Incremental Data Transformations on Wide-Column Stores with NotaQL

M. Sc. Johannes Schildgen
schildgen@cs.uni-kl.de
"A DBA walks into a NoSQL bar, but turns and leaves because he couldn't find a table"
<table>
<thead>
<tr>
<th>RowId</th>
<th>info</th>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RowId</td>
<td>info</td>
<td>children</td>
</tr>
<tr>
<td>-------</td>
<td>------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td>Peter</td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td>Lisa</td>
<td>1997</td>
<td>BSIT</td>
</tr>
</tbody>
</table>
HBase API

put 'pers', 'Carl',
'info:born', '1982'

put 'pers', 'Carl',
'info:school', 'BSIT'

put 'pers', 'Carl',
'info:school', 'BUIT'

get 'pers', 'Carl'
Jaspersoft HBase QL

\[ \sigma_{\text{school}='BSIT'} \text{pers} \]

```json
{
    "tableName": "pers",
    "deserializerClass": "com.jaspersoft...DefaultDeserializer",
    "filter": {
        "SingleColumnValueFilter": {
            "family": "info",
            "qualifier": "school",
            "compareOp": "EQUAL",
            "comparator": {
                "SubstringComparator": {
                    "substr": "BSIT"
                }
            }
        }
    }
}
```

Phoenix

\[ \sigma_{\text{school='BSIT'}} \text{pers} \]

SELECT * FROM pers WHERE school = 'BSIT'

„Parent of each person?“
<table>
<thead>
<tr>
<th>Column</th>
<th>RowID</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output Cell</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Column</td>
<td>RowID</td>
<td>Value</td>
</tr>
</tbody>
</table>

The diagram shows a flow of data from the input cell to the output cell through the application of NotaQL.
<table>
<thead>
<tr>
<th>Input Cell</th>
<th>Output Cell</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Cell" /></td>
<td><img src="image2.png" alt="Cell" /></td>
</tr>
</tbody>
</table>

Notation: 

- **r**: Input value
- **v**: Output value
- **c**: Calculation

Diagram: NətaQL
\[ \pi_{\text{born pers}} \]

Input Cell

<table>
<thead>
<tr>
<th>_r</th>
<th>_v</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisa</td>
<td>1997</td>
</tr>
</tbody>
</table>

Output Cell

<table>
<thead>
<tr>
<th>_r</th>
<th>_v</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisa</td>
<td>1997</td>
</tr>
</tbody>
</table>

\[ \text{OUT.}_r \leftarrow \text{IN.}_r, \]
\[ \text{OUT.born} \leftarrow \text{IN.born}; \]
\[ \pi_{\text{born}, \text{school}} \text{ pers} \]

### Input Cell

<table>
<thead>
<tr>
<th>_r</th>
<th>_v</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisa</td>
<td>1997</td>
</tr>
</tbody>
</table>

### Output Cell

<table>
<thead>
<tr>
<th>_r</th>
<th>_v</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisa</td>
<td>1997</td>
</tr>
</tbody>
</table>

**NotaQL**

\[
\text{OUT}._r \leftarrow \text{IN}._r, \\
\text{OUT}.\text{born} \leftarrow \text{IN}.\text{born}, \\
\text{OUT}.\text{school} \leftarrow \text{IN}.\text{school};
\]
\[ \sigma_{\text{school}='BSIT'} \text{ pers} \]

Input Cell

<table>
<thead>
<tr>
<th>_r</th>
<th>_v</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisa</td>
<td>1997</td>
</tr>
</tbody>
</table>

Output Cell

<table>
<thead>
<tr>
<th>_r</th>
<th>_v</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lisa</td>
<td>1997</td>
</tr>
</tbody>
</table>

\[ \text{OUT.}_r \leftarrow \text{IN.}_r, \]
\[ \text{OUT.}$($\text{IN.}_c$) \leftarrow \text{IN.}_v; \]
\( \sigma_{\text{school} = 'BSIT'} \) pers

\[
\begin{array}{|c|c|}
\hline
\_r & \_v \\
\hline
Lisa & 1997 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|}
\hline
\_r & \_v \\
\hline
Lisa & 1997 \\
\hline
\end{array}
\]

\text{IN-FILTER: school='BSIT',}
\text{OUT.}_r \leftarrow \text{IN.}_r,
\text{OUT.$\$(IN.}_c) \leftarrow \text{IN.}_v;
That was:

Selection and Projection
Now: Grouping
Salary sum of each company.

