DBToaster
Higher-Order Delta Processing for Dynamic, Frequently Fresh Views

Yanif Ahmad  Johns Hopkins
Oliver Kennedy  University at Buffalo
Christoph Koch  EPFL
Milos Nikolic  EPFL
Realtime Monitoring Programs...

...Monitor The State of the World

...React to Conditions in that State
Realtime Monitoring Programs are Everywhere
Monitoring Programs
Monitoring Programs

Problem: People write monitoring programs by hand
Monitoring Programs

**Problem:** People write monitoring programs by hand
Monitoring Programs
Monitoring Programs

- An Aggregate Representation of the State of the World
- Maintained in Realtime as the State of the World Changes
- Needs to React to Changes In the World Quickly

Not just Views

Frequently Fresh Views
Monitoring Programs
Monitoring Programs

(The Current State of the Art)
Monitoring Programs

Spec

Javac
Monitoring Programs

The DBToaster Compiler

9

Thursday, December 6, 12
The Viewlet Transform
The Viewlet Transform

Use Auxiliary Views to Speed Up View Maintenance
The Viewlet Transform

Use Auxiliary Views to Speed Up View Maintenance

The Delta of a Query Can Be Materialized!
The Viewlet Transform

SELECT SUM(R.A * S.C) FROM R, S WHERE R.B = S.B

A Simple 2-Way Join Aggregate
The Viewlet Transform

$q[] := \text{SELECT SUM(R.A} \ast \text{S.C)}$

\text{FROM} \quad \text{R, S}

\text{WHERE} \quad \text{R.B} = \text{S.B}

A Simple 2-Way Join Aggregate
The Viewlet Transform

\[ q[] \; += \; SELECT \; \text{SUM}(\partial A \times S.C) \]
\[ \text{FROM} \; \; S \]
\[ \text{WHERE} \; \; \partial B = S.B \]

Materialize and Incrementally Maintain The Query
The Viewlet Transform

\[
\begin{align*}
\text{ON } & +R(\partial A, \partial B): \\
q[] & += \text{SELECT SUM}(\partial A \times S.C) \\
& \text{FROM } S \\
& \text{WHERE } \partial B = S.B
\end{align*}
\]

The Delta of the Original Query

Materialize and Incrementally Maintain The Query

Thursday, December 6, 12
The Viewlet Transform

ON \(+R(\partial A, \partial B)\):

\[ q[] \ += \ \partial A \times \left( \begin{array}{c} \text{SELECT} \\ \text{SUM}(S.C) \\ \text{FROM} \\ S \\ \text{WHERE} \\ \partial B = S.B \end{array} \right) \]
The Viewlet Transform

ON \( +R(\partial A, \partial B) : \)

\[ q[] += \partial A \times \left( \begin{array}{c}
\text{SELECT SUM(S.C)} \\
\text{FROM S} \\
\text{WHERE } \partial B = S.B
\end{array} \right) \]
The Viewlet Transform

\[ \text{ON } +R(\partial A, \partial B): \]
\[ q[] + = \partial A \times \left( \begin{array}{c}
\text{SELECT } S.B, \text{SUM}(S.C) \\
\text{FROM } S \\
\text{GROUP BY } S.B
\end{array} \right)[\partial B] \]
The Viewlet Transform

\[
\text{ON } +R(\partial A, \partial B): \quad q[] = \partial A \ast \left( \left( \begin{array}{c}
\text{SELECT } S.B, \text{SUM}(S.C) \\
\text{FROM } S
\end{array} \right) \right)_{\partial B}^{[\partial B]}
\]

Just an Ordinary Query

Optimize
The Viewlet Transform

\[
\text{ON } +R(\partial A, \partial B): \\
q[ ] += \partial A * mR[\partial B]
\]

\[
mR[B] := \text{SELECT } S.B, \text{SUM}(S.C) \\
\text{FROM } S \\
\text{GROUP BY } S.B
\]

Extract and Materialize The Delta View
The Viewlet Transform

ON +R(∂A, ∂B):

q[] += ∂A * mR[∂B]

