**Problem Set 2: Building Databases and Loading Data**

Follow the steps below to create tables in a local database. Note that we will use the tables you create in a future problem set, so do not delete the tables after you turn in this assignment.

**1.** Install Postgres locally and create a database, using the instructions we saw in class. You can name your database whatever you like.

**2.** Download the datasets for this assignment, which contain data about fruit and vegetable prices for the year 2020.

**3.** Before going further, add a column to the vegetable data file called “legume,” which will contain a “Y” if the vegetable is a legume, and a “N” if it is not. Fill in this column. (Hint: legumes are any kind of beans, lentils, or peas. If you mislabel a legume, it will not count against your grade.)

**4.** Create two tables in your database, fruit\_prices and vegetable\_prices. Create these tables so that you can use them to import the data you downloaded, including your new column in the vegetable table. All columns must be represented, each column must have a specific data type, and primary keys must be defined.

**5.** Import the fruit and vegetable data, respectively, into your two created tables.

Turn in the following deliverables:

*Deliverable 1:* Submit your CREATE TABLE statements for each table.

**CREATE TABLE fruit\_prices (**

**Fruit VARCHAR,**

**Form VARCHAR,**

**RetailPrice DECIMAL,**

**RetailPriceUnit VARCHAR,**

**CupEquivalentSize DECIMAL,**

**CupEquivalentUnit VARCHAR,**

**CupEquivalentPrice DECIMAL,**

**PRIMARY KEY (Fruit, Form)**

**);**

**CREATE TABLE vegetable\_prices (**

**Vegetable VARCHAR,**

**Form VARCHAR,**

**RetailPrice DECIMAL,**

**RetailPriceUnit VARCHAR,**

**CupEquivalentSize DECIMAL,**

**CupEquivalentUnit VARCHAR,**

**CupEquivalentPrice DECIMAL,**

**Legume BOOLEAN,**

**PRIMARY KEY (Vegetable, Form)**

**);**

*Deliverable 2:* Write and submit a SQL query against the fruit table that will correctly calculate the retail price of 17 pounds of fresh blackberries.

**select (retailprice \* 17) as price**

**from fruit\_prices**

**where fruit = 'Blackberries'**

**and form = 'Fresh';**

*Deliverable 3:* Write and submit a SQL query against the vegetable table that will correctly calculate the average retail price per pound of Dried legumes.

**select avg(retailprice) as avgprice**

**from vegetable\_prices**

**where legume = 'T'**

**and form = 'Dried';**

*Deliverable 4:* Write and submit a SQL query against the vegetable table that will (1) select all vegetables with a “Dried” form, and (2) calculate the difference in retail price per pound between the Dried form and the Canned form, if there is a Canned form. If a vegetable has a Dried form but no Canned form, still include it in the results even though your price difference calculation will be null.

**select dried.vegetable, (dried.retailprice - canned.retailprice) AS pricediff**

**from vegetable\_prices as dried**

**left join vegetable\_prices as canned**

**on dried.vegetable = canned.vegetable**

**and canned.form = 'Canned'**

**where dried.form = 'Dried';**

All code from each deliverable must run without error in order to be considered correct.