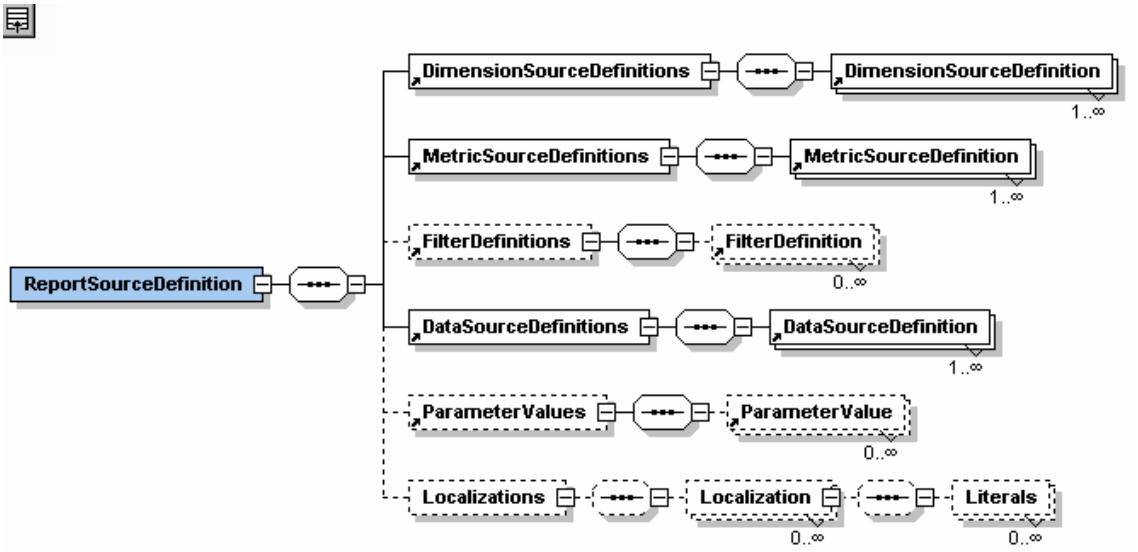
REPORTSOURCEDEFINITION.XML
Describes a data source



ReportSourceDefinition describes generic properties of the data source

Attributes:

Id: Unique identifier. Is used to reference a RSD from the RD. Example: `RSD_DISTRIBUTION_SALARY_CUBE`.

Cached: Boolean. If true the data obtained from data sources will be temporarily saved to disk. For example, if I have a report with a SQL data source, I can use de real query once and save that result. Using that cached result avoids overcharging the data base server. I may want to update the saved data, for example every 10 days, that's the pourpose of the next property.

Expiration: if the cached option is true, the expiration is the amount of **hours** that the saved data is going to be valid. After that period, the saved data won't be usable, and the data sources will be used again, making a new saving. Example: 240 (ten days).

MultiLanguage: Boolean. (It should be used with localizations). If true, it will try to translate every description String in the XML to a value in Localizations, searching the codes that may appear in descriptions. For example, I can use as a dimension description the value "01" (a code), and add 2 localizations to translate the value to ZONE or ZONA depending on the Locale configured.

MaxRowsDimensionReference: Es el nombre de la dimensión que se usará para obtener los primeros n valores, utilizando un filtro de tipo TOP. Ejemplo: CLIENTE

MaxRowCount: indicates how many values are going to use. Used for Top filter. Over that value, data will be ignored.

DimensionSourceDefinition: describes properties for the dimensions of the report.

Attributes:

Name: name of the dimension. It must be unique. Example: CLIENT.

Description: description of the dimension. This value will be the title of the dimension on the report, unless is overrided in the Report Definition. Example: Client.

DataType: type of data that will have the dimension. Possible values are: STRING, DATE, DATETIME, FLOAT, INTEGER, BOOLEAN. While data is obtained from data source, it will be converted to its specific type.

Calculated: dimensions values can be calculated from the values of other dimensions. This Boolean attribute indicates whether the dimension is calculated or not. In case is true, there is no data source for this dimension, because it's value is a calculation made from a previous dimension.