<table>
<thead>
<tr>
<th>Input Cell</th>
<th>Output Cell</th>
</tr>
</thead>
<tbody>
<tr>
<td>IN.cmpny</td>
<td>OUT.salsum</td>
</tr>
<tr>
<td>_r Peter</td>
<td>_r IBM</td>
</tr>
<tr>
<td>_v IBM</td>
<td>_v 645k</td>
</tr>
</tbody>
</table>

**NotaQL**

```
OUT._r <- IN.cmpny,
OUT.salsum <- SUM(IN.salary);
```
<table>
<thead>
<tr>
<th>RowId</th>
<th>info</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Eve</td>
<td>born</td>
<td>cmpny</td>
<td>salary</td>
</tr>
<tr>
<td></td>
<td>1965</td>
<td>IBM</td>
<td>70k</td>
</tr>
<tr>
<td>Carl</td>
<td>born</td>
<td>cmpny</td>
<td>job</td>
</tr>
<tr>
<td></td>
<td>1966</td>
<td>IBM</td>
<td>intern</td>
</tr>
<tr>
<td>Julia</td>
<td>born</td>
<td>cmpny</td>
<td>salary</td>
</tr>
<tr>
<td></td>
<td>1967</td>
<td>IBM</td>
<td>80k</td>
</tr>
<tr>
<td>Lisa</td>
<td>born</td>
<td>school</td>
<td>salary</td>
</tr>
<tr>
<td></td>
<td>1997</td>
<td>BSIT</td>
<td>1k</td>
</tr>
</tbody>
</table>

```
OUT._r <- IN.cmpny,
OUT.salsum <- SUM(IN.salary):
```
Advanced Transformations: More Filters
<table>
<thead>
<tr>
<th>RowId</th>
<th>info</th>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td>Peter</td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td>Lisa</td>
<td>1997</td>
<td>BSIT</td>
</tr>
<tr>
<td>RowId</td>
<td>info</td>
<td>children</td>
</tr>
<tr>
<td>-------</td>
<td>--------------</td>
<td>----------</td>
</tr>
<tr>
<td></td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td>Peter</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lisa</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1997</td>
<td>BSIT</td>
</tr>
</tbody>
</table>

**NotaQL**

```sql
OUT._r <- IN._r,
OUT.$(IN._c) <- IN._v;
```
<table>
<thead>
<tr>
<th>RowId</th>
<th>info</th>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td></td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td>Lisa</td>
<td>Carl</td>
</tr>
<tr>
<td></td>
<td>€5</td>
<td>€0</td>
</tr>
<tr>
<td>Lisa</td>
<td>born</td>
<td>school</td>
</tr>
<tr>
<td></td>
<td>1997</td>
<td>BSIT</td>
</tr>
</tbody>
</table>

**IN-FILTER:** COL_COUNT(children) > 0
**OUT._r <- IN._r,**
**OUT.$(IN._c) <- IN._v;**
<table>
<thead>
<tr>
<th>RowId</th>
<th>info</th>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td>Peter</td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td>Lisa</td>
<td>1997</td>
<td>BSIT</td>
</tr>
</tbody>
</table>

IN-FILTER: COL_COUNT(children) > 0
OUT._r <- IN._r,
OUT.$(IN.children._c) <- IN._v;
<table>
<thead>
<tr>
<th>RowId</th>
<th>Info</th>
<th>Children</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td>Peter</td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td>Lisa</td>
<td>1997</td>
<td>BSIT</td>
</tr>
</tbody>
</table>

**NotaQL**

```
IN-FILTER: COL_COUNT(children)>0
OUT._r <- IN._r, cell predicate
OUT.$(IN.children._c?(@>5)) <- IN._v;
```
<table>
<thead>
<tr>
<th>RowId</th>
<th>info</th>
<th>children</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>born</td>
<td>cmpny</td>
</tr>
<tr>
<td>Peter</td>
<td>1965</td>
<td>IBM</td>
</tr>
<tr>
<td>Lisa</td>
<td>1997</td>
<td>BSIT</td>
</tr>
</tbody>
</table>

