# TimeTravel for PostgreSQL

Michele Mariani <michele.mariani@databtech.com>
Sergio Mior <sergio.mior@databtech.com>
*Database & Technology (D&T) S.r.l. Italia*

## Abstract

*TimeTravel* is an added functionality to PostgreSQL which we developed by drawing inspiration from a similar Oracle functionality: *FlashBack*. Since human errors are among the first causes of data loss, it would be a good idea to maintain historical details of all transactions, for a limited time, to be able to rollback a committed transactions when it will be needed.

Maybe you have deployed an application you realize it's damaging your data, and you realize that only some days later its deployment: what can you do? Nowadays you can make a restore from the last backup before the deployment, but if you do so all transactions will be missed: yours and all those committed by other users since the deployment. It wouldn't be great if you'd be able to rollback only the transactions committed by your application, while maintaining the ones committed by all other users in the meanwhile? It's exactly what you can do by enabling TimeTravel, which archives opposite operations to obtain metadata and historical data for transactions. Thanks to TimeTravel you can act only on data (rows) you need to restore and execute the opposite operation to the previous one, on a per-operation basis.

## 1. Introduction

In this document we show a group of features developed for PostgreSQL that allow:

- search past data;
- retrieve metadata that shows a detailed history of changes to the database;
- recover tables or rows in a particular instant back on the time.

## 2. Storage and performance impact

At any record change (insertion, updating or removal) we need to save in a dedicated log table (by using a trigger) the operation involved and its opposite one, with old and new data and some other attributes like transaction id, timestamp, current account and so on. Thus you can forecast the amount of additional space required to enable the TimeTravel functionality on your database by estimating the amount of changed data in the timeframe you wish to be able to rollback your transactions within.

The TimeTrael log table is populated each time you commit or alter any data, thus enabling this function would theoretically doubles the effort to commit changes to your database, but the performance impact due to increasing storage I/O can be considerably reduced by using a separate tablespace for this purpose.

## 3. Future development

Here is a list of the improvements we plan to add to the next releases, which are under development.

- Extend the intervention of TimeTravel to DDL operations (drop and alter table).
- Compress log table to save space.
- Reduce log table archiving to a sliding window, automatically handled by TimeTravel feature.
- Archive the original operation with WHERE predicate, not a row for every row involved in the operation itself.

## 4. Mapping functionality

In the following paragraphs we explain all functionalities of the first release of TimeTravel and the features we are going to add.

### 4.1 TimeTravel activation

Function tt_on () is used to activate TimeTravel for a table, all tables in a schema, all tables in a database. For each activated table, a new table is created with the name tt.schemaname_tablename_log to archive all DML statements committed since the activation. The log contains the original statemente and the UnDo statement to rollback itself and some other information like xid, timestamp, user and so on. In addition, if no primary key is defined for a table, a column (tt_rowid) containing a distnct value is added to the original table itself.

```
select tt_on ();
select tt_on (schema <schema_name>);
select tt_on (schema <schema_name>,
table <table_name>);
```

### 4.2 TimeTravel query

Used to retrieve the data in a specified time in the past.

It needs 2 steps to obtain the result.

```
SELECT tt.query(schema s, table t,
timestamp p);
```

The above command will create the table named

tt.<schema_name>_<table_name>_past, which contains data of table “t” at time “p”:

```
SELECT values FROM
tt.<schema_name>_<table_name>_past;
```

to know values of table “t” at time “p”.

## 4.3 TimeTravel restore

Used to restore a table to a specified time in the past.

## 4.4 TimeTravel de-activation

Function tt_off() is used to deactivate TimeTravel for a table, all tables in a schema, all tables in a database. The table tt.schemaname_tablename_log is dropped for eevery table involved in the deactivation. If column tt_rowid has been added to original table, it would be dropped. All objects useful for mantaining TimeTravel wolud be dropped.

```
select tt_off ();
select tt_off (schema <schema_name>);
select tt_off (schema <schema_name>,
table <table_name>);
```

## 4.5 Feature installation

To install it for a PostgreSQL cluster database, run :

```
psql –f install_tt.sql
```

it creates schema “tt” with objects to use TimeTravel feature.

## 4.6 Feature uninstall

The following command drops tt schema if no one table is under TimeTravel feature.

```
psql –f uninstall_tt.sql
```

## 5 Transaction log table

This table is in the schema tt, is created during the activation of the TimeTravel activation on a table. Thre is one Transaction Log Table for every activated table.

Its name is : schemaname_tablename_log.

Its task is to record all transactions occurring on that activated table.

It is used to know the metadata of the transactions that took place on the activated table.

Its attributes are :

TT_MODE: type of operation (insert / delete / update)

TT_TUPLE: new if the tuple has been added, if old instead has been eliminated

TT_TIME: timestamp of when the transaction was

TT_USER: user who performed the transaction

TT_REDO instruction or block of instructions which generated the transaction

TT_UNDO: the opposite action, row by row, of the transaction

## 6 Details of functionalities

## 6.1 TimeTravel query

Used to retrieve the data in a specific time in the past Market.

To use it, you have to give the following command [source code 3.7]:

```
SELECT tt.query (schemaname, table-
name, timestamp);
```

The above command creates in the schema “tt” the table named schemaname_tablename_past, which contains all data of that table at the chosen time.

Now, to query the table schemaname_tablename_past to get the desired information:

```
SELECT values FROM
tt.schemaname_tablename_past;
```

## 6.2 TimeTravel restore

Used to return a table as it was at any given time t1.

You can, once made the restore to a time t1, invoke the restore before or after t1 all changes made on the table.

If a table has foreign key constraints, they are disabled and they must be re-enabled manually after the restore. By querying the table named schemaname_tablename_fk, created in the schema tt before disabling the foreign key constraints , you can learn the steps to turn off all constraints.

To restore a table [source code 3.8] use the following command:

```
SELECT tt.restore (schemaname , table-
name , timestamp);
```

You can also restore to all tables in a single schema [source code 3.9] with :

```
SELECT tt.restore (schemaname ,
timestamp);
```

or all tables in the database [source code 3.10] with the following command:

```
SELECT tt.restore (timestamp);
```

## 7 Management of TimeTravel with PHP application

It has also been created a php application to be installed on Apache server to allow any user to view the tables on which it is active Timetravel.

### 7.1 Install

Execute from WINDOWS o UNIX terminal:

```
> psql -f install_tt.sql
```

### 7.2 Uninstall

execute from WINDOWS o UNIX terminal:

```
> psql -f unistall_tt.sql
```

### 7.3 Activation

Execute from psql terminal:

```
select tt.on(schemaname, tablename);
```

for 'table' tablename in 'schemaname' schema;

```
select tt.on(schemaname);
```

for all table in 'schemaname' schema;

```
select tt.on();
```

for all user table of database.

