Logical and physical data structures for very small databases (VSDB)

Fabio A. Schreiber  Letizia Tanca
Cristiana Bolchini  Fabio Salice

Politecnico di Milano

DATABASES AND SYSTEMS

VSDB

LARGE

SYSTEM

SMALL

DB  VLDB

?
PRESENTATION OUTLINE

• MOTIVATIONS
• VSDB DESIGN METHODOLOGY
• LOGICAL AND PHYSICAL DATA STRUCTURES
• ACCESS MODES
• COMPLEXITY AND POWER CONSUMPTION ANALYSIS
• WORK IN PROGRESS ...

MOTIVATIONS

• FROM PC DOWNWARDS
  ➢ PALM AND HANDHELD PC
  ➢ SMART CARDS
  ➢ CELLULAR PHONES (SIM CARDS)
• PROFESSIONAL EMBEDDED SYSTEMS
  ➢ DEVICE CONTROL AND TELEMETRY
  ➢ SYSTEMS AND INDUSTRIAL PROCESS CONTROL
  ➢ DISTRIBUTED NETWORK MANAGEMENT
Physic Storage Medium

Flash EEPROM

- Technology Issues
  - Constraints on Modify (Erase/Write) Operations
    - Write = Programming
    - Write only on Virgin or Erased Locations
    - Erase only at Block Level
  - Timing Issues
  - Block Allocation/Management
  - Endurance

- Perspective From
  - Performance Viewpoint
  - Power Consumption Viewpoint

<table>
<thead>
<tr>
<th></th>
<th>EEPROM</th>
<th>Flash-EEPROM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read (nsec)</td>
<td>150</td>
<td>80 @ 5V, 150 @ 3V</td>
</tr>
<tr>
<td>Program (µsec/byte)</td>
<td>157 (64 B), 625 (16 B)</td>
<td>10 @ 5V, 17 @ 3V</td>
</tr>
<tr>
<td>Erase (sec/block)</td>
<td>N.A.</td>
<td>0.45 @ 5V, 0.5 @ 3V</td>
</tr>
<tr>
<td>Cell Size (µm² @ 0.4µm technology)</td>
<td>4.2</td>
<td>2</td>
</tr>
<tr>
<td>Cost per bit</td>
<td>Medium</td>
<td>Low</td>
</tr>
<tr>
<td>Endurance (program, erase operations)</td>
<td>10K to 100K write cycle /byte</td>
<td>10K to 100K erase cycle /block</td>
</tr>
</tbody>
</table>
PHYSICAL STORAGE MEDIUM: THE SMART CARD EXAMPLE

FLASH MEMORY ORGANIZATIONS

Symmetrically blocked component

Asymmetrically blocked component

Logical/Physical Data Structures in VSDB
VSDB DESIGN METHODOLOGY

1. IDENTIFY INFORMATION AREAS AND DEFINE VIEWS
2. INTEGRATE VIEWS INTO THE CONCEPTUAL SCHEMA AND TRANSLATE THEM INTO THE LOGICAL DATA MODEL
3. DEFINE LOGICAL VIEWS AND ALLOCATE THEM INTO INFORMATION AREAS
4. ALLOCATE EACH LOGICAL VIEW "ON CARD" OR TO AN OUTER SYSTEM
5. DEFINE PRIVACY PROFILE AND REFINE VIEWS
6. DESGIN MEMORY PROTECTION MECHANISM
7. CHOOSE ENCRYPTION ALGORITHMS
8. ESTIMATE LOGICAL VIEW SIZE
9. FOR EACH FRAGMENT ALLOCATED "ON CARD"
10. ESTIMATE ACCESS TYPE AND DATA VOLATILITY
11. SELECT THE LOGICAL DATA STRUCTURE
12. SELECT THE PHYSICAL DATA STRUCTURE
13. DEFINE LOGICAL VIEWS AND ALLOCATE THEM INTO INFORMATION AREAS
14. ESTIMATE LOGICAL VIEW SIZE

CASE STUDY:
PORTABLE INTERNET ACCESS DATABASE

- web_login
  - id
  - login
  - password

- web_access
  - id
  - web_site
  - description
  - url
  - mvmnt
    - id
    - description
    - amount
  - date
  - time
  - category
  - visited_on
  - last_visited

- history
  - web_access

- bookmark
  - web_access

- web_purse
  - id
  - balance
Logical/Physical Data Structures in VSDB

- **Heap Relation**
  - Limited cardinality (<= 10 records)

- **Sorted Relation**
  - Medium cardinality (~100 - ~1000 records)
  - Sorted w.r.t. a field

- **Circular List Relation**
  - Medium cardinality
  - Stored and managed as a circular list
  - Typically sorted by date/time

