PURDUE UNIVERSITY

PURDUE POLYTECHNIC INSTITUTE
*Department of Computer and Information Technology*

# CNIT 27200: Lab #4

25 pts

## Due Date

- **Part A** is due within the Lab session 10 pts.
- **Part B** is due **the evening before your next lab by 11:59 p.m.** 15 pts. It must be submitted via Blackboard.

## Objectives

Be able to use the DUAL table to test functions. Be able to use single-value functions in a query, including:

- DESCRIBE (DESC also works)
- Numeric functions
  ABS(n), CEIL(n), FLOOR(n), MOD(value, devisor), POWER(value, exponent),
  ROUND(value, precision), TRUNC(value, precision), SQRT(n), SIGN(n)
- Character functions
  LOWER(string), UPPER(string), INITCAP(string)
  INSTR(string, set, start, occurrence), SUBSTR(string, start, count),
  LENGTH(string), LPAD(string, length, 'set'), RPAD(string, length, 'set')
- Date functions
  SYSDATE, TO_CHAR(date, 'format'), TO_DATE(string, 'format'),
  NEXT_DAY(date, 'day'), LAST_DAY(date),
  ADD_MONTHS(date, count), MONTHS_BETWEEN(date2, date1),
  ROUND(date, 'format'), TRUNC(date, 'format')
- Transformation functionsNVL(value, substitute)
- Other functions: USER, SYSDATE

Be able to use group-value functions and related clauses in a query, including:

- Functions COUNT, SUM, AVG, MIN, MAX
- A GROUP BY clause to group data appropriately
- A HAVING clause to restrict the return of grouped data

Your answer template should follow a standard format, to be compiled in a .sql file. This is for saving your FINAL SQL statements and the output. Fill in the header information and insert the question number to separate the responses – we will be using comments to store all our non-statement content such as question numbers, results, etc.

You can comment in SQL by beginning the line with a --

Alternatively, for multi-line comments, surround the block by /* and */.

The following is a template:

```
/* <Begin Comment>
Your Name
CNIT 27200 Spring 2020
Lab Time:
Duration:
**************************************************
Question 1
<End comment> */
Select * from ...

/* Results:
     <paste table output>

   Explanation: In this query...
*/
/*
**************************************************
Question 2 */
etc.
```

Answer the questions listed below, test each SQL statement and verify that it is working correctly, and finally run the SQL Output option at the end. Save to your final SQL file and then add the formatting above to include the SQL statements into your answer template for submission.

# PART A (10 points)

## *For Part A, please use the Lunches Database (CreateLunchDB_s20.sql)*

**Question 1.** Take a look at the Metadata in SQL. The metadata is stored in Oracle just like a database:

A) List the columns of the table called USER_TABLES to review the data dictionary view maintained by Oracle. Type the following in SQL Developer:
   a. DESCRIBE USER_TABLES

B) Then display the names of all the tables (rows) within the **USER_TABLES** table by typing the following SQL statement:
   a. Select Table_Name from USER_TABLES;

**Question 2.** Type this line first: **set linesize 200;**

*It helps with formatting. Then drop to the next line and write your SQL statement.*

A) Using the DUAL table in your SQL statement, round 476.74729 to a) no decimal places, b) 2 decimal places, c) nearest 10s place (ROUND function in slides).

B) Using the DUAL table in your SQL statement, truncate 476.74729 to a) no decimal places, b) 2 decimal places, c) nearest 10s place (TRUNC function in slides).

**Question 3.** In the result set, list the supplier name, then the length of the supplier name (use SN_LENGTH as the column alias), and then the city displayed in all uppercase letters (use CITY as the column alias). Only include the food suppliers from the cities of Orland Park, Chicago, and Aurora. Use the FOOD_SUPPLIER table. (SQL row functions in slides).

In this question, use column formatting:

1. On the first line, type **set linesize 200;**
2. Then on the second line, you are going to format the supplier city column by using the column alias and setting the column width to a15 meaning that it is a width of 15 and alphanumeric. Type **COL CITY FORMAT a15;**
3. Then on the fourth line, you would type your **SELECT** clause to continue with the sql query.

