"stopwatch" code-kit

Because you're reading this, you will already have unzipped *stopwatch.zip* on any convenient directory and changed into its top directory.

Confirm that "ls -1" now shows this:

```
0.sql
README.pdf
cr-cross-session-stopwatch.sql
cr-duration-as-text.sql
cr-stopwatch.sql
test-cross-session-stopwatch.sql
test-cross-session-stopwatch.txt
test-duration-as-text-0.txt
test-duration-as-text.sql
test-duration-as-text.txt
test-stopwatch-0.txt
test-stopwatch.sql
test-stopwatch.txt
```

The scripts install the code and the test results that are explained in the section *Case study—implementing a stopwatch with SQL* within the overall *Date and time data types* section in the YSQL documentation.

Simply start *ysqlsh*, connecting as an ordinary non-Superuser, say *utils*, and (ideally) create a schema, say *utils_s*, in which this user can create and drop objects at will without interfering with any other projects that the database you're connected to might be hosting. Ensure that *"show search_path;"* returns *utils_s* as first in the path.

One of the tests, implemented in *test-cross-session-stopwatch.sql*, includes these two metacommands:

```
\c demo u1
\c demo u1
```

It doesn't create objects. Rather, it simply invokes *pg_sleep()* after starting each new session to demonstrate that the client-side variable *stopwatch_s0* does indeed survive and preserve its value across session boundaries as promised.

Then invoke the script *0.sql*. You can do this time and again, from a cold start or after having already run it. It will always finish silently in a few seconds. (The time is consumed by *pg_sleep()* invocations whose purpose is to give the stopwatch tests sensibly long durations to time.)

Each spool file that the tests produce has a partner reference copy whose name has *"-0"* appended, like this:

```
test-duration-as-text-0.txt
```

Check with your favorite GUI *diff* application that each of your spool files is close-to pairwise identical with its reference copy. There are bound to be differences because of randomly occurring variations in the reported durations.

Any database user that you intend to be able to use the stopwatch code should have the *utils_s* schema in its path, should have the usage privilege on the *utils_s* schema and the *execute* privilege on each of the code units.

If you intend users to be able to do cross-session timings, then you should define these variables (to act as shortcuts) in the *psqlrc* file (on the *postgres/etc directory* under the directory where you've installed the YugabyteDB client code).

```
\set start_stopwatch 'select extract(epoch from clock_timestamp())::text as s0 \\gset stopwatch_'
\set stopwatch_reading 'select stopwatch_reading(:stopwatch_s0);'
```