Database - Bug #3138

case-sensitivity propagation

06/28/2016 03:06 PM - Ovidiu Maxiniuc

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**case_num:**	
vendor_id:	GCD	**version:**	

Description

History

#1 - 06/28/2016 03:30 PM - Ovidiu Maxiniuc

In 4GL the case sensitivity of some character parameters is propagated to the result of the character functions.

This is handled in base language functions but it incorrectly implemented in database related generated code. With the commit of revision 11055 some work was done for fixing usage of character functions in queries, but the implementation is rather heuristic than correct: the result is case-sensitive if any of the character arguments of a function is also case-sensitive.

The situation in 4GL is following:

- almost all of character functions propagate the case-sensitiveness (like: upper / CAPS, LOWER / LC, CODEPAGE-CONVERT, FILL, if, maximum, minimum, quoter, replace, including functions that do not process directly the parameter: dbparam, dbtype) regardless of the position of such argument;
- trim, left-trim and right-trim: if the 1st argument is sensitive, then the result is, too. The case sensitiveness of the second optional parameter is always ignored;
- entry: the character parameters are 2nd and 3rd. The result is always ignore-case.

A procedure with testcases was uploaded as uast/case-sensitivity-functions.p.

The result is quite unexpected. My feelings were telling me that if the processed value is case sensitive the result should also be. But if you pass a case-sensitive list to entry() the result is case insensitive :(. OTOH, I did not expected that the result of some functions like dbparam and dbtype to preserve this attribute because I think that they only use the argument as a key to some kind of internal map and return the string associated with it.

I guess the correct way to implement this is to check each of the above cases. The current solution only works in vast majority of cases because calling trimming functions with a case-sensitive value as optional parameter is quite unusual.