Assignment 1

Pratyush Pandey, 2017EE10938
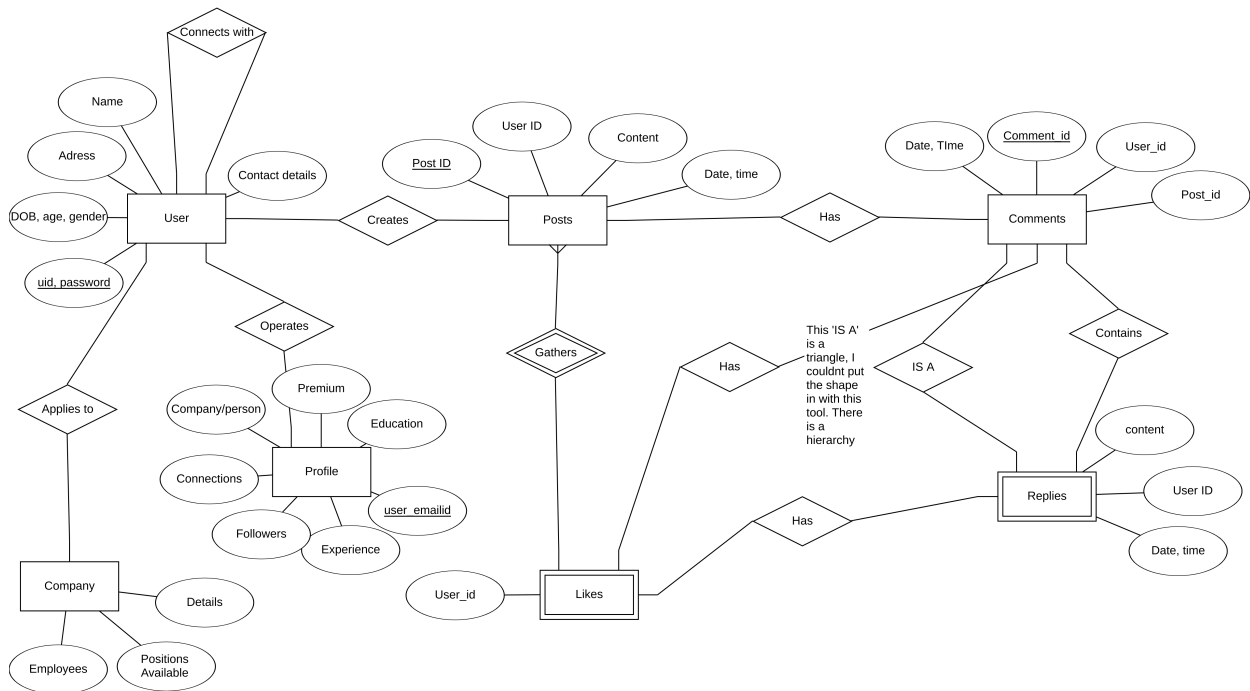
January 8, 2019

Contents

1 Answer 1

1.1 ER Diagram

1.2 Set of relations

1.3 Keys and FDs

1.4 Sample Data

1.5 List of various relationships

2 Answer 2

3 Answer 3

3.1 Schema

The following are the links to the Datasets used in this demonstration.

- DataSet 1 - `https://www.kaggle.com/kashnitsky/mlcourse#medium_posts.csv`
  Size - 70k x 13

- DataSet 2 - `https://www.kaggle.com/kashnitsky/mlcourse#medium_posts.csv`
  Size - 100k x 9

- DataSet 3 - `https://www.kaggle.com/kashnitsky/mlcourse#medium_posts.csv`
  Size - 92.2k x 3

3.2 Bulk Loading

The following SQL Files were used for the process.

