**TERM : Summer 2025**

**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#2 -Version2

**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 keep 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.**

Graphical user interface, text, application

Description automatically generated

**Demo : How to submit answer to professor with answer inside this same Word document ?**

**Demo Question:**

List of employee last name, first name , department name and city they work for all the employees?

**Demo Answer:**

It is copy paste of actual command from SQL Developer’s editor screen . DO NOT PASTE SCREEN SHOT here as I need to copy paste and test /execute your command.

*SELECT last\_name, first\_name, department\_name, city*

*FROM hr\_employees e JOIN hr\_departments d*

*ON e.department\_id=d.department\_id*

*JOIN hr\_locations l*

*ON d.location\_id=l.location\_id*

Demo Screen shot with Snipping tool

Copy paste screen shot of same command and output

A screenshot of a computer

Description automatically generated

1. **Review below code.. Which lines have problem and explain why?**

1 DECLARE

2 TYPE ss\_dept IS RECORD

3 ( department\_name hr\_departments.department\_name%TYPE , manager\_id hr\_departments.manager\_id%TYPE );

4 cv\_deptSub ss\_dept;

5 cv\_deptFull hr\_departments%ROWTYPE;

6 BEGIN

7 SELECT \* INTO cv\_deptSub FROM hr\_Departments where department\_id=90;

8 SELECT \* INTO cv\_deptFull FROM hr\_Departments where department\_id=90;

9 SELECT department\_name,manager\_id INTO cv\_deptSub FROM hr\_Departments where department\_id=90;

10 SELECT manager\_id,department\_name INTO cv\_deptSub FROM hr\_Departments where department\_id=90;

11 SELECT department\_name INTO cv\_deptSub FROM hr\_Departments where department\_id=90;

12 END;  
  
**Line 7 is wrong because cv\_deptSub is a record of ss\_dept which has 2 fields and we are trying to SELECT all (\*) from hr\_departments which has 4 columns. The assignment of columns into the record is ambiguous and cannot happen.**

**Line 10 is wrong because the cv\_debtSub expects columns in the order of department\_name, manager\_id and the selection of columns is in the order of manager\_id, department\_name.**

**Line 11 is wrong because cv\_debtSub expects two columns and the selection is only giving one.**

1. **Review below code.. Which lines have problem and explain why?**

1 DECLARE

2 TYPE t\_rec IS RECORD

3 (sv\_sal number(8),

4 sv\_minsal number(8) default 1000,

5 sv\_hire\_date hr\_employees.hire\_date%type,

6 cv\_recemp %rowtype);

7 cv\_myrec t\_rec;

8 BEGIN

9 cv\_myrec.sv\_sal := cv\_myrec.sv\_minsal + 500;

10 cv\_myrec := sysdate;

11 IF sv\_sal > 1000 THEN

12 SELECT \* INTO cv\_myrec.cv\_recemp FROM hr\_employees WHERE employee\_id = 100;

13 DBMS\_OUTPUT.PUT\_LINE( cv\_recemp.last\_name ||' '|| to\_char(cv\_myrec.sv\_hire\_date) ||' '|| to\_char(cv\_myrec.sv\_sal));

14 END IF;

15 END;

**Line 6 is wrong as %rowtype is not reference any column, the syntax is completely wrong. Needs to be in format of table.column%type or variable%type.**

**Line 10 is wrong because cv\_myrec is a record type and you cannot assign the record variable to a date.**

**Line 11 is wrong because sv\_sal is not a variable on its own. It is a variable inside the record and can be referenced only by a record variable.**

**Line 12 would be right is cv\_recemp was valid, but since cv\_recemp is not valid, this line is not valid.**

**Line 13 is wrong because cv\_recemp is an invalid variable in the record and is not a table or record itself. What could be valid is if cv\_recemp was cv\_myrec (assuming the record itself was correct).**

1. **Delivery cost calculation**  
   Delivery prices calculations are typically based on the number of items to be delivered.

For example:   
0 —99 $1.50 per item   
100 — 499 $2.28 per item   
500 — 749 $2.27 per item   
750 — 1000 $2.26 per item   
over 1000 $2.25 per item

Ask end user which province to deliver as well

Create a program that ask end user state and also how many items to be delivered…

Then program will prompt the user for the number of items to be deliver and then displays the price per item and the total price for the delivery job.

