

ICode elements

```
%=====
%
%
% Input Clauses:
% Clauses which generate entity instances:
% generate(Class,Instance)
% generate(Function,ArgTypeList,ArgList,ResultType,Result)
%
% Functions over entities which return scalars:
% restrict(Function,ArgTypeList,ArgList,Result)
%
% Clauses describing the subqueries of DAPLEX:
% restrict_subquery(SubqueryType,SetCodeList,PredCodeList)
% where,
%     SubqueryType = {all(RestrictedVar), some(RestrictedVar),
%                     no(RestrictedVar),
%                     exactly(RestrictedVar, Value),
%                     least(RestrictedVar, Value),
%                     most((RestrictedVar, Value))}
%
% generate_subquery(Class,GeneratedVar,SubqueryType,
%                  ThenCodeList, ElseCodeList)
%
% where,
%     SubqueryType = if(ConditionCode), union(VarA, VarB),
%                   intersection(VarA, VarB), difference(VarA, VarB)
%
% or:
% generate_subquery(Class,GeneratedVar,SubqueryType,
%                  [], [])
%
% where,
%     SubqueryType = explicit(A_List_Of_CodeList) | the(CodeList)
%
% Boolean expressions:
% logical_and(Lhs, Rhs) | logical_or(Lhs,Rhs) | logical_not(CodeList)
%
% Expressions which do not access the database:
% expression([], RequiredVars, expr(Op, Lh, Rh))
%     Op = {=, >, >=, <, =<, \== } (i.e. comparison operations)
%     e.g. "X > Y" ==> expression([], [X,Y], expr(>, X, Y))
%
% expression(Result, RequiredVars, expr(Op, Lh, Rh))
%     Op = {= (assignment), +, -, *, /} (i.e. arithmetic operations)
%
% expression(Result, [RequiredVar], relative(Class, RequiredVar, Result))
% NB: "Class" in "relative/3" is the entity class of "Result".
%
% expression(Result, RequiredVars, int_set(Start, End, Result))
%
% expression(Result, [Var], expr(Var))
% expression(Result, [], expr(Constant))
% (NB: this forms of expression/3 may appear in "explicit"
%     generate_subquery, e.g. "for each i in {1, position(p), ...}")
%
% Expressions associated with aggregate functions:
% expression(Result, RequiredVars, Op, Code)
%     Op = {maximum(Var), minimum(Var), count(Var),
%          average(Var), total(Var)}
%
%=====
```