Click or tap here to enter text.

**TERM :**

**A picture containing logo

Description automatically generated**

| **Course & Section Code:** | **COMP214** |
| --- | --- |
| **Course Name:** | **Advanced Database Concept** |
|  |  |
| **Q&A / Virtual Office Hour:** |  |
| **Instructor Name & Email:** | Ersan Cam  [ecam@my.centennialcollege.ca](mailto:ecam@my.centennialcollege.ca) |

Assigment#3 \_Version2

(Database Programming Fundamentals)

**Due Date :**

Instructions for delivery of this Labs file back to instructor.

**Step 1:** Download this word copy of Lab document.

**Step2:** Work on your question in SQL Developer.

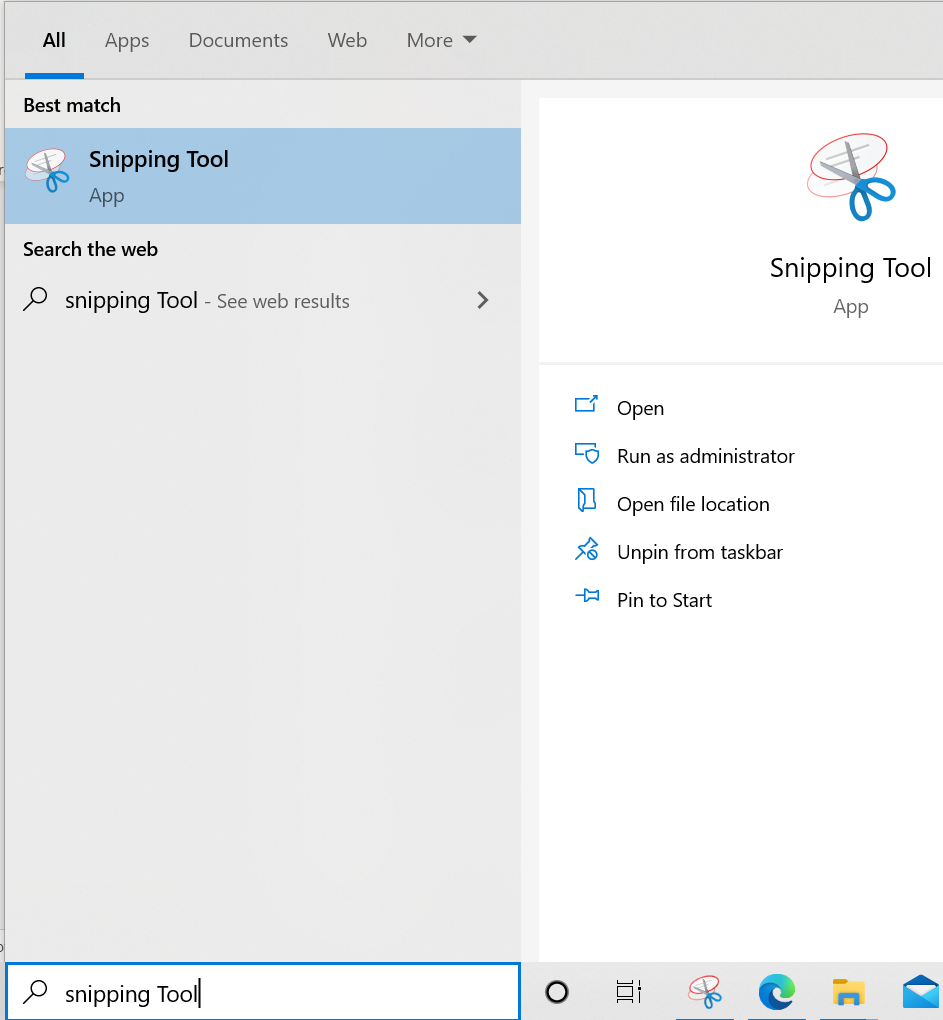
**Step3:** Once you solve the problem copy paste the code under each question and Highlight with RED color

**Step4:** Also go to your Sql Developer and capture screen entire screen with command you execute and result at the bottom. Use Snipping tool in windows to capture screen shot. Below picture shows how to open free windows based snipping screen capture tool

Step5: Drop your finalized & saved word document to respective Lab dropbox assignment folder.

