**Importing and Exporting Data**

If data exists in delimited text file (with one table row per line of text and each column value separated by a comma or other character), PostgreSQL can import the data in bulk via its COPY command.

**Decennial U.S. Census Population tally by county**

Important steps for imports:

* **Prep the source data** in the form of a delimited text file
* **Create a table** to store the data
* **Write a copy script** to perform the import

Most software can import from and export to a delimited text file, which is a common data format that serves as a middle ground.

**What if the value in a column includes a comma?**

Unless the system for delimiting accounts for that extra comma, during import the line will appear to have an extra column and cause the import to fail.

In such cases, usually the delimited file wraps columns that contain a delimiter character with an arbitrary character called a **text qualifier** that tells SQL to ignore the delimiter character held within. Most of the time, the text qualifier is the double quote.

**When importing, PostgreSQL by default ignores delimiters inside double quoted columns, but you can specify a different text qualifier if your import requires it.**

**Handling Header Rows**

Header rows serve a few purposes. For one, the values in the header row identify the data in each column. Some database managers tend to use the header row to map columns in the delimited file to correct columns in the import table. However PostgreSQL does not do this.

Hence we use the HEADER option in the COPY command to exclude the Header row.

**Using COPY to import data**

COPY table\_name

FROM ‘path to csv file’

WITH (FORMAT CSV, HEADER)

In the WITH statement, we specify that we want to import from a CSV file, and HEADER states that the Header row is to be ignored.

**Options for WITH**

Input and output file format

* Format names are CSV, TEXT or BINARY.

Presence of a Header Row

Use HEADER to specify that the source file has a header row. We can also say HEADER ON, which tells the database to start importing with the second line of the file, preventing the unwanted import of the header.

Delimiter

Lets you specify which character the import or export file uses as a delimiter. E.g. DELIMITER ‘|’

Quote Character

Commas inside a single column value will mess up your import unless the column value is surrounded by a character that serves as a text qualifier. E.g. QUOTE ‘””’

**Importing Census Data Describing Counties**

**Importing Data from a table which doesn’t contain all the columns required in the table**

* First create the required table
* Then import while specifying the columns to import

**Adding a default value to a column during import**

If we want to populate a column during the import, even though the column is missing in the CSV file, we can do so by a temporary table. Temporary tables only exists until you end the database session.

* First we create the temporary table (CREATE temp\_table LIKE normal\_table)
* Then we import the data into the temporary table (COPY temp\_table, FROM …., WITH…)
* Then from the temp table, we insert into the main table (INSERT INTO normal\_table (col names) SELECT (col\_names, ‘filler) FROM temp-table

**Using COPY to Export Data**

Instead of using FROM, we use TO, to show that we are exporting data.

**Exporting All Data**

The simplest export sends everything in a table to a file.