### 7.4 Deactivation

Execute from psql terminal:

```
select tt.off(schemaname, tablename);
```

for 'table' tablename in 'schemaname' schema;

```
select tt.off(schemaname);
```

or all table in 'schemaname' schema;

```
select tt.off();
```

for all user table of database.

### 7.5 Usage

Execute from psql terminal to view table's past data:

```
select tt.query(schemaname, tablename,
timestamp);
```

creates the table 'tt.schemaname_tablename_past', which contains the 'tablename' table data at the time 'timestamp'.

To restore, execute from psql terminal:

```
select tt.restore(schemaname, table-
name, timestamp);
```

for restore 'table' tablename in 'schemaname' schema at the time 'timestamp';

```
select tt.restore(schemaname, time-
stamp);
```

for restore all table in 'schemaname' schema at the time 'timestamp';

```
select tt.restore(timestamp);
```

for restore all user table of database at the time 'timestamp'.

### 7.6 PHP application setup

Copy 'tt' directory in 'www' Apache directory, modify 'conf.php' file with personal parameters and go to the address http://<IP-address>:8080/tt.

## 8. Acknowledgments

This work has been developed and fully sponsored by Database and Technology (D&T) S.r.l.
(*http://www.databtech.com*)

## 9. License and Contributions



Contributions to TimeTravel will be published on our website and, of course, are welcome. Please contact dt.marketing@databtech.net for any request about this matter.

**CODE APPENDIX**

# 1 INSTALL TIMETRAVEL

## *1.1 install_tt.sql*

```
CREATE SCHEMA IF NOT EXISTS tt;
CREATE TABLE tt.tt_tables (schema_name text, table_name text);
\ir fun/on.sql;
\ir fun/off.sql;
\ir fun/rec.sql;
\ir fun/query.sql;
\ir fun/restore.sql;
\ir fun/event_trigger.sql;
```

# 2 UNISTALL TIMETRAVEL

## *2.1 unistall_tt.sql*

```
select tt.off();
drop schema tt cascade;
```

# 3 TIMETRAVEL FUNCTION

## *3.1 tt.on(schemaname,tablename)*

```
CREATE OR REPLACE FUNCTION tt.on(text, text) RETURNS void AS $$
DECLARE
        t_name          ALIAS FOR $2;
        t_schema                ALIAS FOR $1;
        t_comp                  text;
        tt_table_name           text;
        tt_table_schema         text;
        tt_table_comp           text;
        nuovo                   integer;
        pk                      integer;
        col_record              record;

BEGIN
        tt_table_schema := 'tt';
        tt_table_name := t_schema||'_'||t_name||'_log';

        t_comp := quote_ident(t_schema)||'.'||quote_ident(t_name);
        tt_table_comp := quote_ident(tt_table_schema)||'.'||quote_ident(tt_table_name);
```

```
--check that tt is not already active on the table
	EXECUTE '
		SELECT
		CASE WHEN count(*) = 1 THEN 1
			ELSE 0
		END
		FROM tt.tt_tables
		WHERE table_name ='||quote_literal(t_name)||'
		AND schema_name = '||quote_literal(t_schema)
	INTO nuovo;

	IF (nuovo=1) then raise exception 'Tabella già presente';
	ELSE
--Check that the table has a primary key
		EXECUTE '
			SELECT
			CASE WHEN c.contype = '||quote_literal('p')||' THEN 1
				ELSE 0
			END
			FROM pg_catalog.pg_constraint c
				join
				pg_catalog.pg_statio_user_tables t
				on c.conrelid = t.relid
			WHERE t.relname ='||quote_literal(t_name)||'
			AND t.schemaname = '||quote_literal(t_schema)
		INTO pk;

		EXECUTE '
			INSERT INTO tt.tt_tables
			VALUES ('
			||quote_literal(t_schema)||','||quote_literal(t_name)||')';
-- If pk not present add the column tt_rowid
		IF (pk<>1) THEN
		EXECUTE '
			ALTER TABLE '	||t_comp
						||' ADD COLUMN tt_rowid BIGSERIAL UNIQUE
NOT NULL ';
		END IF;
-- Create the log table
		EXECUTE '
			CREATE TABLE '||tt_table_comp
			||'(LIKE '||t_comp
			||', tt_mode text'
			||', tt_tuple text'
			||', tt_time TIMESTAMP'
			||', tt_user text'
			||', tt_redo text '
			||', tt_undo text '
			||')';
--Create a trigger that will insert the changes in the log table
		EXECUTE '
			CREATE TRIGGER "tt_trigger" AFTER UPDATE OR INSERT OR
```

```
DELETE ON '
                        ||t_comp||' FOR EACH ROW EXECUTE PROCEDURE tt.rec('
                        ||quote_literal(t_name)
                        ||','
                        ||quote_literal(t_schema)
                        ||')';
        END IF;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.2 tt.on(schemaname)*

```
CREATE OR REPLACE FUNCTION tt.on(text) RETURNS void AS $$
DECLARE
        select_schema           ALIAS FOR $1;
        tab_record              record;

BEGIN
--For each table in selected schema
        FOR tab_record IN
                SELECT schemaname, tablename
                FROM pg_catalog.pg_tables t1
                WHERE schemaname = select_schema
                AND tablename NOT IN (
                        SELECT t2.table_name
                        FROM tt.tt_tables t2
                        WHERE t1.schemaname=t2.schema_name)
        LOOP
-- call the function tt.on
                EXECUTE '
                        SELECT tt.on('||quote_literal(tab_record.schemaname)
                                        ||','
                                        ||quote_literal(tab_record.tablename)
                                        ||')';
        END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.3 tt.on()*

```
CREATE OR REPLACE FUNCTION tt.on() RETURNS void AS $$
DECLARE
        tab_record      record;

BEGIN
--For all user table of database
        FOR tab_record IN
                SELECT schemaname, tablename
                FROM pg_catalog.pg_tables t1
                WHERE schemaname <> 'pg_catalog'
```

```
                AND schemaname <> 'information_schema'
                AND schemaname <> 'tt'
                AND tablename NOT IN (
                        SELECT t2.table_name
                        FROM tt.tt_tables t2
                        WHERE t1.schemaname=t2.schema_name)
        LOOP
-- call the function tt.on
                EXECUTE '
                        SELECT tt.on('||quote_literal(tab_record.schemaname)
                                ||','
                                ||quote_literal(tab_record.tablename)
                                ||')';
        END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.4 tt.off(schemaname, tablename)*

