HW5: PL/SQL

Due Nov 19, 2021 by 11:59pm **Points** 100 **Submitting** a file upload
File Types sql, txt, docx, and pdf **Available** until Nov 20, 2021 at 11:59pm

This assignment was locked Nov 20, 2021 at 11:59pm.

Instructions:

- Download this file (https://utexas.instructure.com/courses/1318318/files/63060161/download?download_frd=1) first and run the script. You can click the hyperlink or find it under Canvas/Files/HW_Files. The file is named: HW5-Starter-DDL-Script.sql (https://utexas.instructure.com/courses/1318318/files/63060161/download?download_frd=1)
- If needed, download, and run this script (https://utexas.instructure.com/courses/1318318/files/63060162/download?download_frd=1) if you need to drop previous tables (optional)

Warning:

- **Do your own work:** This is a team assignment, but your team must do original work create own SQL statements. If you are caught cheating on this or using someone else's work, you will fail in this class and be reported to the Dean of Students. Also, this homework prepares you for the exam coming up so doing the work now will help you learn and do well when it counts more.

PL/SQL Questions:

1. Do all tasks below.

(a) Write a script that uses an anonymous block of PL/SQL code to declare a variable called count_reservations and set it to the count of all reservations placed by the customer whose ID is 100002. If the count is greater than 15, then the block should display a message that says, "The customer has placed more than 15 reservations." Otherwise, the block output should say "The customer has placed 15 or fewer reservations." Make sure that you set the server output to be on before the PL/SQL block of code and include that at the top of your submission.
(b) Delete the record that has reservation ID = 318 from the reservation_details and reservation tables. Do NOT commit.

(c) Run your PL/SQL again. You should get a different output.
(d) Rollback your deletion.

2. Run the “set define on;” command to allow substitution variables. Update the previous statement but this time prompt the user to enter a customer ID and dynamically use that input to pull the count of reservations for the customer ID the user entered. Also update the output as well to include the customer ID and the count of reservations like so:
e.g. When you prompt the user and they enter 100002, it should return something like “The customers with customer ID: 100002, has placed more than 15 reservations.” If you enter a customer ID = 100003, it should say something like “The customers with customer ID: 100003, has placed 15 or fewer reservations.”. Hint: you’ll need to adjust the SELECT to pull in customer ID with the count which will create a new to aggregate (i.e. group) data.

3. Write a script that uses an anonymous block of PL/SQL code that attempts to insert a new customer into the customer table. Just utilize the customer_id_seq to assign the customer_id. Make up your own data. Only fill in the fields that require a value. Use a column list to complete the insert statement. Also commit after the data has been inserted. If the insert is successful, the PL/SQL code should display this message:

```
1 row was inserted into the customer table.
```

If the update is unsuccessful, the procedure should display this message:

```
Row was not inserted. Unexpected exception occurred.
```

4. Write a script that uses an anonymous block of PL/SQL code that uses a bulk collect to capture a list of available features that begin with the letter P. The rows in this result set should be sorted by feature name. Then, the code should display a string variable for each feature and its feature. NOTE: We’re to just going to assign a number for the feature based on it’s place in the list and not its actual feature_id. Your output should look like this:

```
Hotel feature 1: Parking Included
Hotel feature 2: Pets Allowed
Hotel feature 3: Pets Not Allowed
Hotel feature 4: Pool
```

5. Write a more complex version of the previous script that uses a cursor to capture the location name, city, and feature name. Then output the rows like the shortened sample list below which is sorted by location name, city, and feature name. ***3 bonus points will be given if you can prompt the user for a city and then change the outputted list to output the location name, city, and feature name for that city.***

Sample Output: (Only first four lines are shown)

```
Balcones Canyonlands Cabins in Marble Falls has feature: Business Center
Balcones Canyonlands Cabins in Marble Falls has feature: Free Breakfast
Balcones Canyonlands Cabins in Marble Falls has feature: Full Kitchen
Balcones Canyonlands Cabins in Marble Falls has feature: Parking Included
…
```

6. Take your script from problem 3 and change it from an anonymous block of PL/SQL code to a named procedure called insert_customer that allows you to insert a new customer by passing in customer ID, first name, last name, email, phone, address_line_1, city, state, and zip. Make sure that you are still using the sequence to generate the customer ID. Handle exceptions generally by rolling back the transaction "when others" occurs. Once your procedure compiles correctly, test it with the following calls.

```
CALL insert_customer ('Joseph', 'Lee', 'jo12@yahoo.com', '773-222-3344', 'Happy street', 'Chicago
', 'Il', '60602');
BEGIN
Insert_customer ('Mary', 'Lee', 'jo34@yahoo.com', '773-222-3344', 'Happy street', 'Chicago', 'Il'
, '60602');
END;
/
```

7. Create a function called hold_count that returned the total number of rooms that a customer has reserved when it is passed a customer_id. Once you have compiled your function, test it using the following select statement:

```
select customer_id, hold_count(customer_id)
from reservation
group by customer_id
order by customer_id;
```

Deliverables (What to turn in: two files)

- **Script in a (.sql) file format.** If you have issues doing this, at least save it as a .txt file. Do not submit in any other format other than .sql and .txt. The SQL questions above will be based on the DDL script that is posted on the Canvas instructions. Download that script and run it before you start.
    - Clearly separate your code for each question. Save your code into one SQL file with the naming format: Group_Number.SQL. Please make sure the team number you use matches what is in Canvas/People/Groups. For example: Team_1.SQL
    - Save your file either as a .sql file or as a .txt file. If you need help doing this, refer to the page linked in the Canvas assignment. Files saved in a different format will be 50% and files in a different format that cannot be read into SQL (example: PDF) will result in a 0%.
    - Submit your .sql file on Canvas before the deadline. Late submissions receive 50% off. No submissions will be accepted 24 hours after the deadline.

.

  - Do not include the DDL in your submission. If you do, you will lose 5 points. Only provide SQL with comments and nothing else. Do this going forward on all other assignments unless noted.
- **Write an executive summary (one page .pdf or .docx)** to explain the code, any assumptions your team made to deliver the SQL code to the customer. Save it as a .pdf or .docx file and submit along with the code.