Guidelines for the manual enrichment of the "Market capitalization of listed companies (current US$)" data set from World Bank Linked Data

To manually enrich a QB data set, you need to use a SPARQL endpoint to redefine the data set according to QB4OLAP and discover the concepts that can be used for the enrichment. In this process, it is necessary that you keep track of:

- The **total time** used in this process.
- Redefinition phase:
  - **The time** used for this phase.
  - The **total number of queries** used for this phase.
  - The number of erroneous queries (if any).
  - The number of time-outed queries (if any).
- Enrichment phase:
  - **The time** used for this phase.
  - The **total number of queries** used for this phase.
  - The number of erroneous queries (if any).
  - The number of time-outed queries (if any).

(Time is to be measured from the moment you run the first query (of a phase) to the moment the last query (of a phase) finishes.)

At the end, please fill in the "QB2OLAP_Evaluation_name_lastname.xlsx" excel sheet with the values of these parameters (the "Manual Enrichment" column). Also, once you are done with all the experiments (including other enrichment option) fill in the "QB2OLAPem survey". Finally, once you have the results for both enrichment options, send all files by email to jvarga@essi.upc.edu. (To recapitulate, you should send 1) the excel file with the metrics you keep, 2) the file with triples produced with the tool, and 3) the filled in survey.)
Assumption: Template queries are designed assuming that all result triples are being stored in the same endpoint as a new QB graph.

Note: When you encounter variable surrounded by question marks (e.g., ?x?), this is a parameter to be replaced by a constant IRI given in the exercise. The necessary input is specified in the query parameter(s).

The target SPARQL endpoint is:

1. Schema redefinition phase

Graph creation

As we are going to store all the new triples generated in the local endpoint, we need to create a new graph for this purpose. Please define a graph name in applying the following pattern "<" + http://example.com/ + "FirstNameLastName" + ">" (for example, <http://example.com/MichaelSmith>) and run the Query 0.1 using it as the parameter:

- ?gIRI? – your graph IRI just defined (e.g., <http://example.com/MichaelSmith>)

```
# QUERY0.1 – create a new graph for the QB4OLAP triples
# INPUT: replace '?gIRI?' with the graph IRI (e.g., http://localhost:8890/NewQB4OLAPGraph )

CREATE GRAPH ?gIRI?
```

NOTE: In the next steps, please watch out **not to mix the graphs** in the parameters.

Redefining schema to QB4OLAP

First, a syntactic transformation from QB to QB4OLAP basic constructs is needed. Here, each QB dimension is transformed into a QB4OLAP level. Please run the query below (Query 1.1) to redefine the initial QB schema according to the QB4OLAP vocabulary, with following parameters:

- ?initialGraphIRI? - <http://localhost:8890/WB>
- ?dsIRI? = <http://worldbank.270a.info/dataset/CM.MKT.LCAP.CD>
- ?dsdIRI? = <http://localhost:8890/NewQB4OLAPGraph.structure>
- ?gIRI? – your graph IRI.

```
# QUERY1.1 – redefine schema according to the QB4OLAP vocabulary
# INPUT: replace ?initialGraphIRI? with the input graph IRI, replace '?dsIRI?' with the data set IRI, "?dsdIRI" with the data structure definition IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
?dsIRI?             a                   qb:DataSet .
?dsIRI?              qb:structure       ?dsdIRI? .
?dsdIRI?            a                   qb:DataStructureDefinition .
?dsdIRI?            qb:component        ?bl . ?bl qb4o:level ?d .
?dsdIRI?            qb:component        ?bm . ?bm qb:measure ?m .
?d                  a                   qb4o:LevelProperty .
?m                  a                   qb:MeasureProperty .
        }
FROM ?initialGraphIRI?
WHERE {
?dsIRI?             a                   qb:DataSet .
?dsd                a                   qb:DataStructureDefinition .
?dsIRI?             qb:structure        ?dsd .
?dsd                qb:component        ?bl . ?bl qb:dimension ?d .
?dsd                qb:component        ?bm . ?bm qb:measure ?m  .
}
```

