

# 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.

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

### 2.1 Examples

#### example-1 PostgreSQL - MAX value for every user

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

#### example-2 SQL exists comparison

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

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

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

### 3 Some observations from the SQLShare Log

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