**Please note that Instructor has the right to call out any students randomly to ask demonstration their solution and walk thru their work and justify their answers in one on one breakout room.**

**The tool to capture screen shot.**



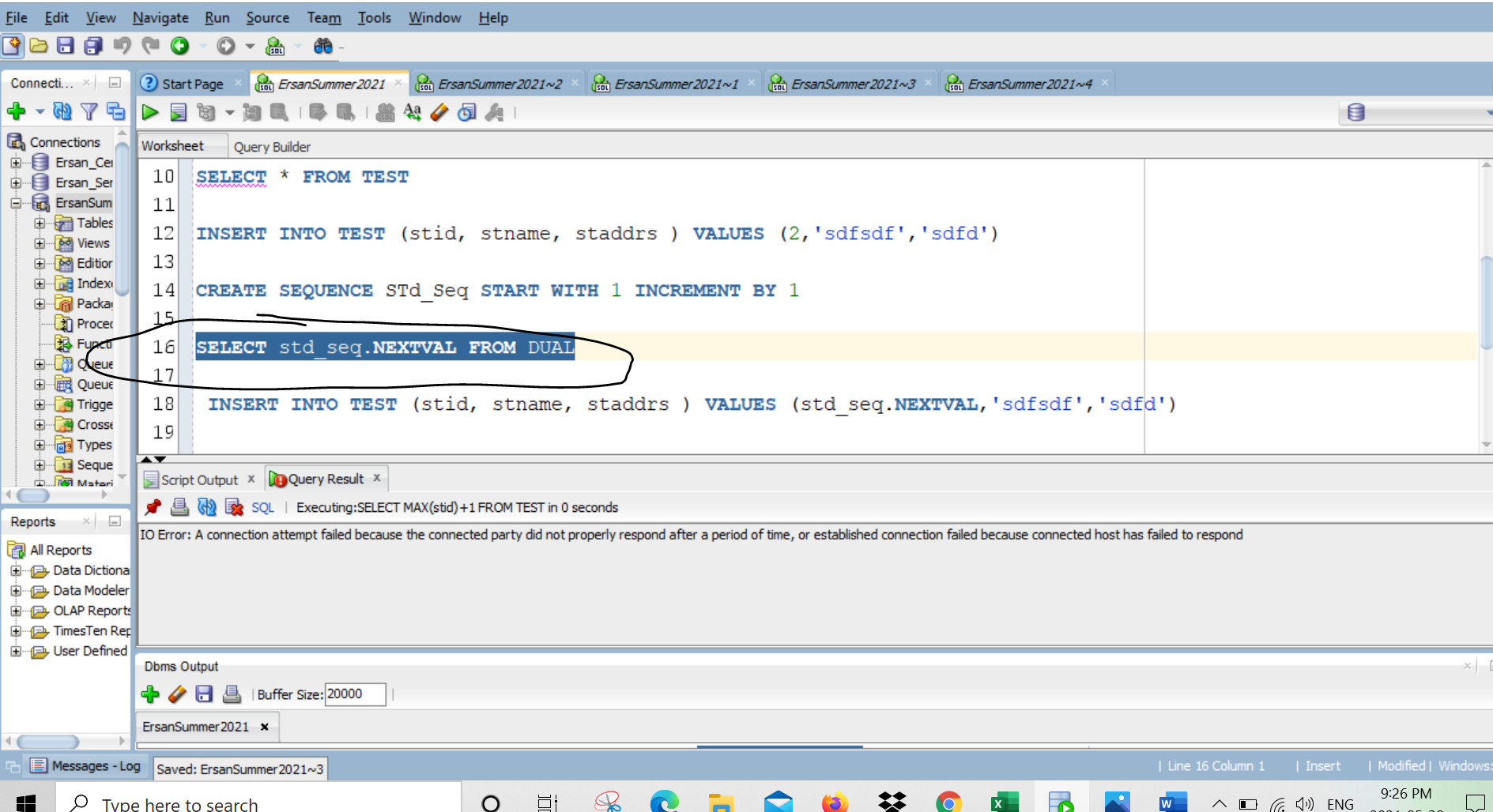
**Demo Question:** How to execute Sequence.Nextval command?

Demo Answer:

Copy paste of actual command: SELECT std\_seq.NEXTVAL FROM DUAL;

Demo Screen shot:

Copy paste screen shot of same command



**Total 4 questions, each 25 points, Total 100**

**Solve Question 1 by working with OT\_XXX database**

1. **Data processing with COMPISITE variables**

1-a) Create High level Flowchart of below problem before you solve it with IF statement

You can use draw.io (https://app.diagrams.net/) free online flow chart diagram tool or draw basic flow on a paper and take a picture and attach image to this same word document

1-b) The OT company wants to display a rating value for customer’s orders for each product

The rating assignments are outlined in the following table:

Develop a SQL statement to find total amount per each product\_id and load this result to a cursor

Hint:

In the OT\_ORDER\_ITEMS table group by based on Product\_id and calculate TOTAL amount ( formula SUM(quantity\*unit\_price) group by product\_id)

Hint for the query:

SELECT product\_id, SUM(quantity\*unit\_price) FROM table

GROUP ….