Expression: only used when calculated is true. Is the expression to get the value of the dimension from other one previously obtained. Example MONTH(DATE_DIMENSION). See appendix: date functions

ExternalData: defines the external name for a dimension in the data source. For example, in a SQL query "select client from… " the ExternalData should be client, no matter what name you use for a dimension, because is the name of the column in the result set. In this way, when you need a value for a dimension, you search the row for the column that has the same name as ExternalData

MetricSourceDefinition: defines properties for the metrics.

Attributes:

Name: name of the metric. Example: PRICE

Description: Name of the metric, visible for the user. Example: Price

AggregateType: Operation used to calculate the value of a metric. Example: SUM. See appendix, AggregateType and GroupFooterType.

GroupFooterType: Operation used to calculate the value of the total of a metric. Example: SUM. See appendix, AggregateType and GroupFooterType.

Calculated: boolean to indicate whether the values of the metric are calculated from other metrics or not. Simple math operations are allowed. Example: false (the metric is not calculated)

AggregateFunction: expression used to calculate the value of a calculated metric. Only used in case that calculated is true. See appendix AggregateFunction and GroupFooterFunction

GroupFooterFunction: expression used to calculate the totals of a calculated metric. Only used in case that calculated is true. See appendix AggregateFunction and GroupFooterFunction

ExternalData: Idem DimensionSourceDefinition

FilterDefinition: defines filters that will be used for the report.

NOTE: the default values for filters of DATE type must be always yyyyMMdd, and then they will be formatted depending on the configured Locale to be shown.
<FilterDefinitionName="DATE_FILTER" FilterParameter="FROM" DefaultValue="19970101"/>

Attributes:

Name: Name of the filter. Example: RANGE_CLIENT

DimensionName: Name of the dimension to filter. Example: CLIENT

FilterType: Type of filter. Possible values are RANGE, GREATERTHAN, LESSTHAN, EQUALTO, BEGINWITH, ENDWITH, INCLUDES, RANKING, IN or EXCLUDEGROUP.

RANGE: filters a range of values, taking an initial and final parameters, excluding values that are outside that range

GREATERTHAN: takes values that are above certain parameter. Excludes values below that parameter.

LESSTHAN: takes values that are below certain parameter. Excludes values above that parameter.

EQUALTO: takes values equal to certain parameter. Exclued values distinct from that parameter.

BEGINWITH: only for dimensions of String type. Takes values that begin with the value of the parameter.

ENDWITH: only for dimensions of String type. Takes values that end with the value of the parameter.

INCLUDES: only for dimensions of String type. Takes values that contain the value of the parameter.

RANKING: makes an order of the dimension, taking 2 optional ways: if the dimension has a RankMetricName (see RD) defined, it sorts the dimension by the values corresponding to that metric y only takes the “n” firsts (n given as parameter). For example if I want to filter the first 5

clients by Amount, in the Client dimension I should define RankMetricName="Amount", and set the parameter to 5. If the dimension doesn't has a metric, the standard order is used (example: alphabetic), and the firsts n are filtered with that order.

IN: in this case, the parameter defines a set of comma separated values (if a value has a comma, it should be encapsulated between single quotations. Example: 'smith, john'). The filter takes vales that are equal to one of this values.

EXCLUDEGROUP: This filter is more complex and will be explained with and example. Let's suppose I want to show all the clients that bought more than 10% of the total sails. This clients will, for example, 3 or 4, because the rest of the clients are the ones who bought for less than 10% of the total. I want to show the rest of clients (less than 10%), in a separated node called "OTHERS", without discrimination, and grouping all the results in one big node. Besides, I have some special clients, that I want to list always, no matter if they reach the 10% limit. All this options are possible with the EXCLUDEGROUP filter. This filter creates the "OTHERS" group considering 2 parameters. The first one declares the operation to make, which possible values are: GreaterThan, LesserThan, GreaterAverage, LesserAverage, Top, RangeAverage, Range. The other one declares the value to consider. For example, the above case, we had as operation the GreaterAverage, and as value 10%. The filter will get the ones who has more than 10% and groups the rest. Finally the "OBLIGARORYVALUE" parameter allows you to define a comma separated list of the values that should appear always (obligatory).

DataType: gives the data type of the filter. In case that no dimension is associated for the filter, and it would be just a parameter for the report, it is possible specify the data type in this value.

