Extended Example

Hospital Knowledge Base

Definitions and Queries
Extended Query Example: Hospital Knowledge Base

(defconcept facility)
(defconcept hospital :is
   (:and facility
      (:at-least 1 ward-capacity)))

(defrelation ward-capacity :domain hospital)

(tell (:about h-1
   (ward-capacity 120)
   (ward-capacity 120)
   (ward-capacity 100)))

(tell (:about h-2
   (ward-capacity 110)
   (ward-capacity 90)))
Is H-1 a Hospital?

(tell (:about h-1
  (ward-capacity 120)
  (ward-capacity 120)
  (ward-capacity 100))))
Is H-1 a Hospital?

(tell (:about h-1
         (ward-capacity 120)
         (ward-capacity 120)
         (ward-capacity 100)))

Yes, for classified instances, because of the :domain entry in

(defrelation ward-capacity
             :domain hospital
             :range ...)

How Many Wards for H-1?

(tell (:about h-1
  (ward-capacity 120)
  (ward-capacity 120)
  (ward-capacity 100)))
How Many Wards for H-1?

(tell (:about h-1
    (ward-capacity 120)
    (ward-capacity 120)
    (ward-capacity 100)))

(pi h-1) ==> 

(:ABOUT H-1...
    (WARD-CAPACITY 120)
    (WARD-CAPACITY 100))

ONLY 2!
What Does This Query Ask?

(retrieve (?x ?y)
    (> (ward-capacity ?x)
        (ward-capacity ?y)))
What Does This Query Ask?

(retrieve (?x ?y)
  (> (ward-capacity ?x)
      (ward-capacity ?y)))

Implicit :for-some wrapped around query, therefore returns:

((((|I|H-1 |I|H-1) (|I|H-1 |I|H-2)
    (|I|H-2 |I|H-1) (|I|H-2 |I|H-2))

(((|I|H-1 |I|H-1) (|I|H-1 |I|H-2)
    (|I|H-2 |I|H-1) (|I|H-2 |I|H-2))
What Is Wrong with This?

(defrelational ward-capacity) ; no :domain

(retrieve (?x ?y)
  (> (ward-capacity ?x)
     (ward-capacity ?y))
What Is Wrong with This?

(retrieve (?x ?y)
  (> (ward-capacity ?x)
      (ward-capacity ?y)))

Performance Warning: Query scans the entire knowledge base to generate bindings for the variables ?X and ?Y.

Query time solution:

(retrieve (?x ?y)
  (:and (hospital ?x)
        (hospital ?y)
        (> (ward-capacity ?x)
           (ward-capacity ?y))))
Find Hospitals Ordered by Their Largest Wards

(defrel ward-capacity :domain hospital)

(retrieve (?x ?y)
  (:and (> (max (ward-capacity ?x))
          (max (ward-capacity ?y))))

==> (([I|H-1] [I|H-2]))
What About All Wards Larger?

(retrieve (?x ?y)
  (:and (> (min (ward-capacity ?x))
          (max (ward-capacity ?y))))))

==> NIL
Hospital with a Ward Larger Than 100 beds?

(retrieve (?x)
  (:for-some (?len)
   (:and (ward-capacity ?x ?len)
     (>= ?len 100))))

Note the explicit :for-some designation!
Hospital with All Wards Larger Than 100?

(retrieve (?x)
   (:for-all (?len)
      (:implies
         (ward-capacity ?x ?len)
         (>= ?len 100)))))
Special Syntax in :for-all

(retrieve (?x)
  (:for-all (?len)
    (:implies
      (ward-capacity ?x ?len)
      (>= ?len 100)))))

Implication used in :for-all to restrict the domain of the quantified variable (?len)

Alternate possibility:

(...(:for-all (?len)
  (:or
    (not (ward-capacity ?x ?len))
    (>= ?len 100)))))
Implication Equivalence

(:implies A B)

(:or (:not A) B)
Hospital with All Wards Larger Than 100?

(retrieve (?x)
   (:for-all (?len)
      (:implies
         (ward-capacity ?x ?len)
         (>= ?len 100))))

Problem: Couldn’t find a closed set of fillers for the role ward-capacity.
Three Possible Solutions

At the individual level:

```
(tell (:about h-1
       (:exactly 2 ward-capacity))
```

At the relation level:

```
(defrelation ward-capacity ...
   :characteristics :closed-world)
```

At the context level:

```
(setf (open-closed-mode
       (current-context))
   :closed)
```
Hospital with All Wards Larger Than 100?

(retrieve (?x)
  (:and (hospital ?x)
    (:for-all (?len)
      (:implies
        (ward-capacity ?x ?len)
        (>= ?len 100)))))

==> (|I|H-1)
Nested Queries Are OK

(retrieve (?x ?y)
  (:and (hospital ?x) (hospital ?y)
       (:for-all (?a)
         (:implies
           (ward-capacity ?x ?a)
           (:for-some (?b)
             (:and (ward-capacity ?y ?b)
               (> ?a ?b)))))
))

==> ((|I|H-1 |I|H-2))
How To Get Multiple Wards of the Same Size for H-1?

(tell (:about h-1
    (ward-capacity 120)
    (ward-capacity 120)
    (ward-capacity 100)))

Need to make wards individuals, so they can be differentiated.
New Domain Model

(defconcept facility)
(defconcept hospital :is
   (:and facility
       (:at-least 1 hospital-ward)))

(defconcept ward :is
   (:and facility
       (:exactly 1 ward-capacity)))

(defrelation hospital-ward
   :domain hospital :range ward
   :characteristics :closed-world)

(defrelation ward-capacity
   :domain ward
   :characteristics :closed-world)
Auxiliary Relation

(defrelation hospital-ward-capacity
  :is (:compose hospital-ward-ward-capacity))
(tell (:about h-1
      (hospital-ward w1)
      (hospital-ward w2)
      (hospital-ward w3)))

(tell (ward-capacity w1 120)
      (ward-capacity w2 120)
      (ward-capacity w3 100))

(tell (:about h-2
      (hospital-ward w4)
      (hospital-ward w5)))

(tell (ward-capacity w4 110)
      (ward-capacity w5 90))
Retrieve Multiple Wards for H-1

\[
(retrieve (?w ?l) \\
  (:and (hospital-ward h-1 ?w) \\
    (ward-capacity ?w ?l)))
\]

\[==> (((W1 120) (W2 120) (W3 100)))\]
Retrieve Multiple Wards for H-1

(retrieve (?w ?l)
  (:and (hospital-ward h-1 ?w)
       (ward-capacity ?w ?l)))

==> ((|I|W1 120) (|I|W2 120) (|I|W3 100))

What about a short-hand notation?

(retrieve ?l
  (hospital-ward-capacity h-1 ?l))

==> (120 100)
Lessons from the Example

Modeling Advice:

- Determine Detail Level
- Use Specialized Operators
- Be Explicit in Queries