….

Go thru this result with CURSOR FOR LOOP and determine each record by assigning category code

If

**N**

|  |  |
| --- | --- |
| **Total amount** | **Category** |
| Between 100,000 and 300,000 | Low |
| Between 300,001 and 700,000 | Mid low |
| Between 700,001 and 999,999 | Mid High |
| Greater than or equal to 1,000,000 | Extreme High |
|  |  |

PL/SQL code has to print each Product\_id, total amount and then also category code for each record

**Product Id total**

1 292932.43 Low

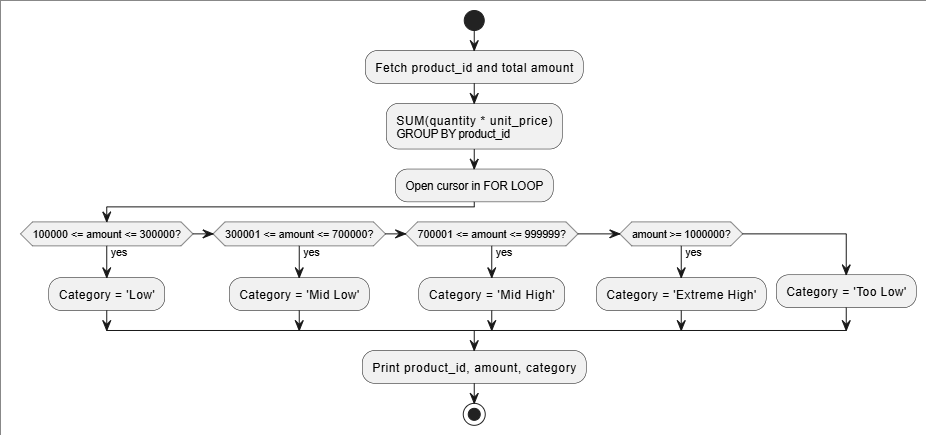
2 585092.71 Mid Low

3 117598.53 Low

4 1190695.59 Extreme High

***………………***

***Explain whole logic that you use in this program and explain your reasoning***



DECLARE

CURSOR c\_orderCat IS

SELECT oi.product\_id, SUM(oi.unit\_price \* quantity) as total

FROM OT\_ORDER\_ITEMS oi

JOIN OT\_PRODUCTS p ON oi.product\_id = p.product\_id

GROUP BY oi.product\_id;

v\_category VARCHAR(20);

BEGIN

DBMS\_OUTPUT.PUT\_LINE(RPAD('Product Id', 12) || ' ' || RPAD('Total',12) || ' ' || RPAD('Category', 15));

FOR rec\_orderCat in c\_orderCat LOOP

IF rec\_orderCat.total BETWEEN 100000 AND 300000 THEN

v\_category := 'Low';

ELSIF rec\_orderCat.total BETWEEN 300001 AND 700000 THEN

v\_category := 'Mid Low';

ELSIF rec\_orderCat.total BETWEEN 700001 AND 999999 THEN

v\_category := 'Mid High';

ELSIF rec\_orderCat.total >= 1000000 THEN

v\_category := 'Extreme High';

ELSE

v\_category := 'Too low';

END IF;

DBMS\_OUTPUT.PUT\_LINE(

RPAD(rec\_orderCat.product\_id, 12) || ' ' ||

RPAD(rec\_orderCat.total, 12) || ' ' ||

RPAD(v\_category, 15)

);

END LOOP;

END;

I declared and assigned a cursor to the product\_id and total of the order items and product tables. I also declared a scalar variable, v\_category to hold the to-be outputted category for the row. I printed table headers at the top of the execution section to make the successive DBMS output (the informational rows) make more sense. The *total* column of each record/ row of when the for loop iterates through the cursor will be tested in the IF ELSIF … ELSE structure and assign v\_category. Then, each row gets printed out with the category.

A screenshot of a computer

AI-generated content may be incorrect.