Next, you need to associate measure with a default aggregate function (Query 1.2), using the following parameter:

- ?gIRI? – your graph IRI.
- ?dsIRI? - <http://worldbank.270a.info/dataset/CM.MKT.LCAP.CD>
- ?aggregateFunctionIRI? - <qb4o:Sum>

```
# QUERY1.2 –define aggregate function for the schema
# INPUT: replace ?gIRI? with your new graph IRI (defined above), replace '?dsIRI?' with the data set IRI (e.g., http://www.example.com/dataset ), ?aggregateFunctionIRI? with the aggregate function IRI (e.g. qb4o:Sum)

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
?blank qb4o:aggregateFunction ?aggregateFunctionIRI?
```

```
}
FROM ?gIRI?
WHERE {
?dsIRI?              a                   qb:DataSet .
?dsd                 a                   qb:DataStructureDefinition .
?dsIRI?              qb:structure        ?dsd .
?dsd                 qb:component        ?blank .
?blank qb:measure ?m  .
}
```

Next, define a QB4OLAP dimension for each initial level (Query 1.3). Thus, 3 times run the query forming the ?dimensionIRI? parameter:

- First time with: ?gIRI? + ":dimension1" (e.g., <http://example.com/MichaelSmith:dimension1>)
- Second time with: ?gIRI? + ":dimension2" (e.g., <http://example.com/MichaelSmith:dimension2>)
- Third with: ?gIRI? + ":dimension3" (e.g., <http://example.com/MichaelSmith:dimension3>)

All 3 times use the following value for the parameter:

- ?gIRI? – your graph IRI.

```
# QUERY1.3 –define dimension for the initial schema
# INPUT: ?dimensionIRI? with the dimension  IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? { ?dimensionIRI? a qb:DimensionProperty . }
```

Next, define a QB4OLAP hierarchy (i.e., an acyclic simple path) for each initial dimension (Query 1.4). Thus, 3 times run the query forming the ?hierarchyIRI? parameter:

- First time with: ?gIRI? + ":hierarchy1" (e.g., <http://example.com/MichaelSmith:hierarchy1>)
- Second time with: ?gIRI? + ":hierarchy2" (e.g., <http://example.com/MichaelSmith:hierarchy2>)
- Third with: ?gIRI? + ":hierarcy3" (e.g., <http://example.com/MichaelSmith:hierarchy3>)

All 3 times use the following values for the parameters:

- ?dimensionIRI? – previously defined dimension IRIs, pair them by order, i.e., dimension 1 with hierarchy 1, dimension 2 with hierarchy 2, and dimension 3 with hierarchy 3.
- ?levelIRI? – assign the levels you redefined (i.e.,
    - o <http://purl.org/linked-data/sdmx/2009/dimension#refArea>,
    - o <http://worldbank.270a.info/property/indicator>, and
    - o <http://purl.org/linked-data/sdmx/2009/dimension#refPeriod>

  such that one is assigned to each dimension-hierarchy pair.
- ?gIRI? – your graph IRI.

```
# QUERY1.4 –define hierarchy for the initial dimensions and their levels
# INPUT: replace ?dimensionIRI? with the dimension IRI, ?hierarchyIRI? with the hierarchy  IRI, ?levelIRI? with the level IRI, and ?gIRI? with your graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>
```

```
INSERT INTO ?gIRI?
{ ?hierarchyIRI? a qb4o:Hierarchy .
                ?dimensionIRI? qb4o:hasHierarchy ?hierarchyIRI? .
                ?hierarchyIRI? qb4o:inDimension ?dimensionIRI? .
                ?hierarchyIRI? qb4o:hasLevel ?levelIRI? .          }
```

For each level defined in Step 1.1, retrieve all level members from available QB observations (Query 1.5). Thus, 3 times run the query using the initial level IRIs, ?levelIRI?:

- <http://purl.org/linked-data/sdmx/2009/dimension#refArea>,
- <http://worldbank.270a.info/property/indicator>, and
- <http://purl.org/linked-data/sdmx/2009/dimension#refPeriod>

