1 Investigation: How expressive should the constraint-lang be?

Discussion question: Which ones are supposed to be supported by our language?

Data Source These functions are summarized based on 1) previously collected Stack Overflow questions, 2) top-rated SQL questions in Stack Overflow, 3) TPC-H, 4) StackExchange examples and 5) some of SQL-Share examples.

Related SQL functions

- Sub-query support: sub-queries in JOIN clause, EXISTS/IN clause, Where clause (comparing with values when sub-query result is a single value table). Sometimes nested level can be high: 3-4 levels.
- Group by and aggregation support: Max, Min, Sum, Average, Count.
- Pivot table.
- Case analysis in SQL.
- Top, Limit, Order by, Distinct.
- Set operations.
- Syntax concatenation. (XML PATH)
- Interpret a single value table as a value.
- Exist, In, All

None-trivial task examples

- Argmax.
- Flat table operations: self-join several times.
  (http://stackoverflow.com/questions/192220/what-is-the-most-efficient-elegant-way-to-parse-a-flat-table-into-a-tree)
- Filtering on multiple fields.
- Finding duplicates.
  (http://stackoverflow.com/questions/2594829/finding-duplicate-values-in-a-sql-table)
- Calculate moving average.
- Find medium value / n-th value in a table.
  (http://stackoverflow.com/questions/1342898/function-to-calculate-median-in-sql-server)
- Arithmetic operation on result.
- Moving average: group by based on window.
  (http://stackoverflow.com/questions/1176011/sql-to-determine-minimum-sequential-days-of-access)
- Date reformatting.
- Order rows.
- PIVOT, string concatenation.
  (http://stackoverflow.com/questions/194852/concatenate-many-rows-into-a-single-text-string)

Other notes The constraint language should support procedural description, e.g. first filter then aggregate.

2 Constraint language design

Discussion question **How to limit the expressiveness of the language?**

Basics Constraint language is designed as an interface between higher level description and lower level SQL queries. We should be able to provide a bidirectional (rule-based) transformation between the constraint language and the SQL queries.

Language design goal:

- Task descriptions can be mapped to the language directly.
- Modularity: complex description can be generated from several simple description.
- Easy to synthesis: can be generated from input easily.

TRC/DRC and language design TRC is a none-procedural language describing SQL queries. Here is a good reference for the language. (Link)

We propose an extension of the language to support the features mentioned above. (Current design only contains an extension of aggregation and subquery) The language now is over-powered: is too expressive and same task can be represented by many different constraints, and I'm working on limiting its expressiveness now.

$$
\begin{array}{rcl}
Q & := & \{r : (c_1, ..., c_n) \mid P(r : (c_1, ..., c_n))\} \\
P & := & c \ relop \ c \\
 & | & c \ relop \ v \\
 & | & c \ relop \ T \\
 & | & R((c_1, ..., c_n)) \\
 & | & \exists r \in T.P(r) \\
 & | & \forall r \in T.P(r) \\
 & | & P \wedge P \\
 & | & P \vee P \\
 & | & \neg P \\
relop & := & < \mid > \mid \geq \mid \leq \mid \neq \mid == \mid \in \\
v & := & \mathsf{const} \\
 & | & \mathsf{table\_to\_value}(T) \\
 & | & \mathsf{aggr}(f, (c_1, ..., c_m), c, T) \\
 & | & f(l) \\
l & | & [v_1, ..., v_n] \\
 & | & [c_i \mid r(c_1, ..., c_i, ..., c_n) \in T] \\
T & := & \mathsf{named}(T) \\
 & | & \{r \in T \mid P(r)\} \\
 & | & T \bowtie T \\
 & | & T \ \mathsf{setop} \ T
\end{array}
$$

2.1 Examples

example-1 PostgreSQL - MAX value for every user

$$Q = \{r : (c_1, c_2, c_3) \mid r \in \texttt{input} \wedge c_3 = \mathsf{aggr}(\mathsf{max}, (\texttt{User}), \texttt{Value}, \{r' \in \texttt{input} \mid r'.\texttt{Value} \leq 8\})\}$$

example-2 SQL exists comparison

$$Q = \{r : (c_1, c_2, c_3) \mid r \in \texttt{input} \wedge c_3 == \text{`}\texttt{TREATED}\text{'} \wedge \exists r' \in \texttt{input}.(r'.\texttt{country} == c2 \wedge r'.\texttt{status} == \text{`}\texttt{UNTREATED}\text{'})\}$$

example-3 Only joining rows where the date is less than the max date in another field

$$Q = \{r : (c_1, c_2, c_3) \mid (c_1, c_3) \in \texttt{table}_1 \wedge (c_1, c_2) \in \texttt{table}_2 \wedge c_2 \leq \texttt{aggr}(\texttt{max}, (\texttt{Emp\_ID}), \texttt{Promo\_Date}, \texttt{table}_2)\}$$

3 Some observations from the SQLShare Log

I have some results on SQLShare log observation. Should show them in slides tomorrow.