A Hash Map (indexed by S.B)

mR[B] := SELECT S.B, SUM(S.C)
FROM S
GROUP BY S.B

Extract and Materialize The Delta View
The Viewlet Transform

ON \ +R(\partial A, \partial B) :
\  q[] \ += \partial A \ast mR[\partial B]

ON \ +S(\partial B, \partial C) :
\  mR[B] \ += \text{SELECT} \ \partial B, \text{SUM}(\partial C)

Incrementally Maintain The Delta View
The Viewlet Transform

\[ \partial A \ast \partial C + = \partial B + R(\partial A, \partial B) : \]

\[ q[ ] += \partial A \ast mR[\partial B] \]

\[ \partial B \ast \partial C : \]

\[ mR[\partial B] += \partial C \]
The Viewlet Transform

ON \( +R(\partial A, \partial B) \):
\[
q[] \, +\, \partial A \ast mR[\partial B] \\
\text{ms}[\partial B] \, +\, \partial A
\]

ON \( +S(\partial B, \partial C) \):
\[
\text{mR}[\partial B] \, +\, \partial C \\
q[] \, +\, \partial C \ast \text{ms}[\partial B]
\]

Repeat for the Other Deltas of the Query
The Viewlet Transform
The Viewlet Transform

- Take the Deltas
  - Optimize and Materialize Them
- Take the Deltas
  - Optimize and Materialize Them
- ...

Thursday, December 6, 12
The Viewlet Transform

- Take the Deltas
  - Optimize and Materialize Them
- Take the Deltas
  - Optimize and Materialize Them
- ...

Thursday, December 6, 12
Performance
• TPC-H Workload
• Simulated Realtime Data Warehouse
• Update Stream Derived from TPC-H Gen

• Financial Benchmark
• 24 hr Trace for an Actively Traded Stock.
TPCH: Q3

Refresh Rate

(3-Way Join)
DBToaster vs Commercial Engines

Refresh Rate (tuples/sec)

TPC-H Schema

Financial Benchmark
DBToaster vs Commercial Engines

DbToaster is consistently 3 OOM better!
Limitations of Commercial Systems

- OLTP IVM is not designed for aggregating Low-Latency/Single-Tuple Updates.
- OLTP IVM doesn’t support our full query workload.
- Stream Processors are not designed for rapidly changing long-lived data.
Limitations of Commercial Systems

• OLTP IVM is not designed for aggregating Low-Latency/Single-Tuple Updates.

• OLTP IVM doesn’t support our full query workload.

• Stream Processors are not designed for rapidly changing long-lived data.

DBToaster opens entirely new application domains!
Conclusions

• The Viewlet Transform generates auxiliary views that make incremental maintenance fast.

• Materializing only part of an auxiliary view can sometimes be faster.

• DBToaster is commonly 3 OoM faster than Commercial Systems.

Download Now: http://www.dbtoaster.org
Cumulus: Hybrid Consistency Aggregation Queries (in the cloud)

Oliver Kennedy

(joint work with JHU’s Yanif Ahmad and Yotam Barnoy)
SELECT...
SELECT...
SELECT SUM(...) FROM A,B,... WHERE ...
SELECT SUM(...) FROM A,B,... WHERE ...
SELECT SUM(...) FROM A,B,... WHERE ...
“Magic Maps”

Map Maintenance Messages

System Design

Hybrid Consistency
$\text{Map}_1[V_1, V_2, ...]$
\[ \text{Map}_1 [V_1, V_2, ...] += f(\text{Map}_2, \text{Map}_3, ..., V_1, V_2, ...) \]
ON Event($V_1, V_2, ...$) DO {
    $Map_1[V_1, V_2, ...] += f(Map_2, Map_3, ..., V_1, V_2, ...)$
}
ON Event(V₁, V₂, ...) DO {
Map₁[V₁, V₂, ...] += f(Map₂, Map₃, ..., V₁, V₂, ...)
Map₂[V₁, V₂, ...] += f(Map₁, Map₃, ..., V₁, V₂, ...)
}
SELECT SUM(R.A * T.D)
FROM   R, S, T
WHERE  R.B = S.B
       AND S.C = T.C
ON Insert_R(V_A,V_B) DO {