```
CREATE OR REPLACE FUNCTION tt.off(text, text) RETURNS void AS $$
DECLARE
        t_name                  ALIAS FOR $2;
        t_schema                        ALIAS FOR $1;
        t_comp                          text;
        tt_table_name                   text;
        tt_table_schema                 text;
        tt_table_comp           text;
        tt_trigger_name                 text;
        tt_table_before_name            text;
        tt_table_before_comp            text;
        fk_table_name           text;
        fk_table_comp                   text;
        tt_rowid                        text;
        nuovo                           integer;

BEGIN
        t_comp := quote_ident(t_schema)||'.'||quote_ident(t_name);

        tt_table_schema := 'tt';
        tt_table_name := t_schema||'_'||t_name||'_log';
        tt_table_comp := quote_ident(tt_table_schema)||'.'||quote_ident(tt_table_name);

        tt_trigger_name         := 'tt_trigger';

        tt_table_before_name := t_schema||'_'||t_name||'_past';
        tt_table_before_comp := quote_ident(tt_table_schema)||'.'||
quote_ident(tt_table_before_name);

        fk_table_name := t_schema||'_'||t_name||'_fk';
        fk_table_comp := quote_ident(tt_table_schema)||'.'||quote_ident(fk_table_name);
```

```
	tt_rowid := t_name||'_tt_rowid_seq';

--check that TimeTravel is active on the table
	EXECUTE '
		SELECT
		CASE WHEN count(*) = 1 THEN 1
			ELSE 0
		END
		FROM tt.tt_tables
		WHERE table_name ='||quote_literal(t_name)||'
		AND schema_name = '||quote_literal(t_schema)
	INTO nuovo;

-- If yes delete all objects created for use Timetravel
	IF (nuovo=0) THEN RAISE EXCEPTION 'TT non attivo su questa tabella';
	ELSE
		EXECUTE ' ALTER EVENT TRIGGER event_trigger_for_drops DISABLE';
		EXECUTE '    DELETE FROM tt.tt_tables
					WHERE table_name ='||quote_literal(t_name)||'
					AND schema_name = '||quote_literal(t_schema);
		EXECUTE 'ALTER TABLE '||t_comp||' DROP COLUMN IF EXISTS tt_rowid';
		EXECUTE 'DROP TRIGGER IF EXISTS '||tt_trigger_name||' ON '||t_comp||'
CASCADE ';
		EXECUTE 'DROP TABLE IF EXISTS '||tt_table_comp||' CASCADE';
		EXECUTE 'DROP TABLE IF EXISTS '||tt_table_before_comp||' CASCADE';
		EXECUTE 'DROP TABLE IF EXISTS '||fk_table_comp||' CASCADE';
		EXECUTE 'DROP SEQUENCE IF EXISTS '||tt_rowid||' CASCADE';
		EXECUTE ' ALTER EVENT TRIGGER event_trigger_for_drops ENABLE';
	END IF;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.5 tt.off(schemaname)*

```
CREATE OR REPLACE FUNCTION tt.off(text) RETURNS void AS $$
DECLARE
	select_schema		ALIAS FOR $1;
	tab_record		record;

BEGIN
--For each table in selected schema
	FOR tab_record IN
		SELECT schema_name, table_name
		FROM tt.tt_tables
		WHERE schema_name=select_schema
	LOOP
-- call the function tt.off
		EXECUTE '
			SELECT tt.off('||quote_literal(tab_record.schema_name)
					||','
					||quote_literal(tab_record.table_name)
```

```
                                        ||')';
        END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.6 tt.off()*

```
CREATE OR REPLACE FUNCTION tt.off() RETURNS void AS $$
DECLARE
        tab_record record;
BEGIN
--For all user table of database
        FOR tab_record IN
                SELECT schema_name, table_name
                FROM tt.tt_tables
        LOOP
-- call the function tt.off
        EXECUTE '
                SELECT tt.off('||quote_literal(tab_record.schema_name)
                                ||','
                                ||quote_literal(tab_record.table_name)
                                ||')';
        END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.7 tt.query(schemaname, tablename, timestamp)*

```
CREATE OR REPLACE FUNCTION tt.query(text, text, timestamp ) RETURNS void AS $$
DECLARE
        t_name                  ALIAS FOR $2;
        t_schema                        ALIAS FOR $1;
        t_comp                          text;
        tt_past_table_name              text;
        tt_past_table_schema    text;
        tt_past_table_comp              text;
        tt_table_name                   text;
        tt_table_schema                 text;
        tt_table_comp                   text;
        data_tt                         ALIAS FOR $3;
        undo                            text;
        nuovo                           integer;

BEGIN
        t_comp := quote_ident(t_schema)||'.'||quote_ident(t_name);

        tt_table_schema = 'tt';
        tt_table_name = t_schema||'_'||t_name||'_log';
        tt_table_comp := quote_ident(tt_table_schema)||'.'||quote_ident(tt_table_name);
```

```
        tt_past_table_name = t_schema||'_'||t_name||'_past';
        tt_past_table_schema = tt_table_schema;
        tt_past_table_comp := quote_ident(tt_past_table_schema)||'.'||
quote_ident(tt_past_table_name);

--check that TimeTravel is active on the table
        EXECUTE '
                SELECT
                CASE WHEN count(*) = 1 THEN 1
                        ELSE 0
                END
                FROM tt.tt_tables
                WHERE table_name ='||quote_literal(t_name)||'
                AND schema_name = '||quote_literal(t_schema)
        INTO nuovo;

        IF (nuovo=0) THEN RAISE EXCEPTION 'TT non attivo su questa tabella';
        ELSE
--Create a temporary view that contains the undo to perform
                EXECUTE '
                        CREATE OR REPLACE TEMP VIEW v AS
                        SELECT *, row_number() OVER () AS row_number
                        FROM  '||tt_table_comp||'
                        WHERE tt_time >'||quote_literal(data_tt)||'
                        ORDER BY row_number DESC';

                EXECUTE '
                        DROP  TABLE IF EXISTS '||tt_past_table_comp;
-- Create the table '_past' equal to input table
                EXECUTE '
                        CREATE TABLE '||tt_past_table_comp||' AS
                        SELECT *
                        FROM  '||t_comp;
--Apply the undo stored in the temporary view to the table '_past'
                FOR undo IN
                        SELECT tt_undo
                        FROM v
                LOOP
                        undo = replace(undo, t_comp , tt_past_table_comp);
                        EXECUTE undo;
                END LOOP;
        END IF;
END;
$$ LANGUAGE plpgsql;
```

