Content Models for RuleML

David Hirtle

2006-05-17, version 0.9

Introduction

This document is a collection of content models, i.e. the content permitted within a particular tag, for all RuleML tags as of version 0.9, organized alphabetically by module name. Each module is a grouping of related (XML) elements and/or attributes (prefixed with “@”). The content models are given in BNF-like DTD syntax. See http://www.ruleml.org/0.9/xsd/modules for the actual XML schemas of the modules and the RuleML glossary for the meaning of each tag.

Since RuleML is a family of sublanguages, it is important to note that the content model of a given tag often varies according to the current sublanguage. In such cases, all variations of the content model are provided along with the corresponding sublanguage(s). The modularization of RuleML, including all sublanguages, is explained at http://www.ruleml.org/modularization.

Content models may also vary depending on context, i.e. surrounding elements (especially parent elements). In these cases, the content models are listed under a heading such as “within x…” where x indicates the context.

For clarification on any RuleML-related topic, including this document, the RuleML-all mailing list may be quite helpful. The RuleML tutorial serves as an introduction.

Index

Atom

Atom

```
(context sensitive; see also the Holog module)

attributes: @closure

in datalog, nafdatalog, nafnegdatalog, negdatalog:
(
 (oid)?, degree?, (op | Rel), (slot)*,
 ( (arg|Ind|Data|Skolem|Var|Reify)+, (slot)* )?
)

in bindatalog:
(
 (oid)?, degree?, (op | Rel), (slot)*,
 ( (arg|Ind|Data|Skolem|Var|Reify), (arg|Ind|Data|Skolem|Var|Reify), (slot)* )?
)

in bindatagroundlog and bindatagroundfact:
(
 (oid)?, degree?, (op | Rel), (slot)*,
 ( (arg|Ind|Data|Skolem|Reify), (arg|Ind|Data|Skolem|Reify), (slot)* )?
)

in hornlog & up (except framehohornlogeq):
(
 (oid)?, degree?, (op | Rel), (slot)*, (resl)?,
 (
  ( ((arg|Ind|Data|Skolem|Var|Reify|Cterm|Plex)+, (repo)?) | (repo) ),
  (slot)*, (resl)?
 )?
)
```

degree

```
in all sublanguages: (Data)
```

op

```
(context sensitive; see also the Holog, Equality and Cterm modules)

within Atom...

in all sublanguages: (Rel)
```

Rel

```
attributes: @uri

in all sublanguages: (#PCDATA)
```

Connective

Implies

```
attributes: @closure, @direction, @kind ( + @mapDirection and @mapClosure in folog & up)

in datalog & down and hornlog:
( oid?, ( head, body) | ( body, head) | ( (Atom | And | Or), Atom ) )

in negdatalog:
( oid?, ( head, body) | ( body, head) | ( (Atom | And | Or | Neg), (Atom | Neg) ))

in nafdatalog & nafhornlog:
( oid?, ( head, body) | ( body, head) | ( (Atom | And | Or | Naf), Atom ) )

in nafnegdatalog:
(oid?, ( head, body) | ( body, head) | ( (Atom | And | Or | Neg | Naf), (Atom | Neg) ))

in hornlogeq:
( oid?, ( head, body) | ( body, head) | ( (Atom | And | Or | Equal), (Atom | Equal) ))

in hohornlog: ( oid?, ( head, body) | ( body, head) | ( (Hterm | And | Or), Hterm) )

in hohornlogeq: ( oid?, ( head, body) | ( body, head) | ( (Hterm|And|Or|Equal), (Hterm|Equal) ) )

in framehohornlogeq:
(
   oid?, ( head, body ) | ( body, head ) |
                (
                 (Atom | Hterm | InstanceOf | SubclassOf | And | Or),
                 (Atom | Hterm | InstanceOf | SubclassOf)
                )
)

in dishornlog: ( oid?, ( head, body) | ( body, head) | ( (Atom | And | Or), (Atom | Or)))

in folog:
(
   oid?, (head, body) | (body, head) |
               (
                 (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists ),
                 (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )
               )
)

in naffolog:
(
   oid?, (head, body) | (body, head) |
               (
                 (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists ),
                 (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )
               )
)

in fologeq:
(
   oid?, (head, body) | (body, head) |
               (
                 (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal ),
                 (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal )
               )
)

in naffologeq:
(
   oid?, (head, body) | (body, head) |
               (
                 (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists | Equal ),
                 (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal )
               )
)
```

