```
(define-syntax define-poly
  (syntax-rules ()
    [(_ (name tyvar) body)
     (define-syntax (name stx)
       (syntax-case stx ()
         [(_ type)
          (with-syntax ([tyvar #'type])
            #'body)])))])
```

(d-p (filter t)
  (letrec ([f (λ (p l)
                ⋮
)])
    f))

(d-p (id t) (λ ([x : t]) : t x))
⟶ (cl-s id ...)

(d id@str (id string)) ⟶ (λ ([x : string]) : string x)
(d id@n (id number)) ⟶ (λ ([x : number]) : number x)
(id @ str "x") → "x"

map-n-s (num -> str) (listof num) -> (listof str)

RANK-1 PARAMETRIC POLYMORPHISM

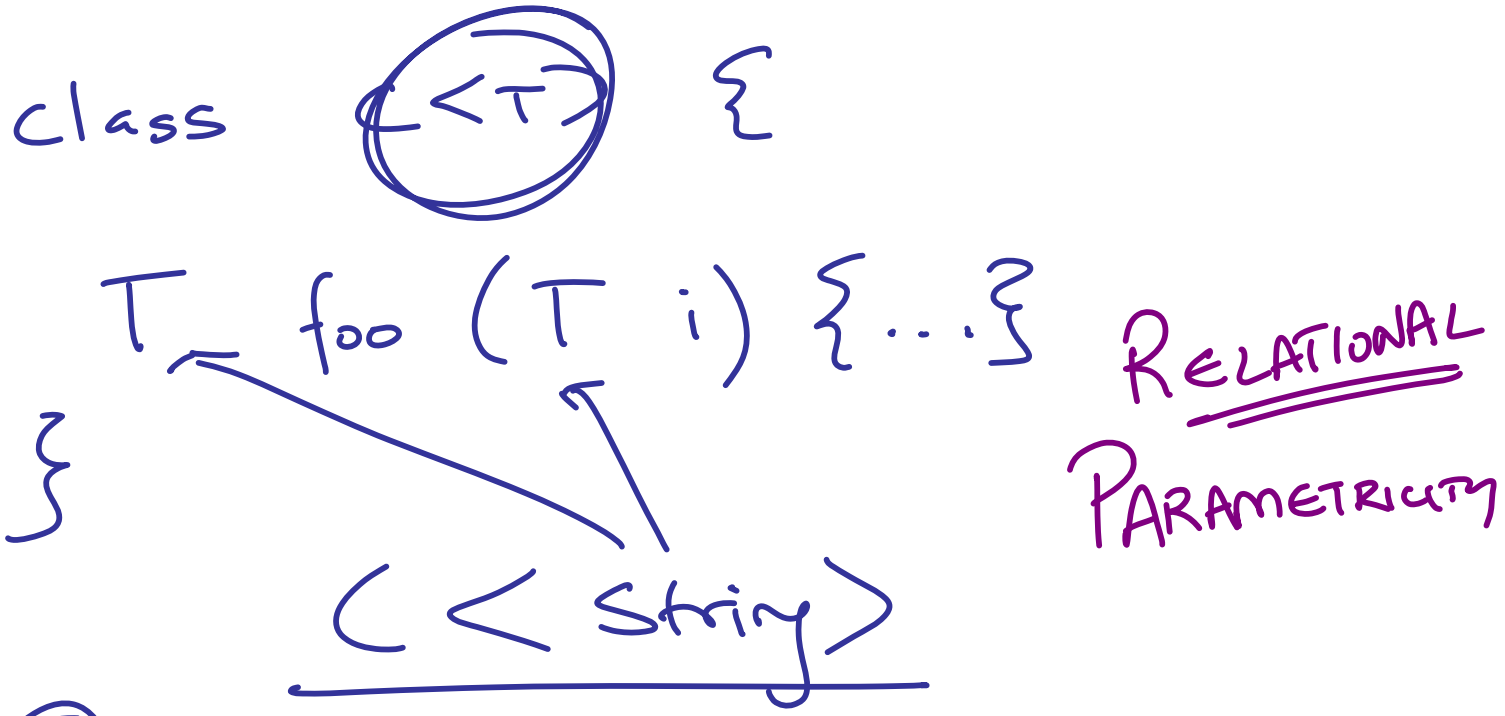
T = num | str | bool
  | (T -> T)
  | Tvar — 'a 'b 'c

