# 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 ***Student Resource Guide Appendix****.* See the Appendix for:   * PSQL Commands – Quick Reference * PSQL Meta Commands – Quick Reference * Surviving LINUX – Quick Reference * R – Quick Reference |

1.1 Accessing Lab Environment

| **Step** | **Action** |
| --- | --- |
| 1 | Utilities such as “putty”, WinSCP and PGadim III can be used to access and update contents in the “Back-End” (be).  Refer to your VM account sheet of the “Back-End” (be) that hosts the databases. Log in to the “be” host (step1) |

1.2 Database Environment – Retail Data

| **Step** | **Action** |
| --- | --- |
| 1 | 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 we only show 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]:-----------------------------------------------------** |
| 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 ***Student Resource Guide 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.   1. Next type: **\l**   Notice a list of databases and record databases named “training\*”.  **Note:** Another way of listing all available databases (without opening a PSQL session) is to call PSQL executable with parameter (-l): **psql -l** |
| 3 | **Connect to the training1 database:**   1. At the PSQL prompt type : **\c training1**   at the OS level prompt type: **psql training1**  To see the schemas you have in this database:   1. Type: **\dn**  * You should see “ddemo” schema, listed. * You should also ensure that this schema is included in the search path.  1. Execute your first PSQL command, type:   **SET search\_path TO ddemo, public;**  **Note:** PSQL commands are terminated with a semi-colon- “;” |
| 4 | You can now view the tables in this database.   1. Type: **\dt** 2. Record the number of tables in the database: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3. Locate the table, “customers\_dim”. 4. Review the column descriptions for this table: 5. Type: **\d+ customers\_dim** 6. Record the column descriptions, their types and column name(s) by which the table is distributed (aka: the distribution key):  |  |  |  | | --- | --- | --- | | **Column Descriptions** | **Type** | **Distribution Key Column(s)** | |  |  |  | |
| 5 | **Analyze the gender distribution of the customer base:**   1. To locate the number of males and females type:   **SELECT gender,count(\*) FROM customers\_dim GROUP BY gender;**   1. Record the number of female customers: \_\_\_\_\_\_\_\_\_\_\_\_ 2. Record the number of male customers: \_\_\_\_\_\_\_\_\_\_\_\_ 3. Record the total number of customers: \_\_\_\_\_\_\_\_\_\_\_\_ |
| 6 | 1. 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:   1. To exit the PSQL environment, use the following meta command, type:   **\q**  You are now at the OS prompt.   1. 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 : \_\_\_\_\_\_\_\_\_\_ Female:\_\_\_\_\_\_\_\_\_\_ |
| 7 | Use the script, “lab1p1step7”, with the appropriate modifications to list the top five product categories ordered by men and women.   |  |  |  | | --- | --- | --- | |  | ***Men*** | ***Women*** | | 1 |  |  | | 2 |  |  | | 3 |  |  | | 4 |  |  | | 5 |  |  | |

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” |
| 3 | Describe the type of data in the database. |
| 4 | Record the number of rows in each table. |
| 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.   1. 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: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2. Record the total number of duplicate records: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 3. Record the total number of unique records: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| 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.   1. 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.     1. If this table already exists execute the following SQL statement:   **DROP TABLE IF EXISTS housing\_nodupes;**   1. 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**   1. Repeat the queries in Step 5 (previous step) to ensure that there are no duplicate records in the housing\_nodupes table. |
| 7 | **Basic Analytics Using the “Housing” Data:**     1. 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: 2. 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**  **;**   1. Record your result: |
| 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. 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  2. 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.   1. Remove the summary line at the end of the output file lab1\_01.txt |
| 9 | **Prepare “Persons” Data for Subsequent Analytic Exercises:**  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  Use the following Education Levels:   |  |  |  | | --- | --- | --- | | 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**  **;** |
| 10 | The code in step 9 is also available at the following location:  /home/gpadmin/LAB01/lab1p2step9.sql  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*