Hannah Roach
2-28-2019

Lab Exercise 1: Introduction to Data Environment

Purpose:	The first lab introduces the *Analytics Lab Environment* you will be working on throughout the course. After completing the tasks in this lab you should able to: • Authenticate and access the Virtual Machine (VM) assigned to you for all of your lab exercises • Use SQL and Meta commands in PSQL to navigate through the data sets • Create subsets of the *data*, using *table joins and filters* to analyze subsequent lab exercises

Tasks:	Tasks you will complete in this lab exercise include: • Exploring databases and datasets • Using PSQL statements and Meta commands. • Creating subsets of data for use in subsequent lab exercises

References:	References used throughout the labs are located in your ***Lab Appendix****.* See the Appendix for: • PSQL Commands – Quick Reference • PSQL Meta Commands – Quick Reference • Common LINUX – Quick Reference • R – Quick Reference

1.1 Accessing Lab Environment

Step	Action
1	**Accessing Your FE client VM:** **1. See pre-lab instructions posted on Blackboard (from week 1)** 2. Your user name, password details are provided by your instructor. **Accessing the LAB** 1. All of your work will be done from the FE client. 2. I will provide you the IP address of your “Back-End” (be) server that hosts the databases and the RStudio environment 3. The RStudio is accessed through the “safari” browser available as a desktop icon on your FE client 4. RStudio is accessed with URL **http:// *Back end server IP*:8787/** *The IP address for your BE server has been emailed to you* 5. Utilities such as “putty”, WinSCP and PGadim III are also available on the “fe” to access and update contents in the “be”. **Use your the lab appendix for additional instructions that may be associated with individual labs.**

1.2 Database Environment – Retail Data

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>1</td><td>

Open Putty on your FE client and log into the BE server (step1). If Putty is not on your machine please download it from https://www.putty.org/

Login: gpadmin
Password: p@ssw0rd

Currently you are logged in as **GPADMIN** and you have administrative access to the *Greenplum Database Environment*, in which you will be working.

You must first verify if the database up and running.

1. Type: `gpstate`

2. Review the output; you should be able to see that the database is active with the following output. *Please note that because of the large output size I am only showing selected lines and that your configuration details may slightly differ from the one below.*

```
[INFO]:-Starting gpstate with args:
[INFO]:-local Greenplum Version: 'postgres (Greenplum Database)
4.1.1.1 build 1'
[INFO]:-Obtaining Segment details from master...
[INFO]:-Gathering data from segments...
[INFO]:-Greenplum instance status summary
[INFO]:-----------------------------------------
[INFO]:-   Master instance                  = Active
[INFO]:-   Master standby                   = No master standby
configured
…
[INFO]:-   Total primary segments                          = 2
[INFO]:-   Total primary segment valid (at master)         = 2
[INFO]:-   Total primary segment failures (at master)      = 0
…
[INFO]:-   Mirrors not configured on this array
[INFO]:----------------------------------------------------
```

</td></tr>
</table>

Step	Action
2	Now you’re ready to open a PSQL session and check all available databases. Refer to the *PSQL Commands – Quick Reference*, *l*ocated in your Lab ***Appendix,*** for the PSQL meta commands. **Note:** PSQL meta commands start with a backslash (\). To review all available meta commands type backslash and question mark (\?). To review all available databases in your environment: 1. Type: **`psql`** This will open a new PSQL session to the default database. 2. Next type: **`\l`** Notice a list of databases and record databases named “training*”.
3	**<u>Connect to the training1 database:</u>** 1. At the PSQL prompt type : **`\c training1`** To see the schemas you have in this database: Type: **`\dn`** • You should see “ddemo” schema, listed. • You should also ensure that this schema is included in the search path. • 2. Execute your first PSQL command, type: **`SET search_path TO ddemo, public;`** **Note:** PSQL commands are terminated with a semi-colon- “;”