All 3 times use the following values for the parameters:

- ?initialGraphIRI? - <http://localhost:8890/WB>
- ?dsIRI? - <http://worldbank.270a.info/dataset/CM.MKT.LCAP.CD>
- ?gIRI? – your graph IRI.

```
# QUERY1.5 – Populate initial level memeber
# INPUT: replace ?levelIRI? with the level IRI, ?dsIRI? with the data set IRI, ?gIRI?  with your graph IRI, ?initialGraphIRI? with the initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
 ?levelMember a qb4o:LevelMember .
 ?levelMember qb4o:memberOf ?levelIRI? .
}
FROM ?initialGraphIRI?
WHERE {
        {
                SELECT DISTINCT ?levelMember WHERE {
                        ?o a qb:Observation .
                        ?o qb:dataSet ?dsIRI? .
                        ?o ?levelIRI? ?levelMember .
                }
        }
}
```

OPTIONAL: copy all observations (Query 1.6) – if you want to also copy observations in the new graph instead of using the existing one, run the query with following parameters:

Use the following values for the parameters:

- ?initialGraphIRI? - <http://localhost:8890/WB>
- ?dsIRI? - <http://worldbank.270a.info/dataset/CM.MKT.LCAP.CD>
- ?gIRI? – your graph IRI.

```
# QUERY1.6 – copy observations
# INPUT: replace ?dsIRI? with the data set IRI, ?gIRI?  with your graph IRI, ?initialGraphIRI? with the initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI?
```

```
{ ?oIRI a qb:Observation .
        ?oIRI ?prop ?val .      }
FROM ?initialGraphIRI?
WHERE {
        ?oIRI a qb:Observation .
        ?oIRI qb:dataSet ?dsIRI? .
        ?oIRI ?prop ?val .
}
```

2. Schema enrichment phase

Retrieve candidate properties (start of enrichment phase)

For each level, retrieve all the properties related to its level members.

First, retrieve all properties for level members of the referenced area level using Query 2.1 with the following parameters:

- ?levelIRI? – <http://purl.org/linked-data/sdmx/2009/dimension#refArea>
- ?initialGraphIRI? - <http://localhost:8890/WB>
- ?gIRI? – your graph IRI.

```
# QUERY2.1
# INPUT: replace ?levelIRI? with the level IRI, ?gIRI?  with your graph IRI, ?initialGraphIRI? with the initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

SELECT DISTINCT ?p

FROM ?initialGraphIRI?
WHERE {
        ?levelMember ?p ?o .
{
        SELECT DISTINCT ?levelMember
        FROM ?gIRI?
        WHERE {
                ?levelIRI? a qb4o:LevelProperty .
                ?levelMember a qb4o:LevelMember .
                ?levelMember qb4o:memberOf ?levelIRI? .
        }
}
}
```

Then, for the two following two properties referring to a region, check they satisfy the to-1 integrity constraint (i.e., the query return *true* if it satisfies, or *false* if it does not satisfy), needed for constructing a hierarchy:

- First check for the administrative region: ?propertyIRI? - <http://worldbank.270a.info/property/admin-region>
- Then, check for the region: ?propertyIRI? - <http://worldbank.270a.info/property/region>

The additional parameters for both queries are:

- ?levelIRI? - <http://purl.org/linked-data/sdmx/2009/dimension#refArea> (the starting level for which we check if we can build a hierarchy)
- ?initialGraphIRI? - <http://localhost:8890/WB>