body

```
in datalog & down and hornlog, dishornlog, and hohornlog: (Atom | And | Or)

in negdatalog: (Atom | And | Or | Neg)

in nafdatalog & nafhornlog: (Atom | And | Or | Naf)

in nafnegdatalog: (Atom | And | Or | Neg | Naf)

in hornlogeq: (Atom | And | Or | Equal)

in hohornlogeq: (Hterm | And | Or | Equal)

in framehohornlogeq: (Atom | Hterm | InstanceOf | SubclassOf | And | Or)

in folog: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

in naffolog: (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists )

in fologeq: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal )

in naffologeq: (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists | Equal )
```

head

```
in datalog & down, nafdatalog, hornlog, and nafhornlog: (Atom)

in negdatalog & nafnegdatalog: (Atom | Neg)

in hornlogeq: (Atom | Equal)

in hohornlog: (Hterm)

in hohornlogeq: (Hterm | Equal)

in framehohornlogeq: (Atom | Hterm | InstanceOf | SubclassOf)

in dishornlog: (Atom | Or)

in folog & naffolog: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

in fologeq: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal )

in naffologeq: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal )
```

Integrity

```
attributes: @closure, @direction

in datalog & down and hornlog and dishornlog: ( oid?, (formula | Atom | And | Or) )

in negdatalog: ( oid?, (formula | Atom | And | Or | Neg) )

in nafdatalog: ( oid?, (formula | Atom | And | Or | Naf) )

in nafnegdatalog: ( oid?, (formula | Atom | And | Or | Neg | Naf) )

in hornlogeq: ( oid?, (formula | Atom | And | Or | Equal) )

in nafhornlog: ( oid?, (formula | Atom | And | Or | Naf) )

in hohornlog: ( oid?, (Hterm | And | Or | Neg | Implies) )

in hohornlog: ( oid?, (Hterm | And | Or | Neg | Implies | Equal) )

in framehohornlogeq:
( oid?, (Atom|Hterm|InstanceOf|SubclassOf|Signature|And|Or|Neg|Implies|Equal) )

in folog: ( oid?, (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists ) )
in naffolog: ( oid?, (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists ) )

in fologeq: ( oid?, (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal ) )

in naffologeq: (oid?, (Atom | And|Or | Neg|Naf | Implies|Equivalent | Forall|Exists | Equal) )
```

Equivalent

```
attributes: @closure ( + @mapDirection and @mapClosure in folog & up)

in datalog & down and up to dishornlog: ( oid?, ( ( torso, torso) | ( Atom, Atom) ) )

in hornlogeq: ( oid?, ( (torso, torso) | ( (Atom | Equal), (Atom | Equal) ) ) )

in hohornlog: ( oid?, ( ( torso, torso) | ( Hterm, Hterm ) ) )

in hohornlogeq: ( oid?, ((torso, torso) | ((Hterm | Equal), (Hterm | Equal))) )

in framehohornlogeq:
(
   oid?, (
          ( torso, torso) |
          (
               (Atom | Hterm | InstanceOf | SubclassOf | Signature | Equal),
               (Atom | Hterm | InstanceOf | SubclassOf | Signature | Equal)
          )
        )
)

in folog and naffolog:
(
   oid?, (torso, torso) |
               (
                  (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists ),
                  (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )
                )
)

in fologeq & naffologeq:
(
   oid?, (torso, torso) |
               (
                  (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal),
                  (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal)
                )
)
```

torso

```
in datalog & down and up to dishornlog: (Atom)

in hornlogeq: ( Atom | Equal )

in hohornlog: ( Hterm )

in hohornlogeq: ( Hterm | Equal )

in framehohornlogeq: (Atom | Hterm | InstanceOf | SubclassOf | Signature | Equal)

in folog and naffolog: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists)

in fologeq & naffologeq: (Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists | Equal)
```