IN-FILTER: COL_COUNT(children) > 0
OUT._r <- IN._r, OUT.?(IN.children._c?(!Carl)) <- IN._v;
NotaQL Transformation Platform: MapReduce
map(rowId, row)

row violates row pred.? yes
no

has more columns? no yes

stop

yes cell violates cell pred.? no

map IN._r,_c,_v}, fetched columns and constants to r,c and v

emit((r, c), v)
\[
\text{reduce}((r, c), \{v\})
\]

\[
\text{put}(r, c, \text{aggregateAll}(v))
\]

\[
\text{Stop}
\]

\[
((\text{IBM}, \text{salsum}), \{70k, 80k, 10k\})
\]

\[
((\text{IBM}, \text{salsum}), 160k)
\]
Incremental Transformations: Self-Maintainability
Salary sum of all people born before 1980 per company.

\[
\text{IN-FILTER: born}<1980, \\
\text{OUT._r <- IN.cmpny,} \\
\text{OUT.salsum <- SUM(IN.salary);} \\
\]

17:45  Execute job

17:47  New people are added

17:50  Execute job again
Salary sum of all people born before 1980 per company.

\begin{align*}
\text{IN-FILTER: born} &< 1980, \\
\text{OUT.}_r &< \text{IN.cmpny}, \\
\text{OUT.salsum} &< \text{SUM(IN.salary)};
\end{align*}

Execute job

IBM 150k
Melissa 1989 IBM 50k
Nora 1977 IBM 80k

New people are added

IBM 230k

Execute job again
Salary sum of all people born before 1980 per company.

IN-FILTER: born<1980, OUT._r <- IN.cmpny, OUT.salsum <- SUM(IN.salary);

17:50  Execute job
17:52  People are deleted
17:55  Execute job again

<table>
<thead>
<tr>
<th>company</th>
<th>salsum</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBM</td>
<td>230k</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>born</th>
<th>cmpny</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1989</td>
<td>IBM</td>
<td>50k</td>
</tr>
<tr>
<td>1977</td>
<td>IBM</td>
<td>80k</td>
</tr>
</tbody>
</table>

IBM 150k
Salary sum of all people born before 1980 per company.

IN-FILTER: born<1980,
OUT._r <- IN.cmpny,
OUT.salsum <- SUM(IN.salary);

17:50
Execute job

17:52
People are updated

17:55
Execute job again

<table>
<thead>
<tr>
<th>born</th>
<th>cmpny</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>1965-1990</td>
<td>IBM</td>
</tr>
</tbody>
</table>

- = 70k
Salary sum of all people born before 1980 per company.

IN-FILTER: born<1980,
OUT._r <- IN.cmpny,
OUT.salsum <- SUM(IN.salary);

People are updated

17:50
Execute job

17:52
Peter 1965 IBM 70k
1990 += 70k

17:55
Execute job again
Salary sum of all people born before 1980 per company.

NotaQL

IN-FILTER: born<1980,
OUT._r <- IN.cmpny,
OUT.salsum <- SUM(IN.salary);

17:50
Execute job

17:52
People are updated

17:55
Execute job again

<table>
<thead>
<tr>
<th>born</th>
<th>cmpny</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>1965</td>
<td>70k</td>
</tr>
</tbody>
</table>

+= 75k-70k
Salary sum of all people born before 1980 per company.

\[
\text{IN-FILTER: born} < 1980, \\
\text{OUT.}_r \leftarrow \text{IN.} \text{cmpny}, \\
\text{OUT.} \text{salsum} \leftarrow \text{SUM(} \text{IN.} \text{.salary)};
\]

Execute job

17:50

People are updated

17:52

Execute job again

17:55

<table>
<thead>
<tr>
<th>born</th>
<th>cmpny</th>
<th>salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peter</td>
<td>1965</td>
<td>IBM SAP 70k</td>
</tr>
</tbody>
</table>
map(rowId, row)

row violates row pred.? yes
  no

has more columns? yes
  no

  Stop

  yes

  cell violates cell pred.? yes
  no

map IN.{_r, _c, _v}, fetched columns and constants to r,c and v

emit((r, c), v)

reduce(((r,c), {v}))

put(r, c, aggregateAll(v))