SELECT SUM(R.A * T.D)
FROM   R, S, T
WHERE  R.B = S.B
       AND S.C = T.C

}
ON Insert_R(V_A,V_B) DO {

    Result[] +=

    SELECT SUM(R.A * T.D)
    FROM R, S, T
    WHERE R.B = S.B
        AND S.C = T.C

}
ON Insert_R(V_A,V_B) DO {

    Result[] +=

    SELECT V_A * SUM(T.D)
    FROM S, T
    WHERE V_B = S.B
    AND S.C = T.C

}
ON Insert_R(V_A, V_B) DO {

    Result[] += V_A * Map[V_B]

}
ON Insert_R(A,B) DO {
    Result[] += A * Map_2[B]
    Map_1[C] += A * Map_3[B,C]
    Map_5[B] += A }

ON Insert_T(C,D) DO {
    Result[] += Map_1[C] * D
    Map_2[B] += D * Map_3[B,C]
    Map_4[C] += D }

ON Insert_S(B,C) DO {
    Result[] += Map_5[B] * Map_4[C]
    Map_1[C] += Map_5[B]
    Map_2[B] += Map_4[C]
    Map_3[B,C] += 1 }
“Magic Maps”

Map Maintenance Messages

System Design

Hybrid Consistency
\texttt{Map}_1[ ] += \texttt{Map}_2[ ] * \texttt{Map}_3[ ]
\[
\text{Map}_1[] += \text{Map}_2[] \times \text{Map}_3[]
\]

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\[ \text{Map}_1[] \ += \ \text{Map}_2[] \ * \ \text{Map}_3[] \]
\[ \text{Map}_1[ ] += \text{Map}_2[ ] \times \text{Map}_3[ ] \]
\[ \text{Map}_1[] \ += \text{Map}_2[] \ast \text{Map}_3[] \]

<table>
<thead>
<tr>
<th></th>
<th>\text{Map}_1</th>
<th>\text{Map}_2 \ast \text{Map}_3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map_1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map_2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map_3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```latex
\text{Map}_1[] += \text{Map}_2[] \times \text{Map}_3[]
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{Map}_1</td>
<td>\text{Map}_2 \times \text{Map}_3</td>
</tr>
<tr>
<td>\text{Map}_2</td>
<td></td>
</tr>
<tr>
<td>\text{Map}_3</td>
<td></td>
</tr>
</tbody>
</table>

Thursday, December 6, 12
\text{Map}_1[] \mathbin{+=} \text{Map}_2[] \times \text{Map}_3[]
$\text{Map}_1[\ ] += \text{Map}_2[\ ] \times \text{Map}_3[\ ]$
Map₁[ ] += Map₂[ ] * Map₃[ ]

Map₂, Map₃

Map₁

Map₁

Thursday, December 6, 12
\[ \text{Map}_1[ ] += \text{Map}_2[ ] \times \text{Map}_3[ ] \]
$$\text{Map}_1[] \ += \ \text{Map}_2[] \ * \ \text{Map}_3[]$$

<table>
<thead>
<tr>
<th>$\text{Map}_2, \text{Map}_3$</th>
<th>$\text{Map}_2 \ * \ \text{Map}_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{Map}_1$</td>
<td>$\text{Map}_1$</td>
</tr>
</tbody>
</table>

Thursday, December 6, 12
ON Insert_R(A,B) DO {
    Result[] += A * Map₂[B]
    Map₁[C]  += A * Map₃[B,C]
    Map₅[B]  += A } 

ON Insert_T(C,D) DO {
    Result[] += Map₁[C] * D
    Map₂[B]  += D * Map₃[B,C]
    Map₄[C]  += D } 

ON Insert_S(B,C) DO {
    Result[] += Map₅[B] * Map₄[C]
    Map₁[C]  += Map₅[B]
    Map₂[B]  += Map₄[C]
    Map₃[B,C] += 1 }
ON Insert_R(A,B) DO {
    Result[] += A * Map2[B]
    Map1[C]   += A * Map3[B,C]
    Map5[B]   += A }

