Query

AQL

'<' <class> '>'

the result is all objects of the specified class

Example:

<"A">

<"Square":"SecondModel001":"My Second Model Type">

'{' <object> '}'

the result is a object of the specified name

Example:

{"A1"}

AQL

<AQL expression> '->' | '<-' | '->>' | '<<-' <Relation>

The result contains all objects which are linked through the given relation with at least one object from the AQL expression

'->'	returns all direct targets of the relation
'<-'	returns all direct start objects of the relation
'->>'	returns all transitive targets of the relation
'<<-'	returns all transitive start objects of the relation

Example:

{"A1"}->"requires "

{"A1"}<-"requires"

<"A">->"requires"

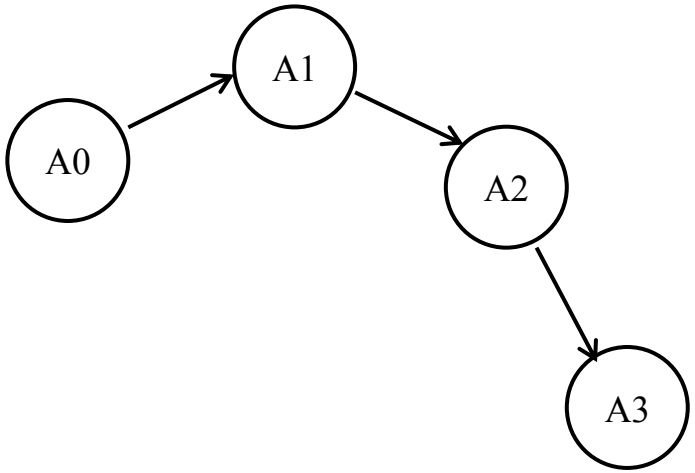
<"A"><-"requires"

{"A0"}->>"requires“

{"A3"}<<-"requires“

({"A2"}->>"requires") -> "has list"

<"A">->"requires" >"B"<



AQL

<AQL expression> '->' | '<-' '<' <Relation> '>'

The result contains all connectors of the specified relation which have as start or target object one of the objects in the AQL expression

'->' returns all connectors originating from the objects of the AQL expression

'<-' returns all connectors ending in the objects of the AQL expression

Please note similarities and differences with before: if you use the '<' and '>' symbols, the result contains the connectors and if you don't use them, it contains the objects

Example:

<"B">-><"requires">

<"A"><-<"requires">

{"List003"} <- <"has list">

{"A1"} -> <"has list">

AQL

<AQL expression> '-->' | '-->>' <Attribute>

The result contains all objects which are referenced in the specified attribute of any of the objects in the AQL expression

The '-->>' operator returns is all objects which are transitively referenced in the specified attribute of any of the objects in the AQL expression

Example:

<"A"> --> "IsRunBy"
<"A"> -->> "IsRunBy"
{"A5"} --> "IsRunBy"
{"A5"} -->> "IsRunBy"
{"A5"} -->> "IsRunBy" >"Rectangle"<

<AQL expression> '<--'

The result contains all objects which refer any of the objects in the AQL expression

Example:

<"Rectangle"> <--

AQL

<AQL expression> '[' <Value> <Operator> <Value> ']'

The result contains all objects, whose attributes fulfill the defined criteria

Constants (numbers, strings) can only be at the right of the operator

To the left of the operator there are only attributes or variable references

Example:

(<"A"> [?"Description" like ""]) AND (<"A"> [?"A_cost" >=10])

<"A">[?"Description" like "*Test*"]

(<"Rectangle">[?"Name" like "M*"]) AND (<"Rectangle">[?"Area" <= 20])

<"A">[?"Name" like "????e?"]

AQL

<AQL expression> '['<Value>']' '['<Value> <Operator> <Value>']'

The result contains all objects of the start query where their record attribute or attribute profile fulfills the defined criteria

The first value specifies the name of the record attribute or attribute profile.

See above the rules for the second expression

Note: In case of a record attribute, the criteria is always fulfilled, if at least a table row of the record attribute meets the defined criteria.