Createtable.sql

```
1  CREATE TABLE persons
2  (
3  id character varying(255),
4  age character varying(255),
5  gender character varying(255),
```

```
6  height character varying(255),
7  weight character varying(255),
8  ap_hi character varying(255),
9  ap_lo character varying(255),
10 cholesterol character varying(255),
11 gluc character varying(255),
12 smoke character varying(255),
13 alco character varying(255),
14 active character varying(255),
15 cardio character varying(255)
16 )
```

Importtable.sql

```
1  COPY persons(id,age,gender,height,weight,ap_hi,ap_lo,cholesterol,gluc,
2  smoke,alco,active,cardio)
3  FROM '/home/pratyush/Downloads/COL319␣HW1/Data1/mlbootcamp5_train_70k.csv'
4  DELIMITER ',' CSV HEADER;
```
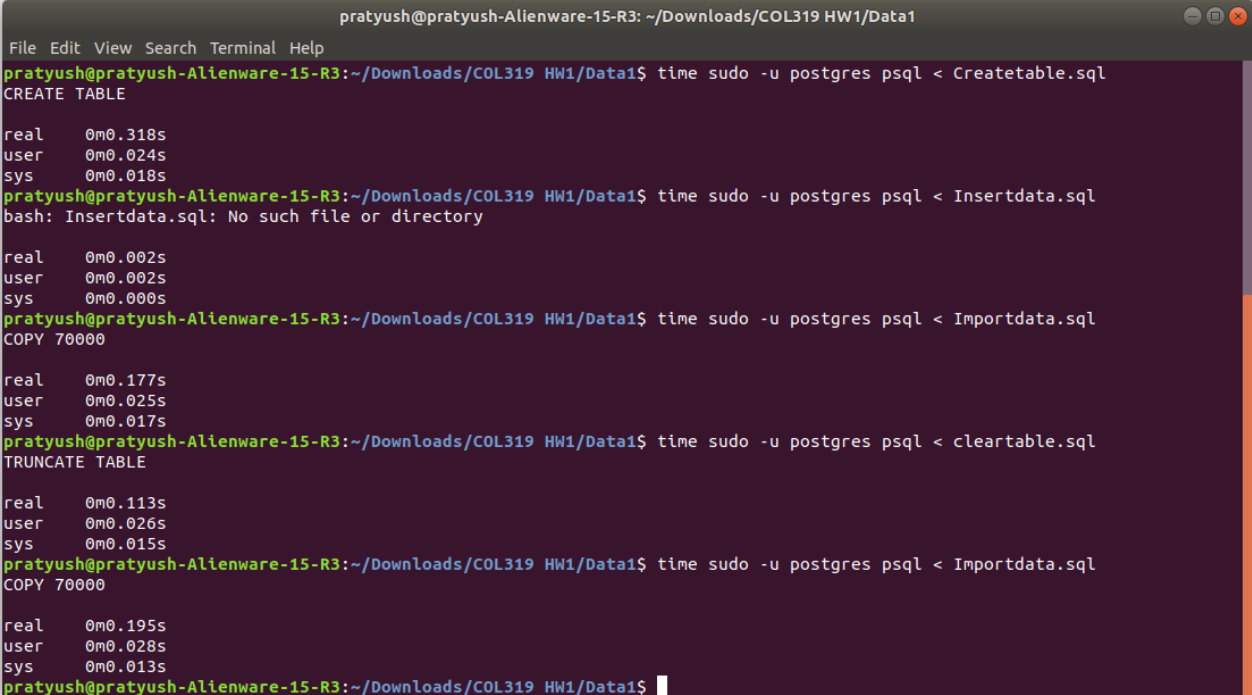
cleartable.sql

```
1  DROP TABLE persons;  /* TRUNCATE TABLE persons;
```

