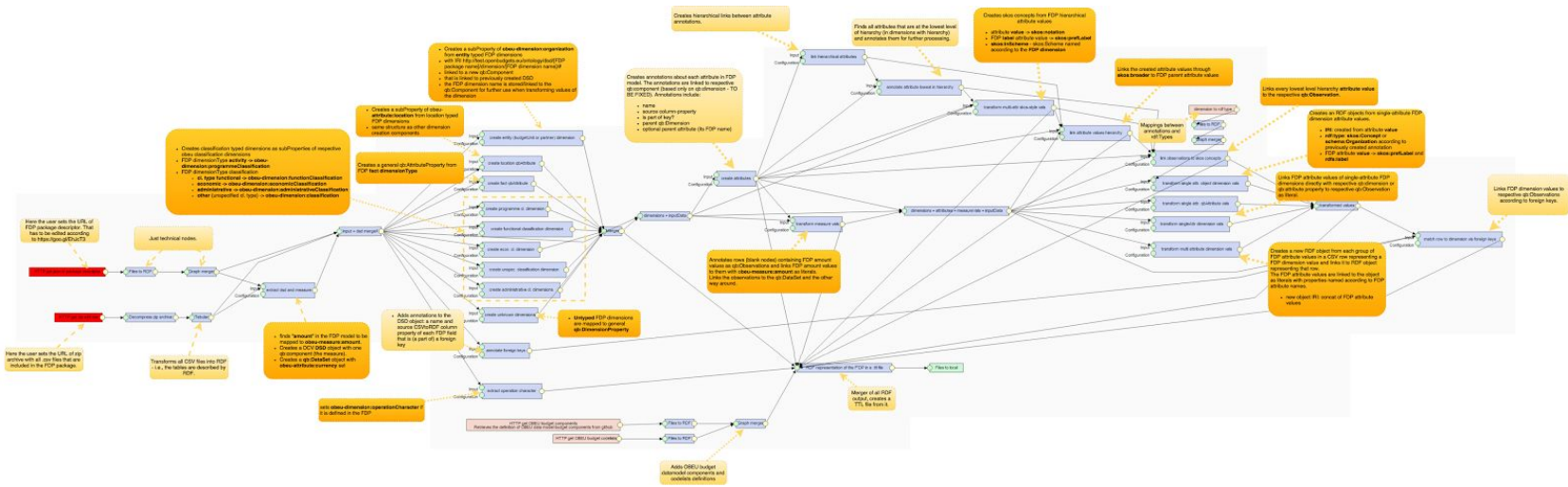
- Attributes in the FDP model (called FDP attributes here)
- qb:AttributeProperties (called qb:Attributes or similarly here)
- Blank nodes linked with the fdprdf namespace properties representing extracted information about the FDP attributes (called fdprdf attributes)

Input

The input is uploaded to the pipeline in POST request as described in separate documentation.

The prefix fdp is used for properties representing the original FDP json descriptor.
CSVs are transformed to RDF by the Tabular node in the following triple format:
[blanknode representing row] file://[CSV_filename]#[column_name_from_CSV_header] [literal value of the cell]

The following part describes all SPARQL CONSTRUCT nodes used in the pipeline in (approximate) order of their application (see the diagram mentioned above to get the whole picture about the pipeline, the diagram contains links back here to corresponding parts of the documentation).



Enlarge the diagram: https://drive.google.com/file/d/0B8L2NYplYPUwa3U3U2g3SG5YZmc/view?usp=sharing (click "open" button at the top of the page after following the link)
Or here in SVG:

Extract dsd and measure

Creates the measure component, data structure definition, dataset object and currency attribute linked to it (FDP has one currency for the whole dataset).
If there are multiple measures in the FDP model, a separate dataset and DSD is created for each one. This has however not been tested yet.

Constructs the following output:

?dsd a qb:DataStructureDefinition; *// ?dsd is constructed as http://test.openbudgets.eu/dsd/[FDP package name]-[FDP measure name]*
qb:component [qb:measure obeu-measure:amount;
fdprdf:source ?measureSource; *// the property linking to values of the measure in the CSV-transformed data*
fdprdf:factor ?measureFactor],
[qb:attribute obeu-attribute:currency;
qb:componentAttachment qb:DataSet] .

