```
/*
Your Name: Noah Zhou
CNIT 27200 Fall 2023
Lab Time: Friday 7:30 AM - 9:20 AM
*/
--
*************************************************************************
********
--Question 1
/*
Type this line first: set linespace 200;
It helps with formatting. Then drop to the next line and write your SQL
statement.
A) Using the DUAL table, round 274.58345 to the following decimal places.
(ROUND function)
A1) 2 decimal places
A2) no decimal places
A3) nearest 10th place

B) Using the DUAL table, truncate 274.58345 to the following decimal
places.
(TRUNC function)
B1) 2 decimal places
B2) no decimal places
B3) nearest 10th place
*/

set linespace 200;
--A1
SELECT ROUND(274.58345,2) "A1" FROM DUAL;
--A2
SELECT ROUND(274.58345,0) "A2" FROM DUAL;
--A3
SELECT ROUND(274.58345,-1) "A3" FROM DUAL;

--B1
SELECT TRUNC(274.58345,2) "B1" FROM DUAL;
--B2
SELECT TRUNC(274.58345,0) "B2" FROM DUAL;
--B3
SELECT TRUNC(274.58345,-1) "B3" FROM DUAL;

/*
Results:
        A1
----------
    274.58

        A2
----------
       275

        A3
----------
```

```
       270

        B1
----------
    274.58


        B2
----------
       274


        B3
----------
       270
*/
--
**************************************************************************
********
--Question 2
/*
In the result set, list the worker ID, worker last name, then the length
of the
worker last name (Label the length as LAST_LENGTH), and the Dept Code
displayed in ALL CAPS. Use the WORKER table. Only include items with dept
codes Sal, Acc, or Tch. (SQL row functions in slides).
In this question, use column formatting:
1. On the first line, type set linespace 200;
2. Then on the second line, you are going to format the food description
length column by using a column alias and setting the column width to
a11 meaning that it is a width of 12 and alphanumeric:
a. Type COL desc_length FORMAT a12;
3. Then on the third line, you would type your SELECT clause to start
your
sql statement.
4. Finally, sort the query by the last name column.
*/

set linespace 200;
COL desc_length FORMAT a12;

SELECT worker_id, last_name, LENGTH(last_name) AS LAST_LENGTH,
UPPER(dept_code)
FROM WORKER
WHERE dept_code='Sal' OR dept_code='Acc' OR dept_code='Tch'
ORDER BY last_name;
/*
Results:
WOR LAST_NAME            LAST_LENGTH UPP
--- -------------------- ----------- ---
565 Cross                          5 TCH
580 Gonzalez                       8 ACC
584 Harney                         6 TCH
576 Jones                          5 TCH
```

```
564 Kingman                        7 SAL
577 Martin                         6 SAL
578 Rayner                         6 TCH
556 Sumner                         6 TCH
582 Templeton                      9 ACC
563 Vought                         6 ACC

10 rows selected.
*/
--
************************************************************************
********
--Question 3
/*
A) List the first name, last name, and city for all employees with a
credit limit
greater than 21. If the employee city is unknown (i.e., not entered or
NULL) print
xxx in the results. Sort by city. (WORKER table; refer to SQL Functions
slide for
the function NVL).

B) Run it again without the NVL row function to see the NULL values under
the
department code column.
*/
--Question 3-A
SELECT first_name, last_name, NVL(city,'xxx')AS city FROM worker
WHERE credit_limit>21
ORDER BY city;

--Question 3-B
SELECT first_name, last_name, city FROM worker
WHERE credit_limit>21
ORDER BY city;


/*
Results:
[Question 3-A]
FIRST_NAME LAST_NAME            CITY
---------- -------------------- ------------------------------
Dane       Shreve               Aurora
Tonya      Montre               Aurora
Avery      Trance               Aurora
Gail       Walsh                Chicago
Jared      Ridgeman             Chicago
James      Kingman              Chicago
Tyler      Harney               Chicago
Blair      Reynolds             Evanston
Katelynn   Rayner               Evanston
Sam        Frank                Evanston
Jose       Sanchez              Glencoe
```

```
FIRST_NAME LAST_NAME            CITY
---------- -------------------- ------------------------------
Jodie      Williams             Glencoe
Tom        Neal                 Hinsdale
Carole     Sumner               Hinsdale
Angie      Templeton            Hinsdale
Keyanna    Jones                Hinsdale
Kerry      Alveral              Oak Brook
Darius     Richards             Oak Brook
Melody     Campbell             Oak Brook
Trey       Vought               Oak Brook
Brooks     Walsh                Oak Brook
Taylor     Young                Wilmette

FIRST_NAME LAST_NAME            CITY
---------- -------------------- ------------------------------
Maria      Bensen               Wilmette
Rita       Gradle               Wilmette
Cleo       White                xxx
Yvonne     Rivera               xxx
Latesha    Cross                xxx
Cassie     Irwin                xxx

28 rows selected.