3.2.1   DataSet 1

3

3.2.2   DataSet 2
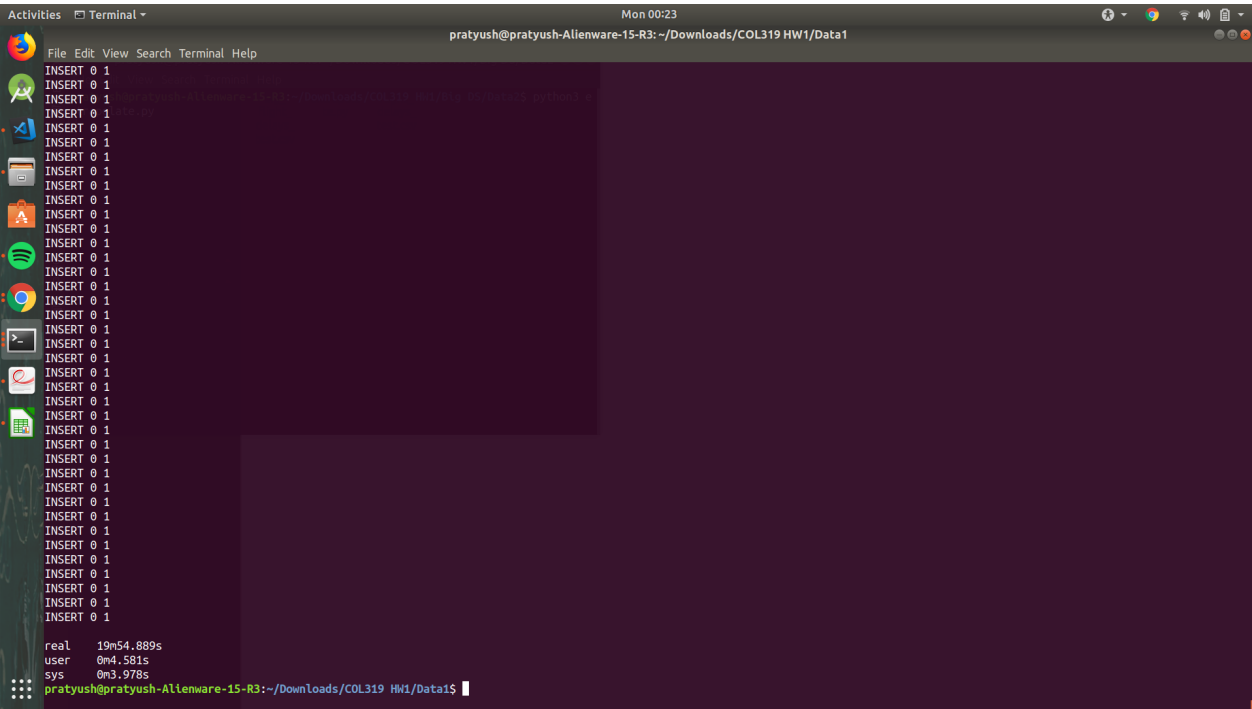

3.2.3   DataSet 3

3.3   Insert statements

**Programmatically Generating the Insert SQL File using Python**

```python
import csv

with open('/home/pratyush/Downloads/COL319
HW1/Data1/mlbootcamp5_train_70k.csv', 'r') as f:
    reader = csv.reader(f)
    next(reader)  # Skip the header row.
    with open("csv2sql.sql", "w") as f1:
        for row in reader:
            f1.write('INSERT INTO persons VALUES (\'%s\', \'%s\', \'%s\',
                \'%s\', \'%s\', \'%s\', \'%s\', \'%s\', \'%s\', \'%s\', \'%s\',
                \'%s\', \'%s\');\n' % (row[0], row[1], row[2], row[3], row[4],
                row[5], row[6], row[7], row[8], row[9], row[10], row[11], row[12]))
```
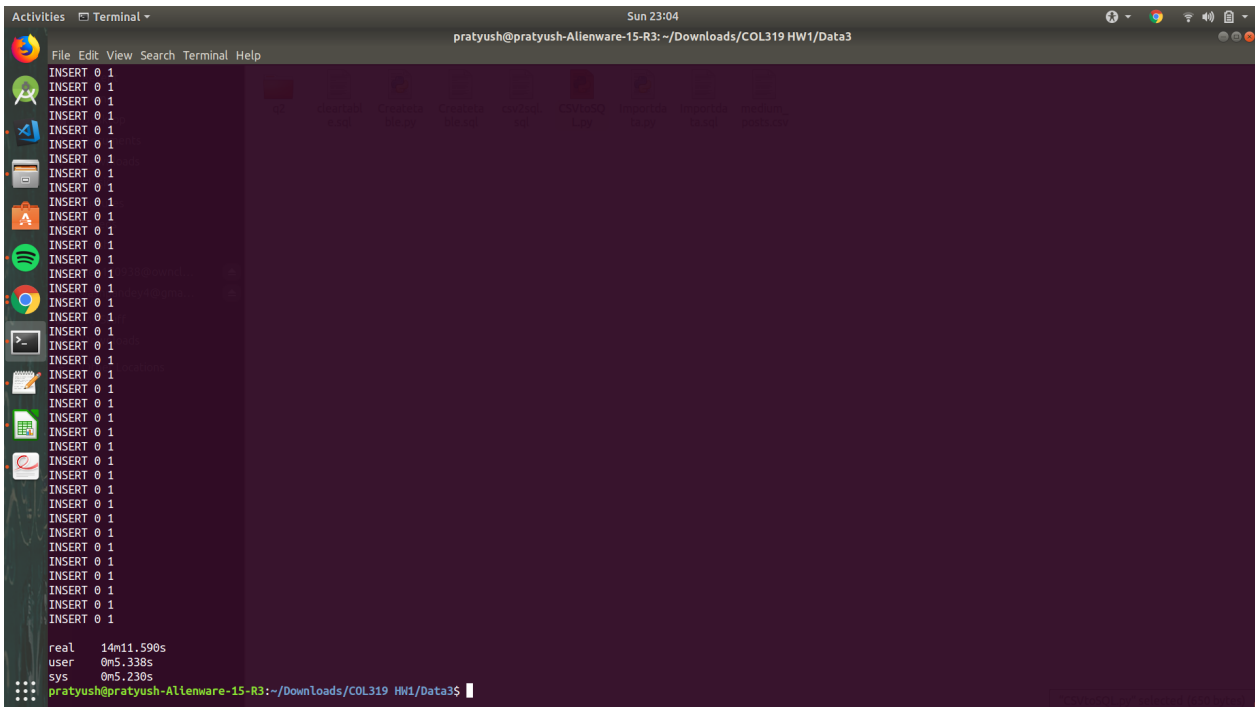
3.3.1   DataSet 1

