SSRS with TMVGate as a Data Source

Technical Note: TA2019009

Contents

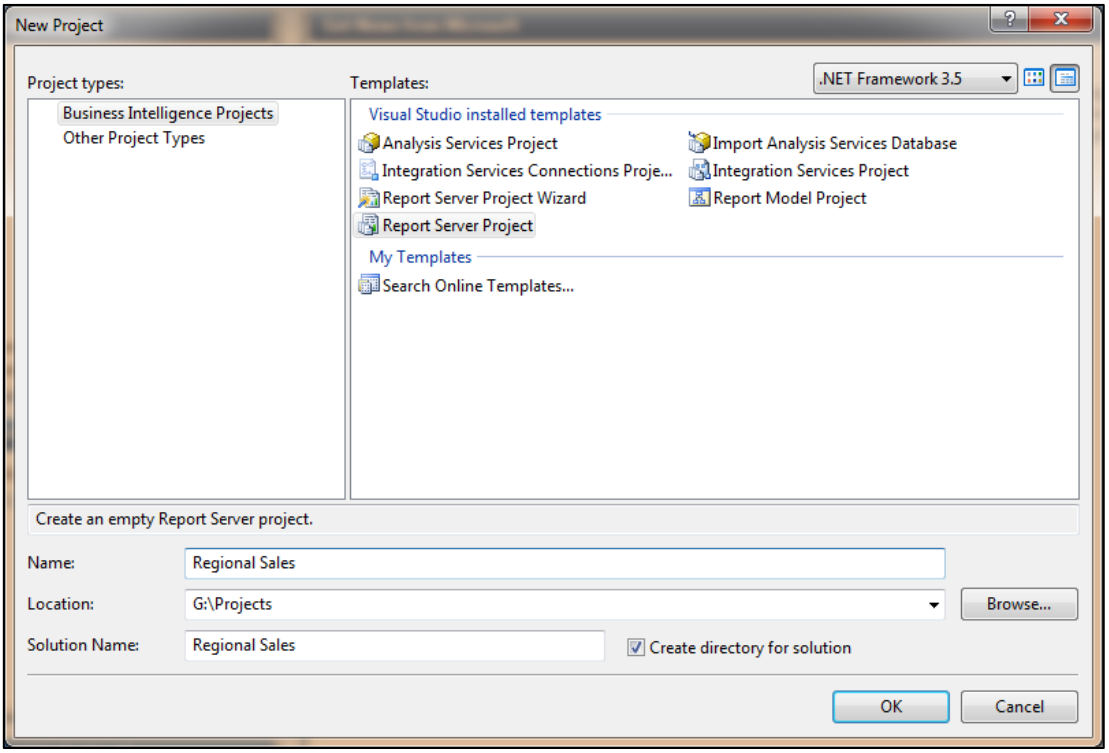
1.0 Overview

This technical document aims to provide a guide on how to connect to TMVGate as a Data Source for Microsoft SQL Reporting Services.

2.0 Creating a Regional Sales Report

Using TMVGate, cube data can be extracted based on a specified cube view, which in turn, can be either private or public. In general, most of the dimension elements specified in the cube view will be at the same level in order for it to make sense in the BI reports. The elements will either be defined in a Subset or explicitly selected in the cube view definition.