<table>
<tr><th>Step</th><th>Action</th></tr>
<tr><td>4</td><td>You can now view the tables in this database.
Type: <code>\dt</code>
1. Record the number of tables in the database: 29
Locate the table, “customers_dim”.
Review the column descriptions for this table:
2. Type: <code>\d+ customers_dim</code>
Record the column descriptions, their types and column name(s) by which the table is distributed (aka: the distribution key):
<table>
<tr><th>Column Descriptions</th><th>Type</th><th>Distribution Key Column(s)</th></tr>
<tr><td>not null default nextval(‘customers_dim_customers_id_seq’::regclass)</td><td>integer</td><td>customer_id</td></tr>
<tr><td>not null</td><td>character varying(100)</td><td></td></tr>
<tr><td>not null</td><td>character varying(200)</td><td></td></tr>
<tr><td></td><td>character(1)</td><td></td></tr>
</table>
</td></tr>
<tr><td>5</td><td><b><u>Analyze the gender distribution of the customer base:</u></b>
To locate the number of males and females type:
<code>SELECT gender,count(*) FROM customers_dim GROUP BY gender;</code>
1. Record the number of female customers: 499041
2. Record the number of male customers: 500959
3. Record the total number of customers: 1000000</td></tr>
</table>

Step	Action
6	Using PSQL, generate a report on the average spending by gender, Type:

```
SELECT
  c.gender
, AVG(o.item_price) AS avg_price
FROM
  ddemo.order_lineitems AS o
JOIN
  ddemo.customers_dim AS c
   ON o.customer_id = c.customer_id
GROUP BY c.gender
;
```

Note: You can find this code in the LAB01 directory. This script can be executed using the following command from the OS prompt:

To exit the PSQL environment, use the following meta command, type:

```
\q
```

You are now at the OS prompt.

To execute the SQL script type at the OS prompt:

```
cd LAB01

psql -d training1 -f lab1p1step6.sql
```

Note 1: In the *psql* command above option "-d" specifies the database name to connect to ("training1"). This is equivalent to specifying *dbname* as the first **non-option argument** on the command line. As a convention we have used the option "-d" throughout this document. However *dbname* can be specified without option "-d" as long as it is the first argument of the *psql* command.

Note 2: This query may take some time to execute as it is processing a million rows of data.

1. Record the average expenditures by gender:

   Male : 33.845   Female: 33.798

1.3 Database Environment-Census Data

Step	Action
1	Follow the steps detailed in, Lab 1 - Data Set 1, to connect to and inspect another database “training2”.
2	Record the tables in database (Schema – Public)“training2” bayes_test, fips, foo, housing, housing_nodupes, income_state, logr_coef, nbtrain, persons, zeta, zeta1
3	Describe the type of data in the database. bayes_test – integer and integer[] fips – integer and text foo – text and numeric housing - integer housing_nodupes - integer income_state – text, numeric logr_coef - integer, double precision[] nbtrain – character varying(8) persons - integer zeta – integer, character varying(255), double precision zeta1 – double precision, double precision[]

Step	Action
4	Record the number of rows in each table. bayes_test – 14 fips – 55 foo – 52 housing - 12515394 housing_nodupes - 6257697 income_state – 52 logr_coef – 1 nbtrain – 10010 persons - 28542588 zeta – 64076 zeta1 - 64076

Step	Action
5	**Data Preparation & Cleanup – 1:**

(Scenario) You realize that the Intern who loaded the “housing” data has copied records into the table twice. Each different row is represented by a unique combination of “serialno“ and “state” columns.
Execute the following code:

```
SELECT
  SUM(c) AS total_records
, SUM(CASE WHEN c>1 THEN c-1 ELSE 0 END) AS
total_dupes
, COUNT(*) AS total_uniques
FROM (
  SELECT
    COUNT(*) AS c
  FROM
    housing
  GROUP BY
    serialno
    , state
) AS dupes
;
```

Note: This code is also available at,

/home/gpadmin/LAB01/countdupes.sql,

1. Record the total number of records in the table: 12515394

2. Record the total number of duplicate records: 6257697

3. Record the total number of unique records: 6257697

Step	Action
6	**Data Preparation & Cleanup – 2:**

To prepare and clean the data you need to create a “housing_nodupes” table.
Make sure that you are in the PSQL environment if you have previously exited to the OS command line.

Check to see if a table already exists with the name (“housing_nodupes”).