3.3.2   DataSet 2
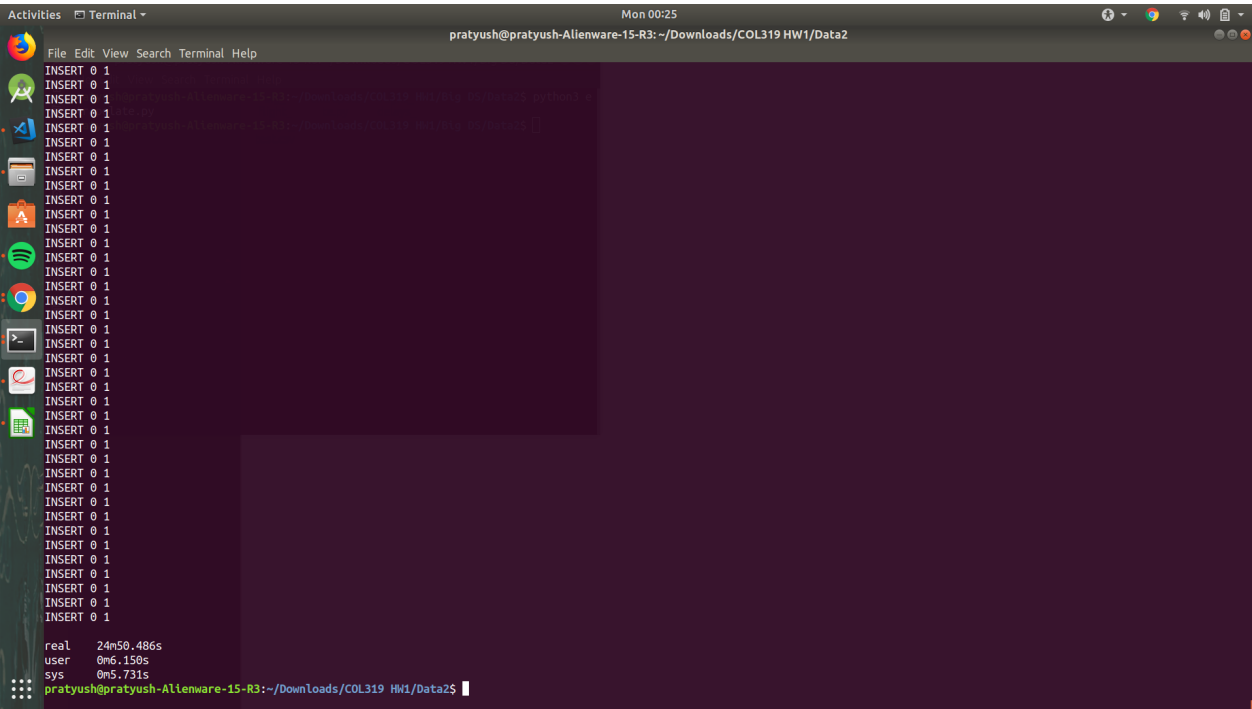
3.3.3 DataSet 3

3.4 Bulk Loading Using Python

We use a package called psycopg2 (Read `http://initd.org/psycopg/docs/` for more information on its protocols) to communicate with out Postgres server.