ON Insert_T(C,D) DO {
    Result[] += Map1[C] * D
    Map2[B]  += D * Map3[B,C]
    Map4[C]  += D }

ON Insert_S(B,C) DO {
    Result[] += Map5[B] * Map4[C]
    Map1[C]   += Map5[B]
    Map2[B]   += Map4[C]
    Map3[B,C] += 1 }
ON Insert_R(A,B) DO {
    Result[] += A * \text{Map}_2[B]
    \text{Map}_1[C] += A * \text{Map}_3[B,C]
    \text{Map}_5[B] += A
}

ON Insert_T(C,D) DO {
    Result[] += \text{Map}_1[C] * D
    \text{Map}_2[B] += D * \text{Map}_3[B,C]
    \text{Map}_4[C] += D
}

ON Insert_S(B,C) DO {
    Result[] += \text{Map}_5[B] * \text{Map}_4[C]
    \text{Map}_1[C] += \text{Map}_5[B]
    \text{Map}_2[B] += \text{Map}_5[C]
    \text{Map}_3[B,C] += 1
}
ON Insert_R(A,B) DO {
  Result[] += A * Map2[B]
  Map1[C]  += A * Map3[B,C]
  Map5[B]  += A
}

ON Insert_T(C,D) DO {
  Result[] += Map1[C] * D
  Map2[B]  += D * Map3[B,C]
  Map4[C]  += D
}

ON Insert_S(B,C) DO {
  Result[] += Map5[B] * Map4[C]
  Map1[C]  += Map5[B]
  Map2[B]  += Map4[C]
  Map3[B,C] += 1
}
ON Insert_R(A,B) DO {
    Result[] += A * Map_2[B]
    Map_1[C] += A * Map_3[B,C]
    Map_5[B] += A }

ON Insert_T(C,D) DO {
    Result[] += Map_1[C] * D
    Map_2[B] += D * Map_3[B,C]
    Map_4[C] += D }

ON Insert_S(B,C) DO {
    Result[] += Map_5[B] * Map_4[C]
    Map_1[C] += Map_5[B]
    Map_2[B] += Map_4[C]
    Map_3[B,C] += 1 }

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Map_1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map_2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map_3</td>
<td></td>
<td></td>
<td></td>
<td>&lt;1,1&gt;→1</td>
</tr>
<tr>
<td>Map_4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map_5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Result</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
ON Insert_R(A,B) DO {
Result[] += A * Map2[B]
Map1[C] += A * Map3[B,C]
Map5[B] += A
}

ON Insert_T(C,D) DO {
Result[] += Map1[C] * D
Map2[B] += D * Map3[B,C]
Map4[C] += D
}

ON Insert_S(B,C) DO {
Result[] += Map5[B] * Map4[C]
Map1[C] += Map5[B]
Map2[B] += Map5[B]
Map3[B,C] += 1
}
ON Insert_R(A,B) DO {
    Result[] += A * Map2[B]
    Map1[C]   += A * Map3[B,C]
    Map5[B]   += A
}

ON Insert_T(C,D) DO {
    Result[] += Map1[C] * D
    Map2[B]  += D * Map3[B,C]
    Map4[C]  += D
}

ON Insert_S(B,C) DO {
    Result[] += Map5[B] * Map4[C]
    Map1[C]   += Map5[B]
    Map2[B]  += Map5[B]
    Map3[B,C] += 1
}
ON Insert_R(A,B) DO {
    Result[] += A * Map[2][B]
    Map[1][C] += A * Map[B,C]
    Map[5][B] += A
}

ON Insert_T(C,D) DO {
    Result[] += Map[1][C] * D
    Map[2][B] += D * Map[B,C]
    Map[4][C] += D
}

ON Insert_S(B,C) DO {
    Result[] += Map[5][B] * Map[4][C]
    Map[1][C] += Map[5][B]
    Map[2][B] += Map[4][C]
    Map[3][B,C] += 1
}
ON Insert_R(A,B) DO {
    Result[] += A * Map2[B]
    Map1[C]  += A * Map2[B,C]
    Map5[B]  += A
}

