COEN 178 Intro to Database Systems Spring 2022

Lab 1 (50 pts)

TA: Email: jcole@scu.edu
Office Hours: 10:30am-12:30pm Thursday
https://scu.zoom.us/j/96535495701?pwd=TWxKdjJ0VmpGREozZVdkc1BnbEprQT09

Objectives: Learn

- Learn the basics of logging into Oracle database
- Use SQLPLUS (Oracle's SQL interface) to create tables.
- Load the tables with values.
- Perform simple SQL queries to retrieve data from the tables.
- Execute SQL from a file.

Note: There are two files you will use for this lab:

a) Lab1_COEN178.pdf b) insertStaffData.sql

Requirements to complete the lab

1. Capture the session (queries and output) into a .log file for each part (only include correct runs!).
2. Submit the source code for all your commands as .sql files and upload to Camino.
3. Submit your answers to questions, observations, and notes as .txt (or .doc/.docx/.pdf) file and upload to Camino.
4. Show the TA correct execution of the SQL programs and files

Logging into Oracle

1. Open PuTTY and connect to Linux m/c.

Host Name: linux.dc.engr.scu.edu

Connection type: SSH

Click on Open button

2. Login to Linux m/c

Enter User name: 8 letters, 1 from your first name and 7 from your last name

Enter Password: Your student ID or other if you have changed it

3. Setup Oracle database

setup oracle

4. Logon to Oracle server

sqlplus <username>@db11g

2

Your Oracle username will be identical to your DC username, but the password is independent. (By default your password is set to your Access Card number with *ALL* the leading zeroes)

Now you are at SQL> prompt

To change password, at SQL prompt, run password as shown below

SQL> password

Check http://wiki.helpme.engr.scu.edu/index.php/Oracle

--

Capturing and Recording your Session

One way to capture your SQL session is to use the spool command that SQLPLUS provides to save query results to a file.

At the SQL> prompt, type spool <*filename*>. For example,

SQL> spool foo;

This will create a **foo.log** file in the current directory and will capture all input and output during your session at SQLPLUS (until you exit SQL).

You can terminate the capture with the command, **spool off;**

Issuing operating system commands from SQL

You can issue host operating system commands from SQL as follows:

Assuming you are accessing Oracle on Unix systems, you can list the contents of the current directory as

SQL> host ls

To see your current directory,

SQL> host pwd

PART 1 (20 pts)

In this part, you will issue SQL statements from the command line.

Do not just directly copy the code from this file to your SQL prompt, or you will meet some error, such as the quotation mark error.

Creating a Table (or two)

At SQL> prompt, **type the following lines to create a table called trial, with two attributes, tnum (an integer) and str (a string of 10 characters).**

```
CREATE TABLE trial (
tnum int,
str char(10)
);
```

Note: SQL is NOT case-sensitive. You may enter text on one line or on several lines. If your command runs over several lines, you will be prompted with line numbers until you type the semicolon that ends any command.

Do you see the message, Table created, at SQL prompt? ------------------------

Inserting Tuples

Now we need to add some data tuples into table, trial.

Type the following SQL statement:

```
INSERT INTO trial VALUES (1,’A test’);
```

You should see the message, “1 row created”.

Insert at least 2 more tuples with values of your choice.

Simple Queries to get data from the table

At the SQL prompt, **type the following statement, to retrieve all the rows in the table, trial.**

```
SELECT * FROM trial;
```

Do you see all the tuples you have entered in the table?

To retrieve only the values for the attribute, tnum, type the following SQL command:

SELECT tnum FROM trial;

Now, try to enter a duplicate tuple (1,'A test') into trial, with the statement:

INSERT INTO trial VALUES (1,'A test');

Did you succeed in adding a duplicate tuple?

Type the statement, SELECT * FROM trial;

You were able to add a duplicate tuple because we did not define a primary key on the table.

Let us create another table with a primary key.

```
CREATE TABLE test (
tnum int,
str char(10),
PRIMARY KEY(tnum)
);
```

Add a tuple into test. Type the following SQL statement:

INSERT INTO test VALUES (1,'A test');

Now, try to enter a duplicate tuple (1,'A test') into test, with the statement:

INSERT INTO test VALUES (1,'A test');

Did you succeed in adding a duplicate row? -------------------------

Getting Meta Information about your Database

The system keeps information about your database (the tables and constraints etc, that you have created) in certain system tables. You can retrieve the names of the tables that you have created from the system table,USER_TABLES.

Type the following query:

SELECT TABLE_NAME

FROM USER_TABLES;

Do you see the tables (trial and test) that you have created so far?

Removing your tables from the database

Now, remove the table, trial from the database with the statement,

DROP TABLE trial;

Now, execute the query to get the table names from USER_TABLES.

SELECT TABLE_NAME

FROM USER_TABLES;

What do you see?

Part 2 (10 pts)

In this part, you will learn to execute SQL commands from a script file.

Step 1: Create a folder structure called COEN178/labs/lab1.

Step 2: Create a text file called **data.sql.** This file will contain the SQL statements that you want to execute.

You have a couple of options to create the text file:

A) Using a text editor (gvim, for example), create the file and include the SQL statements.

B) At SQL prompt, you can type the commands

SQL> **DEFINE_EDITOR = "vim"**

SQL>edit <filename>. This will open up the default editor (vi, for example,in Unix environment) and you can type the SQL statements and save the file under COEN178/labs/lab1 directory.

Now type the following statements to insert values into the table test.

INSERT INTO test VALUES (10,' ten');

INSERT INTO test VALUES (11,' eleven');

You can type more statements to insert values of your choice.

Now, to execute the commands from the script file, data.sql as follows:

SQL> start <*Path/filename*>

or

SQL> @<*Path/filename*>.

For example,

SQL> start /home/<user>/COEN178/labs/lab1/data.sql

or

SQL> @/home/<user>/COEN178/labs/lab1/data.sql

Now, check if the data you have entered has been loaded into the table successfully, by issuing the SQL query,

SELECT * from test;

Part 3 (20 pts)

Oracle SQL Datatypes, Reference: http://www.ss64.com/orasyntax/datatypes.html

You should capture the session (queries and output) into a file called, lab1_part3.log.

Step 1:

Create a table called Staff as follows:

```
CREATE table Staff (Last VARCHAR(20),First VARCHAR(20),salary
NUMBER(10,2), title VARCHAR(60));
```

Step 2:

Load values into the table Staff from using the SQL script file, insertStaffData.sql.

Step 3:

Type the command below from SQL prompt:

Select * from Staff;

How many rows are retrieved from the table ----------------

Type the command below from SQL prompt:

Commit;

This will ensure the data you have entered will be in the table even if the system crashes.

Now you are ready to issue queries against the table **staff**.

• Write a SQL query to show all lastnames from the table staff.

• Write a SQL query to show all first names concatenated with last names (concatenation **operator is ||) and the column in the result renamed as fullname, from the table staff.**