### *3.8 tt.restore(schemaname, tablename, timestamp)*

```
CREATE OR REPLACE FUNCTION tt.restore(text, text, timestamp) RETURNS void AS $$
DECLARE
        t_name          ALIAS FOR $2;
        t_schema                ALIAS FOR $1;
        t_comp                  text;
```

```
        tt_table_name           text;
        tt_table_schema         text;
        tt_table_comp           text;
        fk_table_name           text;
        fk_table_schema         text;
        fk_table_comp           text;
        data_tt                 ALIAS FOR $3;
        undo                    text;
        nuovo                   integer;
        fk                      record;
        fk_redo                 text;

BEGIN
        t_comp := quote_ident(t_schema)||'.'||quote_ident(t_name);

        tt_table_schema = 'tt';
        tt_table_name = t_schema||'_'||t_name||'_log';
        tt_table_comp := quote_ident(tt_table_schema)||'.'||quote_ident(tt_table_name);

        fk_table_schema = 'tt';
        fk_table_name = t_schema||'_'||t_name||'_fk';
        fk_table_comp := quote_ident(fk_table_schema)||'.'||quote_ident(fk_table_name);

--check that TimeTravel is active on the table
        EXECUTE '
                SELECT
                CASE WHEN count(*) = 1 THEN 1
                        ELSE 0
                END
                FROM tt.tt_tables
                WHERE table_name ='||quote_literal(t_name)||'
                AND schema_name = '||quote_literal(t_schema)
        INTO nuovo;

        IF (nuovo=0) THEN RAISE EXCEPTION 'TT non attiva su questa tabella';
        ELSE
--Create a temporary view that contains the undo to perform
                EXECUTE '
                        create OR REPLACE TEMP VIEW v AS
                        SELECT *, row_number() over () AS row_number
                        FROM  '||tt_table_comp||'
                        WHERE tt_time >'||quote_literal(data_tt)||'
                        ORDER BY row_number DESC';

                EXECUTE 'DROP TABLE IF EXISTS '||fk_table_comp;

                EXECUTE 'CREATE TABLE '||fk_table_comp||' (redo_fk text)';
--Drop any foreign keys and saving them into '_fk' table
                FOR fk IN
                        SELECT t4.oid, t1.conname, t2.relname,t2.schemaname,
pg_get_constraintdef(t4.oid) AS redo
                        FROM pg_catalog.pg_constraint t1 , pg_catalog.pg_statio_user_tables t2 ,
```

```
pg_catalog.pg_statio_user_tables t3, pg_constraint t4
                        WHERE t1.conrelid = t2.relid
                        AND t1.confrelid = t3.relid
                        AND t1.contype = 'f'
                        AND t1.conname = t4.conname
                        AND
                        (
                                (
                                        t2.relname = quote_ident(t_name)
                                        AND t2.schemaname = quote_ident(t_schema)
                                )
                                OR
                                (
                                        t3.relname = quote_ident(t_name)
                                        AND t3.schemaname = quote_ident(t_schema)
                                )
                        )
                LOOP
                        fk_redo = 'ALTER TABLE '||fk.schemaname||'.'||fk.relname||' ADD
CONSTRAINT '||fk.conname||' '||fk.redo||';';

                        EXECUTE '
                                insert into '||fk_table_comp||'
                                values ('||quote_literal(fk_redo)||')';

                        EXECUTE '
                                ALTER TABLE '||fk.schemaname||'.'||fk.relname||'
                                DROP CONSTRAINT '||fk.conname;
                END LOOP;

--Apply the undo stored in the temporary view to the table 'tablename'
                FOR undo IN
                        SELECT tt_undo
                        FROM v
                LOOP
                        EXECUTE undo;
                END LOOP;
        END IF;
END;
$$ LANGUAGE plpgsql VOLATILE;
```

### *3.9 tt.restore(schemaname, timestamp)*

```
CREATE OR REPLACE FUNCTION tt.restore(text, timestamp) RETURNS void AS $$
DECLARE
        select_schema           ALIAS FOR $1;
        data_tt                 ALIAS FOR $2;
        tab_record              record;
BEGIN
--For each table in selected schema
        FOR tab_record IN
                SELECT schema_name, table_name
```

```
                FROM tt.tt_tables
                WHERE schema_name=select_schema
        LOOP
-- call the function tt.restore
                EXECUTE '
                        SELECT tt.restore('    ||quote_literal(tab_record.schema_name)||','
                                               ||quote_literal(tab_record.table_name)||','
                                               ||quote_literal(data_tt)||')';
        END LOOP;
        RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.10 tt.restore(timestamp)*

```
CREATE OR REPLACE FUNCTION tt.restore(timestamp) RETURNS void AS $$
DECLARE
        data_tt         ALIAS FOR $1;
        tab_record      record;
BEGIN
--For all user table of database
        FOR tab_record IN
                SELECT schema_name, table_name
                FROM tt.tt_tables
        LOOP
-- call the function tt.restore
                EXECUTE '
                        SELECT tt.restore('    ||quote_literal(tab_record.schema_name)||','
                                               ||quote_literal(tab_record.table_name)||','
                                               ||quote_literal(data_tt)||')';
        END LOOP;
RETURN;
END;
$$ LANGUAGE plpgsql;
```

### *3.11 tt.rec()*

```
CREATE OR REPLACE FUNCTION tt.rec() RETURNS trigger AS $$
DECLARE
        t_name                  TEXT            =tg_argv[0];
        t_schema                TEXT            =tg_argv[1];
        t_comp                  TEXT;
        tt_table_name TEXT;
        tt_table_schema         TEXT;
        tt_table_comp           TEXT;
        tt_mode                 TEXT            =TG_OP;
        tt_tuple                TEXT;
        tt_time                 TIMESTAMP =statement_timestamp();
        tt_user         TEXT            =CURRENT_USER;
        tt_redo                 TEXT            =current_query();
        tt_undo                 TEXT;
        col_record              RECORD;
```

```
        col_name1           TEXT;
        col_name2           TEXT;
        ins_colname         text            = '( ';
        ins_val             text            = '( ';
        undo1               text            = ' ';
        undo2               text            = '( ';

BEGIN
        tt_table_schema = 'tt';
        tt_table_name = t_schema||'_'||t_name||'_log';

        t_comp := quote_ident(t_schema)||'.'||quote_ident(t_name);
        tt_table_comp := quote_ident(tt_table_schema)||'.'||quote_ident(tt_table_name);

--Case Update
        IF(tt_mode='UPDATE') THEN
                tt_tuple = 'NEW';
                tt_undo='UPDATE '||t_comp||' SET ';