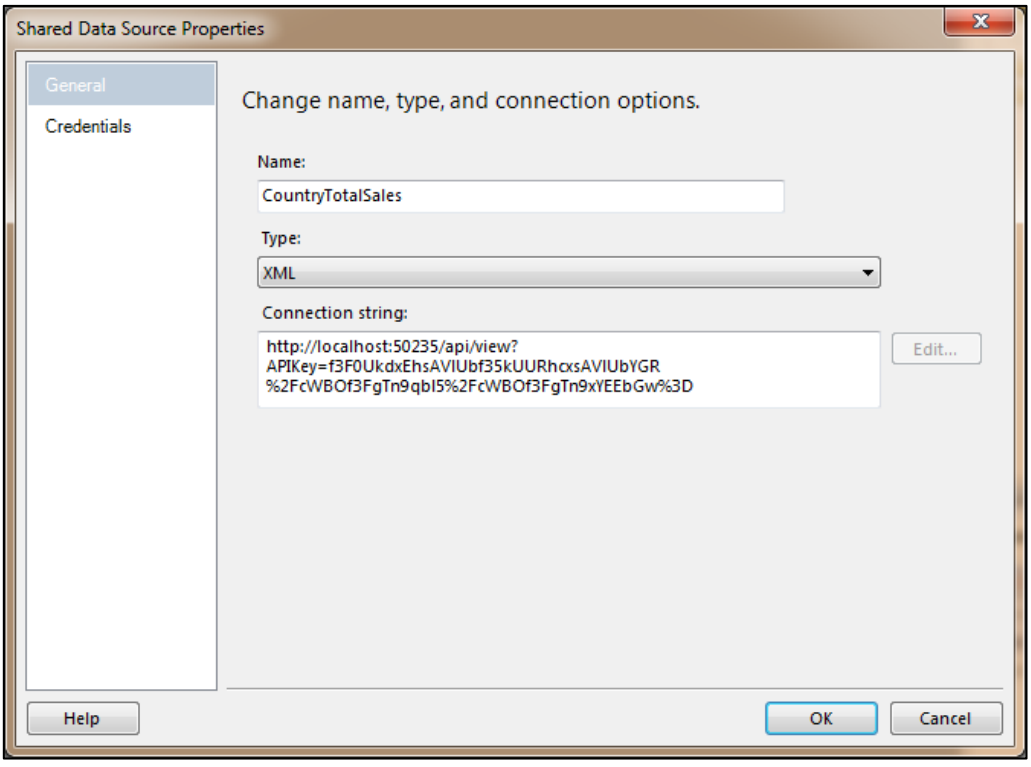
Take for example the TM1 install default SData server, specifically the cube “SalesCube”. The following screen shows a view with the “Region” dimension at the row dimension with N level elements (ie Level 0). Likewise, the “Month” dimension at the column dimension with N level elements is shown.