- ?gIRI? – your graph IRI.

```
# QUERY 2.2 – CHECKING IF THE PROPERTY IS FUNCTIONAL (i.e., to-1)
# INPUT: replace ?levelIRI? with the level IRI, ?propertyIRI? with the property IRI, ?gIRI?  with your graph IRI, ?initialGraphIRI? with the initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>


ASK {
        {
                SELECT (COUNT (?levelMember) AS ?lmWithUniqueObject )
                FROM ?gIRI?
                WHERE {
                        { #get the number of unique object per level member for the input property (and level)
                                SELECT ?levelMember (COUNT (DISTINCT ?obj) AS ?uniqueObjNum)
                                FROM ?initialGraphIRI?
                                WHERE {
                                        ?levelMember ?propertyIRI? ?obj .
                                        {SELECT ?levelMember
                                        FROM ?gIRI?
                                        WHERE {
                                                ?levelIRI? a qb4o:LevelProperty .
                                                ?levelMember a qb4o:LevelMember .
                                                ?levelMember qb4o:memberOf ?levelIRI? .
                                        }}
                                } GROUP BY ?levelMember
                        }
                FILTER ( ?uniqueObjNum = 1)
                }
        }
        { #get the total number of level members for a level
                SELECT (COUNT (DISTINCT ?lm) AS ?totalLevelMemberNumber)
                FROM ?gIRI?
                WHERE {
                        ?levelIRI? a qb4o:LevelProperty .
                        ?lm a qb4o:LevelMember .
                        ?lm qb4o:memberOf ?levelIRI? .
                }
        }
        FILTER (?lmWithUniqueObject = ?totalLevelMemberNumber)
}
```

For the *region* property for which the query returns *true* for the to-1 integrity constraint (i.e., can be new level or attribute), check if it can be used as level attribute 1:1 (Query 2.3) or level M:1 (Query 2.4).

Parameters for Query 2.3 and Query 2.4 are:

- ?levelIRI? – <http://purl.org/linked-data/sdmx/2009/dimension#refArea>
- ?propertyIRI? – <http://worldbank.270a.info/property/region>
- ?initialGraphIRI? - <http://localhost:8890/WB>
- ?gIRI? – your graph IRI.

```
# QUERY 2.3 – CHECKING IF THE PROPERTY IS CANDIDATE FOR ATTRIBUTES (i.e., 1:1 cardinality)
# INPUT: replace ?levelIRI? with the level IRI, ?propertyIRI? with the property IRI, ?gIRI?  with your graph IRI, ?initialGraphIRI? with the initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

ASK {
```

```
	{ #get the number of object per level member for the input property (and level)
		SELECT (COUNT (DISTINCT ?levelMember) AS ?totalLevelMemberNumber) (COUNT (DISTINCT ?obj) AS
?uniqueObjNum)
		FROM ?initialGraphIRI?
		WHERE {
			{SELECT ?levelMember
			FROM ?gIRI?
			WHERE {
				?levelIRI? a qb4o:LevelProperty .
				?levelMember a qb4o:LevelMember .
				?levelMember qb4o:memberOf ?levelIRI? .
			}}
			?levelMember ?propertyIRI? ?obj .
		}
	}
	FILTER (?uniqueObjNum = ?totalLevelMemberNumber)
}


# QUERY 2.4 – CHECKING IF THE PROPERTY IS CANDIDATE FOR NEW LEVEL (i.e., M:1 cardinality)
# INPUT: replace ?levelIRI? with the level IRI, ?propertyIRI? with the property IRI, ?gIRI?  with your graph IRI, ?initialGraphIRI? with the
initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

ASK {
	{ #get the number of object per level member for the input property (and level)
		SELECT (COUNT (DISTINCT ?levelMember) AS ?totalLevelMemberNumber) (COUNT (DISTINCT ?obj) AS
?uniqueObjNum)
		FROM ?initialGraphIRI?
		WHERE {
			{SELECT ?levelMember
			FROM ?gIRI?
			WHERE {
				?levelIRI? a qb4o:LevelProperty .
				?levelMember a qb4o:LevelMember .
				?levelMember qb4o:memberOf ?levelIRI? .
			}}
			?levelMember ?propertyIRI? ?obj .
		}
	}
	FILTER (?uniqueObjNum < ?totalLevelMemberNumber/2)

	{ #check that objects are not literals
		SELECT (COUNT (DISTINCT ?obj2) AS ?notLiteralObj)
		FROM ?initialGraphIRI?
		WHERE {
			{SELECT ?lm
			FROM ?gIRI?
			WHERE {
				?levelIRI? a qb4o:LevelProperty .
				?lm a qb4o:LevelMember .
				?lm qb4o:memberOf ?levelIRI? .
			}}
			?lm ?propertyIRI? ?obj2 .
			FILTER isIRI(?obj2)
		}
	}
	FILTER (?uniqueObjNum = ?notLiteralObj)
}
```

Now we proceed with forming a new level or level attributes depending on the outcome of previous queries.