**Rules:**

If end user entered province as **ON (ONTARIO)** calculation is

0 —99 $1.50 per item   
100 — 499 $2.28 per item   
500 — 749 $2.27 per item   
750 — 1000 $2.26 per item   
over 1000 $2.25 per item

Or if user entered province as **BC** calculation is

0 —99 $1.70 per item   
100 — 499 $2.35 per item   
500 — 749 $2.45 per item   
750 — 1000 $2.50 per item   
over 1000 $2.75 per item

All other provinces

0 —99 $0.70 per item   
100 — 499 $1.35 per item   
500 — 749 $1.45 per item   
750 — 1000 $1.50 per item   
over 1000 $1.75 per item

**Explain your reasoning to support your program logic**

**-- 3. Delivery cost calculation**

**DECLARE**

**v\_state CHAR(2) := &state;**

**v\_num\_items NUMBER := &number\_of\_items;**

**v\_price\_item NUMBER;**

**v\_total\_price NUMBER;**

**BEGIN**

**IF v\_state = 'ON' THEN**

**v\_price\_item := CASE**

**WHEN v\_num\_items BETWEEN 0 AND 99 THEN 1.50**

**WHEN v\_num\_items BETWEEN 100 AND 499 THEN 2.28**

**WHEN v\_num\_items BETWEEN 500 AND 749 THEN 2.27**

**WHEN v\_num\_items BETWEEN 750 AND 1000 THEN 2.26**

**WHEN v\_num\_items > 1000 THEN 2.25**

**ELSE 0**

**END;**

**ELSIF v\_state = 'BC' THEN**

**v\_price\_item := CASE**

**WHEN v\_num\_items BETWEEN 0 AND 99 THEN 1.70**

**WHEN v\_num\_items BETWEEN 100 AND 499 THEN 2.35**

**WHEN v\_num\_items BETWEEN 500 AND 749 THEN 2.45**

**WHEN v\_num\_items BETWEEN 750 AND 1000 THEN 2.50**

**WHEN v\_num\_items > 1000 THEN 2.75**

**ELSE 0**

**END;**

**ELSE**

**v\_price\_item := CASE**

**WHEN v\_num\_items BETWEEN 0 AND 99 THEN 0.70**

**WHEN v\_num\_items BETWEEN 100 AND 499 THEN 1.35**

**WHEN v\_num\_items BETWEEN 500 AND 749 THEN 1.45**

**WHEN v\_num\_items BETWEEN 750 AND 1000 THEN 1.50**

**WHEN v\_num\_items > 1000 THEN 1.75**

**ELSE 0**

**END;**

**END IF;**

**v\_total\_price := v\_price\_item \* v\_num\_items;**

**DBMS\_OUTPUT.PUT\_LINE('Price per item: $' || v\_price\_item);**

**DBMS\_OUTPUT.PUT\_LINE('Total price of delivery: $' || v\_total\_price);**

**END;**

**A screenshot of a computer

AI-generated content may be incorrect.**

**I declared 4 variables, 2 of which to prompt the user to enter values and 2 to hold values for later calculations.**

**In the execution section I have an IF, ELSIF, ELSE structure to test v\_state. I test v\_state for ‘ON’, ‘BC’ and if v\_state does not test true for ‘ON’ or ‘BC’ then we the ELSE block will execute.**

**In each block of the IF structure, a case structure runs that tests the number of items that was inputted by the user. The result of the case structure will be assigned to v\_price\_item, the price of individual items.**

**Then, the calculation for the total price of items in the delivery which is v\_price\_item, the price of individual items, multiplied by v\_num\_items, the number of items in the delivery.**

**Lastly, the price per item and the total price of the delivery is printed out to the DBMS output.**

Develop a code where you have one FOR LOOP from 10 to 30

Program must skip all with CONTUNUE the iteration numbers from 20 to 30 . First section , from 10 to 20 no iteration will be skipped but after 20 to 30 all of those iterations must be skipped

Declare a variable (v\_sum with DEFAULT value as 1 )to calculate total value before CONTINUE and after CONTINUE

Hint:

FOR i IN firstNumber..LastNumber LOOP

V\_sum calculation

Print (‘before CONTINUE total value’ );

CONTINUE IF iteration is 20 or more

V\_sum calculartion

Print (‘after CONTINUE total value ‘ );

END LOOP;

As an example of output

Let’s say loop starts from 10, for first iteration.