 --Retrieve the columns names of the table
                FOR col_record IN
                        SELECT ordinal_position, column_name, data_type
                        FROM information_schema.columns
                        WHERE table_schema = quote_ident(TG_TABLE_SCHEMA)
                        AND table_name = quote_ident(TG_TABLE_NAME)
                        ORDER BY ordinal_position
                LOOP
-- Retrieve new and old values
                        EXECUTE '
                                SELECT ($1).' || col_record.column_name || '::text'
                        INTO col_name1 USING OLD;
                        EXECUTE '
                                SELECT ($1).' || col_record.column_name || '::text'
                        INTO col_name2 USING NEW;
--Create UNDO string
                        IF(col_record.ordinal_position>1) THEN
                                ins_val:=ins_val||' , ';
                                undo1:=undo1||' , ';
                                undo2:=undo2||' AND ';
                                ins_colname:=ins_colname||' , ';
                        END IF;

                        ins_colname:=ins_colname||col_record.column_name;

                        IF(col_name1 IS NOT NULL) THEN
                                undo1:=undo1||col_record.column_name||' = '||
quote_literal(col_name1);
                        ELSE
                                undo1:=undo1||col_record.column_name||' = NULL';
                        END IF;

                        IF(col_name2 IS NOT NULL) THEN
```

```
                undo2:=undo2||col_record.column_name||' = '||
quote_literal(col_name2);
                ins_val:=ins_val||quote_literal(col_name2);
            ELSE
                undo2:=undo2||col_record.column_name||' IS NULL';
                ins_val:=ins_val||'NULL';
            END IF;
        END LOOP;

        tt_undo:=tt_undo||undo1||' WHERE '||undo2||');';
    END IF;

--Case Insert
    IF(tt_mode='INSERT') THEN
        tt_tuple = 'NEW';
        tt_undo='DELETE FROM '||t_comp||' WHERE ( ';

--Retrieve the columns names of the table
        FOR col_record IN
            SELECT ordinal_position, column_name, data_type
            FROM information_schema.columns
            WHERE
            table_schema = quote_ident(TG_TABLE_SCHEMA)
            AND table_name = quote_ident(TG_TABLE_NAME)
            ORDER BY ordinal_position
        LOOP
-- Retrieve new values
            EXECUTE '
                SELECT ($1).' || col_record.column_name || '::text'
            INTO col_name1 USING NEW;
--Create UNDO string
            IF(col_record.ordinal_position>1) THEN
                tt_undo:=tt_undo||' AND ';
                ins_val:=ins_val||' , ';
                ins_colname:=ins_colname||' , ';
            END IF;

            ins_colname:=ins_colname||col_record.column_name;

            IF(col_name1 IS NOT NULL) THEN
                tt_undo:=tt_undo||col_record.column_name||' = '||
quote_literal(col_name1);
                ins_val:=ins_val||quote_literal(col_name1);

            ELSE
                tt_undo:=tt_undo||col_record.column_name||' IS NULL ';
                ins_val:=ins_val||'NULL';
            END IF;
        END LOOP;

        tt_undo:=tt_undo||' );';
    END IF;
```

```
--Case Delete
        IF(tt_mode='DELETE') THEN
                tt_tuple = 'OLD';
                tt_undo='INSERT INTO '||t_comp;

--Retrieve the columns names of the table
                FOR col_record IN
                        SELECT ordinal_position, column_name, data_type
                        FROM information_schema.columns
                        WHERE table_schema = quote_ident(TG_TABLE_SCHEMA)
                        AND table_name = quote_ident(TG_TABLE_NAME)
                        ORDER BY ordinal_position
                LOOP
-- Retrieve old values
                        EXECUTE '
                                SELECT ($1).' || col_record.column_name || '::text'
                        INTO col_name1 USING OLD;
--Create UNDO string
                        IF(col_record.ordinal_position>1) THEN
                                ins_val:=ins_val||' , ';
                                ins_colname:=ins_colname||' , ';
                        END IF;

                        ins_colname:=ins_colname||col_record.column_name;

                        IF(col_name1 IS NOT NULL) THEN
                                ins_val:=ins_val||quote_literal(col_name1);
                        ELSE
                                ins_val:=ins_val||'NULL';
                        END IF;
                END LOOP;

                tt_undo:=tt_undo||ins_colname||') VALUES'||ins_val||');';
        END IF;

-- Insert the information obtained in the log table
        EXECUTE '
        INSERT INTO '
        || tt_table_comp
        || ins_colname
        ||' ,tt_mode, tt_tuple, tt_user, tt_time, tt_undo, tt_redo) VALUES '
        || ins_val||','
        || quote_literal(tt_mode)||','
        || quote_literal(tt_tuple)||','
        || quote_literal(tt_user)||','
        || quote_literal(tt_time)||','
        || quote_literal(tt_undo)||','
        || quote_literal(tt_redo)
        || ')';

RETURN NEW;
```

```
END;
$$ LANGUAGE plpgsql;
```

### *3.12 tt.tt_dropped()*

```
CREATE or replace FUNCTION tt.tt_dropped() RETURNS event_trigger AS $$
DECLARE
        obj                             record;
        t_name                  text;
        t_schema                        text;
        t_comp                          text;
        tt_table_name                   text;
        tt_table_schema                 text;
        tt_table_comp           text;
        tt_table_before_name text;
        tt_table_before_comp            text;
        fk_table_name           text;
        fk_table_comp                   text;
        tt_rowid                        text;
        nuovo                           integer;

BEGIN
--For each object deleted
        FOR obj IN
                SELECT * FROM pg_event_trigger_dropped_objects()
        LOOP
-- If the object is a table
                IF(obj.object_type='table') THEN
                        EXECUTE '
                                SELECT
                                CASE WHEN count(*) = 1 THEN 1
                                        ELSE 0
                                END
                                FROM tt.tt_tables
                                WHERE table_name ='||quote_literal(obj.object_name)||'
                                AND schema_name = '||quote_literal(obj.schema_name)
                        INTO nuovo;
-- If on the table was turned on Timetravel
                        IF (nuovo<>0) THEN
                                t_name = obj.object_name;
                                t_schema = obj.schema_name;
                                t_comp := quote_ident(t_schema)||'.'||quote_ident(t_name);

                                tt_table_schema = 'tt';
                                tt_table_name = t_schema||'_'||t_name||'_log';
                                tt_table_comp := quote_ident(tt_table_schema)||'.'||
quote_ident(tt_table_name);

                                tt_table_before_name := t_schema||'_'||t_name||'_past';
                                tt_table_before_comp := quote_ident(tt_table_schema)||'.'||
quote_ident(tt_table_before_name);
```