Add new level

For the sake of simplicity, we are identifying three leveled dimensions, i.e., hierarchies consisting of the existing QB level, one new level on top of the existing one, plus the all level. Moreover, we are going to create two such hierarchies in the same dimension.
Each property chosen among the ones detected as candidate rollup property (i.e., the ones satisfying the M:1 integrity constraint), we must define as level with its level members and add it to the schema. This can be achieved with Queries 3.1 – 3.3.

First, you should define a new level for region, by using Query 3.1 with following parameters:

- ?levelIRI? - <http://purl.org/linked-data/sdmx/2009/dimension#refArea>
- ?propertyIRI? – <http://worldbank.270a.info/property/region>
- ?initialGraphIRI? - <http://localhost:8890/WB>
- ?gIRI? – your graph IRI.

Then, you need to create an initial hierarchy step from area to the region by using Query 3.2 with following parameters:

- ?hsIRI? - ?gIRI? + ":newHS1" (e.g., "<http://localhost:8890/MichaelSmith:newHS1>")
- ?propertyIRI? - <http://worldbank.270a.info/property/region>
- ?levelIRI? - <http://purl.org/linked-data/sdmx/2009/dimension#refArea>
- ?gIRI? – your graph IRI.

To illustrate the case when addition of new level requires building of new hierarchy, you should add the income-level as higher level to referenced area (optionally, you may use queries from Step 2 to check if this property satisfies the M:1 integrity constraint for referenced area level). Thus, we first define income-level as new level (with Query 3.1) and then create new hierarchy and hierarchy step (Query 3.3) using the following parameters:

- Query 3.1 parameters:
  - ?levelIRI? - <http://purl.org/linked-data/sdmx/2009/dimension#refArea>
  - ?propertyIRI? – <http://worldbank.270a.info/property/income-level>
  - ?initialGraphIRI? - <http://localhost:8890/WB>
  - ?gIRI? – your graph IRI.
- Query 3.3 parameters:
  - ?hIRI? - ?gIRI? + ":hierarchy4" (e.g., <http://example.com/MichaelSmith:hierarchy4>)
  - ?levelIRI? - <http://purl.org/linked-data/sdmx/2009/dimension#refArea>
  - ?propertyIRI? – <http://worldbank.270a.info/property/income-level>
  - hsIRI? - ?gIRI? + ":newHS2" (e.g., "<http://example.com/MichaelSmith:newHS2>")
  - ?gIRI? – your graph IRI.

Finally, we need to add the *all* level on top of both new levels by running the following queries:

- Query 3.4 to define the all level and all level member. Parameters to run it are:
  - ?gIRI? – your graph IRI
  - ?allLevelIRI? - ?gIRI? + ":dimension1AllLevel" (e.g., "<http://example.com/MichaelSmith:dimension1AllLevel>")

    - ?allLevelMemberIRI? - ?gIRI? + ":dimension1AllLevelMember" (e.g., "<http://example.com/MichaelSmith:dimension1AllLevelMember>")
- Query 3.5 to link lower level members with the all level member. It needs to be run twice, one for each new level:
    - First run parameters:
        - ?gIRI? – your graph IRI
        - ?levelIRI? - <http://worldbank.270a.info/property/region>
        - ?allLevelMemberIRI? – previously defined IRI
    - Second run parameters:
        - ?gIRI? – your graph IRI
        - ?levelIRI? - <http://worldbank.270a.info/property/income-level>
        - ?allLevelMemberIRI? – previously defined IRI
- Query 3.2 to create hierarchy steps linking the new level with the all level. It needs to be run twice, one for each new level:
    - First run parameters:
        - ?hsIRI? - ?gIRI? + ":newHS3" (e.g., "<http://localhost:8890/MichaelSmith:newHS3>")
        - ?propertyIRI? – the ?allLevelIRI? previously defined
        - ?levelIRI? - <http://worldbank.270a.info/property/region>
        - ?gIRI? – your graph IRI.
    - Second run parameters:
        - ?hsIRI? - ?gIRI? + ":newHS4" (e.g., "<http://localhost:8890/MichaelSmith:newHS4>")
        - ?propertyIRI? - the ?allLevelIRI? previously defined
        - ?levelIRI? - <http://worldbank.270a.info/property/income-level>
        - ?gIRI? – your graph IRI.