The method to load a file into a table is called copy_from. Like the execute() method, it is attached to the Cursor object. However, it differs quite a bit from the execute() method due to its parameters.

The copy_from arguments requires a file to load (without the header), the tablename it should load into, as well as a delimiter (the key argument sep). Then, running commit(), the file is transferred into this is the most efficient, and recommended, way to load CSV files into a Postgres table.

```python
import psycopg2

conn = psycopg2.connect("host=localhost dbname=postgres user=root password=nahibataunga")
cur = conn.cursor()
with open('/home/pratyush/Downloads/COL319 HW1/Data1/mlbootcamp5_train_70k.csv', 'r')
as f:
    # Notice that we don't need the 'csv' module.
    next(f)  # Skip the header row.
    cur.copy_from(f, 'persons', sep=',')

conn.commit()
```

6

3.4.1 DataSet 1

3.4.2 DataSet 2

3.4.3 DataSet 3

3.5 Line by Line Iterations using Python

We begin by loading the CSV file in Python using the csv module. Then, we'll run the INSERT query for each row and then commit the transcation:

```python
import csv
import psycopg2

conn = psycopg2.connect("host=localhost dbname=postgres user=root password=cencored")
cur = conn.cursor()

with open('/home/pratyush/Downloads/COL319 HW1/Data1/mlbootcamp5_train_70k.csv', 'r') as f:
    reader = csv.reader(f)
    next(reader)  # Skip the header row.
    for row in reader:
        cur.execute(
            "INSERT INTO persons VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s,
            %s, %s, %s, %s)",
            row
        )
conn.commit()
```

While this accomplishes the task of loading in our data, it's actually not the most efficient way of doing it. As you can see, we had to loop through every single row from the file just to insert them into the database! Luckily for us (and my machine), Postgres has a command specific for loading files into tables.

3.5.1 DataSet 1

3.5.2 DataSet 2

3.5.3 DataSet 3

3.6   Statistics

Process Used	DataSet1	DataSet2	DataSet3
Memory	8.9MB : 70K x 13	9.3MB : 100K x 9	22.3MB : 92.8K x 4
Bulk Loading	0.195s	0.107s	0.084s
Bulk Loading using Python	0.196s	0.208s	0.447
Insert Statements	19m 54.885s	24m 50.486s	14m 11.590s
Line by Line insertions using Python	5.433s	6.855s	5.488s
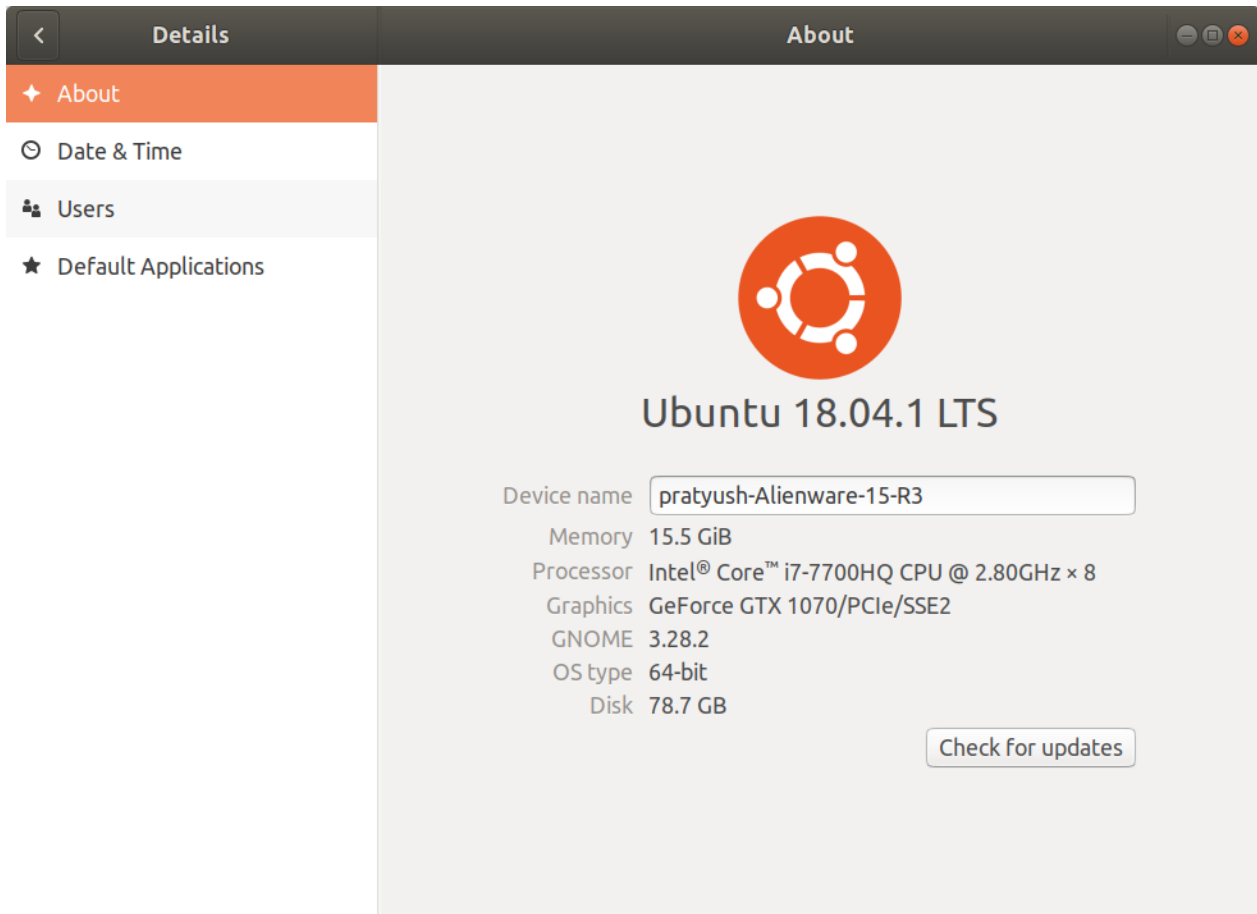
Figure 1: System Specifications

