University of Zurich[UZH]

Department of Informatics

University of Zürich
Department of Informatics
Binzmühlestr. 14
CH-8050 Zürich
Phone. +41 44 635 43 11
Fax +41 44 635 68 09
www.ifi.uzh.ch/dbtg

UZH, Dept. of Informatics, Binzmühlestr. 14, CH-8050 Zürich

Christine Graff
Mohit Narang

Prof. Dr. Michael Böhlen
Professor

Phone +41 44 635 43 33
Fax +41 44 635 68 09
boehlen@ifi.uzh.ch

Zürich, 27. Mai 2015

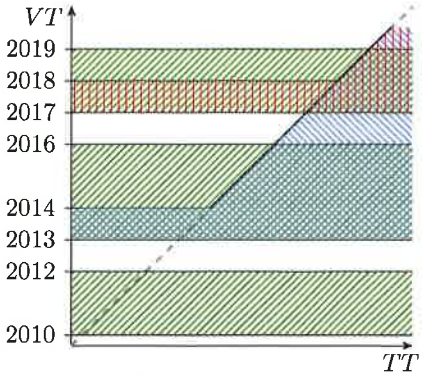
MSc Project
Topic: Querying Now-Relative Databases With Generally Valid Results

Temporal databases allow keeping track of time-varying data. Valid time databases allow storing the information when a tuple is valid in the real world. The valid time interval can either be in the past, present, or future.

Now is a constant whose value evolves over time. When used as the lower or upper bound of a tuple's valid time it allows modeling that a tuple's valid time changes over time rather than being bounded by two fixed values. The interpretation of *Now* as the upper bound is that a tuple's valid time is open-ended. There exists several real-world applications which require *Now* for properly modeling valid times. One application deals with employment contracts: a permanent employment requires an open-ended valid time and thus *Now* as the upper bound. The following table contains the employment contracts of five employees with their name, pay grade, and when they were employees.

Relation **r**

	N	P	T'
r_1	John	4	$[2010, 2012)$
r_2	Alex	3	$[2013, 2016)$
r_3	John	5	$[2017, 2019)$
r_4	Ann	3	$[2013, \langle 2014 \mid Now \rangle)$
r_5	Jane	6	$[2017, \langle 2018 \mid Now \rangle)$



Applying a query to such a now-relative relation can either result in a point-in-time result or a generally valid one. A point-in-time result is only valid at specific point in time. Thus, whenever the same query is stated at another points in time, the result has to be re-calculated. In contrast, a generally valid result is valid at every point in time and therefore does not suffer from the drawback of a point-in-time result. The following table contains the generally valid result of the query *What is the average pay grade of the employees?* applied to relation r.

$_P\vartheta^T_{AVG(P)}(r)$

Derived from	P	T'
r_1	4	$[2010, 2012)$
r_2, r_4	3	$[2013, \min^V(2016, \langle 2014 \vert Now\rangle))$
r_2	3	$[\min^V(2016, \langle 2014 \vert Now\rangle), 2016)$
r_4	3	$[2016, \min^V(2017, \langle 2016 \vert Now\rangle))$
r_3, r_4, r_5	4.67	$[2017, \min^V(2019, \langle 2017 \vert Now\rangle))$
r_3, r_5	5.5	$[\min^V(2018, \langle 2017 \vert Now\rangle), 2018)$
r_3	5	$[\min^V(2019, \langle 2018 \vert Now\rangle), 2019)$
r_4, r_5	4.5	$[2019, \langle 2019 \vert Now\rangle)$

The goal of this Master's project is to provide the functionality of querying now-relative databases with generally valid query results in PostgreSQL.

Tasks

1. Get familiar with the concept of *Now* in temporal databases [2]: how is *Now* defined, in which application scenarios is *Now* a required concept and why, and how does *Now* affect the query processing.
2. Get familiar with temporal alignment [3]: how are the temporal primitives defined and what do they intuitively mean. Integrated temporal alignment into the PostgreSQL 9.4 kernel (affected parts: grammar, parser, analyzer, optimizer, and executor) by porting the existing implementation [1] to PostgreSQL 9.4.
3. Integrate Now into the PostgreSQL kernel, supporting the time domain $\Omega^V = T \cup \{\langle t|Now\rangle, \min^V(t', \langle t|Now\rangle)\}$ with $\langle t|Now\rangle = \max^V(t, t_{curr})$.
4. Adapt the kernel parts parser and executor, resulting in a now-relative temporal alignment.
   - Range Functions: implement generally valid minimum, maximum, intersection, difference, and equals functions. Determine whether additional range functions have to be adapted and implement these adaptations.
   - Executor: Design and implement the algorithms for the temporal alignment and temporal normalization operator based on the generally valid range functions.
   - Parser: Adapt the construction of the parse tree, so that the resulting parse tree provides the information required/expected in the executor.

5. Evaluate the implementation with a set of reference queries that are performed on the constructed now-relative database.
6. Technical report (20 pages).

References

[1] http://www.ifi.uzh.ch/dbtg/research/align.html.

[2] Clifford, James and Dyreson, Curtis and Isakowitz, Tomás and Jensen, Christian S. and Snodgrass, Richard Thomas. On the Semantics of Now'in Databases. *ACM Trans. Database Syst.*, 1997.

[3] A. Dignös, M. Böhlen, and J. Gamper. Temporal alignment. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 433–444. ACM, 2012.

Supervisor: Yvonne Mülle (muelle@ifi.uzh.ch)

Start date:

End date:

University of Zurich
Department of Informatics

Prof. Dr. Michael Böhlen
Professor