

Copyright ©1999 Timothy Howard Merrett

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and full citation in a prominent place. Copyright for components of this work owned by others than T. H. Merrett must be honoured. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, or to redistribute to lists, requires prior specific permission and/or fee. Request permission to republish from: T. H. Merrett, School of Computer Science, McGill University, fax 514 398 3883.

The author gratefully acknowledges support from the taxpayers of Québec and of Canada who have paid his salary and research grants while this work was developed at McGill University, and from his students (who built the implementations and investigated the data structures and algorithms) and their funding agencies.

T. H. Merrett

©99/11

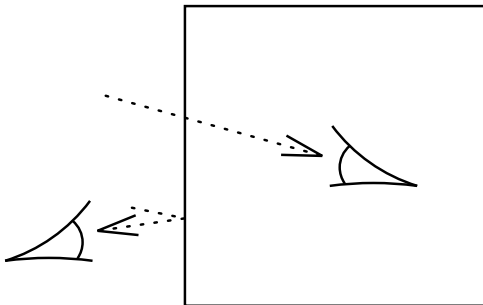
Scoping

Scoping in P.L.

```

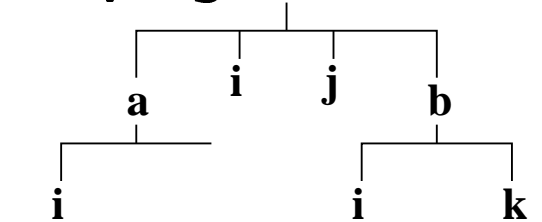
begin int  $i := 1, j := 2$ 
  begin int  $i := 3$ 
  end
  begin int  $i := 4, k := 5$ 
  end
end

```

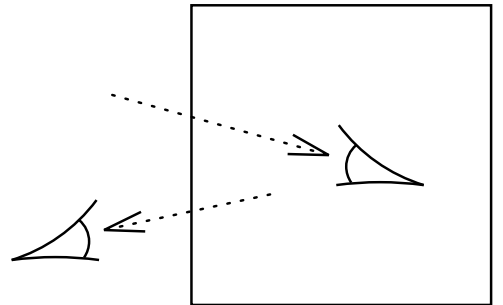


T. H. Merrett

Scoping in O.S.



$i, j, a/i, b/i, b/k$
 $i, .. / i, .. / j, .. / b / i, .. / b / k$
 $i, k, .. / i, .. / j, .. / a / i$



©99/11

DBPL is persistent, interactive:
follow O.S., not P.L.

```
begin relation  $i(w, x)$ ; relation  $j(y, z)$   
   $i, j, a.i, b.i, b.k$   
  begin  $a$  relation  $i(x, y)$   
     $i, \text{up } i, \text{up } j, \text{up } b.i, \text{up } b.k$   
  end  
  begin  $b$  relation  $i(w, z), \text{relation } k(x, y)$   
     $i, j, \text{up } i, \text{up } j, \text{up } a.i$   
  end  
end
```

Note resemblance to nested relations:
a database is a scope, is a tuple (but only one).

Note *navigational* syntax.

This gives homogenous **multidatabases**.

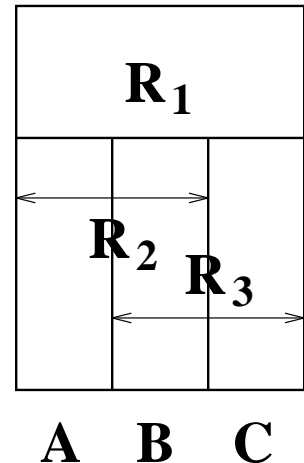
It can be extended to **distributed databases** using Randell's "UNIX United" (or "the Newcastle Connection")
e.g., as implemented in AFS

- Each file hierarchy is a tree;
- file hierarchy for many machines is a forest;
- common root unites the forests into a tree.
- (The tree could start with the I.P. address levels.)

Partitioning in Distributed Databases

Fragmentation

$$R = R_1 \text{ ujoin } (R_2 \text{ ijoin } R_3)$$

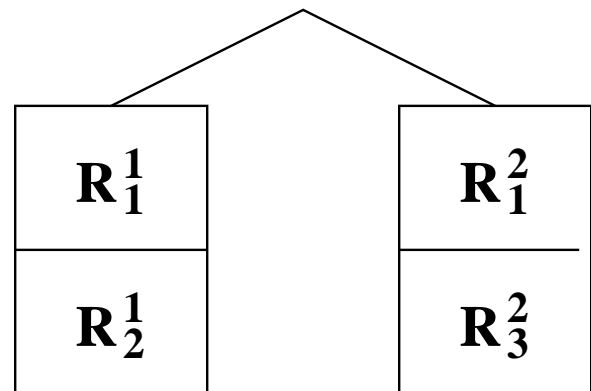


Distribution

R_1 at sites 1, 2

R_2 at site 1

R_3 at site 2



Working from site 1:

$$R_1^1, R_2^1, \text{ up } 2.R_1^2, \text{ up } 2.R_3^2$$

Working from site 2:

$$\text{up } 1.R_1^1, \text{ up } 1.R_2^1, R_1^1, R_3^2,$$

Now Query It

(the programmer controls locations)

[B] where $A = 2$ and $C = 1$ in R
= [B] where $A = 2$ and $C = 1$ in
 $(R_1 \text{ ujoin } (R_2 \text{ ijoin } R_3))$
= ([B] where $A = 2$ and $C = 1$ in R_1) ujoin
(([B] where $A = 2$ in R_2) ijoin
[B] where $C = 1$ in R_3)
= ([B] where $A = 2$ and $C = 1$ in R_1^1) ujoin
(([B] where $A = 2$ in R_2^1) ijoin
up 2.[B] where $C = 1$ in R_3^2)
= ([B] where $A = 2$ and $C = 1$ in R_1^2) ujoin
((up 1.[B] where $A = 2$ in R_2^1) ijoin
[B] where $C = 1$ in R_3^2)

Semijoins for efficiency

$$S^1 \text{ ijoin } T^2 = \text{up } 1.(S^1 \text{ ijoin up } 2.[Y] \text{ in } T^2) \text{ ijoin } T^2$$

T. H. Merrett

©99/11