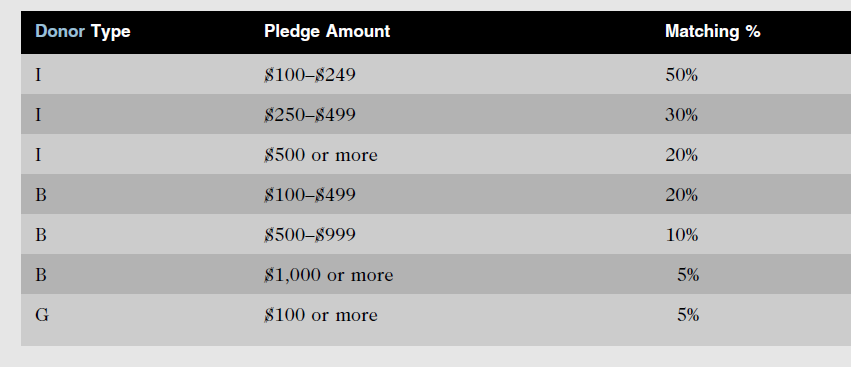
**2. Using Nested IF Statements ( Main IF will have another IF THEN ELSE as nested (child-Inner) inside)**

An organization has committed to matching pledge amounts based on the donor type and

pledge amount. Donor types include I = Individual, B = Business organization, and G = Grant

funds. The matching percents are to be applied as follows:

**Donor**



Create a PL/SQL block using nested IF statements to accomplish the task. Input values for

the block are the donor type code and the pledge amount.

***Explain whole logic that you use in this program and explain your reasoning***

ACCEPT donorType CHAR PROMPT 'Enter donor type (I/B/G): '

ACCEPT pledgeAmount NUMBER PROMPT 'Enter pledge amount: '

DECLARE

v\_donorType CHAR(1) := '&donorType';

v\_donorTypeFull VARCHAR2(30);

v\_pledgeAmt NUMBER := &pledgeAmount;

v\_matching NUMBER;

v\_total NUMBER;

BEGIN

IF v\_donorType = 'I' THEN

v\_donorTypeFull := 'Individual';

IF v\_pledgeAmt BETWEEN 100 AND 249 THEN

v\_matching := 0.5;

ELSIF v\_pledgeAmt BETWEEN 250 AND 499 THEN

v\_matching := 0.3;

ELSIF v\_pledgeAmt >= 500 THEN

v\_matching := 0.2;

ELSE

-- 0, not one of the options

v\_matching := 0;

END IF;

ELSIF v\_donorType = 'B' THEN

v\_donorTypeFull := 'Business';

IF v\_pledgeAmt BETWEEN 100 AND 499 THEN

v\_matching := 0.2;

ELSIF v\_pledgeAmt BETWEEN 500 AND 999 THEN

v\_matching := 0.1;

ELSIF v\_pledgeAmt >= 1000 THEN

v\_matching := 0.05;

ELSE

-- 0, not one of the options

v\_matching := 0;

END IF;

ELSIF v\_donorType = 'G' THEN

v\_donorTypeFull := 'Grant funds';

IF v\_pledgeAmt >= 100 THEN

v\_matching := 0.05;

ELSE

-- 0, not one of the options

v\_matching := 0;

END IF;

ELSE

-- invalid type

DBMS\_OUTPUT.PUT\_LINE(v\_donorType || ' is an invalid Donor Type.');

END IF;

IF v\_matching = 0 THEN

DBMS\_OUTPUT.PUT\_LINE(v\_pledgeAmt || ' is too low.');

ELSIF v\_donorType IN ('I', 'B', 'G') THEN

DBMS\_OUTPUT.PUT\_LINE('Donor Type of (' || v\_donorType || ')' || v\_donorTypeFull || ' for $' || v\_pledgeAmt);

DBMS\_OUTPUT.PUT\_LINE('The organization is matching ' || v\_matching \* 100 || '% ($' || v\_pledgeAmt \* v\_matching || ') ');

v\_total := v\_pledgeAmt + (v\_pledgeAmt \* v\_matching);

DBMS\_OUTPUT.PUT\_LINE('Total donation is $' || v\_total);

END IF;  
END;  
  
  
  
I added in some ACCEPT PROMPT statements to make the prompts more understandable.

I then gave v\_donorType a CHAR(1) type as that is the most the donorType is supposed to need.

The v\_donorTypeFull variable is to hold the eventual full donor type name.

The v\_pledgeAmt holds the number given by the user from the prompt.

The v\_matching will hold the percentage, in decimal format, for usage in printing and for calculating the donor match.

