Step 1: Connected to Postgresql Command prompt after specifying the server,username, password.

Step 2: Once it is successfully connected, i have created below databases and tables

CREATE DATABASE foo;

\c foo; # To login into Foo database

CREATE TABLE source(a INT,b INT,c INT);

\d #To Verify whether "source" table has been created

CREATE DATABASE bar;

\c bar; # To login into bar database

CREATE TABLE dest(a INT,b INT,c INT);

\d #To Verify whether "dest" table has been created

Step 3:

I have written below Python Program using Visual Studio Code which handles below features:

- **open a connection to the database foo**

- fill the table source with 1 million rows where:

- column a contains the numbers from 1 to 1e6

- column b has a % 3

- column c has a % 5

- open a connection to the database bar

- copy the data from table source in foo to table dest in bar using postgresql copy command

- start an embedded web server that has two endpoints: ./dbs/foo/tables/source and ./dbs/bar/tables/dest

- upon a GET request to either of the two it must respond with contents of a corresponding table serialized as CSV

- Required Module (psycopg2, flask, requests) needs to be installed for above requirement using PIP command

  PIP install flask

  PIP install requests

  PIP install psycopg2

Code :

```
import psycopg2
from psycopg2 import Error
import StringIO
from flask import Flask, Response
import requests
io = StringIO.StringIO('')
src_connection = psycopg2.connect(user="postgres",
                                  password="Berlin",
                                  host="127.0.0.1",
                                  port="5432",
                                  database="foo")
dest_connection = psycopg2.connect(user="postgres",
                                  password="Berlin",
                                  host="127.0.0.1",
                                  port="5432",
                                  database="bar")
cursor = src_connection.cursor() #Establishes Source Connection
postgres_insert_query = """ INSERT INTO source (a,b,c) VALUES (%s,%s,%s)"""
for i in range(1,10):
   record_to_insert = (i, i%3, i%5)
   cursor.execute(postgres_insert_query, record_to_insert) #loads Source Table with 1 million records
src_connection.commit()
print("Source Table has been loaded")
cursor.copy_expert("""COPY source TO STDIN;""",io)  #Copies data from source table into STDIN
cursor.close()
src_connection.close()
io.seek(0)
output_cur = dest_connection.cursor() #Establishes Target Connection
output_cur.copy_expert("""COPY dest from STDIN;""",io) #loads data into dest table from STDIN
```

```
dest_connection.commit()

print("Destination Load has been completed")

#Copies data from the Destination Table into CSV File

sql = "COPY (SELECT * FROM dest ) TO STDOUT WITH CSV DELIMITER ',''' #Creates CSV File required for webserver

with open("C:\Python27\sample.csv", "wb") as file:

   output_cur.copy_expert(sql, file)

   file.close()

output_cur.close()

dest_connection.close()

app = Flask(__name__)

@app.route('/dbs/foo/tables/source', methods = ['GET'])  # Method to autodownload the table content as CSV file for end point /dbs/foo/tables/source

def source_output():

   with open("C:\Python27\sample.csv") as fp:

      csv = fp.read()

      fp.close()

   return Response(

      csv,

      mimetype="text/csv",

      headers={"Content-disposition":

            "attachment; filename=table_data.csv"})

@app.route('/dbs/bar/tables/dest', methods = ['GET']) #Method to autodownload the table content as CSV file for end point /dbs/bar/tables/dest

def dest_output():

   with open("C:\Python27\sample.csv") as des:

      csv = des.read()

      des.close()

   return Response(

      csv,

      mimetype="text/csv",

      headers={"Content-disposition":
```

```
            "attachment; filename=table_data.csv"})


if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0')
```

Step 4: I have saved the above code with .py extension and excuted the code in the command prompt using below command

python script.py

Step 5: After the execution of script, I have opened the browser with the below endpoints, which has auto downloaded the table content as CSV Files.

http://127.0.0.1:5000/dbs/bar/tables/dest

http://127.0.0.1:5000/dbs/foo/tables/source