And

```
attributes within Query only: @closure ( + @mapDirection and @mapClosure in folog & up)

in datalog & down, hornlog and dishornlog: ( oid?, (formula | Atom | And | Or)* )

in negdatalog: ( oid?, (formula | Atom | And | Or | Neg)* )

in nafdatalog: ( oid?, (formula | Atom | And | Or | Naf)* )

in nafnegdatalog: ( oid?, (formula | Atom | And | Or | Naf | Neg)* )

in hornlogeq: ( oid?, (formula | Atom | And | Or | Equal)* )

in nafhornlog: ( oid?, (formula | Atom | And | Or | Naf)* )

in hohornlog: ( oid?, (formula | Hterm | And | Or | Neg)* )

in hohornlogeq: ( oid?, (formula | Hterm | And | Or | Neg | Equal)* )

in framehohornlogeq:
( oid?, (formula|Atom|Hterm|InstanceOf|SubclassOf|Signature|And|Or|Neg|Equal)* )

in folog:
( oid?, (formula | Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists)* )

in naffolog:
(oid?, (formula | Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists)*)

in fologeq:
(oid?,(formula | Atom | And|Or | Neg | Implies|Equivalent | Forall|Exists | Equal)*)

in naffologeq:
(oid?,(formula | Atom | And|Or | Neg|Naf | Implies|Equivalent | Forall|Exists | Equal)*)
```

Or

```
attributes within Query only: @closure ( + @mapDirection and @mapClosure in folog & up)

in datalog & down, hornlog and dishornlog: ( oid?, (formula | Atom | And | Or)* )

in negdatalog: ( oid?, (formula | Atom | And | Or | Neg)* )

in nafdatalog: ( oid?, (formula | Atom | And | Or | Naf)* )

in nafnegdatalog: ( oid?,  (formula | Atom | And | Or | Naf | Neg)* )

in hornlogeq: ( oid?, (formula | Atom | And | Or | Equal)* )

in nafhornlog: ( oid?, (formula | Atom | And | Or | Naf)* )

in hohornlog: ( oid?, (formula | Hterm | And | Or | Neg)* )

in hohornlogeq: ( oid?, (formula | Hterm | And | Or | Neg | Equal)* )

in framehohornlogeq:
( oid?, (formula|Atom|Hterm|InstanceOf|SubclassOf|Signature|And|Or|Neg|Equal)* )

in folog:
( oid?, (formula | Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists)* )

in naffolog:
(oid?, (formula | Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists)*)

in fologeq:
(oid?,(formula | Atom| And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal)*)

in naffologeq:
(oid?,(formula | Atom | And|Or | Neg|Naf | Implies|Equivalent | Forall|Exists | Equal)*)
```