ON Insert_T(C,D) DO {
    Result[] += Map1[C] * D
    Map2[B]  += D * Map2[B,C]
    Map4[C]  += D
}

ON Insert_S(B,C) DO {
    Result[] += Map5[B] * Map4[C]
    Map1[C]  += Map5[B]
    Map2[B]  += Map4[C]
    Map3[B,C] += 1
}
ON Insert_R(A,B) DO {
    Result[] += A * Map[B]
    Map1[C]  += A * Map1[B,C]
    Map5[B]  += A
}

ON Insert_T(C,D) DO {
    Result[] += Map1[C] * D
    Map2[B]  += D * Map2[B,C]
    Map4[C]  += D
}

ON Insert_S(B,C) DO {
    Result[] += Map5[B] * Map4[C]
    Map1[C]  += Map5[B]
    Map2[B]  += Map4[C]
    Map3[B,C] += 1
}

<table>
<thead>
<tr>
<th>Map1</th>
<th>&lt;1&gt;&lt;1&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map2</td>
<td>&lt;1&gt;&lt;1&gt;</td>
</tr>
<tr>
<td>Map3</td>
<td>&lt;1,1&gt;&lt;1&gt;</td>
</tr>
<tr>
<td>Map4</td>
<td>&lt;1&gt;&lt;1&gt;</td>
</tr>
<tr>
<td>Map5</td>
<td>&lt;1&gt;&lt;1&gt;</td>
</tr>
<tr>
<td>Result</td>
<td></td>
</tr>
</tbody>
</table>
ON Insert_R(A,B) DO {
    Result[] += A * Map2[B]
    Map1[C]  += A * Map3[B,C]
    Map5[B]  += A
}

ON Insert_T(C,D) DO {
    Result[] += Map1[C] * D
    Map2[B]  += D * Map3[B,C]
    Map4[C]  += D
}

ON Insert_S(B,C) DO {
    Result[] += Map5[B] * Map4[C]
    Map1[C]  += Map5[B]
    Map2[B]  += Map4[C]
    Map3[B,C] += 1
}

<table>
<thead>
<tr>
<th>Map1</th>
<th>&lt;1&gt;→1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map2</td>
<td>&lt;1&gt;→1</td>
</tr>
<tr>
<td>Map3</td>
<td>&lt;1,1&gt;→1</td>
</tr>
<tr>
<td>Map4</td>
<td>&lt;1&gt;→1</td>
</tr>
<tr>
<td>Map5</td>
<td>&lt;1&gt;→1</td>
</tr>
<tr>
<td>Result</td>
<td>??</td>
</tr>
</tbody>
</table>
“Magic Maps”

Map Maintenance Messages

System Design

Hybrid Consistency
Epoch: 0
Epoch: 0
Epoch: 0

\[ E \langle 0,0,1 \rangle \]

\[ E \langle 0,1,0 \rangle \]
$\text{Map}_2[1,3] += 2$

$\langle 0, 2, 4 \rangle$

$\text{E}$

$\langle 0, 1, 3 \rangle$
$\mathbf{E}<0,1,3> \xrightarrow{\text{History}} \mathbf{E}<0,1,3> \xrightarrow{\text{Map}_2[...]} \text{Map}_3[...] * \text{Map}_4[...]$
\[ E <0,1,3> \rightarrow \text{History} \rightarrow \text{Map}_2[...] += \text{Map}_3[...]*\text{Map}_4[...]\]
$E <0,1,3>$

$\Sigma_\delta \text{Map}_3 [...] * \text{Map}_4 [...]$

$\text{Map}_2 [...] += \text{Map}_3 [...] * \text{Map}_4 [...]$

$\text{Map}_2 [...] += \text{...} <0,1,3>$

Thursday, December 6, 12
→ Map$^2_{[1,3]} += 2$

$<0,2,4>$
$\text{Map}_2[1,3] += 2$

$<0,2,4>$

$<1,3> \rightarrow 2 \quad 4 \quad 2 \quad 3 \quad 1$

$<0,1,2> \quad <0,1,3> \quad <0,2,4> \quad <0,2,5> \quad <0,3,1>$
\[ \text{Map}_2[1,3] += 2 \]

\[ <0,2,4> \]

\[ <0,2,5> \]

\[ <0,3,1> \]

\[ E \]

\[ \delta \]

\[ <0,2,5> \]

Thursday, December 6, 12
Coord.

