COMP3311 22T1 Database Systems

# Assignment 1
## Checking Script

Last updated: **Sunday 27th February 3:07pm**

Most recent changes are shown in red ... older changes are shown in brown.

[Assignment Spec] [Database Design] [Schema in SQL] **[check1.sql]** [Examples] [Fixes+Updates]

An SQL script is available to assist with testing:

```
/web/cs3311/22T1/assignments/ass1/check1.sql
```

Warning: this script creates a bunch of functions and tables in your MyMyUNSW database. The functions are called `ass1_XXX` or `check_qXX` and the tables are called `qXX_expected`. If you have any functions or tables with names like these, you will need to edit the `check1.sql` file and change the names to avoid name clashes.

In order to use the testing framework, first load the script into your MyMyUNSW database via e.g.

```
$ cd /localstorage/YourZid
$ cp /web/cs3311/22T1/assignments/ass1/check1.sql .
$ psql mymy -f check1.sql
```

or, alternatively,

```
$ psql mymy -f /web/cs3311/22T1/assignments/ass1/check1.sql
```

If you're working on your home machine, download the `check1.sql` onto your home machine and load it from there.

The first time you do this, you'll receive a bunch of notices like:

```
CREATE FUNCTION
CREATE FUNCTION
CREATE FUNCTION
CREATE FUNCTION
CREATE FUNCTION
CREATE FUNCTION
psql:check.sql:143: NOTICE:  type "testingresult" does not exist, skipping
DROP TYPE
CREATE TYPE
CREATE FUNCTION
CREATE FUNCTION
psql:check.sql:185: NOTICE:  table "q1_expected" does not exist, skipping
DROP TABLE
CREATE TABLE
COPY 16
... plus a lot more lines ...
```

The `NOTICE`s are harmless, and will not appear if you reload the script. If you get any `ERROR` messages, let us know.

Once the functions and tables are loaded, you can run tests by invoking the checking functions, e.g.

```
mymy=# select check_q1();
  check_q1
-------------
 q1: correct
(1 row)

mymy=# select check_q6b();
  check_q6b
--------------
 q6b: correct
(1 row)
```

If a test fails, you can check the expected results via, e.g.

```
mymy=# select * from q2_expected;
 unswid  |       name       | course_cnt
---------+------------------+------------
 9511757 | Rochayah Machali |         39
(1 row)

mymy=# select * from q9a_expected;
 objtype | objcode
---------+---------
 stream  | BINFA1
(1 row)
```

Note that the checks use result *sets*, so the order of tuples does not matter.

If you want to run all of the tests, you can do it via:

```
mymy=# select * from check_all();
 test |                 result
------+----------------------------------------
 q1   | q1: correct
 q2   | q2: correct
 q3   | q3: correct
 q4   | q4: correct
 q5a  | q5a: correct
 q5b  | q5b: correct
 q6a  | No q6 function; did it load correctly?
 q6b  | No q6 function; did it load correctly?
 q6c  | No q6 function; did it load correctly?
 q7a  | q7a: incorrect result tuples
 q7b  | q7b: correct
 q7c  | q7c: correct
 q8a  | q8a: incorrect result tuples
 q8b  | q8b: incorrect result tuples
 q8c  | q8c: incorrect result tuples
 q9a  | q9a: correct
 q9b  | q9b: correct
 q9c  | q9c: correct
 q10a | q10a: correct
 q10b | q10b: correct
```

```
 q10c | q10c: correct
(21 rows)
```

This may take a while, especially if some of your queries are slow.

Note that it checks for missing functions and also gives feedback about incorrect results. It has messages other than `incorrect result tuples`; it also checks for missing or extraneous result tuples. If you have errors, you can always look at the expected results and compare them manually to your results. If you think that any of our expected results tables are incorrect, let us know and we can make a new version of `check1.sql` that you can reload.

The `check2.sql` script can be derived by replacing "check1" with "check2" in the path. For instance, if you are using the server, the path of `check2.sql` is **/web/cs3311/22T1/assignments/ass1/check2.sql**. If you are using your home machine, `check2.sql` can be downloaded here.