DBEnrich operator

This operator can be used to enrich stream objects with information from a database.

State of the operator is alpha. In case of error, please report.

Parameter

- connection: The database connection to use. Look at Database Feature for a detailed description.
- query: The SQL query that should be used to enrich den current tuple. The query can contain placeholder for variables that should come from the current input tuple. A placeholder is defined (similar to PreparedStatements in Java) with '?': "SELECT name, description FROM categories WHERE id=?". The attributes used in the SQL string are attached to the current event (here name and description)
- attributes:A list of attributes from the input schema that should be used in the query, i.e. for each '?' there must be exactly one corresponding attribute in the list (order is important!). If you want to use an attribute value multiple times, you will need to repeat the attribute.
- outerJoin (Boolean): If for a tuple no result is found in the database, the corresponding tuple is removed. If set to true, evet with no corresponding entry in the database are not remove.
- multiTupleOutput (Boolean): If set to false, only the first result of the database is attached, else each return will be attached with the current tuple, i.e. multiple output tuple will be created.

- caching (Boolean): Should elements from the datebase be cached, i.e. instead of query the database for each input element, look inside the cache first.
- removalStrategy: If the cache reaches its limit, what is the strategy that should be used, to remove elements from the cache.
- expirationTime: How long should elements be cached, before they are removed automatically.
- cacheSize: How many elements should be cached
- uniqueKeys: To optimize processing, the attributes from the database that are prime (i.e. distinct each value from each other)

Beta-Options (use with care, may not work in any cases!):

- noOfWorkers: Typically, this enrich operator will not use any worker, i.e. the source thread will be blocked, until there is a result. To allow concurrent accesses to the enrichment source, the number of workers can be raised. Because order is still kept, no enriched element will overtake another one.
- keepOrder (Boolean, default: true): When set to false and in combination with noOfWorkers, the output order is no longer guaranteed and the worker that delivers the results first, will send there output directly.

Example

```
// Using with attributes from the source to retrieve values from database
#PARSER PQL
#RUNQUERY
out = dbenrich({connection='connectionName', query='SELECT name, description FROM categories WHERE id=?',
attributes=['category']}, nexmark:auction))
```

Another example uses a timer source as trigger to retrieve values from a database in a regular fashion:

```
// Using a timer as trigger to retrieve values
#PARSER PQL
#RUNQUERY
timer = TIMER({PERIOD = 1000, SOURCE = 'source'})
out = dbenrich({connection='connectionName', query='SELECT * FROM mytable', attributes=[]}, timer))
```