3.6.1 DataSet 3

4 Answer 4

4.1 New DataSets

Following Python was used to increase the size of existing Data Sets to the required 5 gb size.

```python
import os
for i in range(1720):
        os.system("cat mlbootcamp5_train_70k.csv >> big_mlbootcamp5_train_70k.csv")
```

Where the counter of the for loop was set as (6000/(Size of the original csv file in MBs))+-100 to generate an input csv file of 5-10 gb.

4.2 Bulk Loading

4.2.1 DataSet 1

4.2.2 DataSet 2

4.2.3 DataSet 3

4.3 Bulk Loading using Python

4.3.1 DataSet 1

4.3.2 DataSet 2


```
real    0m2.635s
user    0m0.036s
sys     0m0.020s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data2$ time sudo
 -u postgres psql < Createtable.sql\
>
CREATE TABLE

real    0m0.214s
user    0m0.030s
sys     0m0.024s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data2$ time pyth
on3 Importdata.py
/home/pratyush/.local/lib/python3.6/site-packages/psycopg2/__init__.py:144: User
Warning: The psycopg2 wheel package will be renamed from release 2.8; in order t
o keep installing from binary please use "pip install psycopg2-binary" instead.
For details see: <http://initd.org/psycopg/docs/install.html#binary-install-from