Example:

record attribute: <"List">[?"Classification"][?"State" = "Authorized"]

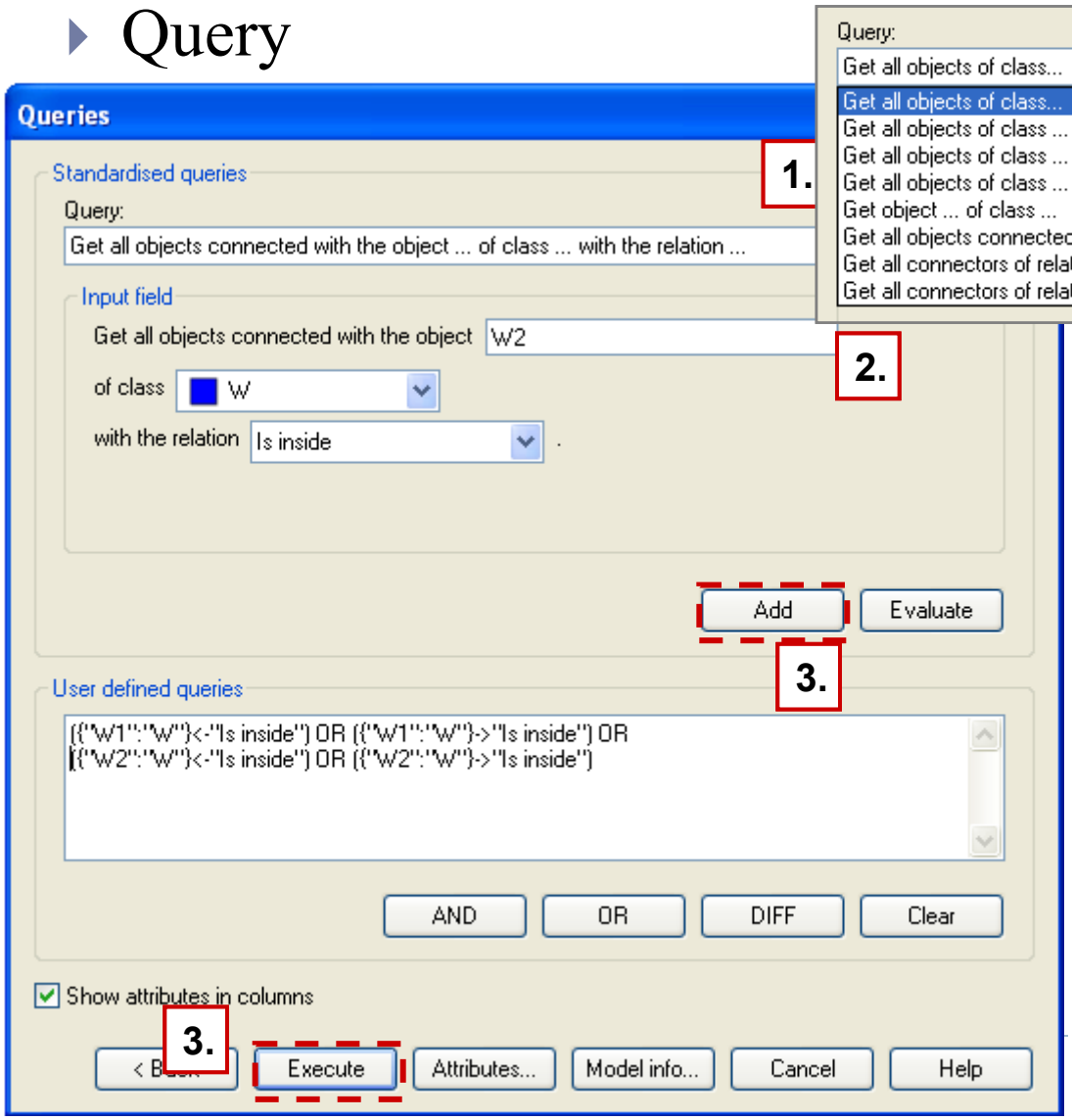
attribute profile: <"A"> [?"Availability"][?"Days per week" >= 3]

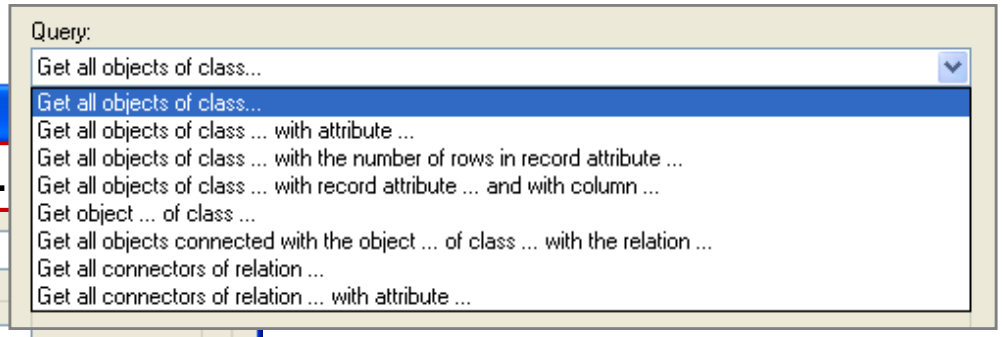
Query

- Query
  - Standardized queries
    - Standardized queries as "to complete text", which are completed by the user. For execution, no AQL knowledge is required.
  - User-defined queries
    - Queries which are defined by the user through standardized queries in AQL syntax. For execution AQL knowledge is required.

