301225121(mabborang) Assignment#2

Advanced Database Concepts (Centennial College)

Studocu is not sponsored or endorsed by any college or university
Downloaded by Vansh Chug (vanshchug1@gmail.com)

Assignment #2 – PL/SQL Block Structures

Due Date: Midnight of Feb 11 (Friday)

Purpose: The purpose of this assignment is to help you:

- Become familiar with the data source: Brewbeans database and DoGood database
- Become familiar with PL/SQL block structures, %TPYE attribute, %ROWTYPE attribute, etc.

Instructions: Be sure to read the following general instructions carefully:

This assignment should be completed individually by all the students. Submit your solution **through the dropbox.** Your submission should include PL/SQL block and the screenshot of block execution result, the submission must be named according to the following rule: **studentID(yourlastname)_Assignment#number.doc**. e.g., 300123456(**smith)_Assignment#2**.doc

Questions [14 marks]

1. **[2 marks]** The Brewbean's application contains a page displaying order summary information, including IDBASKET, SUBTOTAL, SHIPPING, TAX and TOTAL columns from BB_BASKET table. Create a PL/SQL block with scalar variables to retrieve this data and then display it. An initialized variable should provide the IDBASKET value. Test the block using any existing basket ID

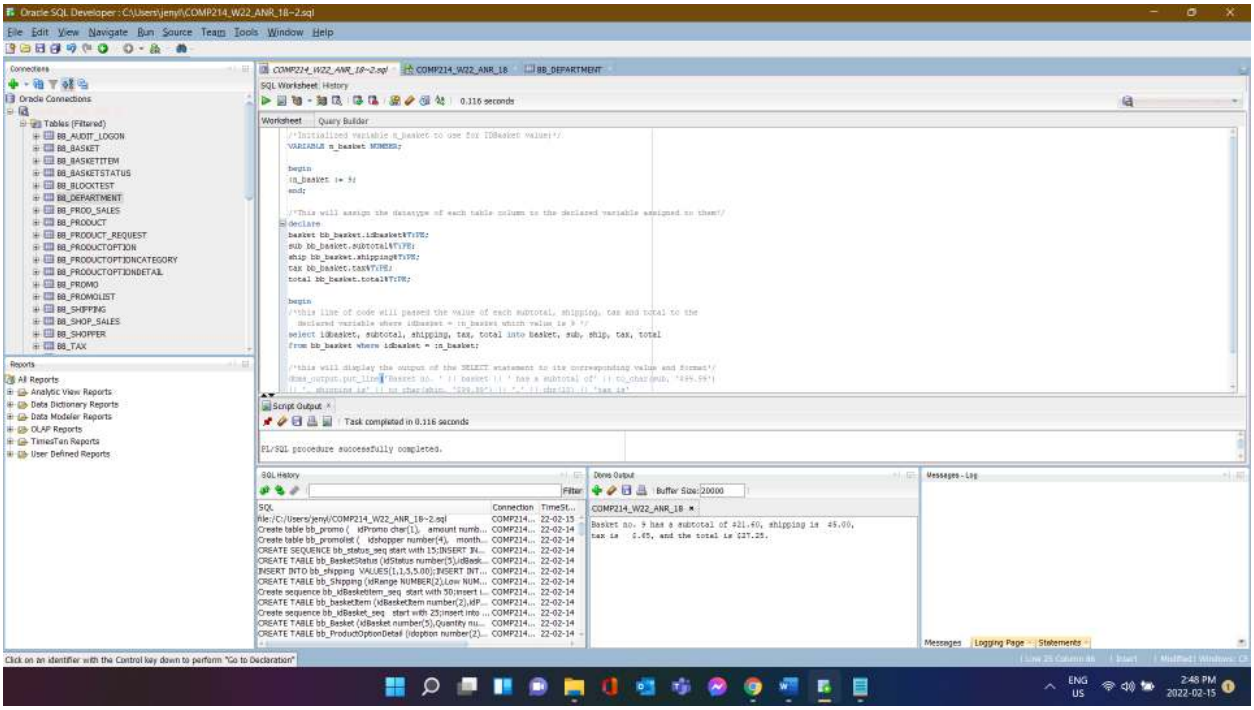
```
/*Initialized variable n_basket to use for IDBasket value)*/
VARIABLE N_BASKET NUMBER;

BEGIN
:N_BASKET := 9;
END;

/*This will assign the datatype of each table column to the declared variable
assigned to them*/
DECLARE
BASKET BB_BASKET.IDBASKET%TYPE;
SUB BB_BASKET.SUBTOTAL%TYPE;
SHIP BB_BASKET.SHIPPING%TYPE;
TAX BB_BASKET.TAX%TYPE;
TOTAL BB_BASKET.TOTAL%TYPE;

BEGIN
/*this line of code will passed 9the value of each subtotal, shipping, tax and total
to the declared variable where idbasket = :n_basket which value is 9 */
SELECT IDBASKET, SUBTOTAL, SHIPPING, TAX, TOTAL INTO BASKET,
SUB, SHIP, TAX, TOTAL
FROM BB_BASKET WHERE IDBASKET = :N_BASKET;
```