Actual | Total | Sales

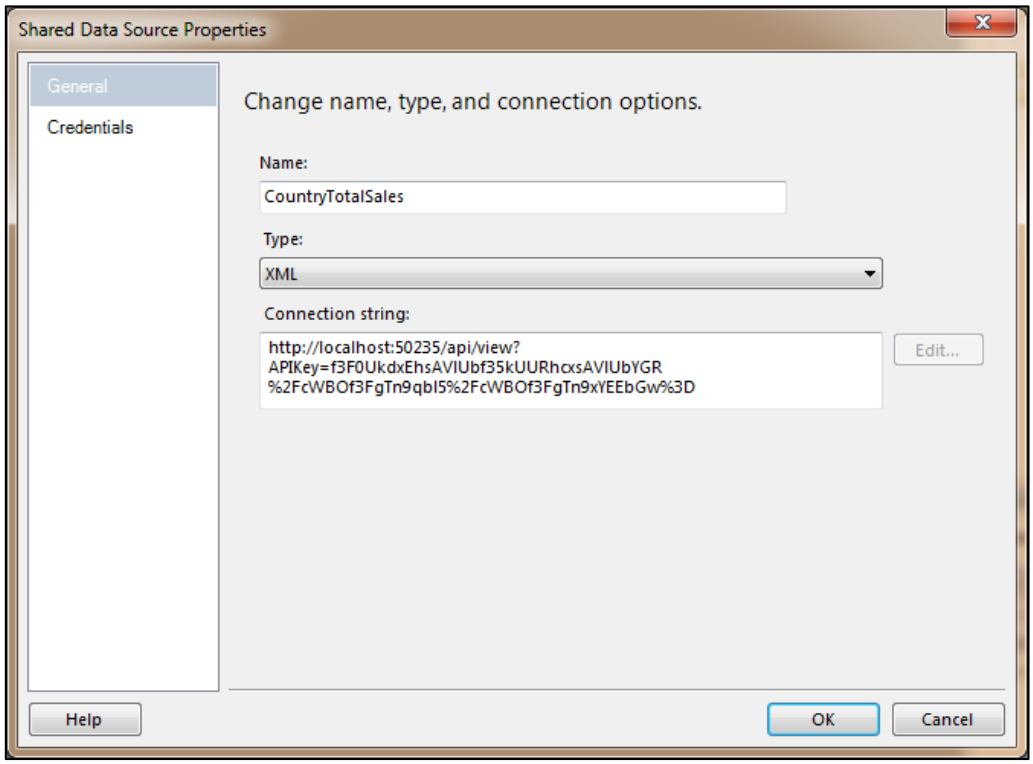
month

region	Jan	Feb	Mar	Apr
Argentina	45618.40033	51067.16187	55668.839	44688.11455
Belgium	53901.30137	63925.65462	63232.54484	52125.11368
Brazil	43804.11399	50807.2124	53491.35628	42946.3177
Canada	65347.51731	72684.8245	78007.84827	68033.93685
Chile	10661.3982	13186.14485	12877.03759	11076.35759
Denmark	16936.51489	19758.32374	20239.1191	17489.06663
France	617796.14493	669530.98355	734462.84256	584892.21415
Germany	679222.88862	719990.7194	821404.85768	665275.37598
Great Britain	242709.56521	259221.66873	269823.47785	237509.72094
Greece	40268.60683	46554.31771	51720.17607	41109.8074
Ireland	23539.08348	25485.42446	27380.86354	24271.45699
Italy	218401.86708	246980.56575	246001.42405	210372.90653
Luxemburg	5657.72544	6202.12676	6796.32624	5231.19613
Mexico	95997.47911	110975.12906	117120.80508	101984.93575
Netherlands	61784.97831	68735.38421	74404.55248	58082.77158
Norway	19490.01612	23803.91823	24200.46925	20748.3992
Portugal	17514.98992	19350.19646	20450.74319	15715.51572



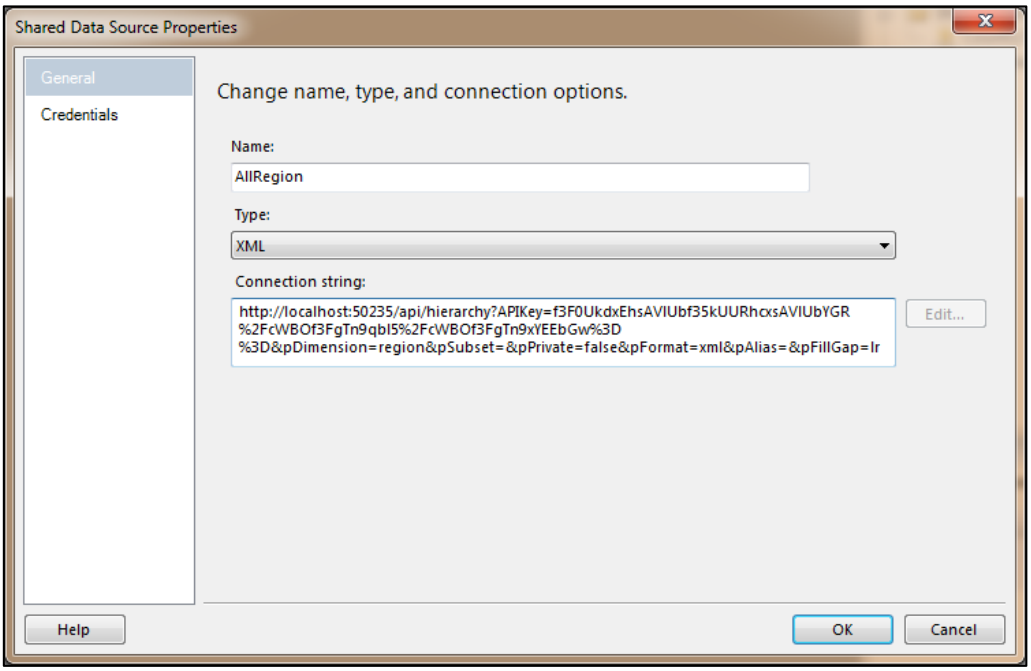
In Power BI, this cube data is translated into simple bar charts as shown below:

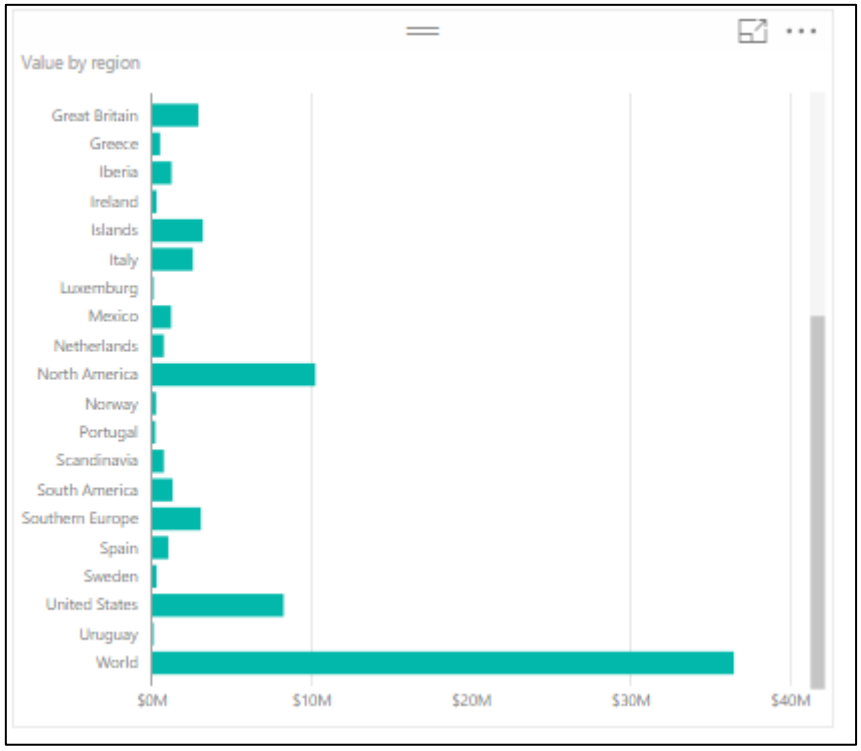
By Country and By Month



Power BI does not recognize the concepts of levels. If you start pulling different level of elements into Power BI, the visualizations can turn into something that can be misleading as shown in the example below:

In this example, we are pulling all the sub-regions, regions, world into Power BI.


Shared Data Source Properties
General
Credentials
Change name, type, and connection options.
Name:
AllRegion
Type:
XML
Connection string:
http://localhost:50235/api/hierarchy?APIKey=f3F0UkdxEhsAVlUbf35kUURhcxsAVlUbYGR
%2FcWBOf3FgTn9qbI5%2FcWBOf3FgTn9xYEEbGw%3D
%3D&pDimension=region&pSubset=&pPrivate=false&pFormat=xml&pAlias=&pFillGap=lr
Edit...
Help
OK
Cancel


Value by region
Great Britain
Greece
Iberia
Ireland
Islands
Italy
Luxemburg
Mexico
Netherlands
North America
Norway
Portugal
Scandinavia
South America
Southern Europe
Spain
Sweden
United States
Uruguay
World
$0M
$10M
$20M
$30M
$40M

Of course, there will be cases in which users would like to pull different level elements into Power BI based on their requirements. There is nothing wrong in doing so except that the user must understand the implication of mixing different levels of elements in their report.