Query

- Query





Not specific for a method and their modelling language, but use the classes and attributes of the modelling language

Generally held queries – "Wording is standardized"

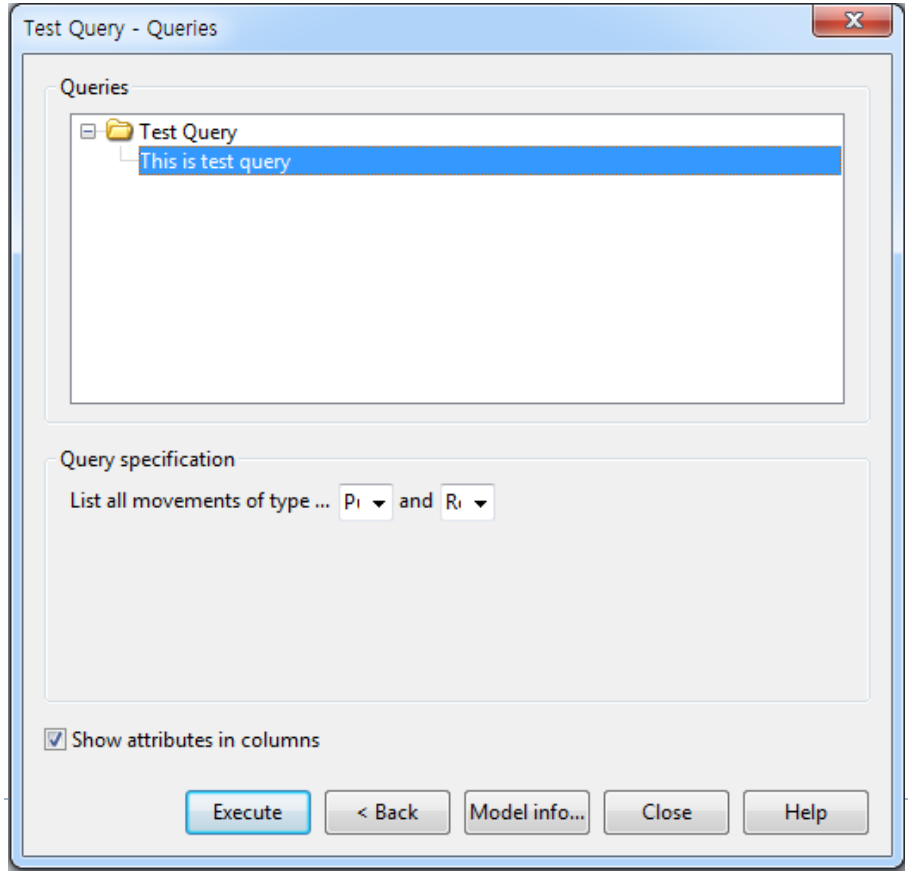
Queries, which are complete defined by the user by using AQL