The v\_total is a variable to hold the calculate total, pledgeAmt + donor match amount.

In the execution block I start the first IF ELSIF … ELSE structure. I chose to test the donorType first as it is a more concise method to test 3 options that have varying inner outcomes (the pledge amount tests are different).

I assign the donorTypeFull to the respective title and test the pledgeAmt on each of the donorType’s cases.

I assign v\_matching to 0 in each ELSE, default, case to use later for printing out that the input was too low.

The ELSE statement for the outer loop will also print that it was an invalid donor type.

At the end, in the last IF structure, I test v\_matching on if it is 0 as a 0 v\_matching variable means that the pledgeAmt was too low and in the case that v\_matching was not 0 and the donorType is I, B, or G, do the calculations and printing out of information.

Multiplying the v\_matching by 100 gets us the percentage value of the v\_matching decimal.

And to get total, I added the pledgeAmt with the pledgeAmt multiplied by the v\_matching decimal.

A screenshot of a computer

AI-generated content may be incorrect.

**Use OT\_XXX Database for Question 3**

**3. Procedure to find Product sales total amount and product name**

Develop a stored procedure named **sp\_product\_total** to return total amount and product name.

This procedure will accept 2 Input parameters… order\_id and product\_id same as IT\_ORDER\_ITEMS

Then in the begin and end , procedure will first find total amount for this order\_id and product id by multiplying quantity \* unit\_price

Then also by using either JOIN or subquery techniques, procedure will have to find product name in OT\_PRODUCTS table for a given product\_id

And procedure will have to return two values back to caller… total and product name

***Explain whole logic that you use in this program and explain your reasoning***

*===============================================================*

CREATE OR REPLACE PROCEDURE SP\_PRODUCT\_TOTAL (

order\_id IN OT\_ORDER\_ITEMS.order\_id%TYPE,

product\_id IN OT\_ORDER\_ITEMS.product\_id%TYPE,

v\_total OUT NUMBER,

v\_product\_name OUT VARCHAR2

)

AS

BEGIN

SELECT SUM(oi.quantity \* oi.unit\_price), p.product\_name

INTO v\_total, v\_product\_name

FROM OT\_ORDER\_ITEMS oi

JOIN OT\_PRODUCTS p ON oi.product\_id = p.product\_id

WHERE oi.order\_id = SP\_PRODUCT\_TOTAL.order\_id

AND oi.product\_id = SP\_PRODUCT\_TOTAL.product\_id

GROUP BY p.product\_name;

END SP\_PRODUCT\_TOTAL;

I created the input variables order\_id and product\_id and then two output variables to return information from the procedure, v\_total and v\_product\_name.

In the execution block, I wrote a select statement to return the total and product name.

I used a JOIN on product\_id instead of a subquery because I thought it was a more concise solution.

The oi\_order\_id had to equal the input variable order\_id and the oi.product\_id had to equal the input variable product\_id. Due to the fact that order\_id is a column on the OT\_ORDER\_ITEMS table and product\_id being a column on both OT\_ORDER\_ITEMS and OT\_PRODUCTS, I had to specify that the variables were from SP\_PRODUCT\_TOTAL scope.

VARIABLE v\_total VARCHAR2(100)

VARIABLE v\_product\_name VARCHAR2(100)

EXEC sp\_product\_total(70, 32, :v\_total, :v\_product\_name)

PRINT v\_total

PRINT v\_product\_name  
  
I then created two variables and executed the procedure with 70 as the order\_id and 32 as the product\_id

Once the procedure finishes running, the total and product name gets printed out.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Use LGXXXX Database for Question 4**

**Use LG\_LargeCo\_ORA.txt scripts under Assignment3 folder to build necessary tables**

**4- Function to find max invoice amount**

Develop a function named **fn\_finding\_max\_invoiceamount** with two input and one return value.

This function will accept one parameter p\_employee\_id same as LGINVOICE and then

In BEGIN and END it will select all the invoice records belong to this particular employee.

Then among all these invoice details your function has to find max INV\_TOTAL of all and return this value to caller program.

You can use simple MAX function to find among all invoices belong to this employee , or use array and sort them and find max , or use Cursor and process records to find max of all. I am OK any of these methods.

Let’s assume function takes parameter p\_employee\_id as 83734 . It has to return 496.38 as max invoice amount of all entries for this employee

Test the function with different employee and share the screen shots

*A screenshot of a spreadsheet

Description automatically generated with medium confidence*

***Explain whole logic that you use in this program and explain your reasoning***