**7 rows selected**

**Question 4.** List the supplier id, description, price and price upcharge for all food with a price greater than $**7**. If the price upcharge is unknown (i.e., NULL), display a value of **999** in the results. (FOOD table; SQL Functions slide for the function NVL).

   a. You can also format a column within the SELECT clause: In the SELECT clause, try formatting the price column to display as currency (use PRICE as the column alias). Use this format: **TO_CHAR((price), '$999.99') as PRICE**

**8 rows selected**

**Question 5.** Show the oldest lunch date (as MIN_DATE) and most recent lunch date (as MAX_DATE) found in the LUNCH table.

a. Format the dates using TO_CHAR to MM-DD- YYYY, Day (Example: 10-17-2019, Thursday). Review the date format document on Blackboard.
b. Use column functions MIN and MAX

**Question 6.** COUNT and AVG Column Function practice:

a. List the supplier id, the count of the number of food items for each supplier id, and the average price for food items. Hint: This requires that you also use the GROUP BY function.

   i. When using count, use the wildcard * in the count function like so: COUNT(*).

   ii. When using the AVG function, round the average price to 2 decimal places (use round from the row functions)

   iii. Hint: FOOD table; COUNT function; AVG function; ROUND row function

   **11 rows selected**

b. Repeat the query above, but when using count this time, use supplier id in the count function like so: COUNT(supplier_id).

   **11 rows selected**

c. Repeat the query above again, but this time when using count, use the attribute price_upcharge in the count function like so: COUNT(price_ upcharge).

   **11 rows selected**

   *The count should be the same in parts a and b because the * counts the number of rows, and the other count was using an attribute from the primary key (supplier_id). In part c we used an attribute in the entity that allowed null values, so the count results would only count those rows with values. The average price will stay the same because it is independent of the count.*

**Question 7.** The company wants to review the credit limits for all workers.

a. SUM the credit limits for each department code. Use GROUP BY to group the department codes and SUM to add the credit limits per department. Sort the result set by the department code. **11 rows selected**
b. Based on the SQL statement in part **a**, add an additional sub group to further group by not only the department code, but also the manager. **24 rows selected**
c. Based on the SQL statement in part **b**, add an additional aggregate function to count the items in each grouping (Hint: use Count(*)). **24 rows selected**
d. Based on the SQL statement in part **c**, only include workers hired after the year 2010. **18 rows selected**
e. Based on the SQL statement in part **d**, only include departments with a total credit limit greater than 25 (HAVING function in slides). **9 rows selected**

**Question 8.** Practice nested subqueries:

a. Using the subquery approach, list the description and price for all food priced less than the average price of all food for sale. (Example found in Functions slides). **18 rows selected**
b. Check the calculation. Run the statement in the nested query as a separate SQL statement so that you can see what the nested query does as its own SQL statement. **1 row selected**

# PART B (15 points) - *Lab 4 Part B utilizes the EAGLE DATABASE (loadEagle_s20.SQL)*

**Use the same answer template as Lab 3.**

**Questions 1 - 10 are worth 1.5 pts. = 15 pts. (For each question, report the SQL statement used, the resulting output from the sql statement, and in the response explain how the function works in SQL)**

*Open the EAGLE ERD on Blackboard for a list of entities and attributes in the database.*

**Question 1.** Using the DUAL table, format the number 6867.65786 in one query:

a. show the USER (you), the ceiling, floor, truncation (to 2 decimal places), and round (to the tens position) for the number.
b. Set the linesize to 200 if needed for improved spacing.
c. Explain: Difference between ROUND and TRUNC? Why use the DUAL table?

**Question 2.** List the employee id, concatenated last name and first name (formatted like "Smith, John" and name the column Employee), hire date, and email address of all employees that were hired in **2013**. If the employee does not have an email address, substitute with '***Need Email***'. Label the column as EMAIL_ADDRESS. **10 Rows**

Explain: When do you use LIKE vs. = in a WHERE clause to filter?