formula

```
(context sensitive)

within And/Or...

        in datalog & down, hornlog and dishornlog: (Atom | And | Or)

        in negdatalog: (Atom | And | Or | Neg)

        in nafdatalog: (Atom | And | Or | Naf)

        in nafnegdatalog: (Atom | And | Or | Naf | Neg)

        in hornlogeq: (Atom | And | Or | Equal)

        in nafhornlog: (Atom | And | Or | Naf)

        in hohornlog: (Hterm | And | Or | Neg)

        in hohornlogeq: (Hterm | And | Or | Neg | Equal)

        in framehohornlogeq: (Atom|Hterm|InstanceOf|SubclassOf|Signature|And|Or|Neg|Equal)

        in folog: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists)

        in naffolog: (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists)

        in fologeq: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal)

        in naffologeq: (Atom | And|Or | Neg|Naf | Implies|Equivalent | Forall|Exists | Equal)


within Assert...

        in datalog & bindatalog and up to folog: ( Atom | Implies | Equivalent | Forall )

        in bindatagroundlog: ( Atom | Implies | Equivalent )

        in bindatagroundfact: ( Atom )

        in hornlogeq: ( Atom | Implies | Equivalent | Forall | Equal )

        in hohornlog: ( Hterm | Implies | Equivalent | Forall )

        in hohornlogeq: ( Hterm | Implies | Equivalent | Forall | Equal )

        in framehohornlogeq:
        ( Atom|Hterm|InstanceOf|SubclassOf|Signature|Implies|Equivalent|Forall|Equal )

        in folog and naffolog:
        ( Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists )

        in fologeq and naffologeq:
        ( Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists | Equal )

within Query...

        in datalog, bindatalog, hornlog and dishornlog: (Atom | And | Or | Exists)

        in bindatagroundlog and bindatagroundfact: (Atom | And | Or)

        in negdatalog: (Atom | And | Or | Exists | Neg)

        in nafdatalog: (Atom | And | Or | Exists | Naf)

        in nafnegdatalog: (Atom | And | Or | Exists | Neg | Naf)

        in hornlogeq: (Atom | And | Or | Exists | Equal)

        in nafhornlog: (Atom | And | Or | Exists | Naf)

        in hohornlog: (Hterm | And | Or | Exists | Neg)

        in hohornlogeq: (Hterm | And | Or | Exists | Neg | Equal)
in framehohornlogeq:
(Atom | Hterm | InstanceOf | SubclassOf | And | Or | Exists|Neg|Equal)

in folog: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

in naffolog: (Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists )

in fologeq: (Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal )

in naffologeq:
(Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists | Equal )
```

@kind

```
[optional] (default:fo | lp)
```

@mapDirection

```
[optional] (forward | backward | default:bidirectional)
```

@direction

```
[optional] (forward | backward | default:bidirectional)
```

@mapClosure

```
[optional] (universal | existential)
```

@closure

```
[optional] (universal | existential)
```

Cterm

Cterm

```
attributes: @type

in hornlog & up (except hohornlog, etc):
(
 oid?, (op | Ctor), (slot)*, (resl)?,
 (
  ( ( (arg|Ind|Data|Skolem|Var|Reify|Cterm|Plex)+, (repo)? ) | (repo) ),
  (slot)*, (resl)?
 )?
)
```

op

```
(context sensitive; see also the Atom, Holog and Equality modules)

within Cterm...

in all sublanguages: (Ctor)
```

Ctor

```
attributes: @uri

in all sublanguages: (#PCDATA)
```

Plex

```
(context sensitive)

within Atom, Plex, slot...

        in hornlog & up (except hohornlog, etc):
        (
         oid?, (slot)*,
         (
          ( (arg|Ind|Data|Skolem|Var|Reify|Cterm|Plex)+, (repo)?, (slot)*, (resl)? )?
          |
          ( (repo), (slot)*, (resl)? ) | (resl)
         )
        )

        in hohornlog & up:
        ( (slot)*, (arg | Con | Skolem | Var | Reify | Hterm)*, (repo)?, (slot)*, (resl)? )

within repo...

        in hornlog & up (except hohornlog, etc):
        ( (arg | Ind | Data | Skolem | Var | Reify | Cterm | Plex | repo)* )

        in hohornlog & up: ( (arg | Con | Skolem | Var | Reify | Hterm | repo)* )

within resl...

        in hornlog & up: ( (slot | resl)* )
```

Desc

oid

```
in datalog & down, negdatalog, nafdatalog and nafnegdatalog: (Ind | Data | Var | Skolem | Reify)
in hornlog & up (except hohornlog, etc): (Ind | Data | Var | Skolem | Reify | Cterm | Plex)
in hohornlog & up: (Con | Data | Skolem | Var | Reify | Hterm)
```