Question [3-B]
FIRST_NAME LAST_NAME            CITY
---------- -------------------- ------------------------------
Dane       Shreve               Aurora
Tonya      Montre               Aurora
Avery      Trance               Aurora
Gail       Walsh                Chicago
Jared      Ridgeman             Chicago
James      Kingman              Chicago
Tyler      Harney               Chicago
Blair      Reynolds             Evanston
Katelynn   Rayner               Evanston
Sam        Frank                Evanston
Jose       Sanchez              Glencoe

FIRST_NAME LAST_NAME            CITY
---------- -------------------- ------------------------------
Jodie      Williams             Glencoe
Tom        Neal                 Hinsdale
Carole     Sumner               Hinsdale
Angie      Templeton            Hinsdale
Keyanna    Jones                Hinsdale
Kerry      Alveral              Oak Brook
Darius     Richards             Oak Brook
Melody     Campbell             Oak Brook
Trey       Vought               Oak Brook
Brooks     Walsh                Oak Brook
Taylor     Young                Wilmette
```

```
FIRST_NAME LAST_NAME            CITY
---------- -------------------- ------------------------------
Maria      Bensen               Wilmette
Rita       Gradle               Wilmette
Cleo       White
Yvonne     Rivera
Latesha    Cross
Cassie     Irwin

28 rows selected.
*/


--
***********************************************************************
********
--Question 4
/*
MIN and MAX practice:
Format both the dates using TO_CHAR to Month DD, YYYY, Day (Example:
February 15, 2022, Tuesday). Review the date format document on
Brightspace.
Use the column formatting above (question 3) to set the two columns to a
width
of a25
A) Show the oldest lunch date (as MIN_DATE) and most recent lunch date
(as
MAX_DATE) found in the LUNCH table.
(Use column function MIN and MAX; Show both in 1 SQL statement)
*/
COL MIN_DATE FORMAT a25;
COL MAX_DATE FORMAT a25;
SELECT TO_CHAR(MIN(lunch_date),'Month DD, YYYY, Day')AS
MIN_DATE,TO_CHAR(MAX(lunch_date),'Month DD, YYYY, Day')AS MAX_DATE FROM
lunch;
/*
Results:
MIN_DATE                  MAX_DATE
------------------------- -------------------------
May      22, 2021, Saturd June     23, 2021, Wednes
ay                        day
*/


--
***********************************************************************
********
--Question 5
/*
COUNT, SUM, and AVG Column Function practice:
a. Use the GROUP BY function to group by supplier id. The result set
should
include the supplier id, the count of the number of food items for each
```

```
supplier id, the total price of all food that is provided by that
supplier, and the
average price of all food. (FOOD table; COUNT function; SUM function; AVG
function)
i. When using count, use the wildcard * in the count function like so:
COUNT(*). This will count all records (per any filters in a WHERE clause)
ii. As for the total price and average price, format as currency
1. Example using the AVG function for formatting currency:
a. TO_CHAR(AVG(price), '$9999.99') as AVG_PRICE
b. This will apply a currency mask to the average price
c. Result Example: $5.50
d. Do this for both the SUM and AVG in your SQL statement
11 rows selected
*/
--6A
SELECT supplier_id, COUNT(*), TO_CHAR(SUM(price),'$9999.99')AS SUM_PRICE,
TO_CHAR(AVG(price),'$9999.99')AS AVG_PRICE FROM food
GROUP BY supplier_id;
--6B
SELECT supplier_id, COUNT(supplier_id), TO_CHAR(SUM(price),'$9999.99')AS
SUM_PRICE, TO_CHAR(AVG(price),'$9999.99')AS AVG_PRICE FROM food
GROUP BY supplier_id;
--6C
SELECT supplier_id, COUNT(price_upcharge),
TO_CHAR(SUM(price),'$9999.99')AS SUM_PRICE,
TO_CHAR(AVG(price),'$9999.99')AS AVG_PRICE FROM food
GROUP BY supplier_id;

/*
Results:
--6A
SUP   COUNT(*) SUM_PRICE AVG_PRICE
--- ---------- --------- ---------
Ard          3    $12.75     $4.25
Blu          2    $10.30     $5.15
Crm          4    $25.05     $6.26
Dpz          3    $10.25     $3.42
Foi          4    $14.75     $3.69
Gls          3     $9.70     $3.23
Hsd          4    $23.50     $5.88
Jd6          3     $7.25     $2.42
Jmd          4    $18.60     $4.65
Lak          3    $14.10     $4.70
Lss          2    $10.80     $5.40

11 rows selected.