First calculation will be

v\_sum := v\_sum + i (v\_sum := 1+10 )🡪 Print 11

After CONTINUE

v\_sum:=v\_sum+i ( 11 (10 +1) + 10)

print 21

Explain your reasoning to support your program logic

-- 4.

DECLARE

v\_sum NUMBER := 1;

BEGIN

FOR i IN 10..30 LOOP

v\_sum := v\_sum + i;

DBMS\_OUTPUT.PUT\_LINE(v\_sum);

CONTINUE WHEN i >= 20;

v\_sum := v\_sum + i;

DBMS\_OUTPUT.PUT\_LINE(v\_sum);

END LOOP;

END;

A screenshot of a computer

AI-generated content may be incorrect.

I first declare and set the default of 1 to v\_sum.

In the execution block, a for loop with the variable i is assigned to the range of 10 to 30.

As the loop runs, v\_sum will always increment by i at least once.

Once i becomes 20, then the second increment of v\_sum by i will not run. Only the ‘before CONTINUE total value’ will print.

When i becomes 30 the loop terminates and since there are no more executable lines, the entire program terminates.

**5) In this task you will ask the end user how many side shape has and based on the entry you will print name of the shape**

For example ;

if user answers 3 for number of side then you need to print Triangle

if user answers 4 for number of side then you need to check again with child IF

IF 4 sides are equal (you can ask secondary prompt to end user asking are they equal or not? If they all equal then print Square otherwise print Rectangle

if user answers 5 for number of side then you need to print Pentagon

if user answers 6 for number of side then you need to print Hexagon

if user answers 8 for number of side then you need to print Octagon

Anything else other than these numbers print No shape available

Explain your reasoning to support your program logic

-- 5. In this task you will ask the end user how many side shape has and based on the entry you will print name of the shape

DECLARE

v\_num\_sides NUMBER := &num\_of\_sides;

BEGIN

CASE

WHEN v\_num\_sides = 3 THEN DBMS\_OUTPUT.PUT\_LINE('Triangle');

WHEN v\_num\_sides = 4 THEN

IF UPPER('&equal\_or\_not') IN ('TRUE', '1') THEN

DBMS\_OUTPUT.PUT\_LINE('Square');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Rectangle');

END IF;

WHEN v\_num\_sides = 5 THEN DBMS\_OUTPUT.PUT\_LINE('Pentagon');

WHEN v\_num\_sides = 6 THEN DBMS\_OUTPUT.PUT\_LINE('Hexagon');

WHEN v\_num\_sides = 8 THEN DBMS\_OUTPUT.PUT\_LINE('Octagon');

ELSE DBMS\_OUTPUT.PUT\_LINE('No Shape Available');

END CASE;

END;

A screenshot of a computer

AI-generated content may be incorrect.

Explaining my output:  
I entered, in order (v\_num\_sides first, equal\_or\_not second):

[ The relevance of 1, or any other number being used doesn’t matter other than when v\_num\_sides is 4)

1, 1 -> No Shape Available

3, 1 -> Triangle

4, 1 -> Square

4, 0 -> Rectangle

5, 1 -> Pentagon

6, 1 -> Hexagon

8, 1 -> Octagon

I first declared v\_num\_sides and prompted for input.

In the execution block, I made a case structure to test v\_num\_sides.

Each case will output the shape that coincides with the number of sides inputted or ‘No Shape Available’.

For v\_num\_sides = 4, a child IF structure is present to prompt the user for if the sides are equal or not. If they are equal, which is a test on the uppercase value of the input with ‘TRUE’ or ‘1’, Square is printed, if the test proves false, Rectangle is printed.

The ELSE statement, the default case, will print No Shape Available which is why when v\_num\_sides is 1 the program prints No Shape Available.