Equality

Equal

```
in hornlogeq, fologeq and naffologeq:
(
 (oid)?, (degree)?,
 (side | Ind | Data | Skolem | Var | Reify | Cterm | Plex | Nano),
 (side | Ind | Data | Skolem | Var | Reify | Cterm | Plex | Nano)
)

in hohornlogeq:
(
 (oid)?, (degree)?,
 (side | Con | Skolem | Var | Reify | Hterm | Nano),
 (side | Con | Skolem | Var | Reify | Hterm | Nano)
)

in framehohornlogeq:
(
 (oid)?, (degree)?,
 (side | Con | Skolem | Var | Reify | Hterm | Nano | Get),
 (side | Con | Skolem | Var | Reify | Hterm | Nano | Get)
)
```

side

```
in hornlogeq, fologeq and naffologeq: ( Ind | Data | Skolem | Var | Reify | Cterm | Plex | Nano )

in hohornlogeq: (Con | Skolem | Var | Reify | Hterm | Nano)

in framehohornlogeq: (Con | Skolem | Var | Reify | Hterm | Nano | Get)
```

Nano

```
in hornlogeq, fologeq and naffologeq:
( oid?, (op | Fun), (arg | Ind | Data | Skolem | Var | Reify | Cterm | Plex)* )

in hohornlogeq: ( oid?, (op | Fun), (arg | Con | Skolem | Var | Reify | Hterm)* )

in framehohornlogeq: ( oid?, (op | Fun), (arg | Con | Skolem | Var | Reify | Hterm | Get)* )
```

op

```
(context sensitive; see also the Atom, Cterm and Holog modules)

within Nano...

in all sublanguages: (Fun)
```

Fun

```
attributes: @uri

in all sublanguages: (#PCDATA)
```

Frame

Set

```
in framehohornlogeq: ( (Con | Skolem | Var | Reify | Hterm | Get)* )
```

InstanceOf

```
in framehohornlogeq:
( (Con|Skolem|Var|Reify|Hterm|Get),(Con|Skolem|Var|Reify|Hterm|Get) )
```

SubclassOf

```
in framehohornlogeq:
( (Con|Skolem|Var|Reify|Hterm|Get),(Con|Skolem|Var|Reify|Hterm|Get) )
```

Signature

```
in framehohornlogeq: ( oid, (op | Con | Skolem | Var | Reify | Hterm)?, slot* )
```

Get

```
in framehohornlogeq: ( oid, SlotProd )
```

SlotProd

```
in framehohornlogeq: ( (Con | Skolem | Var | Reify | Hterm | Get)+ )
```

Holog

Hterm

```
in hohornlog & hohornlogeq:
(
   oid?, (op | Con | Skolem | Var | Reify | Hterm), (slot)*,
   (arg | Con | Skolem | Var | Reify | Hterm)*, (repo)?, (slot)*, (resl)?
)

in framehohornlogeq:
(
   oid?, (op | Con | Skolem | Var | Reify | Hterm), (slot)*,
   (arg | Con | Skolem | Var | Reify | Hterm | Get)*, (repo)?, (slot)*, (resl)?
)
```

Atom

```
(context sensitive; see also the Atom module)

within SWSL sublanguages...
in framehohornlogeq: ( oid, (op | Con | Skolem | Var | Reify | Hterm)?, slot* )
```

op

```
(context sensitive; see also the Atom, Cterm and Equality modules)

within Hterm...
in hohornlog & up: (Con | Skolem | Var | Reify | Hterm)
```