--6B
SUP COUNT(SUPPLIER_ID) SUM_PRICE AVG_PRICE
--- ------------------ --------- ---------
Ard                  3    $12.75     $4.25
Blu                  2    $10.30     $5.15
Crm                  4    $25.05     $6.26
Dpz                  3    $10.25     $3.42
Foi                  4    $14.75     $3.69
```

```
Gls                  3     $9.70     $3.23
Hsd                  4    $23.50     $5.88
Jd6                  3     $7.25     $2.42
Jmd                  4    $18.60     $4.65
Lak                  3    $14.10     $4.70
Lss                  2    $10.80     $5.40

11 rows selected.
--6C
SUP COUNT(PRICE_UPCHARGE) SUM_PRICE AVG_PRICE
--- -------------------- --------- ---------
Ard                     0    $12.75     $4.25
Blu                     1    $10.30     $5.15
Crm                     4    $25.05     $6.26
Dpz                     1    $10.25     $3.42
Foi                     2    $14.75     $3.69
Gls                     1     $9.70     $3.23
Hsd                     3    $23.50     $5.88
Jd6                     3     $7.25     $2.42
Jmd                     3    $18.60     $4.65
Lak                     1    $14.10     $4.70
Lss                     0    $10.80     $5.40

11 rows selected.
*/


--
**********************************************************************
********
--Question 6
/*
Group by more than one attribute.
Use the query from 6a, but add an additional subgroup to further group
the items
by not only the supplier id but also the price upcharge. Sort by supplier
ID and
price upcharge.
25 rows selected
This means that you are first grouping by the supplier_ID, but within the
supplier_ID, you are then grouping by the Price_Upcharge.
*/
SELECT supplier_id, COUNT(*), TO_CHAR(SUM(price),'$9999.99')AS SUM_PRICE,
TO_CHAR(AVG(price),'$9999.99')AS AVG_PRICE FROM food
GROUP BY supplier_id,price_upcharge;

/*
Results:
SUP   COUNT(*) SUM_PRICE AVG_PRICE
--- ---------- --------- ---------
Hsd          1     $5.75     $5.75
Hsd          1     $4.25     $4.25
Crm          1     $5.25     $5.25
Crm          1     $7.20     $7.20
Crm          1     $4.00     $4.00
Jd6          2     $5.70     $2.85
Ard          3    $12.75     $4.25
Foi          1     $4.00     $4.00
Foi          1     $4.75     $4.75
Jmd          2    $10.85     $5.43
Lak          2     $8.60     $4.30

SUP   COUNT(*) SUM_PRICE AVG_PRICE
--- ---------- --------- ---------
Crm          1     $8.60     $8.60
Jd6          1     $1.55     $1.55
Lss          2    $10.80     $5.40
Dpz          2     $9.25     $4.63
Blu          1     $2.80     $2.80
Hsd          2    $13.50     $6.75
Jmd          1     $6.25     $6.25
Dpz          1     $1.00     $1.00
Blu          1     $7.50     $7.50
Jmd          1     $1.50     $1.50
Gls          1     $3.50     $3.50

SUP   COUNT(*) SUM_PRICE AVG_PRICE
--- ---------- --------- ---------
Foi          2     $6.00     $3.00
Gls          2     $6.20     $3.10
Lak          1     $5.50     $5.50

25 rows selected.
*/


--
****************************************************************************
********
--Question 7
/*
The company wants to review the credit limits for all workers.
(Worth 2 Questions)
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 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 city.
29 rows selected
C) Based on the SQL statement in part b, add an additional aggregate
function
to count the items in each group.
29 rows selected
D) Based on the SQL statement in part c, add a filter to only include
workers
```

```
hired after the year 2017.
8 rows selected
E) Based on the SQL statement in part d, add to the query to only include
departments with a total credit limit greater than 30
(HAVING function using SUM).