```
                                fk_table_name := t_schema||'_'||t_name||'_fk';
                                fk_table_comp := quote_ident(tt_table_schema)||'.'||
quote_ident(fk_table_name);

                                tt_rowid :=     t_name||'_tt_rowid_seq';
--Drop all objects created for use Timetravel
                                EXECUTE ' ALTER EVENT TRIGGER event_trigger_for_drops
DISABLE';
                                EXECUTE '   DELETE FROM tt.tt_tables
                                                WHERE table_name ='||
quote_literal(t_name)||'
                                                AND schema_name = '||
quote_literal(t_schema);
                                EXECUTE 'DROP TABLE IF EXISTS '||tt_table_comp||'
CASCADE';
                                EXECUTE 'DROP TABLE IF EXISTS '||tt_table_before_comp||'
CASCADE';
                                EXECUTE 'DROP TABLE IF EXISTS '||fk_table_comp||'
CASCADE';
                                EXECUTE 'DROP SEQUENCE IF EXISTS '||tt_rowid;
                                EXECUTE ' ALTER EVENT TRIGGER event_trigger_for_drops
ENABLE';
                        END IF;
                END IF;
        END LOOP;
END
$$ LANGUAGE plpgsql;

CREATE EVENT TRIGGER event_trigger_for_drops
        ON sql_drop
        EXECUTE PROCEDURE tt.tt_dropped();
```

# 4 PHP APPLICATION

### *4.1 index.php*

```
<?php

include_once('funzioni.php');
ini_set('display_errors','Off');
$db=connection_pgsql() or die('Connessione al DBMS non riuscita');

print('<html>');
print('<head>');
        print('<title>TimeTravel </title>');
print('</head>');
print('<body>');
print('<div align="center">');
        print('<img  src="immagini/titolo.png"  width ="30%" />');
print('</div>');
print('<br><br><br><br>');
```

```
//CASE ACTIVE ON SINGLE TABLE
if($_POST['function']=='attiva')
{
	$_table = $_POST['table'];
	$_schema = $_POST['schema'];
	$sql="SELECT tt.on('$_schema','$_table')";
	pg_query($db, $sql);
}
//CASE DISABLE ON SINGLE TABLE
if($_POST['function']=='disattiva')
{
	$_table = $_POST['table'];
	$_schema = $_POST['schema'];
	$sql="SELECT tt.off('$_schema','$_table')";
	pg_query($db, $sql);
}
//CASE ACTIVE ON ALL TABLES OR MORE
if($_POST['function']=='attiva_all')
{
	if($_POST['rest_schema']=='all')
	{
		$sql="SELECT tt.on()";
		pg_query($db, $sql);
	}
	else
	{
		$_schema = $_POST['rest_schema'];
		$sql="SELECT tt.on('$_schema')";
		pg_query($db, $sql);
	}
}
//CASE DISABLE ON ALL TABLES OR MORE
if($_POST['function']=='disattiva_all')
{
	if($_POST['rest_schema']=='all')
	{
		$sql="SELECT tt.off()";
		pg_query($db, $sql);
	}
	else
	{
		$_schema = $_POST['rest_schema'];
		$sql="SELECT tt.off('$_schema')";
		pg_query($db, $sql);
	}
}
//SHOW ALL USER TABLES OF DB
$sql=" SELECT t.table_name as table, t.table_schema as schema,
		CASE WHEN t.table_name in (
			SELECT table_name
			FROM tt.tt_tables c
			WHERE t.table_schema = c.schema_name)
```

```
                THEN 'si'
                ELSE 'no'
                END as attivo
                FROM information_schema.tables t
                WHERE t.table_schema NOT IN ('pg_catalog','information_schema','tt')
                ORDER BY table_schema,table_name";
$resource=pg_query($db, $sql);
$numcol = pg_num_fields($resource);
$numrows=pg_num_rows($resource);
$posizione=0;

if($numrows==0)
{
        print('<br><h2>Non risulta installata la TimeTravel</h2><br>');
}
//CREATE THE TABLE TO MANAGE TT
else
{
        print('<table border="1" rules="rows" width="90%" align="center">');
                print('<tr align="center">');
                        print('<th  bgcolor="silver">');
                                print('Tabella');
                        print('</th>');
                        print('<th  bgcolor="silver">');
                                print('Schema');
                        print('</th>');
                        print('<th  bgcolor="silver">');
                                print('Mostra Dati');
                        print('</th>');
                        print('<th  bgcolor="silver">');
                                print('TimeTravel attiva?');
                        print('</th>');
                        print('<th bgcolor="silver">');
                                print('Attiva TimeTravel');
                        print('</th>');
                        print('<th  bgcolor="silver">');
                                print('Disattiva TimeTravel');
                        print('</th>');
                        print('<th  bgcolor="silver">');
                                print('Usa TimeTravel');
                        print('</th>');
                print('</tr>');
                for($i=0;$i<$numrows;$i++)
                {
                        $row=pg_fetch_array($resource, NULL, PGSQL_BOTH);
                        $posizione++;

                        print('<tr align="center">');
                                print('<td     >');
                                        print(".($row[table]).");
                                print('</td>');
                                print('<td     >');
```

```
                                        print(".($row[schema]).");
                                print('</td>');
                                print('<td valign="middle"  >');
                                        print('
                                        <br>
                                        <form method="post" action=index.php>
                                                <input name="schema" type="hidden" value="'.
$row[schema].'">
                                                <input name="table" type="hidden" value="'.
$row[table].'">
                                                <button name="function" type="submit"
value="showtable" >
                                                        Guarda
                                                </button>
                                        </form>
                                        ');
                                print('</td>');
                                print('<td     >');
                                        print(".($row[attivo]).");
                                print('</td>');
                                print('<td   >');
                                        if($row[attivo]=='no')
                                        {
                                                print('
                                                <br>
                                                <form method="post" action=index.php>
                                                        <input name="schema" type="hidden"
value="'.$row[schema].'">
                                                        <input name="table" type="hidden" value="'.
$row[table].'">
                                                        <button name="function" type="submit"
value="attiva">
                                                                Attiva
                                                        </button>
                                                </form>
                                                ');
                                        }
                                print('</td>');
                                print('<td    >');
                                        if($row[attivo]=='si')
                                        {
                                                print('
                                                <br>
                                                <form method="post" action=index.php>
                                                        <input name="schema" type="hidden"
value="'.$row[schema].'">
                                                        <input name="table" type="hidden" value="'.
$row[table].'">
                                                        <button name="function" type="submit"
value="disattiva">
                                                                Disattiva
                                                        </button>
```