Con

```
attributes: @uri, @type

in hohornlog & up: (#PCDATA)
```

Naf

Naf

```
attributes: none ( + @mapDirection and @mapClosure in naffolog & up)

in nafdatalog: ( oid?, (weak | Atom) )

in nafnegdatalog: ( oid?, (weak | Atom | Neg) )

in hohornlog ( oid?, (weak | Hterm) )

in naffolog: ( oid?, (weak | Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists) )

in naffologeq:( oid?, (weak | Atom | And|Or | Neg | Implies|Equivalent | Forall|Exists | Equal) )
```

weak

```
in nafdatalog: ( Atom )

in nafnegdatalog: ( Atom | Neg)

in hohornlog ( Hterm )

in naffolog: ( Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

in naffologeq:( Atom | And|Or | Neg | Implies|Equivalent | Forall|Exists | Equal )
```

Neg

Neg

```
 attributes: none ( + @mapDirection and @mapClosure in folog & up)

in negdatalog and nafnegdatalog: ( oid?, (strong | Atom) )

in hohornlog: ( oid?, (strong | Hterm) )

in hohornlogeq & up: ( oid?, (strong | Hterm | Equal) )

in folog and naffolog: (oid?, (strong | Atom|And|Or|Neg | Implies|Equivalent | Forall | Exists) )

in fologeq and naffologeq:
(oid?, (strong | Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal) )
```

strong

```
in negdatalog and nafnegdatalog: ( Atom )

in hohornlog: ( Hterm )

in hohornlogeq & up: ( Hterm | Equal )

in folog and naffolog: ( Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

in fologeq and naffologeq: (Atom | And|Or | Neg | Implies|Equivalent | Forall|Exists | Equal)
```

Performative

RuleML

```
in all sublanguages: ( oid?, (Assert | Query | Protect)* )
```

Assert

```
 attributes: @mapDirection and @mapClosure

in datalog & bindatalog and up to folog: (oid?,(formula | Atom | Implies|Equivalent | Forall)* )

in bindatagroundlog: ( oid?, (formula | Atom | Implies | Equivalent)* )

in bindatagroundfact: ( oid?, (formula | Atom)* )

in hornlogeq: ( oid?, (formula | Atom | Implies | Equivalent | Forall | Equal)* )

in hohornlog: ( oid?, (formula | Hter m | Implies | Equivalent | Forall)* )

in hohornlogeq: ( oid?, (formula | Hterm | Implies | Equivalent | Forall | Equal)* )

in framehohornlogeq:
(oid?, (formula |Atom|Hterm|InstanceOf|SubclassOf|Signature|Implies|Equivalent|Forall|Equal)* )

in folog and naffolog:
( oid?, (formula | Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists)* )

in fologeq and naffologeq:
( oid?, (formula | Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists | Equal)* )
```

Query

```
 attributes: @closure ( + @mapDirection and @mapClosure in folog & up)

in datalog, bindatalog, hornlog and dishornlog: ( oid?, (formula | Atom | And | Or | Exists)* )

in bindatagroundlog and bindatagroundfact: ( oid?, (formula | Atom | And | Or)* )

in negdatalog: ( oid?, (formula | Atom | And | Or | Exists | Neg)* )

in nafdatalog: ( oid?, (formula | Atom | And | Or | Exists | Naf)* )

in nafnegdatalog: ( oid?, (formula | Atom | And | Or | Exists | Naf | Neg)* )

in hornlogeq: ( oid?, (formula | Atom | And | Or | Exists | Equal)* )

in nafhornlog: ( oid?, (formula | Atom | And | Or | Exists | Naf)* )

in hohornlog: ( oid?, (formula | Hterm | And | Or | Exists | Neg)* )

in hohornlogeq: ( oid?, (formula | Hterm | And | Or | Exists | Neg | Equal)* )

in framehohornlogeq:
(oid?, (formula | Atom|Hterm|InstanceOf|SubclassOf|Signature|And|Or|Exists|Neg|Equal)* )

in folog:
( oid?, (formula | Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists)* )

in fologeq:
( oid?, (formula | Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists | Equal)* )

in naffolog:
(oid?, (formula | Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists)* )

in naffologeq:
( oid?,(formula | Atom | And | Or | Neg | Naf | Implies | Equivalent | Forall | Exists | Equal)*)
```