?dataset a qb:DataSet; *// ?dsd is constructed as http://test.openbudgets.eu/dataset/[FDP package name]-[FDP measure name]*
fdprdf:datasetShortName ?packageName ;
obeu-attribute:currency ?currencyUri ; *// http://dbpedia.org/resource/[currency code from FDP descriptor]*
qb:structure ?dsd .

Qb Dimensions creation

A set of parallel SPARQL CONSTRUCT nodes creates qb:DimensionProperties and qb:AttributeProperties based on the FDP model dimensions and their types. The dimension properties are created as subProperties of respective OBEU model properties. All queries have almost identical structure, only the parent property from OBEU data model and the fdprdf:valueType are different (and of course respective parts of the WHERE sections of the queries)

Example of output: (for entity type FDP dimension)

?dsd qb:component [qb:dimension ?dimensionProperty;
fdprdf:valueType fdprdf:organization]. *// used later to control values transformation*
?dimensionProperty a rdf:Property, qb:DimensionProperty;
rdfs:subPropertyOf obeu-dimension:organization;
fdprdf:name ?dimensionName .

Annotate foreign keys

Creates an annotation for each field that is a part of some foreign key.

Output:
?dsd fdprdf:foreignKey [
  fdprdf:name ?foreignKeyField; *// the name of the field in the FDP descriptor*
  fdprdf:source ?foreignKeySource *// property linking to values of the field*
 ] .

Extract operation character

Creates the triple specifying obeu-dimension:operationCharacter of the whole dataset according to the FDP direction property (if it is present).

Output:
?dsd qb:component [qb:dimension obeu-dimension:operationCharacter ;
        qb:componentAttachment qb:DataSet] .
?dataset obeu-dimension:operationCharacter ?specifiedDirection . *//obeu-operation:expenditure for FDP "direction": "expenditure", obeu-operation:revenue for other values of FDP direction (only "revenue" is possible)*

Create attributes

Creates annotations about each attribute in FDP model. The annotations are linked to respective qb:component (based only on qb:dimension - TO BE FIXED) and control further linking of attribute values to qb:dimensions.

Output:

?component
    fdprdf:attribute [fdprdf:name ?attributeName; *// the name of the attribute in the FDP model*
        fdprdf:source ?attributeSource; *// the property linking to the actual values of the attribute in the RDF representation of the input CSV*
        fdprdf:iskey ?isKeyAttribute; *// true if this attribute is a part of the FDP dimension key as specified in the FDP model*
         fdprdf:type ?fieldType; *// not used in the current version*
         fdprdf:format ?fieldFormat; *// not used in the current version*
        fdprdf:labelfor ?labelfor; *// the name of the FDP attribute that this attribute is a label for*
        fdprdf:valueProperty ?attributeValueProperty; *// the property that will be used to link FDP attribute values to qb:DimensionProperty values*
        fdprdf:parentFDPAttribute ?parentAttribute; *// the name of the parent FDP attribute (if exists)*
        fdprdf:parentDimension ?dimensionProperty; *// the qb:DimensionProperty associated with this FDP attribute*
        fdprdf:createdFromFDPAttribute ?someAttribute] . *// the object (blank node) representing the original FDP attribute in the RDF version of the FDP descriptor*
?component fdprdf:valueType ?valueType . *// if the value type wasn't created before for the dimension based on FDP value type, it may be guessed now - if there is a hierarchy between FDP attributes, the value type is set to skos*

Link hierarchical attributes

Creates hierarchical links between the fdprdf:attribute objects according to the previous annotations (which just link to literal names).

Output:
?attribute fdprdf:parentAttribute ?parentAttribute . // *(?attribute and ?parentAttribute were created in the previous step)*

Annotate attribute lowest in hierarchy

Annotates fdprdf attributes that have a parent but are not a parent of any other attribute.

Output:
?lowestAttribute fdprdf:isLowestInHierarchy true .

Transform multi-attr skos-style vals

Creates skos:Concepts from hierarchical FDP attribute values.