This document is available free of charge on studocu

Downloaded by Vansh Chug (vanshchug1@gmail.com)

```
/*this will display the output of the SELECT statement to its corresponding value
and format*/
DBMS_OUTPUT.PUT_LINE('BASKET NO. ' || BASKET || ' HAS A
SUBTOTAL OF' || TO_CHAR(SUB, '$99.99')
|| ', SHIPPING IS' || TO_CHAR(SHIP, '$99.99') || ',' || CHR(10) || 'TAX IS'
|| TO_CHAR(TAX, '$99.99') || ', AND THE TOTAL IS' || TO_CHAR(TOTAL,
'$99.99') || '.');
END;
```

OUTPUT FOR NUMBER 1:



2. **[5 marks]** An organization has committed to matching pledge amounts based on the donor type and pledge amount. Donor types include I (for Individual), B (for Business organization), and G (for Grant funds). The matching percentage is shown below:

Donor type	Pledge Amount	Matching %
I	$500 or more	30%
I	$300-$499	40%
I	$100-$299	50%
B	$1000 or more	10%
B	$500-$999	20%
B	$100-$499	40%

Downloaded by Vansh Chug (vanshchug1@gmail.com)

G	$100 0r more	10%

Create a PL/SQL anonymous block to accomplish the task. Input values for the block are the donor type code and the pledge amount.