For the sake of simplicity we omit adding of all levels for remaining two dimensions as that can be performed with existing queries.

```
# QUERY 3.1 – ADDING A NEW LEVEL TO SCHEMA
# INPUT: replace ?levelIRI? with the level IRI (i.e., child level), ?propertyIRI? with the property IRI (i.e., new level), ?gIRI? with your graph IRI, ?initialGraphIRI? with the initial graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>
prefix skos: <http://www.w3.org/2004/02/skos/core#>

INSERT INTO ?gIRI? {
        ?propertyIRI? a qb4o:LevelProperty .
        ?obj a qb4o:LevelMember .
        ?obj qb4o:memberOf ?propertyIRI? .
        ?levelMember2 skos:broader ?obj2 .
                }
WHERE {
        {
        SELECT DISTINCT ?obj
        FROM ?initialGraphIRI?
        WHERE {
                {SELECT ?levelMember
                FROM ?gIRI?
                WHERE {
```

```
                    ?levelIRI? a qb4o:LevelProperty .
                    ?levelMember a qb4o:LevelMember .
                    ?levelMember qb4o:memberOf ?levelIRI? .
            }}
            ?levelMember ?propertyIRI? ?obj .
    }}
    {
    SELECT ?levelMember2 ?obj2
    FROM ?initialGraphIRI?
    WHERE {
            {SELECT ?levelMember2
            FROM ?gIRI?
            WHERE {
                    ?levelIRI? a qb4o:LevelProperty .
                    ?levelMember2 a qb4o:LevelMember .
                    ?levelMember2 qb4o:memberOf ?levelIRI? .
            }}
            ?levelMember2 ?propertyIRI? ?obj2 .
    } GROUP BY ?levelMember2 ?obj2
    }
}
```

Create the initial hierarchy step in the hierarchy (if there is no existing hierarchy steps) or the top hierarchy step (e.g., adding all level at the end) (Query 3.2).

```
# QUERY 3.2 – CREATING INITIAL HIERARCHY STEP / CREATING TOP HIERARCHY STEP
# INPUT: replace ?levelIRI? with the level IRI, ?propertyIRI? with the property IRI, ?hsIRI? with the hierarchy step IRI (typically blank node, .e.,g "_:b1"), ?gIRI?  with your graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
    ?hsIRI? a qb4o:HierarchyStep .
    ?hsIRI? qb4o:inHierarchy ?h .
    ?hsIRI? qb4o:parentLevel ?propertyIRI? .
    ?hsIRI? qb4o:childLevel ?levelIRI? .
    ?hsIRI? qb4o:pcCardinality qb4o:ManyToOne .
    ?h qb4o:hasLevel ?propertyIRI? .
}
FROM ?gIRI?
WHERE {
    ?h a qb4o:Hierarchy .
    ?h qb4o:hasLevel ?levelIRI? .
}
```