-pypi>.
  """)

real    6m46.843s
user    0m6.444s
sys     0m3.249s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data2$
```
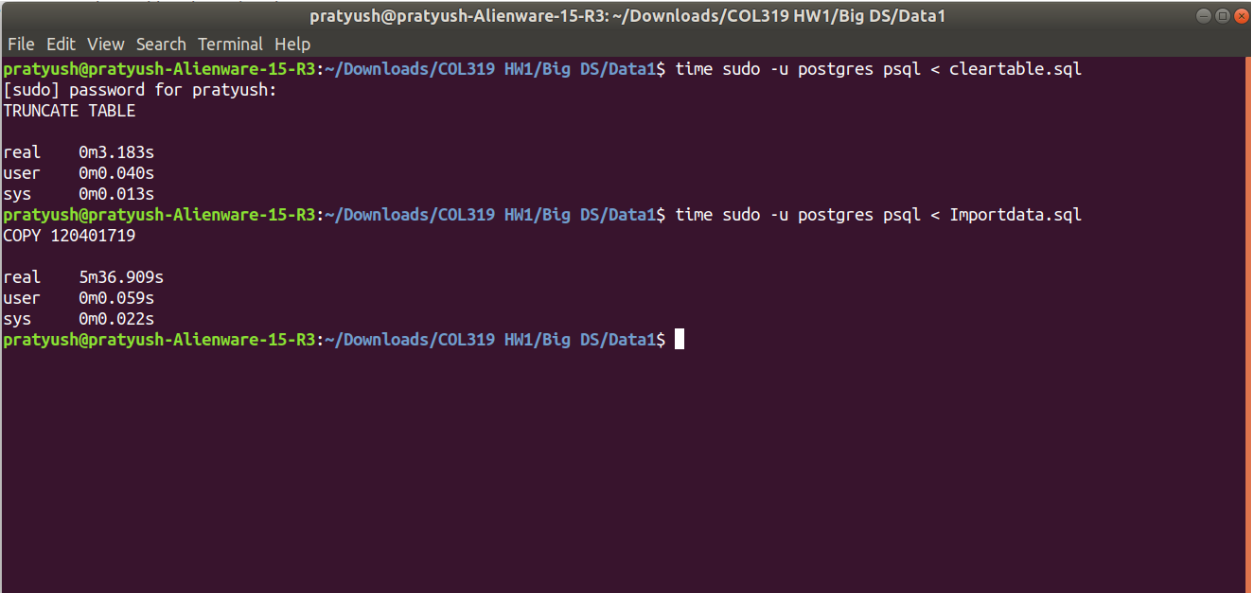
4.3.3 DataSet 3

```
pratyush@pratyush-Alienware-15-R3: ~/Downloads/COL319 HW1/Big DS/Data3

File  Edit  View  Search  Terminal  Help
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data3$ time sudo
 -u postgres psql < cleartable.sql
[sudo] password for pratyush:
TRUNCATE TABLE

real    0m3.585s
user    0m0.037s
sys     0m0.022s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data3$ time pyth
on3 Importdata.py
/home/pratyush/.local/lib/python3.6/site-packages/psycopg2/__init__.py:144: User
Warning: The psycopg2 wheel package will be renamed from release 2.8; in order t
o keep installing from binary please use "pip install psycopg2-binary" instead.
For details see: <http://initd.org/psycopg/docs/install.html#binary-install-from
-pypi>.
  """)

real    5m33.969s
user    0m9.599s
sys     0m5.142s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data3$ █
```
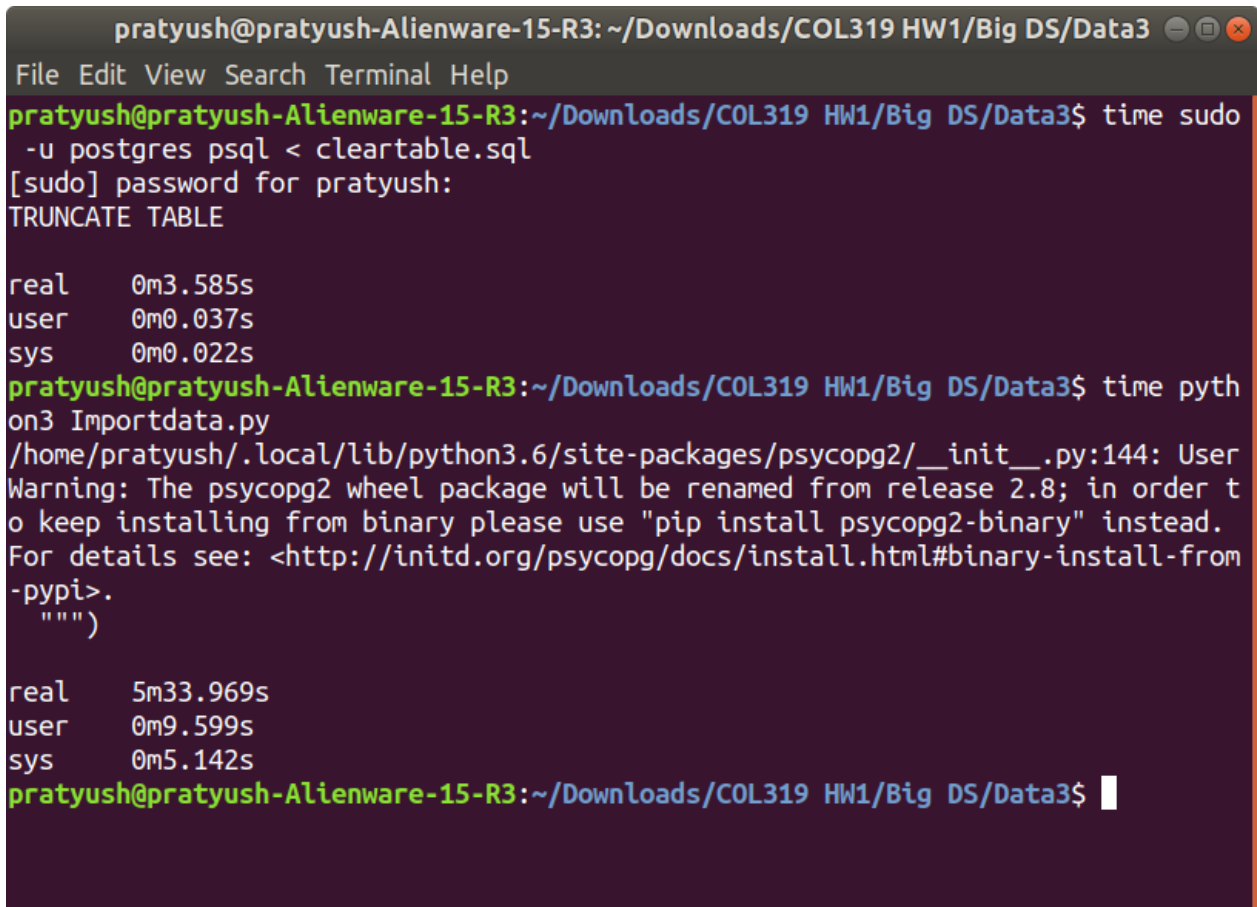
4.3.4 Line by Line Insertions, with or without Python.....