Type

```
\dt
```

Note: the command \dt will list all tables in the database. \dt public.* will list all tables in the public schema.

If this table already exists execute the following SQL statement:

```
DROP TABLE IF EXISTS housing_nodupes;
```

Execute the following SQL statement:

```
CREATE TABLE housing_nodupes AS
SELECT DISTINCT ON
  (serialno, state) *
FROM
  housing
DISTRIBUTED BY (serialno, state)
;
```

Note: This code is also available at, **/home/gpadmin/LAB01/lab1p2step6.sql**

Repeat the queries in Step 5 (previous step) to ensure that there are no duplicate records in the housing_nodupes table.

Step	Action
7	**Basic Analytics Using the “Housing” Data:**

Execute the following SQL statement to calculate correlation between household income and number of rooms:

```
SELECT
  corr(hinc, rooms)
FROM
  housing_nodupes
WHERE
  state = 25
;
```

1. Record your result:

0.374485423827578

Execute the following SQL statement calculate the R-squared of the regression line of household income and number of rooms::

```
SELECT
  regr_r2(hinc, rooms)
FROM
  housing_nodupes
WHERE
  state = 25
;
```

2. Record your result:

0.140239332659321

Step	Action
8	**Prepare “Housing” Data for Subsequent Analytic Exercises:**

You need to prepare data from the, “housing_nodupes” and “persons” tables, for subsequent analysis with “R” in the next module.

1. 1.Run the following commands and SQL query to move (pipe) the results into a text file **Note:** Use the meta commands to render your output to a file and remove the white spaces (formatting)

   ```
   \a
   \o lab1_01.txt
   SELECT
     serialno
   , hinc
   , rooms
   FROM
     housing_nodupes
   WHERE
     hinc > 0
     AND state = 25
   ;
   ```

Note: The SQL query is also available at the following location:

/home/gpadmin/LAB01/lab1p2step8.sql

Alternatively you can execute the following command from the OS prompt:

```
psql  -d  training2 -f lab1p2step8.sql
```

Now, your data is ready for the lab exercise in the next module.

2. Remove the summary line at the end of tpwdhe output file lab1_01.txt

Step	Action
9	<u>**Prepare “Persons” Data for Subsequent Analytic Exercises:**</u>

Prepare a summary table with the number of people by race and by education level.

Note: Use the following Races: White, Black, American Indian/Alaska Native, Asian, Hawaiian /Pacific Islander, and Others.

```
 (white) White,
(black) Black,
(aian)  American_Indian_Alaska_native,
(asian) Asian,
(nhpi)  Hawaii_pacific_islander,
(other) Others
```

<u>Use the following Education Levels:</u>

01. No schooling completed
02. Nursery school to 4th grade
03. 5th grade or 6th grade
04. 7th grade or 8th grade
05. 9th grade
06. 10th grade
07. 11th grade
08. 12th grade, no diploma
09. High school graduate
10. Some college, but less than 1 year
11. One or more years of college, no degree
12. Associate degree
13. Bachelor’s degree
14. Master’s degree
15. Professional degree
16. Doctorate degree

1. Create a table with columns for Races and rows for Educational Level. (The cells denote the number of “persons” for each category.) Prepare a text file with headers to use in the next module. SQL code necessary for this task is presented below:

```
\a
\o lab1_02.txt
SELECT
  educ AS Education_Level
  , SUM(white) AS White
  , SUM(black) AS Black
  , SUM(aian) AS American_Indian_Alaska_Native
  , SUM(asian) AS Asian
  , SUM(nhpi) AS Hawaii_Pacific_Islander
  , SUM(other) AS Others
FROM
  persons
WHERE
  age > 17
  AND educ > 0
GROUP BY educ
ORDER BY educ
;
```

Step	Action
10	The code in step 9 is also available at the following location: /home/gpadmin/LAB01/lab1p2step9.sql 2. Execute the following command from the OS prompt: `psql  -d  training2 -f lab1p2step9.sql` Remove the last “summary” line as you did in Step 8 and prepare the file “lab1_02.txt” for the lab exercise in the next module.

End of Lab Exercise Submit this completed worksheet to Blackboard for Grading