Interactive - 14 - data types

This is really the stuff from chapter 3 in the book.

When we create a table, (Already created in the last homework when you ran the file)

```
CREATE TABLE us_state (
    state_id UUID NOT NULL DEFAULT uuid_generate_v4() primary key,
    state_name text NOT NULL,
    state char varying (2) NOT NULL,
    FIPS_code char varying (2) NOT NULL,
    area_rank int not null default 9999,
    area_sq_mi numeric not null default 0
);
```

we specify that the columns have a "type" and constraints on the data.

The most common set of types are (ordered from my own code on how frequently I use each type):

Type	Description
text	a string of 0 or more characters.
varchar(n)	a string of 0 to `n` characters.
uuid	a UUID.
date	a date - not a time.
time	a time value.
bigint, int8	8 byte integer
timestamp	a date and a time - usually with a time zone.
numeric	a large number with (think 130k digits, before/after decimal point)
decimal	a large number with (think 130k digits, before/after decimal point)
float	4 byte - a floating point number with an exponent
real	4 byte - a floating point number with an exponent
json, jsonb	JSON data
double precision	8 byte - a floating point number with an exponent
bit, varbit	variable length bit string values
serial	a 4 byte auto-generated integer that counts up
interval	a time interval can be +/- 178million years
bytea	byte array - binary data
boolean	true/false
point, line	geometric data
box, path	geometric data
polygon, cirlce	geometric data
tvector, tsquery	full text search type
inet, cidr	interget address tyeps, IPv4 122.4.3.22 or IPV6
user defined tyeps	The type system is extensible!

Also there is a Geographic Information Package for PostgreSQL that is very good. It is built on top of the geometric data types and GIN indexes.

We are not going to use all of these types. Just a few of the most common ones like timestamp.

Often we want a column to show when a new row is inserted or updated in a table. Let's add these to our name_list table.

```
ALTER TABLE name_list add updated timestamp ;
ALTER TABLE name_list add created timestamp
    default current_timestamp not null  ;
```

To make the 'updated' column work we need have PostgreSQL run some code when an update happens. This kind of a thing is called a "trigger".

Let's add a trigger to the table.
Run the file `hw14_2.sql`. The contents of the file is shown below.

```
CREATE OR REPLACE function name_list_upd()
RETURNS trigger AS
$$
BEGIN
  NEW.updated := current_timestamp;
  RETURN NEW;
END
$$
LANGUAGE 'plpgsql';


CREATE TRIGGER name_list_trig
    BEFORE update ON name_list
    FOR EACH ROW
    EXECUTE PROCEDURE name_list_upd();
```

Now when we do an update on this table it will set the updated field to the current time.

FilesToRun: hw14_2.sql