Coord.

Coord.

Epoch Commit
Commit OK
ON Insert_R(A,B) DO {
    Result[] += A * Map_2[B]
    Map_1[C] += A * Map_3[B,C]
    Map_5[B] += A }

ON Insert_T(C,D) DO {
    Result[] += Map_1[C] * D
    Map_2[B] += D * Map_3[B,C]
    Map_4[C] += D }

ON Insert_S(B,C) DO {
    Result[] += Map_5[B] * Map_4[C]
    Map_1[C] += Map_5[B]
    Map_2[B] += Map_4[C]
    Map_3[B,C] += 1 }
Map

1

Result

2

3

4

5

Map

1

Map

4

Map

2

Map

3

Result

✓

✓

✓

✓

✓

✓

✓

✓

Thursday, December 6, 12
Cumulus

- Additive deltas streamline distribution of query processing.
- Key/Value stores and query processing engines complement each other.
- Cumulus’ hybrid consistency engine supports both low-latency and high-precision applications.
Laasie (and BarQL)

(with Luke Ziarek, Sumit Agarwal, and Daniel Bellinger)
Collaborative Web Applications
Collaborative Web Applications

- Dropbox
- Google Docs
- Google Wave
- Office 365
- Twitter
- Facebook

Logos are the property of their respective owners
Collaborative Web Applications
Collaborative Web Applications

The Application Lives in the Browser

Image Source: openclipart.org, OpenIconLibrary
Collaborative Web Applications

The Application Lives in the Browser
The Server Just Relays and Persists Application State

Image Source: openclipart.org, OpenIconLibrary
Collaborative Applications

Request Updates for $Q(state)$

{Update 1, Update 2, ...}

Post Update: $state := Q(state)$

Success!

Image Source: openclipart.org, OpenIconLibrary
State as a Log
State as a Log

Clients Log State Updates
The Server Relays Log Entries

Image Source: openclipart.org, OpenIconLibrary

Thursday, December 6, 12
State as a Log

Clients Can Leave Unexpectedly

Image Source: openclipart.org, OpenIconLibrary
State as a Log

Clients Can Leave Unexpectedly
When Clients Arrive, the Server Can Restore Their State
State as a Log

But Clients Can Leave For Other Reasons

Image Source: openclipart.org, OpenIconLibrary
State as a Log

But Clients Can Leave For Other Reasons
...And Only Need The Latest Changes When It Returns
Log as a Service
Log as a Service

I need all docs

Image Source: openclipart.org, OpenIconLibrary
Log as a Service

I need all docs

... about VLDB
... by Bob
... modified in the last 2 days

Client state is a View!

Image Source: openclipart.org, OpenIconLibrary

Thursday, December 6, 12
Laasie

(1) App Requests Views
(2) Proxy Collates Views
(3) Proxy Requests Master View
(4) Server Sends View Updates
(5) Proxy Updates Local State
(6) Proxy Notifies App of Changes

Application  LaaS Proxy  LaaS Service

Image Source: openclipart.org, OpenIconLibrary
Laasie

Making the Log Scale: Indexing

Send me all docs by Bob

Index on /docs/*/creator

Image Source: openclipart.org, OpenIconLibrary
Laasie

Making the Log Scale: Materialization

How many unread messages?

Materialized
COUNT(filter(/msgs))
Laasie

Making the Log Scale: Log Rewrites

Image Source: openclipart.org
Active Research

- Cumulus -- Distributed Monitoring
- Laasie -- Collaborative Applications
- Mìmisbrunnr -- Managing Uncertain Knowledge
- GraphDBs -- Graphs as First Class Objects

http://okennedy.xthemage.net/?page=research