Queries which are combined by the user through usage of standardized queries

Session dependent, regarding of evaluation parameters

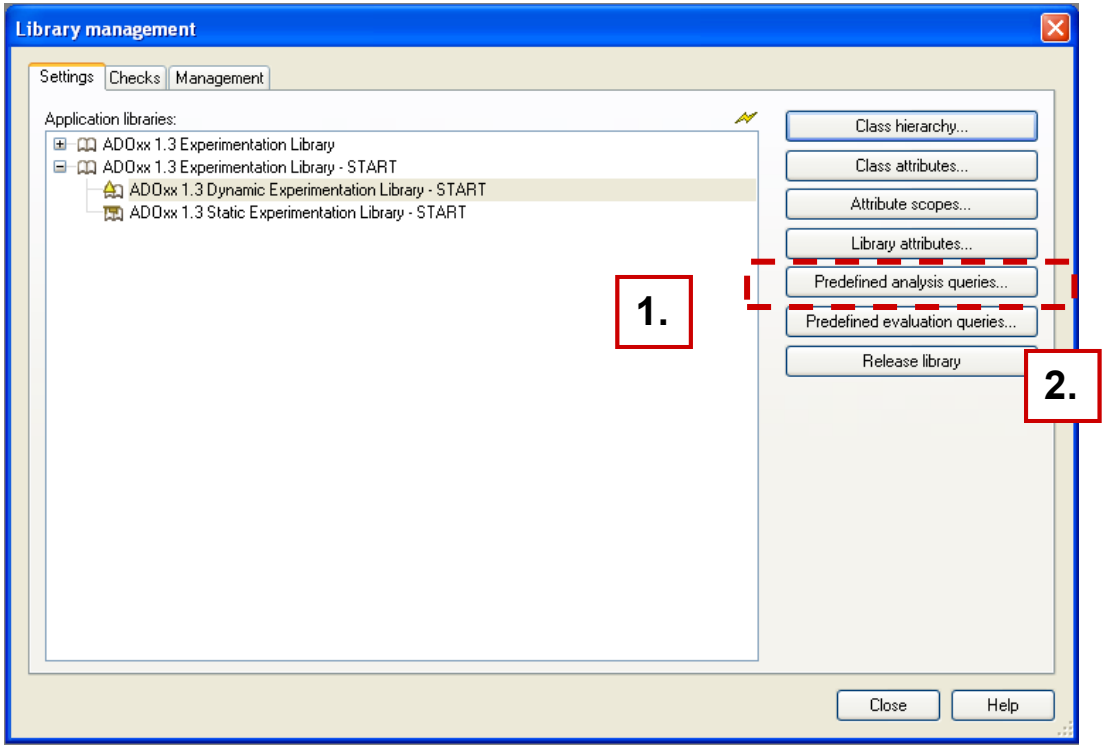
Query

- Query
  - Predefined queries
    - Professional queries, which are business or method specific defined.