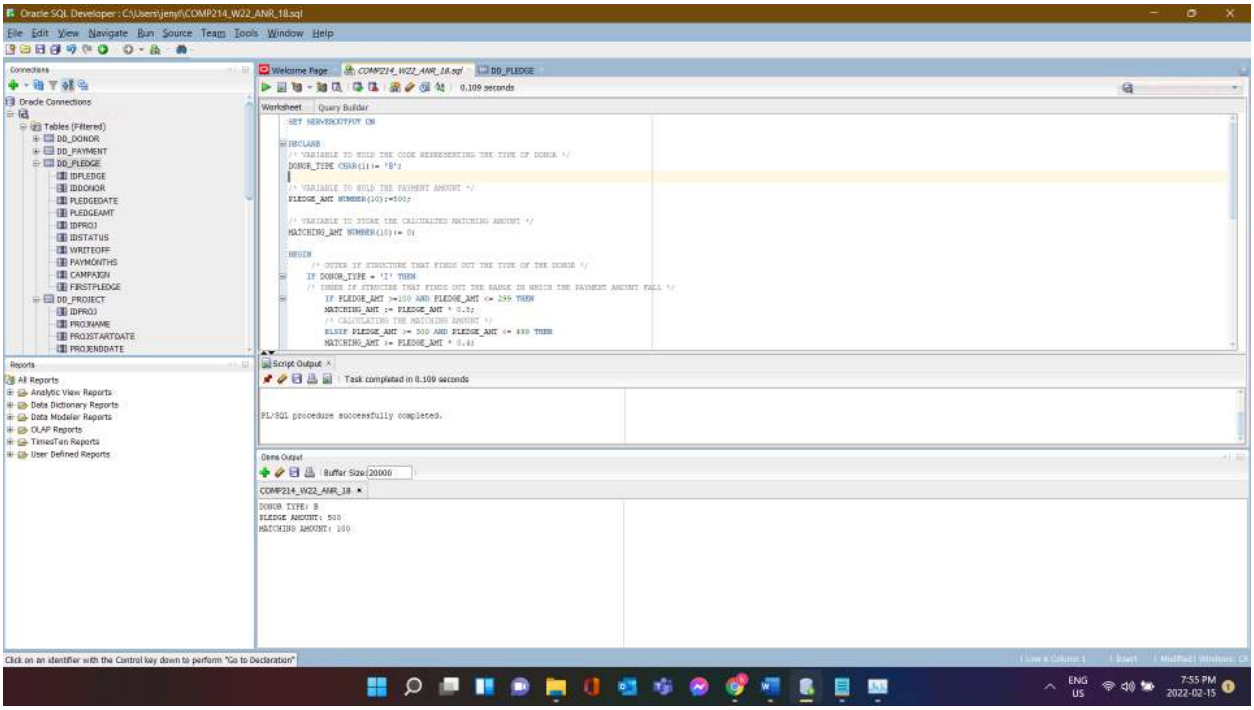
```
SET SERVEROUTPUT ON

DECLARE
/* VARIABLE TO HOLD THE CODE REPRESENTING THE TYPE OF DONOR */
DONOR_TYPE CHAR(1):= 'B';

/* VARIABLE TO HOLD THE PAYMENT AMOUNT */
PLEDGE_AMT NUMBER(10):=500;

/* VARIABLE TO STORE THE CALCUALTED MATCHING AMOUNT */
MATCHING_AMT NUMBER(10):= 0;

BEGIN
    /* OUTER IF STRUCTURE THAT FINDS OUT THE TYPE OF THE DONOR */
   IF DONOR_TYPE = 'I' THEN
   /* INNER IF STRUCTRE THAT FINDS OUT THE RANGE IN WHICH THE
PAYMENT AMOUNT FALL */
      IF PLEDGE_AMT >=100 AND PLEDGE_AMT <= 299 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.5;
      /* CALCULATING THE MATCHING AMOUNT */
      ELSIF PLEDGE_AMT >= 300 AND PLEDGE_AMT <= 499 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.4;
      ELSIF PLEDGE_AMT >= 500 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.3;
      END IF;

   ELSIF DONOR_TYPE = 'B' THEN
      IF PLEDGE_AMT >=100 AND PLEDGE_AMT <= 499 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.4;
      ELSIF PLEDGE_AMT >= 500 AND PLEDGE_AMT <= 999 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.2;
      ELSIF PLEDGE_AMT >= 1000 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.1;
      END IF;

   ELSIF DONOR_TYPE = 'G' THEN
      IF PLEDGE_AMT >= 100 THEN
      MATCHING_AMT := PLEDGE_AMT * 0.1;
      END IF;
```


This document is available free of charge on studocu

Downloaded by Vansh Chug (vanshchug1@gmail.com)

```
END IF;
/* END OF OUTER IF STRUCTURE */

/* DISPLAYING THE CALCULATED MATCHING AMOUNT */
   DBMS_OUTPUT.PUT_LINE('DONOR TYPE: ' || DONOR_TYPE || CHR(10) ||
'PLEDGE AMOUNT: ' || PLEDGE_AMT ||
   CHR(10) ||'MATCHING AMOUNT: ' || MATCHING_AMT);

END;
```

OUTPUT FOR NUMBER 2:



3. **[3 marks]** Create a PL/SQL anonymous block to insert a new project in DoGood Donor database. Create and use a sequence to handle generating and populating the project ID. The first number issue by the sequence should be 600, and no caching should be used. Use a record variable to handle the data to be added. Data for the new row should be the following: project name is "Covid-19 relief fund", start date: Feb 1, 2022, end date: Jun 30, 2022, and fundraising goal is half million. Any columns not addressed in the data list are currently unknown.

