Interactive - 11 - add a index on the name table that is case insensitive.

Let's address the case sensitive problem first.

We can create indexes on functions of data. This allows us to have an index where we convert the upper/lower case to all lower case and then index this result.

```
CREATE INDEX user_real_name_ci_idx1 ON name_list ((lower(real_name)));
```

Now when we search the table using the WHERE clause we can use the `lower()` function on both sides of the equal and use the index to find the data.

```
SELECT *
    FROM name_list
    WHERE lower(real_name) = lower('Bob True')
;
```

Indexes are also a disadvantage. Every insert of a row now requires an insert to each of the indexes. They take up storage space. This means that we are using space and time at data insert or update to save time when we select the data back. Most interactive databases have 10 to 20 times as many selects as they do insert/update operations. So this is usually a good tried off.

We can also get the database to tell us the set of indexes that are not used. We may want to get rid of unused indexes. We can do that with

```
drop index <index_name>;
```

Care should be taken. If the index is a "UNIQUE" index and you remove it you are also removing the unique constraint on the column of data.

Let's get rid of the index that is upper-lower case on the table.

```
drop index name_list_idx1;
```

This is the query that I used to find unused indexes (it's a bit complicated): We will use this in a homework on database performance later in the class.

```
SELECT
    idstat.relname AS TABLE_NAME,
    indexrelname AS INDEX_NAME,
    idstat.idx_scan AS index_scans_count,
    pg_size_pretty(pg_relation_size(indexrelid)) AS index_size,
    tabstat.idx_scan AS table_reads_index_count,
    tabstat.seq_scan AS table_reads_seq_count,
    tabstat.seq_scan + tabstat.idx_scan AS table_reads_count,
    n_tup_upd + n_tup_ins + n_tup_del AS table_writes_count,
    pg_size_pretty(pg_relation_size(idstat.relid)) AS table_size
FROM pg_stat_user_indexes AS idstat
```

```
    JOIN pg_indexes ON indexrelname = indexname
            AND idstat.schemaname = pg_indexes.schemaname
    JOIN pg_stat_user_tables AS tabstat ON idstat.relid = tabstat.relid
WHERE indexdef !~* 'unique'
ORDER BY
    idstat.idx_scan DESC,
    pg_relation_size(indexrelid) DESC
;
```

Please cut/past the above query and run it once.

Things to note:

1. The index name has to be unique.
2. You can have more than one index on a table.
3. Function based indexes require that the function be run on every insert. That takes time.
4. We can create our own functions and index on them as well. We will return to this subject later with `soundex()`

Tags: index,"create index","lower"

Validate: SQL-Select,"select 'PASS' as x"