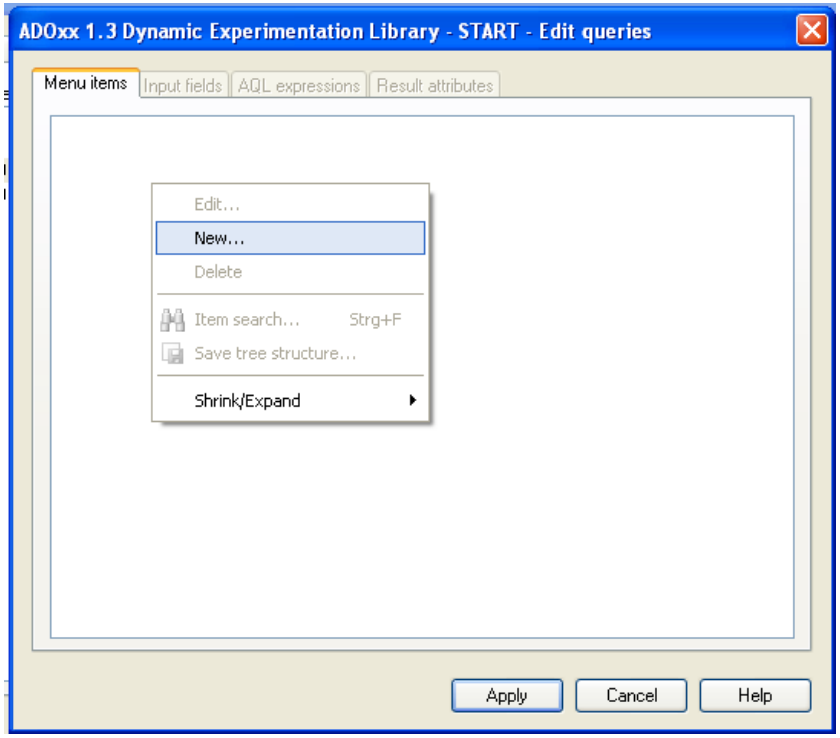
Query

- Query
  - Predefined queries



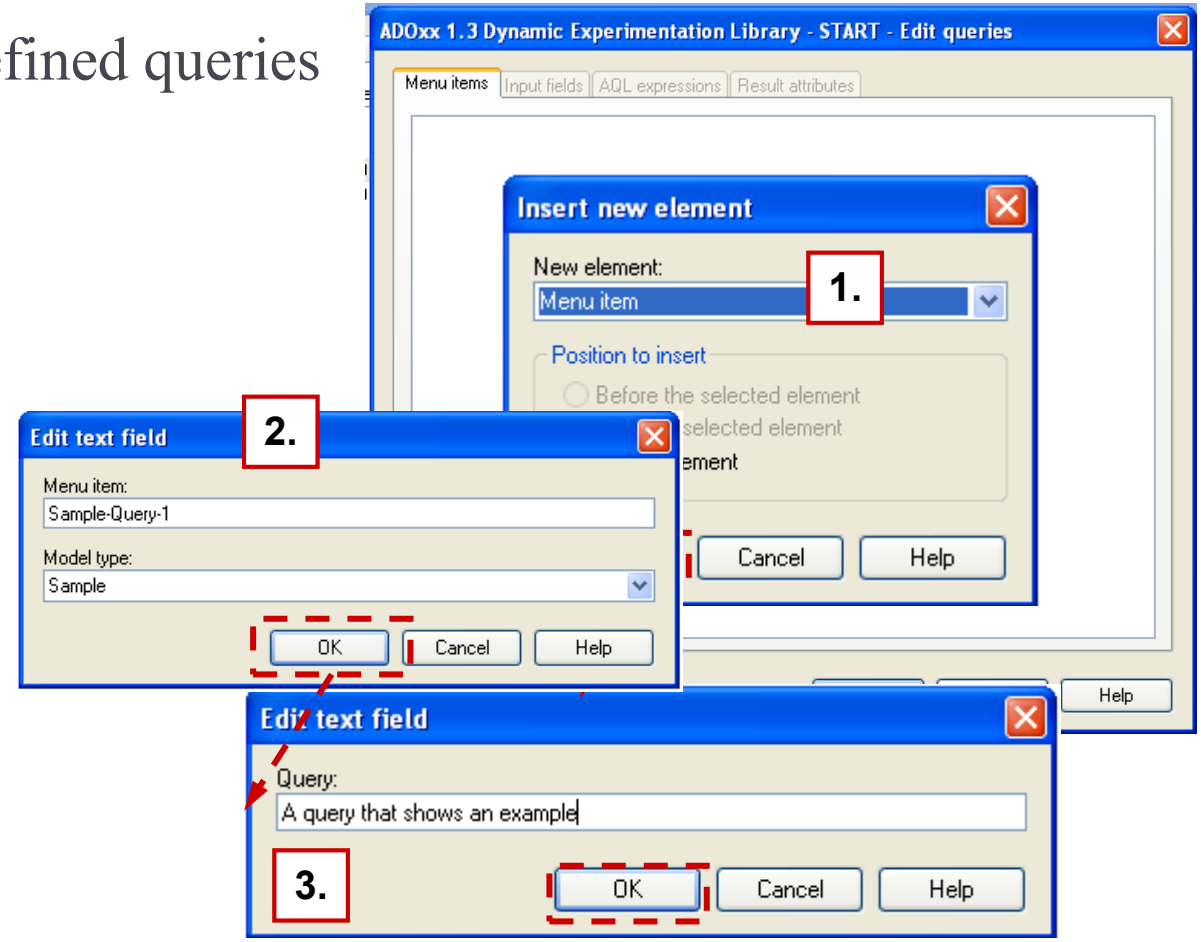
Query

- Query
  - Predefined queries



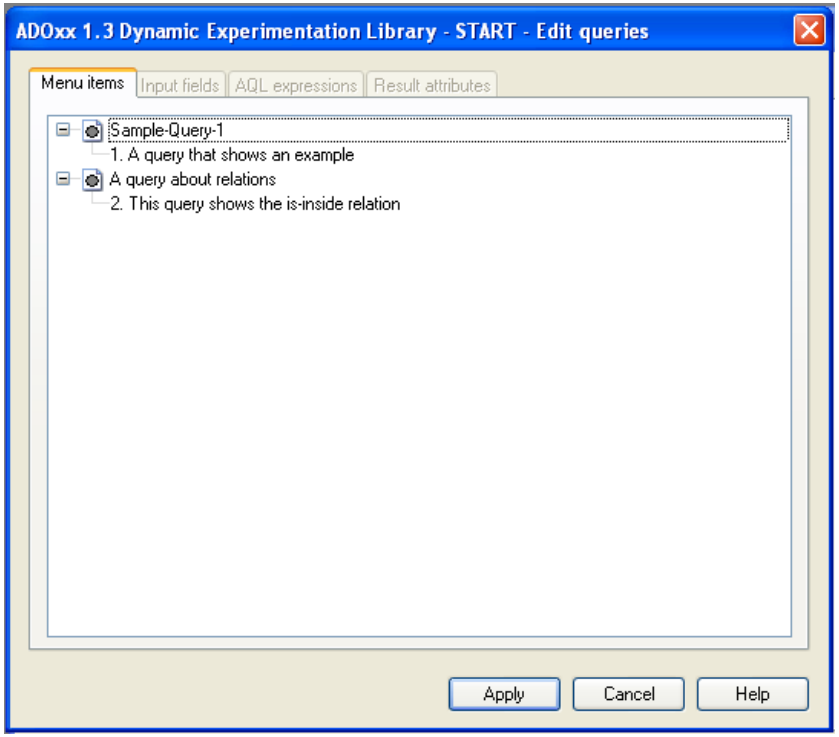
Query

- Query
  - Predefined queries



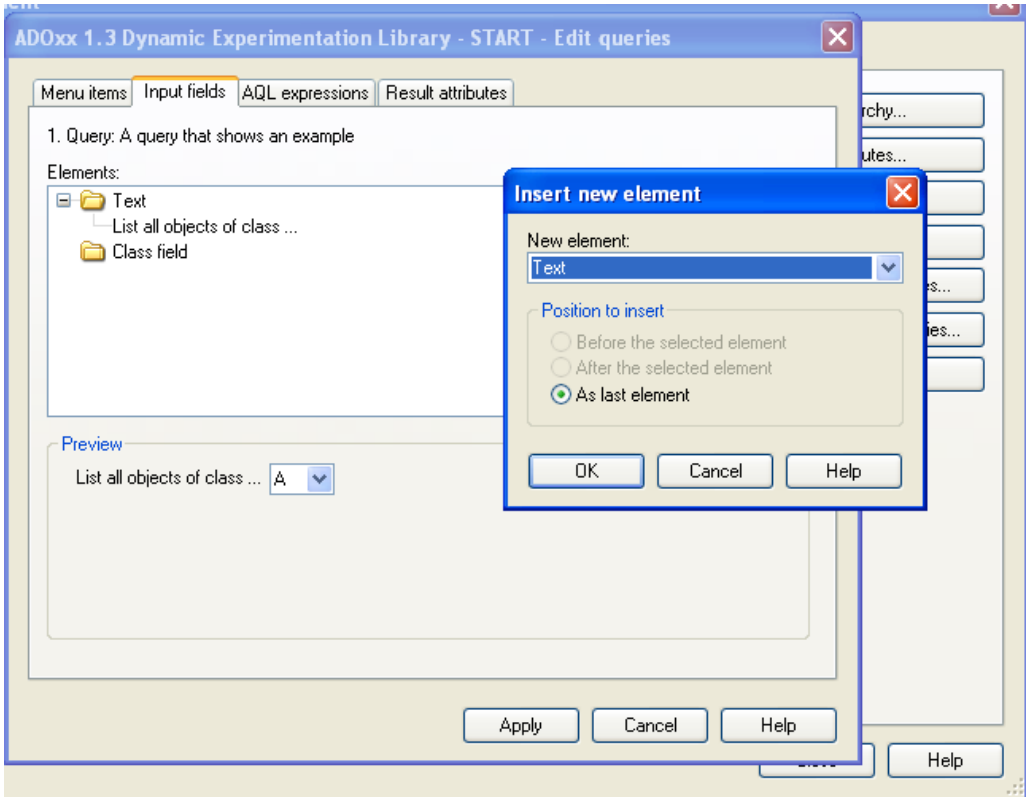
