1 Syntax

In this language, constraints should be built only using provided components, you can only:

- Use a relation operator (relop) from OP.
- Compare a tuple field with another field in the tuple.
- Compare a tuple field with a constant provided in the value set V.
- Compare a tuple with value calculated from aggregation, and the aggregation should be performed on a table in the table set $\bar{T}$.
- Join or union tables only using tables from $\bar{T}$.
- Compare a tuple with a table T, where tables are restricted to basic tables T and joined/unioned tables from multiple basic tables.

The constraint language given synthesis components $(\bar{T}, O, V, OP, F)$.

$$
\begin{array}{rcl}
Q & := & \{r : (c_1, ..., c_n) \mid P(r)\} \\
P & := & cmp \\
src & := & T(\overline{r.c}) \\
 & | & r.c \leftarrow aggrv(AGR)(r) \\
 & | & \texttt{CASE}\ vrel\ \texttt{THEN}\ src\ \texttt{ELSE}\ src \\
vrel & := & c\ \mathsf{relop}\ c \\
 & | & c\ \mathsf{relop}\ v \\
 & | & c\ \mathsf{relop}\ aggrv(T, f, (\overline{T.c}), T.c)(r) \\
trel & := & (\overline{r.c}) \in T \\
 & | & \exists\ r' \in T.(r\ \mathsf{relop}\ r') \\
 & | & \forall\ r' \in T.(r\ \mathsf{relop}\ r') \\
T & := & T\ (T \in \bar{T}) \\
 & | & T \bowtie T \\
 & | & T\ \mathsf{setop}\ T \\
\mathsf{relop} & := & r\ (r \in OP) \\
v & := & v\ (v \in V)
\end{array}
$$

$$V_0 = \left\{c \mid c \in \bar{T}\right\}$$

2 Meta-Constraint Language

As we only allow query constraints write with provided components during the synthesis phase, users need to provide extra components through interaction to enable synthesis of complex queries.

A meta constraint M is a tuple $(\bar{T}, O, V, OP, F)$, containing input table set $\bar{T}$, output set O, value set V, operator set OP and aggregation function set F.

$$
\begin{array}{rcll}
M & := & (\bar{I}, O, V_0, OP_0, F) & \text{(Basic synthesis framework)} \\
 & | & \texttt{extend}\ M.V_0\ \texttt{with}\ \overline{v} & \text{(Extend value set)} \\
 & | & \texttt{extend}\ M.OP_0\ \texttt{with}\ \overline{op} & \text{(Extend operator set)} \\
 & | & \texttt{extend}\ M.F\ \texttt{with}\ \overline{f} & \text{(Extend aggregation functions)} \\
 & | & \texttt{extend}\ M.\bar{T}\ \texttt{with}\ Syn[M'](M'.\bar{T})\{O \subset Syn[M'](M'.\bar{T})\} & \text{(Sub-procedure extension)}
\end{array}
$$

2.1 User interactions

When a user provides extra synthesis ingredients (besides default input output pairs), the synthesizer should be able to 1) enhance the expressiveness of the synthesized query, and 2) change the ranking of candidate programs.

Users can extend meta-constraints through the following interactions:

- Providing a value/operator/aggrfunction as a keyword in the description. (Extend value/operator/function set)
- Describing a sub-procedure: there exists an intermediate result O_m such that O is contained by O_m and O_m is synthesized using original synthesis components; and we can synthesize O by synthesizing a filter for O_m using given synthesize component.

  Example: "1. SELECT * WHERE from_user <> id 2. GROUP BY conversation_id 3. SELECT in every group row with MAX(timestamp) **(if there are two same timestamps in a group use second factor as highest message_id)** !!! 4. then results SORT BY timestamp"
- Describing a computation of on a special case, most (all?) of this case are describing a computation related to aggregation. In this case, we will generalize the computation in to group by flavor queries.

  Example: "in a single query, given a particular ID, I want to compute the proportion of entries that has action Foo. For example, given A, there are 2 Foos out of 3 actions. So it'd be 66%."

3 Examples

3.1 Example #1

description How do I select one row per id and only the greatest rev?

input

id	rev	content
1	1	A
2	1	B
1	2	C
1	3	D

output

col1	col2	col3
1	3	D
2	1	B

Synthesis Process

- Build meta-constraint: $M = \text{extend } (input, output, V_0, OP_0, F_0).F_0 \text{ with } max$.
  i.e. $M = (input, output, V_0, OP_0, F_0 \cup \{max\})$.
- Extend the input table with max aggregation value, yield a new table T'.

id	rev	content	max(id, (rev))	max(rev, (id))
1	1	A	2	3
2	1	B	2	1
1	2	C	1	3
1	3	D	1	3

- Solve the constraints built by basic relations forms: c relop c, c relop v, c relop $\bar{T}$

4 Legacy..

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