} monotypes

P = ∀ Tvar ... . T

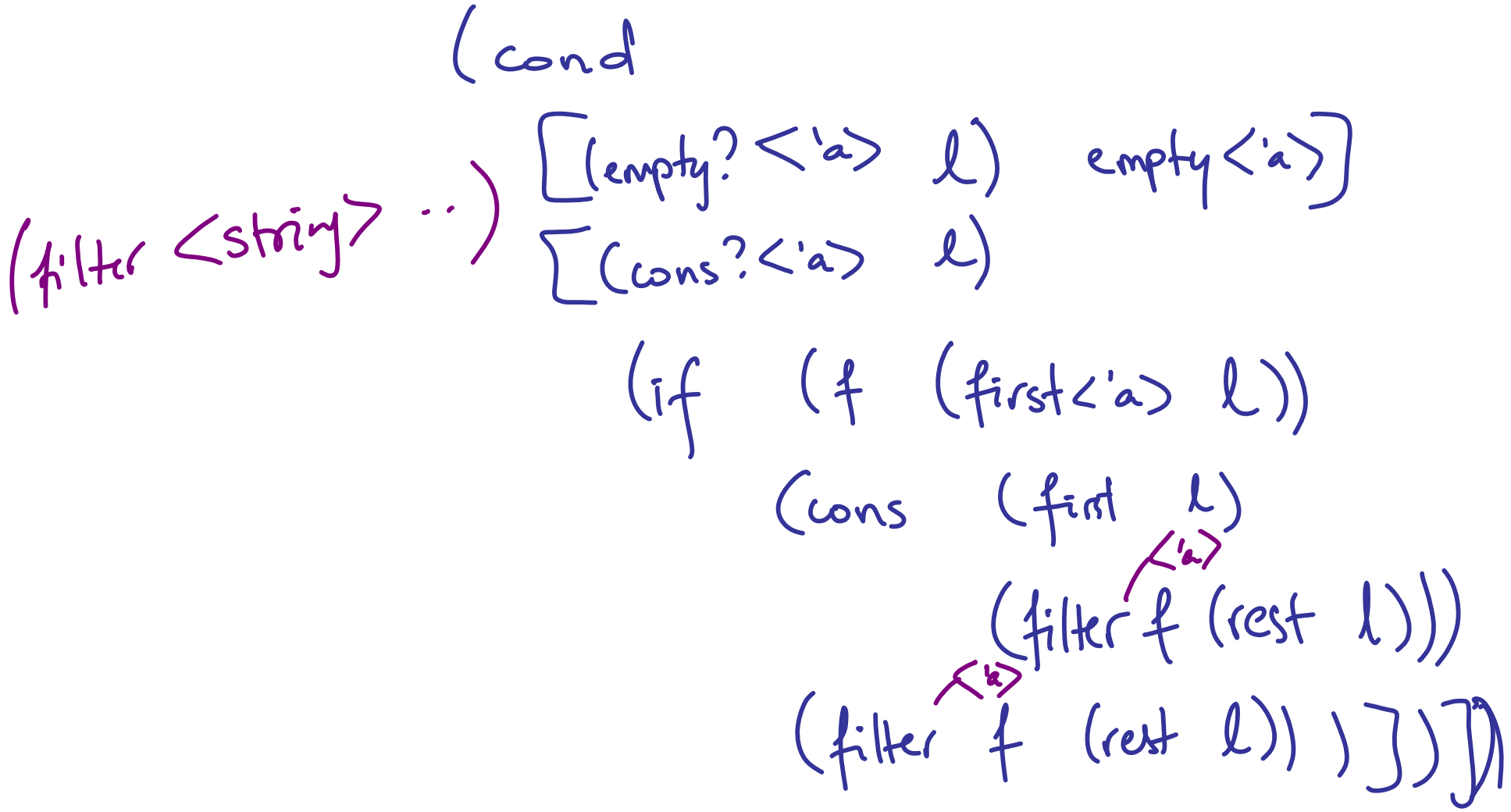
} polytypes

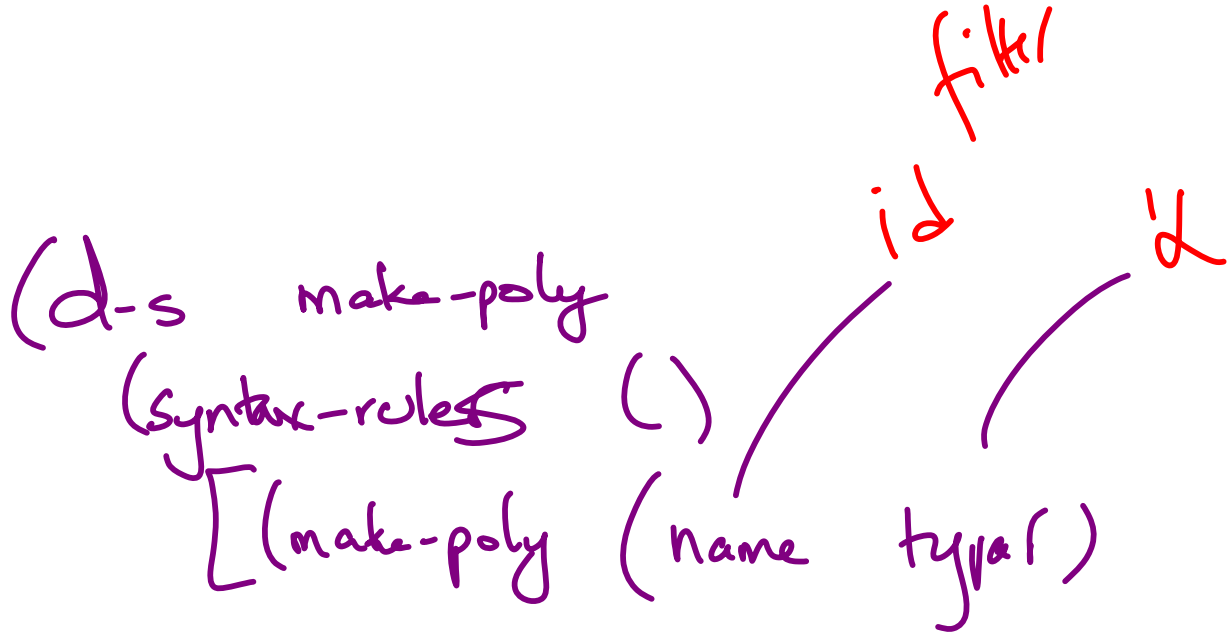
λ Tvar ... . T



∀'a ('a → boolean) (listof 'a) → (listof 'a)

```
(define-poly ('a)
  (filter [f : ('a → boolean)]
          [l : (listof 'a)]) : (listof 'a)
```


(filter <string> ..)
(cond
[(empty? <'a> l) empty<'a>]
[(cons? <'a> l)
(if (f (first <'a> l))
(cons (first l)
(filter f (rest l)))
(filter f (rest l)))])
<'a>
<'a>


(d-s make-poly
(syntax-rules ()
[(make-poly (name typar)
id
'α