**6.** **Ask the user their name and how many times they want to output their name (using a for loop).**

Hint:

*DECLARE*

*v\_name varchar(50) := '&name';*

*v\_hmtimes number :=&numb;*

*BEGIN*

*LOOP*

*END;*

-- 6. Ask the user their name and how many times they want to output their name (using a for loop).

DECLARE

v\_name varchar(50) := '&name';

v\_num\_times number :=&numb;

BEGIN

FOR i in 1..v\_num\_times LOOP

DBMS\_OUTPUT.PUT\_LINE(v\_name);

END LOOP;

END;

A screenshot of a computer

AI-generated content may be incorrect.

Using the HINT code, I modified v\_hmtimes to be the name of v\_num\_times just because I wanted to. The names don’t matter for any functionality, just for readability purposes.

In the execution block, a FOR loop is created that assigned i to the range of 1 to v\_num\_times, the number the user inputs after they input their name.

In the screenshot, I entered Lucas for v\_name and 7 for v\_num\_times.

This resulted in Lucas being outputted 7 times.

The reason I started the FOR loop at 1 and not 0 is because the loop will run for the upper bound of v\_num\_times. If I started the loop at 0, 8 Lucas’s would have been outputted.

7.

Create a visitors application that prompts the user for the number of visitors each day for the past 5 days and then displays the average number of visitors per day. (use & to accept new value)

Use PL/SQL Array (NUMBER type basic composite array type ) and array has to have 5 members only. The application output should look like same as following:

Test: If you receive below numbers from user then your average should be 234. Display average on screen.

1: 150

2: 220

3: 180

4: 300

5: 320

The average number of visitors is: 234

Hint :

*Once you create Array*

*DECLARE*

*TYPE my\_array\_type IS TABLE OF number INDEX BY PLS\_INTEGER;*

*v\_n\_array my\_array\_type;*

*Then in the BEGIN and END block accept user entry as new value for each array members. DO NOT Use LOOP to accept new & as it is not design to get new value via loop.. Instead use separate 5 different &substitution value to get each values from user input*

*BEGIN*

*v\_n\_array(1):= &n1*

*v\_n\_array(2):=&n2*

*…………….*

*…………*

*Collect all v\_n\_array members (5 different numbers) and then do calculations like finding average.*

*v\_avg := total of all v\_n\_array(x) / 5;…………*

*Then Display them*

*DBMS\_OUTPUT.PUT\_LINE (‘*The average number of visitors is: ‘ || v\_avg *)*

*END;*

Explain your reasoning to support your program logic

DECLARE

type num\_visitors\_type IS TABLE OF NUMBER INDEX BY PLS\_INTEGER;

v\_n\_array num\_visitors\_type;

v\_total\_sum NUMBER := 0;

v\_avg NUMBER;

BEGIN

v\_n\_array(1) := &num1;

v\_n\_array(2) := &num2;

v\_n\_array(3) := &num3;

v\_n\_array(4) := &num4;

v\_n\_array(5) := &num5;

for i in 1..v\_n\_array.COUNT LOOP

v\_total\_sum := v\_total\_sum + v\_n\_array(i);

END LOOP;

v\_avg := v\_total\_sum / v\_n\_array.COUNT;

DBMS\_OUTPUT.PUT\_LINE('The average number of visitors is: ' || v\_avg);

END;

First, the array type is defined to be a table of numbers indexed by integer.

A variable is defined with the array type.

Two variables are also defined and v\_total\_sum is assigned, to hold the total sum of the array’s elements and to hold the average later in the program.

In the execution block, 5 array values are assigned to values inputted by the user.

Then, a FOR loop was created with the range of 1 to the number of elements (COUNT) in the array. To sum the elements, the value of the array at index ‘I’ is added to the v\_total\_sum variable.

Lastly, the average is calculated by dividing the v\_total\_sum by the number of elements in the array and is outputted.