**Question 3.** Per each supplier id, list the minimum ordered unit cost, maximum ordered unit cost, average ordered unit cost, and count the number of purchased items for each supplier id that starts with the letter **C** or the letter **P**. Round the average ordered unit cost column to 2 decimal places. Create column aliases for the MIN, MAX, AVG, and COUNT columns. Format the min, max and avg columns with a width of a15 (see part A, question 3 for help). Sort by supplier id.

**9 Rows**

Explain**:** What is the difference between a row function and a column function?

**Question 4** Use the same SQL statement from question 3, except add a clause that will only list supplier ids with an average ordered unit cost greater than **$800**. **4 Rows**

Explain the difference between WHERE and HAVING clauses.

**Question 5.** List the part number, part description (mask the description in the result set as all lowercase, and using LPAD, right justify the part description to 40 characters and fill the empty spaces with a '#'), category id , weight, and stock price for inventory parts with a weight between 2.50 and 10 pounds and that have a stock price less than the average stock price of ALL parts. **27 Rows.**

In your response explanation, find the rounded average stock price of all parts to two decimal places. Include the query and the result.

**Question 6.** List the customer id, order id , and order date for the oldest customer order. Hint: You will need to find the oldest order date in a nested query.

a. use the CUSTORDER table
b. Explain how nested subqueries work, as well as how MIN and MAX column functions work with dates (oldest vs most recent)
c. **4 rows selected**

**Question 7.** Using SUBSTR, parse the part number to only include the first three characters. In addition to the parsed part number, also include the operation step, and a count of part numbers per operation step in the SELECT clause. (need to use Group by to get the count)

a. Limit the result set to only operation steps – **BOX**, **WRAP**, or **LABEL**
b. Provide a count of supplied parts per parsed part number and operation step.
c. Limit the results to a count of parts greater than or equal to **10**.
d. Order by the parsed part number
e. Explain how SUBSTR works.
f. **15 Rows**

**Question 8.** Using the TRUNC and MONTHS_BETWEEN functions: **14 rows selected**

1. List the employee job title, the average age when hired in years, and the count of employees per job title.
2. To determine the age of the employee when hired in whole years, you will need to use the following function:

   TRUNC(MONTHS_BETWEEN(HireDate, BirthDate)/12))

   *This will find the months between the hire date and birthdate, then divide by 12 to determine the age of the employee when hired.*
3. Now that you have the age at hire, you need to add the AVG function to this calculation. Finally, Round it to 2 decimal places.
4. Hint: You are grouping by the job title.
5. Explain how you constructed this query.

**Question 9.** The accounting department is auditing customer orders by status.

a. Using the GROUP BY function, list the status, count of orders, and order subtotal for all customer order lines. An order subtotal is when you multiply the unit price by the order quantity and sum it by the status. Call the SUM calculated expression SUBTOTAL, and round it to 2 decimal places. **5 rows selected**
b. Now check the calculation to see where it is coming from in the database.
   i. Select the unit price, order quantity, and order id from the customer order line for orders with a **PACKED** status. **8 rows selected**
   ii. Once you multiple each unit price by its quantity, add up the amounts and it should equal the **PACKED** line in the **9a** SQL statement with the calculation built in.
c. Based on the SQL statement in part **9a**, using GROUP BY and HAVING functions, list the status, count and subtotal for any orders with a total less than or equal to $30000. (HAVING function). **3 rows selected**
d. Explain aggregate (column) functions (explain GROUP BY)

**Question 10. (Review TO_CHAR and TO_DATE slides for the following 3 separate queries:**

a. Query 1: Using the DUAL table, display **today**'s date in the format "February 15, 2020".
b. Query 2: Display today's day of the week (by name).

c. Query 3: Using the DUAL table, show the user (you), and then using the TO_DATE data conversion, on what day of the week you were born. (Hint: You will use both to_char and to_date to convert the results)

d. Explain the difference between TO_CHAR and TO_DATE