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

TBD

Prof. Dr. Michael Böhlen
Professor
Phone +41 44 635 43 33
Fax +41 44 635 68 09
boehlen@ifi.uzh.ch

Zürich, November 21, 2011

Facharbeit / BSc thesis
Datenbanktechnologie

Topic: Timestamp Extraction from PostgreSQL Procedural Functions

The presence of historical information in a database is of huge value. Databases that provide support for time varying data are called Temporal Databases [1, 3]. An essential aspect to handle time varying information is, determining constant time intervals in the database. Those are intervals where no data changes take place. Extraction of time intervals from a temporal database in a brute-force manner is inefficient. The aim of this project is to extract only timestamps that are used within a procedural function.

A brute-force time interval extraction algorithm determines all constant time intervals in the database. The goal of this project is to extend PostgreSQL [2] with a built-in function that extracts time intervals from a given procedural function. This reduces the number of time intervals extracted, compared to brute-force extraction.

Tasks

- Literature review on the procedural language of PostgreSQL (PL/pgSQL) [1]
- Implement a small application using PL/pgSQL (ex: Library reservation system). That will be used to benchmark your implementation
- Whenever PostgreSQL encounters a query that calls a procedural function, traverse its' body to determine its used relations. Extract a list of time points from the `TS` and `TE` columns
- Generate a list of time intervals from the extracted time points.

[1] `http://www.postgresql.org/docs/8.3/static/plpgsql.html`

The anticipated outcome is a thesis that has a strong emphasis on the implementation performed in PostgreSQL. The student will give a 10 minute oral presentation infront of the database technology group.

Example: For the query shown in Listing 1, the invoked function contains a SQL query that interacts with the EMPLOYEE relation. The timestamps extracted from that relation are {1,2,3,4,6,8,NULL}. The resulting time intervals should then be: {(1,2), (2,3), (3,4), (4,6), (6,8), (8,NULL)}.

Listing 1: Query with function call

```
1 SELECT    getMaxSalary();
```

EMPLOYEE relation

Name	Salary	TS	TE
A	9000	1	2
B	5000	3	8
C	6000	4	6
C	7000	6	NULL

Listing 2: Procedural function

```
1  CREATE FUNCTION getMaxSalary()
2  RETURNS INTEGER AS $$
3  DECLARE
4   empTuple employee%ROWTYPE;
5   maxSalary INTEGER := -1;
6  BEGIN
7   FOR empTuple IN (SELECT * FROM employee) LOOP
8    IF empTuple.salary > maxSalary
9     THEN
10     maxSalary := empTuple.salary;
11    END IF;
12   END LOOP;
13  RETURN maxSalary;
14 END; $$ LANGUAGE PLPGSQL;
```

Supervisor: Amr Noureldin (noureldin@ifi.uzh.ch)
Start date: TBD
Duration: 3 months

University of Zürich
Department of Informatics

Prof. Dr. Michael Böhlen

References

[1] Chris J. Date, Hugn Darwen, and Hugh Darwen. *Temporal Data and the Relational Model*. Morgan Kaufman Publ Inc, 1 edition, 2002.

[2] Korry Douglas. *PostgreSQL (2nd Edition)*. Sams, Indianapolis, IN, USA, 2005.

[3] O. Etzion S. Jajodia and S. Sripada (Eds.). *Temporal Databases: Research and Practice*. Springer-Verlag, 1998.