```
                                        </form>
                                        ');
                                }
                        print('</td>');
                        print('<td    >');
                                if($row[attivo]=='si')
                                {
                                        print('
                                        <br>
                                        <form method="post" action=tt.php>
                                                <input name="schema" type="hidden"
value="'.$row[schema].'">
                                                <input name="table" type="hidden" value="'.
$row[table].'">
                                                <button name="function" type="submit"
value="home">
                                                        Usa
                                                </button>
                                        </form>
                                        ');
                                }
                        print('</td>');
                print('</tr>');
        }
print('</table>');

print('<table align="center">');
        print('<tr>');
                print('<td align="center">');
                        print('<br>Per ogni tabella dello schema:<br>');
                        print('<form method="post" action=index.php>');
                                print '<select name="rest_schema">';
                                        $db=connection_pgsql() or die ('Connessione al
DBMS non riuscita');
                                        $sql=" SELECT        distinct schemaname
                                                FROM pg_statio_user_tables
                                                WHERE schemaname<>'tt'
                                                ORDER BY schemaname";
                                        $resource=pg_query($db,$sql);
                                        $numrows = pg_num_rows($resource);
                                        print('<option value="all">tutti');
                                        for($i=0;$i<$numrows;$i++)
                                        {

$row=pg_fetch_array($resource,NULL,PGSQL_BOTH);
                                                print('<option value="'.
$row['schemaname'].'">'.htmlentities($row['schemaname']).'</option>');
                                        }
                                print '</select>';
                                print('<br>
                                <button name="function" type="submit" value="attiva_all">
                                        Attiva TimeTravel
```

```
                                        </button>
                                        <button name="function" type="submit"
value="disattiva_all">
                                                Disattiva TimeTravel
                                        </button>');
                                print '</form>';
                        print('</td>');
                print('</tr>');
        print('</table>');
}

//CASE SHOW TABLE
if($_POST['function']==showtable)
{
$_table = ($_POST['schema']).'.'.($_POST['table']);
print('<br><h2>Valori correnti della tabella '.$_table.': </h2>');

$sql="SELECT * FROM $_table";
$resource=pg_query($db, $sql);
$numcol = pg_num_fields($resource);
$numrows=pg_num_rows($resource);

print('<table border="1" rules="rows"  width="90%" align="center"   >');
        print('<tr>');
                for($i=0;$i<$numcol;$i++)
                {
                        print('<th bgcolor="silver">');
                                print(".(pg_field_name($resource,$i)).");
                        print('</th>');
                }
        print('</tr>');
        for($i=0;$i<$numrows;$i++)
        {
                $row=pg_fetch_array($resource, NULL, PGSQL_BOTH);
                print('<tr>');
                        for($a=0;$a<$numcol;$a++)
                        {
                                print('<td align="center" width="'.(100/$numcol).'%">');
                                        print(".($row[$a]).");
                                print('</td>');
                        }
                print('</tr>');
        }
print('</table>');
}
pg_free_result($resource);
pg_close($db);

print('<br><br>');
print('</body>');
print('</html>');
?>
```

### *4.2 tt.php*

```
<?php

include_once('funzioni.php');
ini_set('display_errors','Off');
$db=connection_pgsql() or die('Connessione al DBMS non riuscita');

print('<html>');
print('<head>');
	print('<title>');
		print('TimeTravel della tabella '.$_POST[schema].'.'.$_POST[table]);
	print('</title>');
print('</head>');
print('<body>');
print('<table width="100%">');
	print('<tr>');
		print('<br>');
	print('</tr>');
	print('<tr>');
		print('<td valign="top" width="15%">');
			print('<table width="100%">');
				print('<tr>');
//BUTTONS ON THE LEFT
					print('<td valign="top" height="75">');
						print('
						<form method="post" action=tt.php>
							<button name="function" type="submit"
value="showtable">
								<img
src="immagini/bottoni/correnti.png"  width ="100%" />
							</button>
						</form>
						');
					print('</td>');
				print('</tr>');
				print('<tr>');
					print('<td valign="top" height="75">');
						print('
						<form method="post" action=tt.php>
							<button name="function" type="submit"
value="showlog">
								<img  src="immagini/bottoni/log.png"
width ="100%" />
							</button>
						</form>
						');
					print('</td>');
				print('</tr>');
				print('<tr>');
					print('<td valign="top" height="75">');
						if(($_POST['function']!
```

```
=query1)and($_POST['function']!=query)){
                                                print('
                                                <form method="post" action=tt.php>
                                                        <button name="function"
type="submit" value="query1">
                                                                <img
src="immagini/bottoni/query.png"  width ="100%" />
                                                        </button>
                                                </form>
                                                ');
                                        }
                                        else
                                        {
                                                print('
                                                        <form method="post" action=tt.php>
                                                        <input name="data" type="date"
value="'.$_POST[data].'".>
                                                        <input name="ora" type="time"
value="'.$_POST[ora].'">
                                                        <button name="function"
type="submit" value="query">
                                                                ok
                                                        </button>
                                                        <button >
                                                                <
                                                        </button>
                                                </form>
                                                ');
                                        }
                                print('</td>');
                        print('</tr>');
                        print('<tr>');
                                print('<td valign="top" height="75">');
                                        if(($_POST['function']!
=restore1)and($_POST['function']!=restore)){
                                                print('
                                                <form method="post" action=tt.php>
                                                        <button name="function"
type="submit" value="restore1">
                                                                <img
src="immagini/bottoni/riporta.png"  width ="100%" />
                                                        </button>
                                                </form>
                                                ');
                                        }
                                        else
                                        {
                                                print('
                                                <form method="post" action=tt.php>
                                                        <input name="data" type="date"
value="'.$_POST[data].'".>
                                                        <input name="ora" type="time"
```

```
value="'.$_POST[ora].'">
                                                <button name="function"
type="submit" value="restore">
                                                        ok
                                                </button>
                                                <button >
                                                        <
                                                </button>
                                        </form>
                                        ');
                                }
                        print('</td>');
                print('</tr>');
                print('<tr>');
                        print('<td valign="top" height="75">');
                                print('
                                <form method="post" action=index.php>
                                        <input name="schema" type="hidden"
value="'.$_SESSION['schema'].'">
                                        <input name="table" type="hidden" value="'.
$_SESSION['table'].'">
                                        <button name="function" type="submit"
value="disattiva">
                                                <img
src="immagini/bottoni/disattiva.png"  width ="100%" />
                                        </button>
                                </form>
                                ');
                        print('</td>');

                print('</tr>');
                print('<tr>');
                        print('<td valign="top" height="75">');
                                print('
                                <form method="post" action=index.php >
                                        <button >
                                                <img
src="immagini/bottoni/torna.png"  width ="100%" />
                                        </button>
                                </form>
                                ');
                        print('</td>');
                print('</tr>');
        print('</table>');
print('</td>');
//INFORMATION ON THE RIGHT
        print('<td valign="top">');