2 rows selected
*/
--8A
SELECT dept_code, SUM(credit_limit) FROM worker
GROUP BY dept_code;
--8B
SELECT dept_code, SUM(credit_limit) FROM worker
GROUP BY dept_code, city;
--8C
SELECT dept_code, SUM(credit_limit), COUNT(*) FROM worker
GROUP BY dept_code, city;
--8D
SELECT dept_code, SUM(credit_limit), COUNT(*) FROM worker
WHERE hire_date>'31-DEC-2017'
GROUP BY dept_code, city;
--8E
SELECT dept_code, SUM(credit_limit), COUNT(*) FROM worker
WHERE hire_date>'31-DEC-2017'
GROUP BY dept_code, city
HAVING SUM(credit_limit)>30;

/*
Results:
--8A
DEP SUM(CREDIT_LIMIT)
--- -----------------
Aud                69
Sal                53
Leg                87
Hmn                66
Acc                72
Tch               137
                  202
Fin                48
Exe                45
Com                22
Srv                89

11 rows selected.
--8B
DEP SUM(CREDIT_LIMIT)
--- -----------------
Aud                22
Hmn                17
Acc                18
Acc                24
                   25
Exe                45
Hmn                49
```

```
Sal                33
Aud                25
Aud                22
Sal                20

DEP SUM(CREDIT_LIMIT)
--- -----------------
                   27
Acc                30
Srv                30
Tch                25
                   20
Tch                25
Tch                60
Leg                32
Srv                33
                   50
Leg                55

DEP SUM(CREDIT_LIMIT)
--- -----------------
Srv                26
                   55
Fin                20
                   25
Com                22
Fin                28
Tch                27

29 rows selected.
--8C
DEP SUM(CREDIT_LIMIT)   COUNT(*)
--- ----------------- ----------
Aud                22          1
Hmn                17          1
Acc                18          1
Acc                24          1
                   25          1
Exe                45          1
Hmn                49          2
Sal                33          1
Aud                25          1
Aud                22          1
Sal                20          1

DEP SUM(CREDIT_LIMIT)   COUNT(*)
--- ----------------- ----------
                   27          1
Acc                30          1
Srv                30          1
Tch                25          1
                   20          1
Tch                25          1
Tch                60          2
```

```
Leg                32         1
Srv                33         1
                   50         2
Leg                55         2

DEP SUM(CREDIT_LIMIT)   COUNT(*)
--- ----------------- ----------
Srv                26         1
                   55         2
Fin                20         1
                   25         1
Com                22         1
Fin                28         1
Tch                27         1

29 rows selected.
--8D
DEP SUM(CREDIT_LIMIT)   COUNT(*)
--- ----------------- ----------
                   25         1
Hmn                49         2
Aud                25         1
Sal                20         1
                   27         1
Tch                25         1
Leg                33         1
Tch                27         1

8 rows selected.
--8E
DEP  SUM(CREDIT_LIMIT)    COUNT(*)
--- ------------------- ----------
Hmn                  49          2
Leg                  33          1

2 rows selected.
*/


--
***************************************************************************
--Question 8
/*
Practice nested subqueries:
A) Using the subquery approach, list the supplier id, product code,
description
and price for all food priced less than the average price of all food for
sale.
(There is an example in the Functions slides).
16 rows selected
B) Check the calculation. Run the statement in the nested subquery as a
separate SQL statement so that you can see what the subquery does as its
own SQL statement. Round the calculation to 3 decimal places.
```

```
1 row selected
*/
--9A
SELECT supplier_id, product_code, description, price FROM food
WHERE price<(SELECT AVG(price) FROM food);
--9B
SELECT ROUND(AVG(price),3) FROM food;
/*
Results:

--9A
SUP PR DESCRIPTION                   PRICE
--- -- -------------------- ----------
Ard Ds PB Cookie                      1.25
Hsd Sp Chicken Soup                   4.25
Crm Br Wheat Bagel                       4
Foi Vt Broccoli Salad                    4
Foi Ff French Fries                    1.5
Jd6 Vr Soda                           2.25
Jd6 Cf Coffee                         1.55
Jd6 Ds Brownie                        3.45
Jmd Vr Iced Tea                       2.85
Jmd Vt Cole Slaw                       1.5
Dpz Br Dinner Roll                       1

SUP PR DESCRIPTION                   PRICE
--- -- -------------------- ----------
Dpz Sc Cheese Sauce                    .75
Gls Ds Sugar Cookie                    3.5
Gls Br Breadstick                     1.25
Blu Cp Chips                           2.8
Lak Br Cheese Stick                   2.35

16 rows selected.
--9B
ROUND(AVG(PRICE),3)
-------------------
              4.487

1 row selected.
*/
```