Visual: indicates if the filter should be prompted to the user for modification, or not (if not the default value of the parameter will be used always).

DataSourceDefinition : declares the data sources. Is the most important part of the RSD.

Attributes:

Name: Name of the data source. Example: MYSQL_CLIENT_DATA

SourceType: Type of the data source. Possible values are: SQL, XML, o EXCEL. SQL type is for database sources, using SQL queries. XML is used for data included in XML files. Excel type is used to get data from an excel document.

Expression: Expression used to get data. In SQL case will be the query ("Select client from… "). In the excel and xml options, this is the path to the file (absolute or relative). The datasources directory in sourcefiles is frequently used to put the data sources files.

ExternalConnectionString: declares the connection to the datasource if necessary. It's better to use the other values for this option, like ExternalConnectionLocalUrl. Example: "DatabaseConfigurationClassName=com.mysql.jdbc.Driver;DatabaseConfigurationLocalUrl=jdbc:mysql://localhost:3306/mydb;DatabaseConfigurationDistributedUrl=java:comp/nv/jdbc/MyUrl;DatabaseConfigurationUser=user;DatabaseConfigurationPassword=password".

ExternalConnectionClassName: declares the name of the jdbc driver to use. It is the full class name, for example:

Sun ODBC: **sun.jdbc.odbc.JdbcOdbcDriver**

Oracle: **oracle.jdbc.driver.OracleDriver**

ExternalConnectionLocalUrl: is the url used to connect to the db with the driver. It's syntax is specified by the product builder.

MySQL: **jdbc:mysql://localhost:3306/foodmart**

Oracle: **jdbc:oracle:thin:@host:port:base_name**

ExternalConnectionDistributedUrl: string connection in case that the data source is remote, connecting to other Server. Example: java:comp/nv/jdbc/intercargo

ExternalConnectionUser: Name of the user of the database. Example: root

ExternalConnectionPassword: Password of the user of the data source. Example: 1

Encrypt option: the name and password to connect to data base are stated in the xml file. As this may be dangerous, the encrypting method of LocalKeyEncrypter class may be used. The result of this method should be saved in the xml. The idea of the LocalKeyEncrypter is to use a key based on local machine options, like machine name. This makes more difficult to decrypt the data if the xml file is taken from the server

DatePattern: specify the format of the DATE data type inside a database. For example, a DATE may be saved with dd/MM/yyyy pattern or mm/DD/yyyy. This value allows the application to transform those dates in native types.

DateTimePattern: idem DatePattern, but for DATETIME fields.

FloatPattern: idem as DatePattern but for Float fields. Specify the format a float is saved in a database.

SheetName (Only for Excel data sources): Gives the name of the sheet to use in an Excel document.

DataInitialCell (Only for Excel data sources): It's a referente to the inicial cell where start taking data. Default is A1. Sintaxis for the cell is like excel, Char|Number

DataEndingCell (Only for Excel data sources): It's a reference to the last cell, where you should stop collecting data. For example, you may want to get the first n rows of the sheet.

ParameterValue: default value and filter behaviour.

Attributes:

FilterDefinitionName: Name of the filter. Example: DESDE_CLIENTE

FilterParameter: identifies the parameter we are using inside the filter. This is because some filters has more than one parameter associated. For example, range filter has parameters FROM and TO. Possible values are: FROM, TO, VALUE, VALUES, OPERATION o OBLIGATORYVALUE

For the filters BEGINWITH, ENDWITH, EQUALTO, GREATERTHAN, INCLUDES, LESSTHAN y RANKING: the only possible parameter is VALUE

For RANGE filters: use FROM and TO

For IN filters: use VALUES

For EXCLUDEGROUP filters: use FROM, TO, VALUE, OPERATION or OBLIGATORYVALUE (the FROM and TO are used if the operation selected is a ranged one).

DefaultValue: Default value that will have the parameter.

Localizations: defines multilanguage based on code translation.

Attibutes:

Locale: indicates the language that will be defined. Example: EN

Literals: defines each text to translate

Attributes:

Code: Code that identifies the corresponding literal. Example: 01

Value: String corresponding for the translation of that code. Example: Client

Using these values, you can for example, use a code (01, 02, 03) as a dimension name, that will be searched in literals, making the translation for its corresponding literal according to the configured locale.