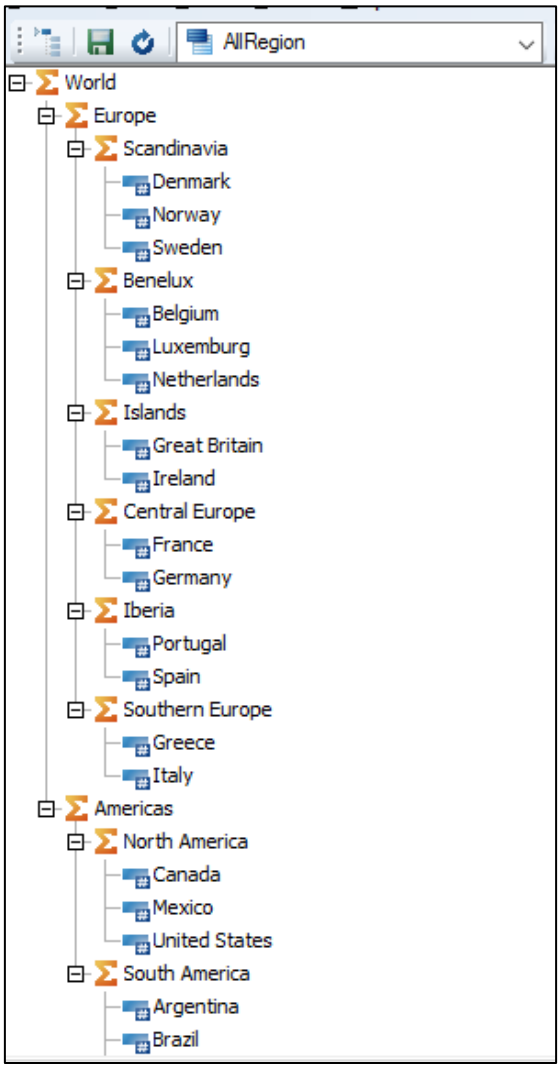
The next topic we will discuss about enriching the data with hierarchy information.

3.0 Enriching the Cube data with hierarchy structure

Using the same cube view data, which has the countries (N level) from the "Region" dimension, we will enrich the Power BI reports and have them show the sub-regions and region hierarchy.

The first thing to do is to create a subset in the "Region" dimension. This will contain the elements in the cube view, and the required parent elements that you would like to be made available in Power BI.

In this example, we have created a subset "AllRegion" that will contain all the N level elements, the respective parents specified in the cube view, and these eventually roll up to the "World". This is important as the subset **MUST** contain all the required elements and parents in order for TMVGate to construct the tabular table for Power BI to consume the data.



The TMVGate URL statement will look something like this:

```
http://.....&pDimension=Region&pSubset=AllRegion&pPrivate=false&pFormat=JSON&pAlias&pFillGap=LR
```

Notice that we are using a fill gap option of LR (Left to right). Please review the user guide on the various fill options.

When we extract this hierarchy information into Power BI, the data table will look something like this:

Region_L3	Region_L2	Region_L1	Region_L0	Region
World	Europe	Scandinavia	Denmark	Denmark
World	Europe	Scandinavia	Norway	Norway
World	Europe	Scandinavia	Sweden	Sweden
World	Europe	Benelux	Belgium	Belgium
World	Europe	Benelux	Luxemburg	Luxemburg
World	Europe	Benelux	Netherlands	Netherlands
World	Europe	Islands	Great Britain	Great Britain
World	Europe	Islands	Ireland	Ireland
World	Europe	Central Europe	France	France
World	Europe	Central Europe	Germany	Germany
World	Europe	Iberia	Portugal	Portugal
World	Europe	Iberia	Spain	Spain
World	Europe	Southern Europe	Greece	Greece
World	Europe	Southern Europe	Italy	Italy
World	Americas	North America	Canada	Canada
World	Americas	North America	Mexico	Mexico
World	Americas	North America	United States	United States
World	Americas	South America	Argentina	Argentina
World	Americas	South America	Brazil	Brazil
World	Americas	South America	Chile	Chile
World	Americas	South America	Uruguay	Uruguay

Using TMVGate, we have flattened the TM1 hierarchy structure into a tabular format which can be consumed by Power BI (or other tools).

With the above example, “Region” column is always the element principal name. “Region_L0” will be the matching lowest level element specified in the subset in the corresponding Alias, if the alias option is specified in the TMVGate extract URL.

The following example illustrates a hierarchy extract with an alias specified. Notice now “Region_L0” and “Region” is showing different values.