A screenshot of a computer

AI-generated content may be incorrect.I opted to use v\_n\_array.COUNT instead of hard coding the number 5 in because, in the case that the program was modified to add a 6th element to the array, we would not run into any errors.  
In the example below, I used the values: 15, 80, 600, 8492, and 34 which averages to 1844.2

**8. Earthquake impact scale measurement program**

The following table contains earthquake magnitude ranges on the Richter scale and their descriptors:

Table

Description automatically generated

 Write a program that reads magnitude from the user and displays the appropriate

descriptor as part of a meaningful message. For example, if the user enters 5.5 then

your program should indicate that a magnitude 5.5 earthquake is considered to be a

moderate earthquake

Explain your reasoning to support your program logic

DECLARE

v\_magnitude NUMBER := &magnitude;

v\_descriptor VARCHAR(20);

BEGIN

CASE

WHEN v\_magnitude < 2.0 THEN v\_descriptor := 'Micro';

WHEN v\_magnitude >= 2.0 AND v\_magnitude < 3.0 THEN v\_descriptor := 'Very minor';

WHEN v\_magnitude >= 3.0 AND v\_magnitude < 4.0 THEN v\_descriptor := 'Minor';

WHEN v\_magnitude >= 4.0 AND v\_magnitude < 5.0 THEN v\_descriptor := 'Light';

WHEN v\_magnitude >= 5.0 AND v\_magnitude < 6.0 THEN v\_descriptor := 'Moderate';

WHEN v\_magnitude >= 6.0 AND v\_magnitude < 7.0 THEN v\_descriptor := 'Strong';

WHEN v\_magnitude >= 7.0 AND v\_magnitude < 8.0 THEN v\_descriptor := 'Major';

WHEN v\_magnitude >= 8.0 AND v\_magnitude < 10.0 THEN v\_descriptor := 'Great';

WHEN v\_magnitude >= 10.0 THEN v\_descriptor := 'Meteoric';

END CASE;

DBMS\_OUTPUT.PUT\_LINE('A magnitude of ' || v\_magnitude || ' is considered to be a ' || v\_descriptor || ' earthquake');

END;

I used a case control structure to test each scenario on the inputted magnitude. I used ‘greater than and equal to’ as well as ‘less than’ logical operators to accurately test the variable. I did this instead of using BETWEEN because BETWEEN is inclusive of the lower and upper bound and that would make the test inaccurate.

I finally outputted the message, concatenating the variables inline to give a nice, personalized to the program’s execution, output.

A screenshot of a computer

AI-generated content may be incorrect.

**9&10**

**Create a table with this command**

**CREATE TABLE A\_TEST (id number );**

Develop a PL/SQL code to insert 10 records into above table.

**9- Use FOR LOOP to insert 10 numbers**

INSERT INTO A\_TEST VALUES (i , ‘times’);

**10- Use WHILE LOOP to insert 10 numbers**

INSERT INTO A\_TEST VALUES (counter , ‘times’);

Explain your reasoning to support your program logic. Explain difference between For loop and While loop in this context

DROP TABLE A\_TEST;

CREATE TABLE A\_TEST(id number);

DECLARE

v\_counter number := 1;

BEGIN

FOR i in 1..10 LOOP

INSERT INTO A\_TEST VALUES (i);

END LOOP;

WHILE v\_counter <= 10 LOOP

INSERT INTO A\_TEST VALUES (v\_counter);

v\_counter := v\_counter + 1;

END LOOP;

END;

I added the DROP statement for testing, it isn’t technically necessary for the questions.

Since we created a table with only one value, id, which has the type of number, I could not insert the ‘times’ in each loop. I omitted it, but could also have modified the A\_TEST table to take in a VARCHAR value to handle the insert of the ‘times’.

The difference between FOR loop and WHILE loop in this context is that the for loop does not need any additional variables as it automatically assigns ‘I’ to the values in the range of 1 to 10, incrementing by 1 after each execution. The WHILE loop needs an additional variable which I have declared and assigned to 0 (v\_counter). This v\_counter is incremented by 1 upon each iteration, by my design. However, v\_counter could be changed to increment, decrement or any arithmetic operation by any number that is chosen. This offers more complex capabilities over the FOR loop’s mandatory increment by 1.

**A screenshot of a computer

AI-generated content may be incorrect.**

**A screenshot of a computer

AI-generated content may be incorrect.**