Protect

```
attributes: @mapDirection and @mapClosure

in datalog & bindatalog and up to folog:
( oid?, (warden | Integrity)+, (formula | Atom | Implies | Equivalent | Forall)* )

in bindatagroundlog: ( oid?, (warden | Integrity)+, (formula | Atom | Implies | Equivalent)* )

in bindatagroundfact: ( oid?, (warden | Integrity)+, (formula | Atom)* )

in hornlogeq:
( oid?, (warden | Integrity)+, (formula | Atom | Implies | Equivalent | Forall | Equal)* )

in hohornlog: ( oid?, (warden | Integrity)+, (formula | Hterm | Implies | Equivalent | Forall)* )

in hohornlogeq:
( oid?, (warden | Integrity)+, (formula | Hterm | Implies | Equivalent | Forall | Equal)* )

in framehohornlogeq:
(
 oid?, (warden | Integrity)+,
 (formula|Atom|Hterm|InstanceOf|SubclassOf|Signature|Implies|Equivalent|Forall|Equal)*
)

in folog and naffolog:
(
 oid?, (warden | Integrity)+,
 (formula | Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists)*
)

in fologeq and naffologeq:
(
 oid?, (warden | Integrity)+,
 (formula | Atom | And|Or | Neg | Implies | Equivalent | Forall | Exists | Equal)*
)
```

warden

```
in all sublanguages: ( Integrity )
```

Quantifier

Forall

```
attributes: none ( + @mapDirection and @mapClosure in folog & up)

in bindatalog, datalog & up to (including) hornlog and dishornlog:
( oid?, (declare | Var)+, (formula | Atom | Implies | Equivalent | Forall) )

in hornlogeq:
( oid?, (declare | Var)+, (formula | Atom | Implies | Equivalent | Forall | Equal) )

in hohornlog: ( oid?, (declare | Var)+, (formula | Hterm | Implies | Equivalent | Forall) )

in hohornlogeq: ( oid?, (declare | Var)+, (formula |Hterm|Implies|Equivalent|Forall|Equal) )

in framehohornlogeq:
( oid?,(declare | Var)+,(formula|Atom|Hterm|InstanceOf|SubclassOf|Implies|Equivalent|Forall) )

in folog and naffolog:
( oid?, (declare | Var)+, (formula|Atom|And|Or|Neg|Implies|Equivalent|Forall|Exists) )

in fologeq and naffologeq:
( oid?, (declare | Var)+, (formula|Atom|And|Or|Neg|Implies|Equivalent|Forall|Exists|Equal) )
```

Exists

```
attributes: none ( + @mapDirection and @mapClosure in folog & up)

in bindatalog, datalog & up to (including) hornlog and dishornlog:
( oid?, (declare | Var)+, (formula | Atom | And | Or | Exists) )

in hornlogeq: ( oid?, (declare | Var)+, (formula | Atom | And | Or | Exists | Equal) )

in hohornlog: ( oid?, (declare | Var)+, (formula | Hterm | And | Or | Exists)

in hohornlogeq: ( oid?, (declare | Var)+, (formula | Hterm | And | Or | Exists | Equal)

in framehohornlogeq:
(oid?, (declare|Var)+, (formula|Atom|Hterm|InstanceOf|SubclassOf|Signature|And|Or|Exists|Equal) )

in folog and naffolog:
( oid?, (declare | Var)+, (formula|Atom|And|Or|Neg|Implies|Equivalent|Forall|Exists) )

in fologeq and naffologeq:
( oid?, (declare | Var)+, (formula|Atom|And|Or|Neg|Implies|Equivalent|Forall|Exists|Equal) )
```