Region_L3	Region_L2	Region_L1	Region_L0	Region
全球	欧洲	斯堪的纳维亚半岛	丹麦	Denmark
全球	欧洲	斯堪的纳维亚半岛	挪威	Norway
全球	欧洲	斯堪的纳维亚半岛	瑞典	Sweden
全球	欧洲	（比利时、荷兰、卢森堡三国经济联盟	比利时	Belgium
全球	欧洲	（比利时、荷兰、卢森堡三国经济联盟	卢森堡	Luxemburg
全球	欧洲	（比利时、荷兰、卢森堡三国经济联盟	荷兰	Netherlands
全球	欧洲	群岛	英国	Great Britain
全球	欧洲	群岛	爱尔兰	Ireland
全球	欧洲	中欧	法国	France
全球	欧洲	中欧	德国	Germany
全球	欧洲	伊比利亚半岛	葡萄牙	Portugal
全球	欧洲	伊比利亚半岛	西班牙	Spain
全球	欧洲	南欧	希腊	Greece
全球	欧洲	南欧	意大利	Italy
全球	美洲	北美	加拿大	Canada
全球	美洲	北美	墨西哥	Mexico
全球	美洲	北美	美国	United States
全球	美洲	南美	阿根廷	Argentina
全球	美洲	南美	巴西	Brazil
全球	美洲	南美	智利	Chile
全球	美洲	南美	乌拉圭	Uruguay

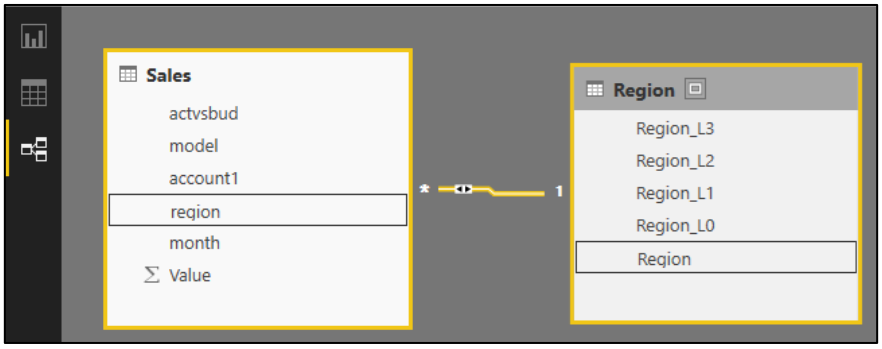
We are now ready to link up the hierarchy with the cube view data in Power BI.

4.0 Linking up the Cube data with hierarchy structure

You can create a link between the cube data table with the hierarchy table to link up the cube data with the hierarchy information.

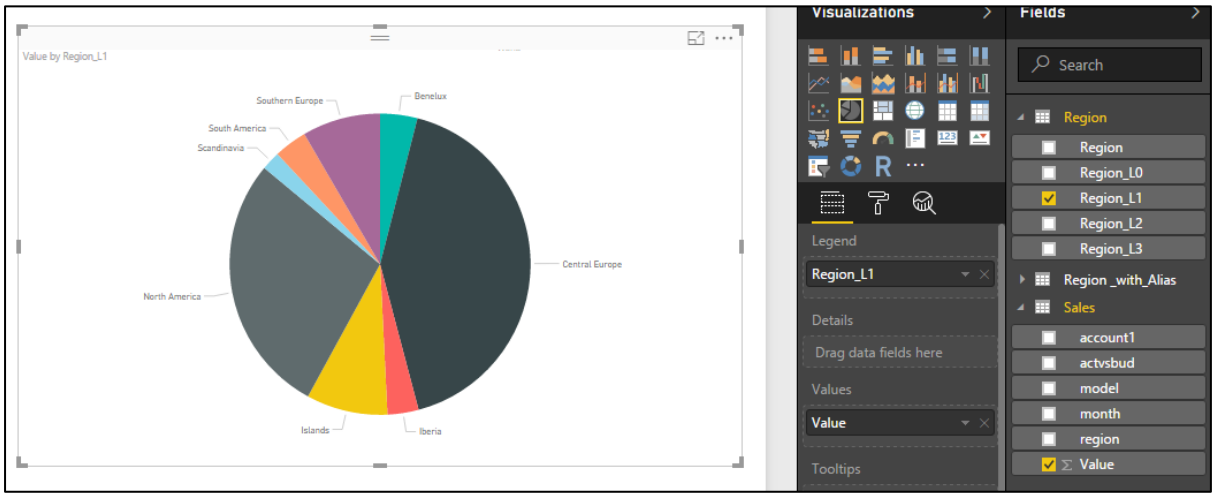
Assume that cube data is extracted without the respective aliases for the “Region” dimension.