Query

- Query
  - Predefined queries



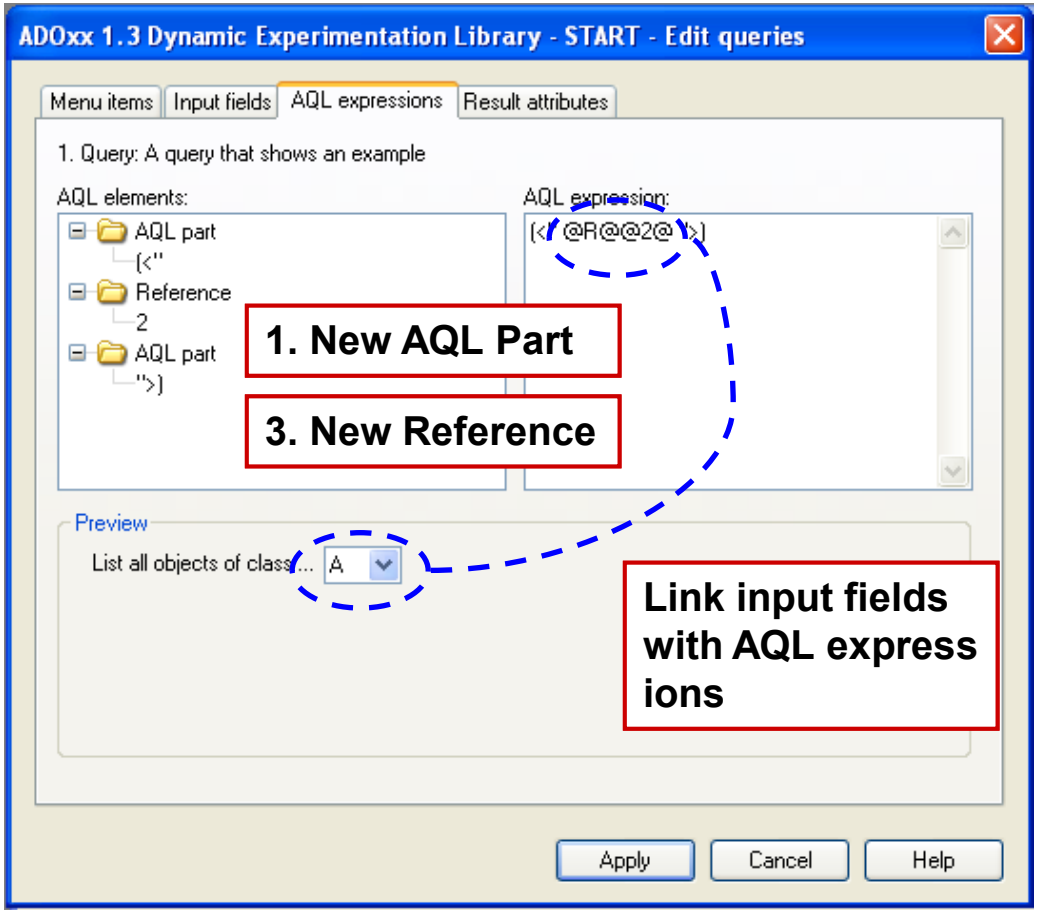
Query

- Query
  - Predefined queries



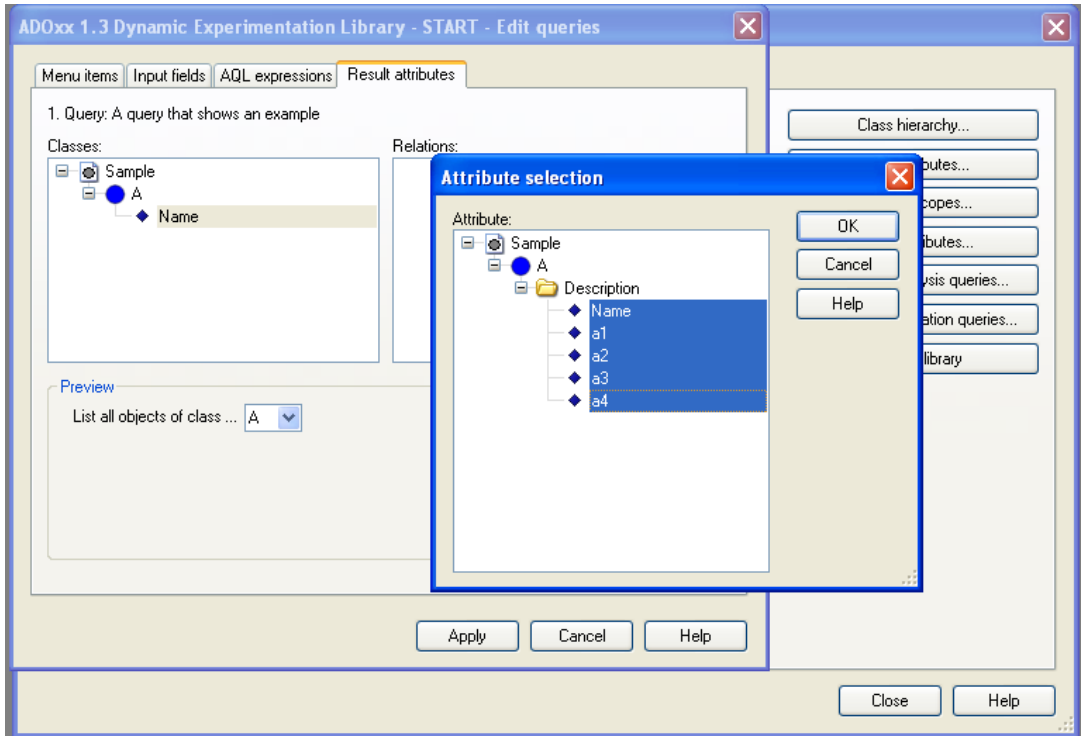
Query

- Query
  - Predefined queries



Query

- Query
  - Predefined queries



Query

