# Reflections

## Safe macro

Assuming the variable is a reference to a object with the interface closeable. The function in the parameter form has to be evaluated before the object can be closed. But the value from the function should also be returned, and the function has to be evaluated last for that to happen. The function also need the object defined in the parameter variable to work. A macro makes this possible. A function with the same structure would not behave as desired. The form parameter would not be able to be evaluated since it needs the object defined in the parameter variable. The macro in contrast does not evaluate it's formparameter until after the object is defined and accesseable.

The parameters in a function is evaluated and then passed to the function body. In a macro the parameters does not get evaluated as they get passed.

## SQL-like macro

It does not seem likely that a function would be a good replacement for the SQL-like macro, at least not if it's supposed to be called as in the assignment description. A specific evaluationorder is required for the desired result, and some functions that's used in the macro body need the returnvalues from other functions as parameters. Evaluation of the parameters need to be postponed for the SQL-like macro to function. The return table can't for example be sorted before there is anything to sort. A function could not achieve the same result and have the same structure as the macro, because the evaluation of the parametervalue can't be postponed in a function.