..... Is where things got really messy, Even before when the dataset was small, these operations were taking time in the magnitudes of seconds and minutes. It would have been hours before the thing completed its journey in a high performance PC. I quit the process at around 45 Minutes (Screenshot attached). Since our other times are much smaller than any time in the order of hours, we shall consider the upper limit as infinity.

```
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data1$ time sudo
 -u postgres psql < Createtable.sql
[sudo] password for pratyush:
CREATE TABLE

real    0m2.953s
user    0m0.043s
sys     0m0.005s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data1$ time pyth
on3 Importdata.py
/home/pratyush/.local/lib/python3.6/site-packages/psycopg2/__init__.py:144: User
Warning: The psycopg2 wheel package will be renamed from release 2.8; in order t
o keep installing from binary please use "pip install psycopg2-binary" instead.
For details see: <http://initd.org/psycopg/docs/install.html#binary-install-from
-pypi>.
  """)
^Z
[1]+  Stopped                 python3 Importdata.py

real    49m1.448s
user    0m0.000s
sys     0m0.000s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data1$
```



```
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data1$ time sudo
 -u postgres psql < Createtable.sql
[sudo] password for pratyush:
CREATE TABLE

real    0m3.100s
user    0m0.045s
sys     0m0.014s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data1$ time pyth
on3 Importdata.py
/home/pratyush/.local/lib/python3.6/site-packages/psycopg2/__init__.py:144: User
Warning: The psycopg2 wheel package will be renamed from release 2.8; in order t
o keep installing from binary please use "pip install psycopg2-binary" instead.
For details see: <http://initd.org/psycopg/docs/install.html#binary-install-from
-pypi>.
  """)
^Z
[1]+  Stopped                 python3 Importdata.py

real    36m7.612s
user    0m0.001s
sys     0m0.000s
pratyush@pratyush-Alienware-15-R3:~/Downloads/COL319 HW1/Big DS/Data1$
```

4.4 Statistics

Process Used	DataSet1	DataSet2	DataSet3
Memory	6,1GB : 120401719 x 13	5.9GB : 150001499 x 9	7.05 GB : 46095999 x 4
Bulk Loading	5m36.989s	18m53.632s	9m24.175s
Bulk Loading using Python	5m36.389s	6m46.843s	5m33.969s