Stop

Just read Delta (ts>tŝ)
map(rowId, row)

- row violates row pred.?  
  - yes: ...and the former Result
  - no: has more columns?
    - no: Stop
    - yes: cell violates cell pred.?
      - yes: map IN.\{r, c, v\}, fetched columns and constants to r, c and v
        - emit((r, c), v)
      - no: reduce(((r, c), \{v\}))
        - put(r, c, aggregateAll(v))
        - Stop
map(rowId, row)

row violates row pred.? no

has more columns? no Stop

Was it satisfied on previous execution? yes? Invert v

map IN.{_r, _c, _v}, fetched columns and constants to r, c and v

cell violates cell pred.? yes emit((r, c), v)

No

reduce(((r, c), {v}))

put(r, c, aggregateAll(v))

Stop
map(rowId, row)

row violates row pred.?

yes

row

no

has more columns?

no

Stop

yes

cell violates cell pred.?

yes

map IN._r,_c,_v, fetched columns and constants to r,c and v

no

emit((r, c), v)

Unchanged?

reduce(((r,c), {v}))

put(r, c, aggregateAll(v))

Stop
Evaluation: Performance
TPCH 1 (selection, projection, sum of prices per status code)
TPCH 6 (selection, projection, sum of all prices)
Reverse Web-Link Graph

Native Hadoop
NotaQL (non-inc.)
NotaQL (0.1% changes, timestamp-based CDC)
NotaQL (10% changes, manual CDC)
Conclusion

NotaqL ≠ SQL

- Selection, Projection  ✔
- Grouping, Aggregation  ✔
- Schema-Flexible  ❌
- Horizontal Aggregation  ❌
- Metadata ↔ Data  ❌
- Graph Processing  ❌
- Text Processing  ❌

Incremental!
Thank you!

\[
\text{\texttt{\langle NotaQL\rangle} \models \ [\text{IN-FILTER: \langle predicate\rangle}, \langle rowspec\rangle, \langle cellspec\rangle (=, \langle cellspec\rangle) \ast \ ;]}
\]

\[
\text{\langle rowspec\rangle} \models \text{OUT.}_{-r} \leftarrow \langle vdata\rangle
\]

\[
\text{\langle cellspec\rangle} \models \text{OUT.}(\langle colname\rangle \mid \$((\langle input\rangle)) \leftarrow ((\langle vdata\rangle \mid \langle aggfunk\rangle((\langle vdata\rangle))
\]

\[
\text{\langle input\rangle} \models (\text{IN.}_{-r} \mid \text{IN.}[\langle colfamily\rangle]:[(-c \mid _v) \mid \text{IN.}(\langle colname\rangle)])[?((\langle predicate\rangle))]
\]

\[
\text{\langle vdata\rangle} \models \langle input\rangle \mid \langle const\rangle \mid \langle vdata\rangle(+ \mid - \mid * \mid /)\langle vdata\rangle
\]

\[
\text{\langle aggfunk\rangle} \models \text{COUNT} \mid \text{SUM} \mid \text{MIN} \mid \text{MAX} \mid \text{AVG}
\]

\[
\text{\langle const\rangle} \models '(A \ldots Z \mid a \ldots z \mid 0 \ldots 9) +' \mid (0 \ldots 9) +
\]

\[
\text{\langle colname\rangle} \models [\langle colfamily\rangle]:([A \ldots Z \mid a \ldots z \mid 0 \ldots 9) +
\]

\[
\text{\langle colfamily\rangle} \models (A \ldots Z \mid a \ldots z \mid 0 \ldots 9) +
\]

\[
\text{\langle predicate\rangle} \models (\langle colname\rangle \mid @ \mid \text{col\_count}([\langle colfamily\rangle]))[\langle op\rangle(\langle colname\rangle \mid \langle const\rangle)]
\]

\[
\quad \mid (\text{NOT} \mid !)\langle predicate\rangle \mid \langle predicate\rangle(\text{AND} \mid \text{OR})\langle predicate\rangle
\]

\[
\text{\langle op\rangle} \models = \mid != \mid < \mid <= \mid > \mid >=
\]