- AdoScript for Query
  - CC “AQL” EVAL_AQL_EXPRESSION expr:*strValue* (modelid:*intValue* | modelscope)
  - Returns
    - ecode: intValue
    - objids: strValue
  - Example

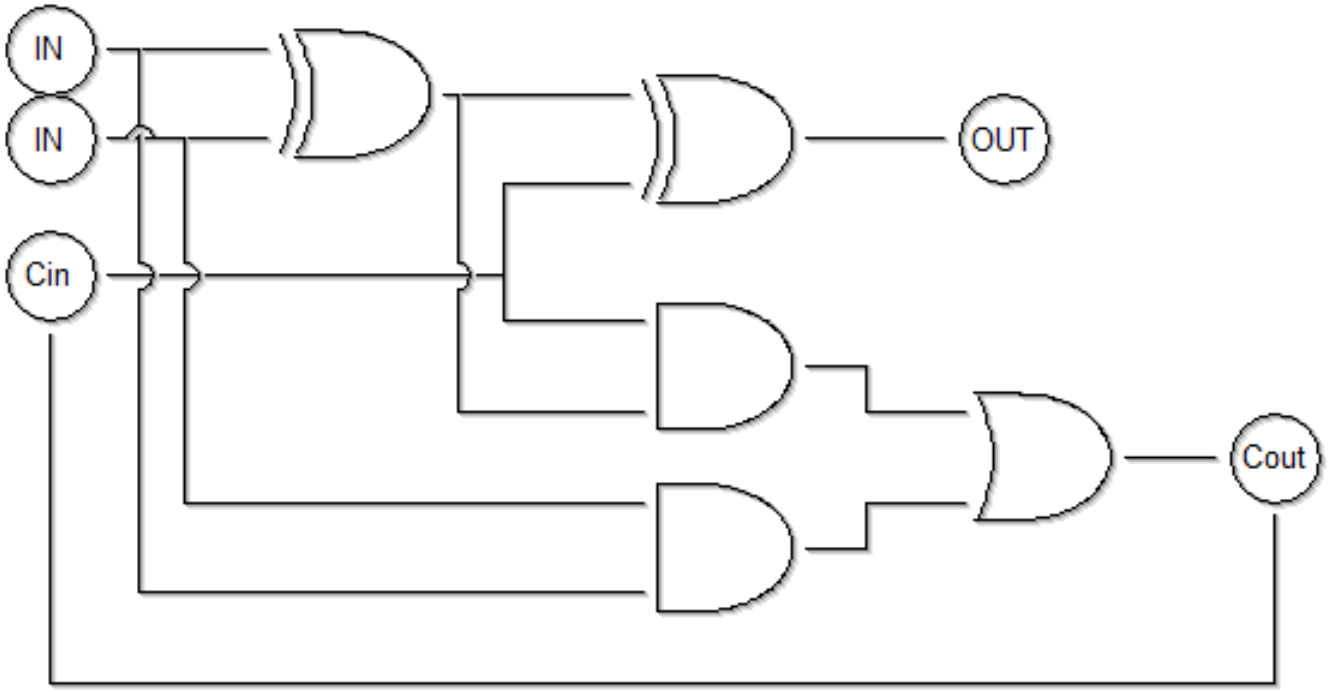
```
CC "AQL" EVAL_AQL_EXPRESSION expr:"<\"A\">" modelid: (modelid)
IF (ecode = 0) {
   CC "AdoScript" INFOBOX ("Found objects: " + objids)
}
ELSE {
   CC "AdoScript" INFOBOX "An error has occured!"
}
```

Practice

VHDL

- 디지털 회로 모델링 및 시뮬레이션
  - 각 논리 게이트 모델링
  - 디지털 회로도 구성
  - 시뮬레이션 수행

VHDL



VHDL

- ADOxx Training week 3 part 3 파일 참조