SIGMOD 2015
Programming Contest

Team CUT_HERE________________
KAIST and Seoul National University
Team Members

• Wonha Ryu: BS student at KAIST
• Seoung Gook Sohn: PhD student at SNU
• Eunjin Song: MS student at SNU
Our Approach

Dataset Feature + Efficient Implementation
Feature 1: Validation Result

The majority of validation results are zero
Feature 2: Operator Type

Equal operators dominate the types of operators
Feature 3: Column Range

<table>
<thead>
<tr>
<th>C₁</th>
<th>C₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>3074</td>
<td>8896128387</td>
</tr>
<tr>
<td>3077</td>
<td>6811549715</td>
</tr>
<tr>
<td>3075</td>
<td>7782881939</td>
</tr>
<tr>
<td>3077</td>
<td>5819204839</td>
</tr>
</tbody>
</table>

Query: $C₁ \equiv 3076$, $C₂ \equiv 5739273029$

Chance of conflict: High

Each column has a fixed-range of values
Feature 4: Transaction Size

... 

$T_n$ Insert rows: 4  
Delete rows: 4  
$T_{n+1}$ Insert rows: 1  
Delete rows: 2  
$T_{n+2}$ Insert rows: 1  
Delete rows: 1  
$T_{n+3}$ Insert rows: 2380  
Delete rows: 1643  
$T_{n+4}$ Insert rows: 1  
Delete rows: 3  
...

Small-sized transactions are substantial
Large-sized transactions are exceptional
Design Overview
Transaction Format

Delete

Rows with PK 2 and 8 in relation A

Insert

Tuple (2, 3, 4) into relation A

Tuple (4, 5, 6, 7) into relation B

\[\text{Transaction}_1 = [\text{A} [2], \text{A} [8], \text{A} [2, 3, 4], \text{B} [4, 5, 6, 7]]\]

Relation A = 
[[2, 8, (2, 3, 4)], T_2, T_3, …]
Storing Transactions

- Represent a relation as a list of transactions
- Save each transaction in column-store
- Group small-sized transactions into one

Relation \(_1 = [ T_1, T_2, T_{3-8}, \ldots ] \)

Small Txs
## Simple Bloom Filter

| Row \| Value 1 | Value 2 | Value 3 |
|---|---|---|---|
| Row 1 | 40 | 20 | 10 |
| Row 2 | 40 | 22 | 10 |

- Construct bloom filters for each row
- \([40, 20, 10] \Rightarrow \{(1, 40), (2, 20), (3, 10)\}\)
- Only a single hash function is used for performance issues
Validation Requests

• Validation requests are given in DNF

  e.g. (Relation 1: \(Q_1 \land Q_2 \land Q_3\)) \lor (Relation 2: \(Q_4 \land Q_5\))

• Independently process each conjunction

• Optimizations:
  • Check if queries are within \([\text{min}, \text{max}]\) range
  • Reorder queries
  • Utilize bloom filters for equal queries
Reordering Queries

Original: \((c_1 < v_1) \land (c_2 == v_2) \land (c_3 == v_3) \land (c_4 >= v_4)\)

Reordered: \((c_2 == v_2) \land (c_3 == v_3) \land (c_1 < v_1) \land (c_4 >= v_4)\)

Reorder queries in the following order:

1. Query with = operator having the widest range of values
2. Queries with = operators
3. Queries with other operators
Bloom Filter

Validation: \( C_1 = 24 \land C_7 = 56 \land C_{12} = 2 \)

\[ h(1, 24) \]
\[ h(7, 56) \]
\[ h(12, 2) \]

BF(Row_1): 0 0 0 0 0 1 0 0 1 0 1 0 ...
Bloom Filter

Validation: \( C_1 = 24 \land C_7 = 56 \land C_{12} = 2 \)

- Using AVX SIMD bitwise AND (\texttt{VANDPS})
Test Scores

column store

threading

bloom filter

input buffer

... optimizations ...
Thank you