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

![Diagram showing the relationship between database and system sizes. VSDB is placed outside the standard database and system categories.]

Logical/Physical Data Structures in VSDB
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
PHYSICAL 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>Physical Storage Medium</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>90K to 100K write cycle /byte</td>
<td>10K to 100K erase cycle /block</td>
</tr>
</tbody>
</table>
**PHYSICAL STORAGE MEDIUM: THE SMART CARD EXAMPLE**

```
<table>
<thead>
<tr>
<th>ROM</th>
<th>RAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flash-EEPROM</td>
<td></td>
</tr>
<tr>
<td>Time Base &amp; Watchdog</td>
<td>CPU</td>
</tr>
<tr>
<td>Random Number Generator</td>
<td>Mod Arithmetic Processor</td>
</tr>
<tr>
<td></td>
<td>Control logic</td>
</tr>
<tr>
<td></td>
<td>Interrupt Controller</td>
</tr>
<tr>
<td>Ck</td>
<td>Rst Gnd Vcc</td>
</tr>
</tbody>
</table>
```

**FLASH MEMORY ORGANIZATIONS**

- Symmetrically blocked component
  - 64 KB Main Block
  - 64 KB Main Block
  - 64 KB Main Block

- Asymmetrically blocked component
  - 16 KB Boot Block
  - 6 KB parameter Block
  - 9 KB parameter Block
  - 96 KB Main Block
  - 128 KB Main Block
  - 128 KB Main Block

VSDB DESIGN METHODOLOGY

IDENTIFY INFORMATION AREAS AND DEFINE VIEWS

INTEGRATE VIEWS INTO THE CONCEPTUAL SCHEMA AND TRANSLATE THEM INTO THE LOGICAL DATA MODEL

DEFINE LOGICAL VIEWS AND ALLOCATE THEM INTO INFORMATION AREAS

ESTIMATE LOGICAL VIEW SIZE

ALLOCATE EACH LOGICAL VIEW "ON CARD" OR TO AN OUTER SYSTEM FOR EACH FRAGMENT ALLOCATED "ON CARD"

DEFINE PRIVACY PROFILE AND REFINE VIEWS

DESIGN MEMORY PROTECTION MECHANISM

CHOOSE ENCRYPTION ALGORITHMS

ESTIMATE ACCESS TYPE AND DATA VOLATILITY

SELECT THE LOGICAL DATA STRUCTURE

SELECT THE PHYSICAL DATA STRUCTURE

CASE STUDY:
PORTABLE INTERNET ACCESS DATABASE

web_login

has

web_access

history

web_purse

url

description

to

category

date

time

id

login

password

(amount)
description

balance

visited_on

last_visited

mvmnt

1:n

1:1

1:n
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
Logical/Physical Data Structures in VSDB

DB ANNOTATION

Table Length: 100B
Cardinality: 10
Limited: NO
PK: ID
Ordered: NO
Access Type Frequency:
- INSERT: LOW
- UPDATE: LOW
- SELECT: LOW

Table Length: 200B
Cardinality: 10
Limited: NO
PK: ID
Ordered: NO
Access Type Frequency:
- INSERT: HIGH
- UPDATE: HIGH
- SELECT: HIGH

PHYSICAL DESIGN: DATA ORGANIZATION

Data structures are assigned on the basis of the annotations on the relations.

Sorted
- Sorted Access
  - PK: ID
  - Description: high_id

Heap
- Heap Access
  - PK: ID
  - Description: high_id

Circular List
- Circular List
  - PK: ID
  - Description: high_id

Logical/Physical Data Structures in VSDB 12

Logical/Physical Data Structures in VSDB 13
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

Logical/Physical Data Structures in VSDB
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)

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></td>
<td>10%-30% 40%-60% 70%-90%</td>
<td>10%-30% 40%-60% 70%-90%</td>
</tr>
<tr>
<td>Heap</td>
<td>Simple</td>
<td>1 1 1</td>
<td>1 1 1</td>
</tr>
<tr>
<td></td>
<td>Delete bit</td>
<td>0 0.38 0.98</td>
<td>0.38 0.54 1.00</td>
</tr>
<tr>
<td></td>
<td>Simple</td>
<td>1 1 1</td>
<td>1 1 1</td>
</tr>
<tr>
<td></td>
<td>Delete bit</td>
<td>0.83 0.68 0.79</td>
<td>0.74 0.71 0.77</td>
</tr>
<tr>
<td></td>
<td>Dummy conc.</td>
<td>0.83 0.51 0.44</td>
<td>0.74 0.57 0.45</td>
</tr>
<tr>
<td></td>
<td>Dummy dist.</td>
<td>0.10 0.12 0.24</td>
<td>0.03 0.06 0.22</td>
</tr>
<tr>
<td>Sorted</td>
<td>Simple</td>
<td>1 1 1</td>
<td>1 1 1</td>
</tr>
<tr>
<td></td>
<td>Delete bit</td>
<td>0 0 0.05</td>
<td>0.07 0.07 0.15</td>
</tr>
<tr>
<td>Circular List</td>
<td>Simple</td>
<td>1 1 1</td>
<td>1 1 1</td>
</tr>
<tr>
<td></td>
<td>Delete bit</td>
<td>0 0 0.05</td>
<td>0.07 0.07 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.