//INITIAL CASE
        if($_POST['function']==home)
        {
                $_SESSION['table'] = $_POST['table'];
```

```
                        $_SESSION['schema'] = $_POST['schema'];
                }

//CALL tt.restore FUNCTION
                if($_POST['function']==restore)
                {
                        $_table = $_SESSION['table'];
                        $_schema = $_SESSION['schema'];
                        $_data = date("d-m-Y", strtotime($_POST['data']));
                        $_timestamp = ($_data).' '.($_POST['ora']);
                        $sql="SELECT tt.restore('$_schema', '$_table', '$_timestamp')";
                        $resource=pg_query($db, $sql);
                }

                if(     ($_POST['function']==showtable)OR
                        ($_POST['function']==home)OR
                        ($_POST['function']==query1)OR
                        ($_POST['function']==restore1)OR
                        ($_POST['function']==restore))
                {
                        $_table = ($_SESSION['schema']).'.'.($_SESSION['table']);
                        print('<h2 align="center">Valori correnti della tabella '.$_table.':
</h2><br><br>');

                        $sql="SELECT * FROM $_table";
                        $resource=pg_query($db, $sql);
                        $numcol = pg_num_fields($resource);
                        $numrows=pg_num_rows($resource);

                        print('<table border="1" rules="rows"  width="90%" align="center"   >');
                                print('<tr>');
                                        for($i=0;$i<$numcol;$i++)
                                        {
                                                print('<th bgcolor="silver">');
                                                        print(".(pg_field_name($resource,$i)).");
                                                print('</th>');
                                        }
                                print('</tr>');
                                for($i=0;$i<$numrows;$i++)
                                {
                                        $row=pg_fetch_array($resource, NULL, PGSQL_BOTH);
                                        print('<tr>');
                                                for($a=0;$a<$numcol;$a++)
                                                {
                                                        print('<td align="center" width="'.
(100/$numcol).'%">');
                                                                print(".($row[$a]).");
                                                        print('</td>');
                                                }
                                        print('</tr>');
                                }
                        print('</table>');
```

```
		}
//SHOW LOG TABLE
		if($_POST['function']==showlog)
		{
			$_table = ($_SESSION['schema']).'.'.($_SESSION['table']);
			print('<h2 align="center">Log della tabella '.$_table.': </h2><br><br>');
			$_table = 'tt.'.$_SESSION['schema'].'_'.$_SESSION['table'].'_log';

			$sql="SELECT * FROM $_table";
			$resource=pg_query($db, $sql);
			$numcol = pg_num_fields($resource);
			$numrows=pg_num_rows($resource);

			print('<table border="1" rules="groups"  width="90%" align="center"   >');
				print('<tr>');
					for($i=0;$i<$numcol;$i++)
					{
						if((pg_field_name($resource,
$i)=='tt_undo')OR(pg_field_name($resource,$i)=='tt_redo'))
						{
							$undo = $i;
							print('<tr >');
							print('<th colspan='.($numcol -2).'
bgcolor="silver"     >');
						}
						else
						{
							print('<th bgcolor="silver"     >');
						}

								print(".(pg_field_name($resource,
$i)).");
							print('</th>');

						if((pg_field_name($resource,
$i)=='tt_undo')OR(pg_field_name($resource,$i)=='tt_redo'))
						{
							print('</tr>');
						}
					}
				print('</tr>');

				$cont = 0;
				for($i=0;$i<$numrows;$i++)
				{
					$row=pg_fetch_array($resource, NULL, PGSQL_BOTH);
					$cont++;
					print('<tbody>');
						print('<tr align="center">');
						for($a=0;$a<$numcol;$a++)
						{
							if(($undo==$a)OR($undo==$a+1))
```

```
                                                {
                                                        print('</tr>');
                                                        print('<tr align="center">');
                                                                print('<td colspan='.($numcol
-2).'>');
                                                }
                                                else
                                                {
                                                                print('<td >');
                                                }
                                                print(".($row[$a]).");
                                                                print('</td>');
                                        }
                                        print('</tr>');
                                print('</tbody>');
                        }
                print('</table>');
        }
//CALL tt.query FUNCTION
        if($_POST['function']==query)
        {
                $_table = $_SESSION['table'];
                $_schema = $_SESSION['schema'];
                $_query = 'tt.'.$_SESSION['schema'].'_'.$_SESSION['table'].'_past';
                $_data = date("d-m-Y", strtotime($_POST['data']));
                $_timestamp = ($_data).' '.($_POST['ora']);

                print('<h2 align="center">Valori nella tabella '.$_schema.'.'.$_table.' alle ore
'.$_POST['ora'].' del giorno '.$_data.': </h2><br><br>');

                $sql="SELECT tt.query('$_schema', '$_table', '$_timestamp')";
                $resource=pg_query($db, $sql);
                $sql="SELECT * FROM $_query";
                $resource=pg_query($db, $sql);
                $numcol = pg_num_fields($resource);
                $numrows=pg_num_rows($resource);

                print('<table border="1" rules="rows"  width="90%" align="center">');
                        print('<tr>');
                        for($i=0;$i<$numcol;$i++)
                        {
                                print('<th bgcolor="silver">');
                                        print(".(pg_field_name($resource,$i)).");
                                print('</th>');
                        }
                        print('</tr>');
                        for($i=0;$i<$numrows;$i++)
                        {
                                $row=pg_fetch_array($resource, NULL, PGSQL_BOTH);
                                print('<tr>');
                                        for($a=0;$a<$numcol;$a++)
                                        {
```

```
                                                    print('<td align="center" width="'.
(100/$numcol).'%">');
                                                        print(".($row[$a]).");
                                                    print('</td>');
                                                }
                                            print('</tr>');
                                        }
                                    print('</table>');
                                }
                                print('</td>');
                        print('</tr>');
print('</table>');

print('<br>');
print('</body>');
print('</html>');
?>
```

### *4.3 conf.php*

```
<?php
define ("myhost", " ");
define ("myuser", " ");
define ("mypsw", " ");
define ("mydb", " ");
?>
```

### *4.4 funzioni.php*

```
<?php

function connection_pgsql()
{
   include_once("conf.php");
   $connection = "host=".myhost." dbname=".mydb." user=".myuser." password=".mypsw;
   return pg_connect ($connection);
}
?>
```