Roadmap

1) Roots: System R and Ingres
2) Implementation: buffering, indexing, q-opt
3) Transactions: locking, recovery
4) Distributed DBMSs
5) Parallel DBMSs: Gamma, Alphasort
6) OODB DBMS
7) Data Analysis - data mining
8) Benchmarks
9) vision statements
   extras (streams/sensors, graphs, multimedia, web, fractals)

INGRES Prototype Overview

- QUEL/EQUEL
- Process structure
- Access methods
- Catalog structure – DBA
- Views/protection/integrity/recovery/CC
- Query optimizer – decomposition
- Utilities

INGRES Prototype

- On Unix PDP-11/40-45-70

QUEL – Tuple Calculus

```
RANGE of E is EMPLOYEE
RANGE of D is DEPT
RETREIVE E.SALARY, E.NAME [INTO…]
WHERE E.DNO=D.DNO
AND DEPT.DNAME="marketing"
```

‘E’ and ‘D’: tuple variables
Embedded QUEL: EQUEL

```plaintext
maint() {
    ## char EMPNAME[20];
    while (READ(EMPNAME)) {
        ## RANGE OF x IS EMP
        ## RETRIEVE …
        ## WHERE x.NAME=EMPNAME
        ## {
            /* block executed */
            PRINT( … )/* for every tuple */
        ## }
    }
}
```

INGRES Prototype Overview

- QUEL/EQUEL
- Process structure
- Access methods
- Catalog structure – DBA
- Views/protection/integrity/recovery/CC
- Query optimizer – decomposition
- Utilities

Process Structure

- Unix – tree structured like file system
  - LRU for page replacement
- INGRES – a user job
- Process – fork – pipes
  - Memory limitations (64K)

File Structure

- How would you organize the db files?

Process Structure (cont.)

- Unix – tree structured like file system
  - LRU for page replacement
- INGRES – a user job
- Process – fork – pipes
  - Memory limitations (64K)
- Unix ± tree structured like file system
  - LRU for page replacement
- INGRES – a user job
- Process – fork – pipes
  - Memory limitations (64K)
File Structure

INGRES

AUX DATADIR BIN SRC TMP
dbname1 dbname-n
user rels DBA rels catalog admin

Everything is a relation (except for admin)

INGRES Prototype Overview

- QUEL/EQUEL
- Process structure
- Access methods
- Catalog structure – DBA
- Views/protection/integrity/recovery/CC
- Query optimizer – decomposition
- Utilities

System Catalog

- Which tables would you have, to store the relation and attribute information?
- Which other tables would you need?

Storage structures

- How would you sort your db files (if at all?)

System Catalogue

RELATION (relid, owner, spec, indexed, …)
ATTRIBUTE (relid, owner, domain-name, domain-no, offset, offset, type, length, keyno)
INDEX (…)
PROTECTION (…)
INTEGRITY (…)
VIEW (…)

Storage Structures

- Keyed
  - Hashed, ISAM, C-hashed, C-ISAM
- Non-keyed
  - Random heap
- Page layout like system R
- Compression
  - Suppresses blanks
  - XOR with preceding tuple
Access Methods Interface

OPENR (descriptor, mode, relation_name)

GET (descriptor, tid, limit_tid, tuple, next_flag)
FIND (descriptor, key, tid, key_type)
INSERT (descriptor, tuple)
REPLACE (descriptor, tuple, new_tuple)
DELETE (descriptor, tutid)

CLOSER (descriptor)

Process 2

· Query modification
  – Integrity
  – Security
  – Views
· Concurrency control
  – How would you do it?

INGRES Prototype Overview

· QUEL/EQUEL
· Process structure
· Access methods
· Catalog structure – DBA
  Views/protection/integrity/recovery/CC (process#2)
· Query optimizer – decomposition
· Utilities

Process 2

· How to implement integrity and security?

Process 2

· Concurrency control (crude)
  – Xact = 1 INGRES command (later >1)
  – LOCK domain of relation
  – Deadlock avoidance (get all locks, then proceed)
  – Locks released at the end
  – Recovery nonexistent
INGRES Prototype Overview

- QUEL/EQUEL
- Process structure
- Access methods
- Catalog structure – DBA
- Views/protection/integrity/recovery/CC

Query optimizer – decomposition (process#3)

- Utilities

---

Process 3

- DECOMP: 1 query = n single-variable queries
  - Tuple substitution
  - One-variable detachment
    \[Q_i(V_i) \text{ AND } Q_d(V_1, V_2, \ldots, V_d)\]
    evaluate \(Q_i(V_i)\) in temp

---

Process 4

- Utilities (Process 4)
  - Deferred updates (why?)
  - A1: eg., REPLACE E(salary = 1.1*E.salary)
  - A2: provides a rudimentary log for recovery

---

INGRES Prototype Overview

- QUEL/EQUEL
- Process structure
- Access methods
- Catalog structure – DBA
- Views/protection/integrity/recovery/CC

Query optimizer – decomposition

- Utilities

---

Process 4

- Utilities (Process 4)
  - Deferred updates (why?)
Conclusions

- No performance measurements
- Wish list (from users)
  - <can you guess?>

Conclusions

- No performance measurements
- Wish list (from users)
  - Speed
  - Recursion
  - User-defined functions
  - Report generator
  - Bulk copy

Conclusions

- Future goals
  - distributed INGRES
  - DBA advisor

Evaluation (Mistakes)

- Interpreted code
- 64 KB is too little memory
- UNIX slows system down
- ISAM fixed directory – B-trees are better
- Query optimizer
  - DECOMP: merge-sort join not easy to fit