The link can be easily created in Power BI by using the Managing Relationships function. Simply link up the “region” field in the cube data table as well as the hierarchy data table.



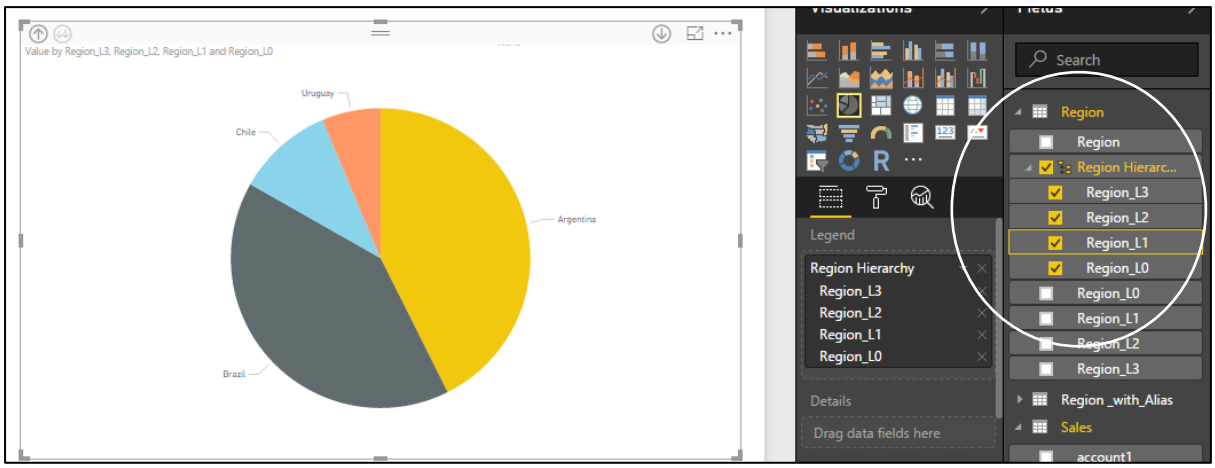
Once the relationship is established, you can start exposing the various levels of the “Region” dimension in Power BI report.

In this example, we are exposing the “Region_L1” data in the report. With all the required levels visible for selection, you can start selecting the hierarchy in Power BI.

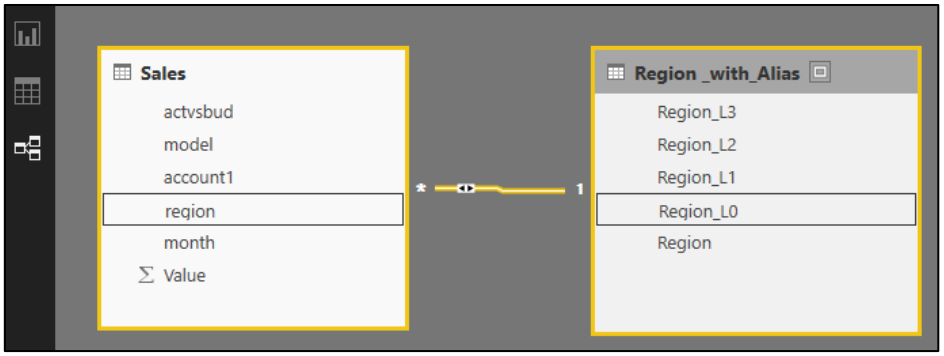


To further enhance the report, you can choose to create a hierarchy structure in Power BI similar to TM1. Please review Power BI guides on how to establish a hierarchy.

The following image shows an example of the hierarchy defined in Power BI. Once a hierarchy has been defined, you can start using the Drill function in Power BI to navigate through the data.



For cube data extracted with alias, you must have the equivalent alias extracted in the hierarchy through TMVGate. Instead of linking the element principal name, which is in the "Region" column, you will now choose the "Region_L0" column to establish the relationship.



By providing this flexibility in TMVGate, users can choose to extract cube data using the element principal name, and enrich it with the corresponding alias using the hierarchy extract.