Output:
?attrValUri a skos:Concept;
        skos:notation ?attrVal; *// the actual value in the CSV cell*
        skos:prefLabel ?label; *// the value from a corresponding FDP attribute that is "labelFor" this one*
        skos:inScheme ?scheme; *// link to a scheme, that is so far represented just by URI constructed from the corresponding FDP dimension name: CONCAT("http://test.openbudgets.eu/resource/", ?packageName,"/codelist/", ?parentDimName,"/"))*

fdprdf:sourceRow ?row; *// the link to the blank node representing the CSV row where this value is*
fdprdf:valueOf ?attribute . *// the link to the parent fdprdf attribute whose value this is*

Link attribute values hierarchy

Creates skos:broader links between newly created skos values according to the previously created annotations between fdprdf attributes.

Output:
?attrValUri skos:broader ?parentAttrValUri .

Link observations to skos concepts

Links every lowest level hierarchy attribute value (values of attributes annotated as fdprdf:isLowestInHierarchy) to the respective qb:Observation.

Output:
?row ?dimension ?attrVal . //?row is the RDF blank node representing the CSV row representing the hierarchy (classification) value - can be either the same row containing the measure value (denormalized table) or a row in a different table (normalized tables) - that will be dealt with in later step;

Transform measure vals

Annotates rows (blank nodes) containing FDP amount values as qb:Observations and links FDP amount values to them with obeu-measure:amount as literals.
Links the observations to the qb:DataSet and the other way around.

Output:
?dataset qb:observation ?row .

?row a qb:Observation; *// ?row is the original blank node representing the CSV row with the measure value*
qb:dataSet ?dataset;
obeu-measure:amount ?finalMeasureVal. *// the literal value from the CSV multiplied by the FDP factor (if there is one)*

Transform single attr. Object dimension vals

Creates RDF objects from single-attribute FDP dimension attribute values that were previously recognized as classifications or organizations (entities)

Output:
?row ?dimensionProp ?attrValUri.

?attrValUri a ?rdfType . *// ?attrValUri is created from the attribute value (CONCAT(str(?dataset),"/",?parentDimName,"/",ENCODE_FOR_URI(str(?attrVal)), ?rdfType is set according to mapping loaded from other node (see below)*

?attrValUri skos:prefLabel ?attrVal; *// ?attrVal is the value in the CSV file*
rdfs:label ?attrVal .

The FDP to RDF type mapping loaded from "dimension to rdf type" node
So far quite simple:
fdprdf:skos fdprdf:rdfType skos:Concept . *//the fdprdf:skos is annotation created for all classification dimensions*
fdprdf:organization fdprdf:rdfType schema:Organization . *// the fdprdf:organization is used for FDP entity typed dimensions*

Transform singleAttr dimension vals

Links FDP attribute values of single-attribute FDP dimensions directly with respective qb:dimension or qb:attribute property to respective qb:Observation as literal.

Output:
?row ?dimensionProp ?attrVal. *// ?row is the blank node representing the CSV row, as this is single-attribute dimension, it is the same row containing the corresponding measure and therefore the corresponding qb:Observation annotated previously*

Transform multi attribute dimension vals

Creates a new RDF object from each group of FDP attribute values in a CSV row representing a FDP dimension value and links it to RDF object representing that row.
The FDP attribute values are linked to the object as literals with properties named according to FDP attribute names.

Output:

?row ?dimensionProp ?dimensionValUri. *// ?dimensionValUri is created from CONCAT(str(?dataset),"/",?dimensionName,"/",ENCODE_FOR_URI(str(?dimensionVal))) where ?dimensionVal is a concatenation of all key field values - this allows for linking based on foreign key later*
?dimensionValUri ?attributeValueProperty ?attrVal; *// ?attributeValueProperty was created in the Create attributes step*
fdprdf:sourceRow ?row. *// link to the CSV for further interlinking in case the values come from a separate (normalized) table*

Match row to dimension via foreign keys

Links FDP dimension values to respective qb:Observations according to foreign keys.

Output:
?measureTabRow ?dimensionProp ?dimensionValUri. *// the corresponding ?measureTabRow (the qb:Observation instance) and ?dimensionValUri (the object representing the qb dimension value) are identified based on the fact that the number of foreign key values from the "measures" table matching the key values from the "dimension" table equals the total number of key FDP attributes of the FDP dimension - there is a slight technological drawback that e.g. a person identified with "John Jack" key will be linked also to person identified as "Jack John" - the order of the values is not taken into account; I believe this is acceptable as this is quite rare situation and keys are usually single numeric ids anyway*