declare

```
in all sublanguages: ( Var )
```

formula

```
(context sensitive; see also the Connective module)

within Forall...

        in bindatalog, datalog & up to (including) hornlog and dishornlog:
        (Atom | Implies | Equivalent | Forall)

        in hornlogeq: ( Atom | Implies | Equivalent | Forall | Equal )

        in hohornlog: (Hterm | Implies | Equivalent | Forall)
        in hohornlogeq: (Hterm | Implies | Equivalent | Forall | Equal)

        in framehohornlogeq:
        (Atom|Hterm|InstanceOf|SubclassOf|Signature|Implies|Equivalent|Forall|Equal)

        in folog and naffolog: ( Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

        in fologeq and naffologeq: (Atom|And|Or|Neg|Implies|Equivalent|Forall|Exists|Equal)

within Exists...

        in bindatalog, datalog & up to (including) hornlog and dishornlog: (Atom|And|Or|Exists)

        in hornlogeq: ( Atom | And | Or | Exists | Equal )

        in hohornlog: (Hterm | And | Or | Exists)

        in hohornlogeq: (Hterm | And | Or | Exists | Equal)

        in framehohornlogeq:
        (Atom | Hterm | InstanceOf | SubclassOf | Signature | And | Or | Exists | Equal)

        in folog and naffolog: ( Atom | And | Or | Neg | Implies | Equivalent | Forall | Exists )

        in fologeq and naffologeq: ( Atom | And|Or|Neg|Implies|Equivalent|Forall|Exists|Equal )
```

Rest

repo

```
in hornlog & up: (Var | Plex)
```

resl

```
in hornlog & up: (Var | Plex)
```

Slot

slot

```
(context sensitive)

attributes: @card, @weight ( + @minCard and @maxCard in framehohornlogeq)

within Atom, etc...

        in bindatalog, datalog & up to hornlog:
        ((Ind|Data|Skolem|Var|Reify),(Ind|Data|Skolem|Var|Reify))

        in bindatagroundlog and bindatagroundfact:
        ( (Ind|Data|Skolem|Reify),(Ind|Data|Skolem|Reify) )

        in hornlog & up (except hohornlog, etc):
        ((Ind|Data|Skolem|Var|Reify|Cterm|Plex), (Ind|Data|Skolem|Var|Reify|Cterm|Plex))

        in hohornlog & hohornlogeq: ( (Con|Skolem|Var|Reify|Hterm),(Con|Skolem|Var|Reify|Hterm) )

        in framehohornlogeq:
        ( (Con|Skolem|Var|Reify|Hterm|Get), (Con|Skolem|Var|Reify|Hterm|Get) )

within Atom-frame...

        in framehohornlogeq:
        ( (Con|Skolem|Var|Reify|Hterm|Get),(Con|Skolem|Var|Reify|Hterm|Get|Set)? )
```

@card

```
[optional] nonNegativeInt
```

@minCard

```
[optional] nonNegativeInt
```

@maxCard

```
[optional] nonNegativeInt
```

@weight

```
[optional] decimal [0,1]
```

Term

arg

attributes: @index

in bindatalog, datalog & up to hornlog: (Ind | Data | Skolem | Var | Reify)

in bindatagroundlog and bindatagroundfact: (Ind | Data | Skolem | Reify)

in hornlog & up (except hohornlog, etc): (Ind | Data | Skolem | Var | Reify | Cterm | Plex)

in hohornlog & hohornlogeq: (Con | Skolem | Var | Reify | Hterm)

in framehohornlogeq: (Con | Skolem | Var | Reify | Hterm | Get)

Ind

attributes: @uri, @type

in all sublanguages: (#PCDATA)

Data

in all sublanguages: (#PCDATA) [optionally datatyped with XSD built-ins]

Var

attributes: @type

in all sublanguages: (#PCDATA)

Skolem

attributes: @type

in all sublanguages: (#PCDATA)

Reify

in all sublanguages: (<xs:any>?)

@type

[optional] string

@index

[required] positiveInt

Uri

@uri

```
[optional] anyURI
```