- **Generic Relation**
  - Multi-index structure

RELATIONAL DB MODEL
Logical/Physical Data Structures in VSDB

DB ANNOTATION

PHYSICAL DESIGN: DATA ORGANIZATION

DATA STRUCTURES ARE ASSIGNED ON THE BASIS OF THE ANNOTATIONS ON THE RELATIONS

SORTED

HEAP

CIRCULAR LIST

HEAP

CIRCULAR LIST

Logical/Physical Data Structures in VSDB

7
ACCESS MODES: QUERY

- **SCAN**: FETCH ALL RECORDS IN THE TABLE
- **SEARCH WITH EQUALITY SELECTION**
  FETCH ALL RECORDS THAT SATISFY AN EQUALITY SELECTION
  "find the VisitedURL for day = #05/01/2001#"
- **SEARCH WITH RANGE SELECTION**
  FETCH ALL RECORDS THAT SATISFY A RANGE SELECTION
  "find the movements with amount between €10 and €100"

ACCESS MODES: UPDATE

- **INSERT**: REQUIRES RECORD SHIFTING WHEN SORTED
  ✓ FETCH BLOCK
  ✓ INCLUDE THE NEW RECORD
  ✓ WRITE THE BLOCK BACK
- **DELETE**: REQUIRES RECORD SHIFTING WHEN SORTED
  ✓ FETCH BLOCK
  ✓ DELETE RECORD
  ✓ WRITE THE BLOCK BACK
- **MODIFY**:  
  ✓ FETCH BLOCK
  ✓ MODIFY RECORD
  ✓ WRITE THE BLOCK BACK
PHYSICAL DESIGN: BLOCK ALLOCATION

GOAL:
• REDUCE THE NUMBER OF ERASE/WRITE OPERATIONS
  ‣ ACHIEVE GOOD PERFORMANCE
  ‣ ACHIEVE LOW POWER CONSUMPTION
  ‣ PRESERVE THE LIFE OF THE FLASH DEVICE

SOLUTIONS:
• ADD A validity_bit PER RECORD
• ADD A deleted_bit PER RECORD
• INSERT DUMMY RECORDS IN BLOCKS
• PERFORM GARBAGE COLLECTION DURING ERASE/PROGRAM CYCLES
PHYSICAL DESIGN: BLOCK ALLOCATION

SORTED DATA STRUCTURE

- HIGH PROBABILITY OF **CASCADED BLOCK ERASURE / REWRITE**
- LEAVE SOME **EMPTY (DUMMY) RECORDS** IN EACH BLOCK
  - ADJACENT (EFFICIENT SEARCH)
  - DISTRIBUTED (EFFICIENT UPDATE)

---

PHYSICAL DESIGN: BLOCK ALLOCATION

SORTED DATA STRUCTURE

- CONCENTRATED ➔ LOWER SEARCH TIME
- DISTRIBUTED ➔ FEWER INSERTION CONFLICTS
COMPLEXITY AND POWER CONSUMPTION ANALYSIS
DIFFERENT MEMORY ALLOCATION SOLUTIONS

EXPERIMENTAL RESULTS

<table>
<thead>
<tr>
<th>Data structure</th>
<th>Strategy</th>
<th>Block Erasures</th>
<th>Transmitted Bits on Bus</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>10%-30%</td>
<td>40%-60%</td>
<td>70%-90%</td>
</tr>
<tr>
<td></td>
<td>10%-30%</td>
<td>40%-60%</td>
<td>70%-90%</td>
</tr>
<tr>
<td>Heap Simple</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Delete bit</td>
<td>0</td>
<td>0.38</td>
<td>0.54</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Sorted Simple</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Delete bit</td>
<td>0.83</td>
<td>0.68</td>
<td>0.74</td>
</tr>
<tr>
<td>Dummy conc.</td>
<td>0.83</td>
<td>0.51</td>
<td>0.74</td>
</tr>
<tr>
<td>Dummy dist.</td>
<td>0.10</td>
<td>0.12</td>
<td>0.03</td>
</tr>
<tr>
<td>Circular List</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Delete bit</td>
<td>0</td>
<td>0</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>0.15</td>
</tr>
</tbody>
</table>

occurred block erasures and transmitted bytes on the system bus w.r.t. “the naive”, no deleted bit, no dummy records solution
WORK IN PROGRESS ...

- ACCURATE LOW POWER ANALYSIS
- USE OF "ADAPTIVE" DATA STRUCTURES
- DETAILED DEFINITION OF THE METHODOLOGY
- DESIGN OF TOOLS SUPPORTING THE AUTOMATIC AND THE SEMIAUTOMATIC STEPS OF THE METHODOLOGY

BIBLIOGRAPHY

- C. Bobineau, L. Bouganim, P. Pucheral, P. Valduriez, "PicoDBMS: Scaling down Database Techniques for Smart card," Proc. 26th Int. Conf. on Very Large Databases (VLDB), 2000, pp.11-20.