If the (existing) level already exists as child level in the initial hierarchy step, you need to create new hierarchy and new starting hierarchy step (Query 3.3).

```
# QUERY 3.3 – CREATING NEW HIERARCHY WITH NEW INITIAL HIERARCHY STEP
# INPUT: replace ?levelIRI? with the level IRI, ?propertyIRI? with the property IRI, ?hsIRI? with the hierarchy step IRI (typically blank node, .e.,g "_:b1"), ?hIRI? with the hierarchy IRI, ?gIRI?  with your graph IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
    ?hIRI? a qb4o:Hierarchy .
    ?hIRI? qb4o:inDimension ?d .
    ?d qb4o:hasHierarchy ?hIRI? .
    ?hsIRI? a qb4o:HierarchyStep .
    ?hsIRI? qb4o:inHierarchy ?hIRI? .
```

```
        ?hsIRI? qb4o:parentLevel ?propertyIRI? .
        ?hsIRI? qb4o:childLevel ?levelIRI? .
        ?hsIRI? qb4o:pcCardinality qb4o:ManyToOne .
        ?hIRI? qb4o:hasLevel ?propertyIRI? .
        ?hIRI? qb4o:hasLevel ?levelIRI? .
        }
FROM ?gIRI?
WHERE {
        ?h a qb4o:Hierarchy .
        ?h qb4o:hasLevel ?levelIRI? .
        ?h qb4o:inDimension ?d .
        ?d a qb:DimensionProperty .
}
```

Finally, to create the *all* level for a dimension we first create all level and its level member (Query 3.4) and link lower level members with all level member (Query 3.5).

```
# QUERY 3.4 – ADDING ALL LEVEL TO A DIMENSION
# INPUT: replace ?gIRI?  with your graph IRI, ?allLevelIRI? with the all level IRI, ?allLevelMemberIRI? with the all level member IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
        ?allLevelIRI? a qb4o:LevelProperty .
        ?allLevelMemberIRI? a qb4o:LevelMember .
        ?allLevelMemberIRI? qb4o:memberOf  ?allLevelIRI? .
}

# QUERY 3.5 – LINK LOWER LEVEL MEMBERS WITH ALL LEVEL MEMBER
# INPUT: replace ?gIRI?  with your graph IRI, ?levelIRI? with the lower level IRI

prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>
prefix skos: <http://www.w3.org/2004/02/skos/core#>

INSERT INTO ?gIRI? {
        ?levelMember skos:broader ?allLevelMemberIRI? .
}
FROM ?gIRI?
WHERE {
        ?levelIRI? a qb4o:LevelProperty .
        ?levelMember a qb4o:LevelMember .
        ?levelMember qb4o:memberOf ?levelIRI? .
}
```

Note that adding a level at arbitrary depth (related to Query 3.3) gets more complex (where we need to copy all previous hierarchy steps before branching). Also, each hierarchy must be acyclic.

OPTIONALLY: Add new attribute

For each property chosen among the ones detected as candidate attributes (satisfying a 1:1 integrity constraint, e.g., preferred label for referenced area), you can define it as level attribute and add it to the level using the following query that adds it to the schema. (Hint: you may check the http://www.w3.org/2004/02/skos/core#prefLabel of the http://purl.org/linked-data/sdmx/2009/dimension#refArea)

```
 # QUERY 4.1 – ADDING A NEW ATTRIBUTE TO THE SCHEMA
# INPUT: replace ?levelIRI? with the level IRI, ?propertyIRI? with the property IRI (i.e., the attribute), ?gIRI?  with your graph IRI,
?initialGraphIRI? with the initial graph IRI
```

```
prefix qb: <http://purl.org/linked-data/cube#>
prefix qb4o: <http://purl.org/qb4olap/cubes#>

INSERT INTO ?gIRI? {
        ?propertyIRI? a qb4o:LevelAttribute .
        ?propertyIRI? qb4o:inLevel ?levelIRI? .
        ?levelIRI? qb4o:hasAttribute ?propertyIRI? .
        ?levelMember ?propertyIRI? ?obj .       }
FROM ?initialGraphIRI?
WHERE {
        {SELECT ?levelMember
        FROM ?gIRI?
        WHERE{
                ?levelIRI? a qb4o:LevelProperty .
                ?levelMember a qb4o:LevelMember .
                ?levelMember qb4o:memberOf ?levelIRI? .
        }}
        ?levelMember ?propertyIRI? ?obj .
}
```