/*The sequence where the Project ID will start */

-
```
CREATE SEQUENCE DD_PROJID_SEQ
START WITH 600
NOCACHE;
```

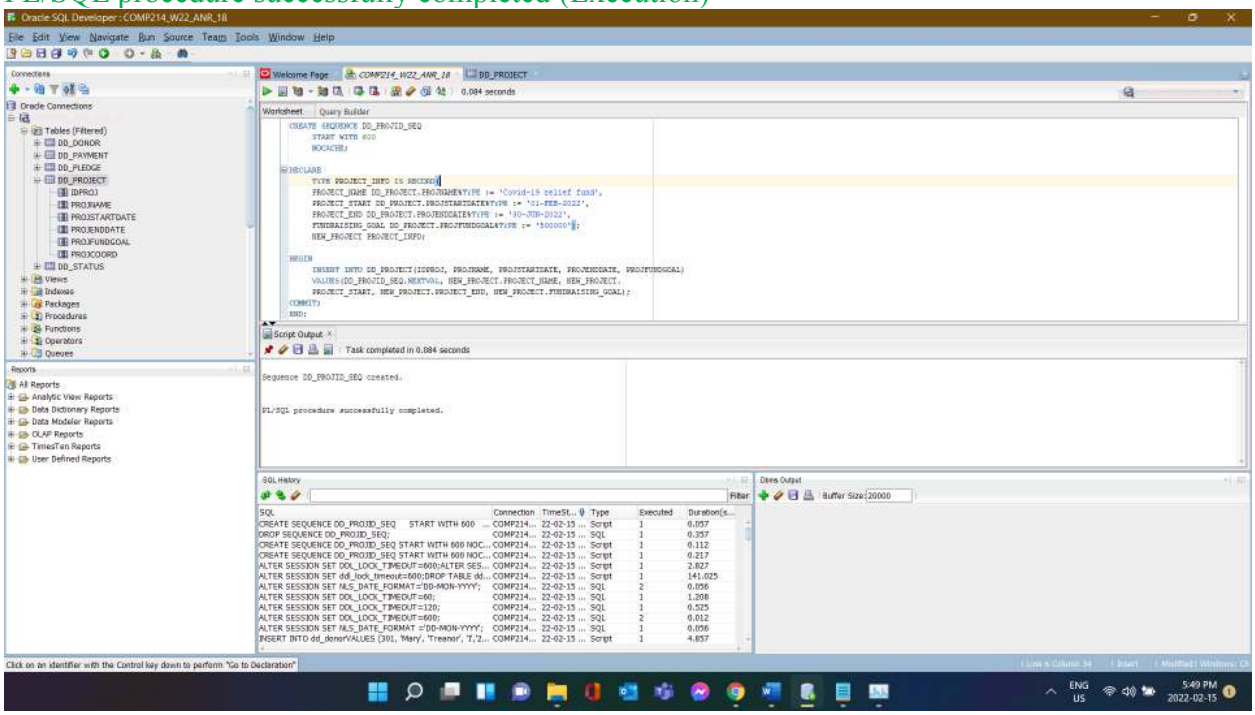
```
DECLARE
```
/*creating the record type Project_info only inside the subprogram*/

Downloaded by Vansh Chug (vanshchug1@gmail.com)

```
    TYPE PROJECT_INFO IS RECORD(
/*assigning value to each column in the record*/
    PROJECT_NAME DD_PROJECT.PROJNAME%TYPE := Covid-19
    relief fund',
    PROJECT_START DD_PROJECT.PROJSTARTDATE%TYPE := '01-
    FEB-2022',
    PROJECT_END DD_PROJECT.PROJENDDATE%TYPE := '30-JUN-
    2022',
    FUNDRAISING_GOAL DD_PROJECT.PROJFUNDGOAL%TYPE :=
    '500000');
    NEW_PROJECT PROJECT_INFO;

BEGIN
/*referring and retrieving value from each column in the record*/
    INSERT INTO DD_PROJECT(IDPROJ, PROJNAME,
    PROJSTARTDATE, PROJENDDATE, PROJFUNDGOAL)
    VALUES(DD_PROJID_SEQ.NEXTVAL,
    NEW_PROJECT.PROJECT_NAME, NEW_PROJECT.
    PROJECT_START, NEW_PROJECT.PROJECT_END,
    NEW_PROJECT.FUNDRAISING_GOAL);
COMMIT;
END;
```

OUTPUT FOR NUMBER 3:

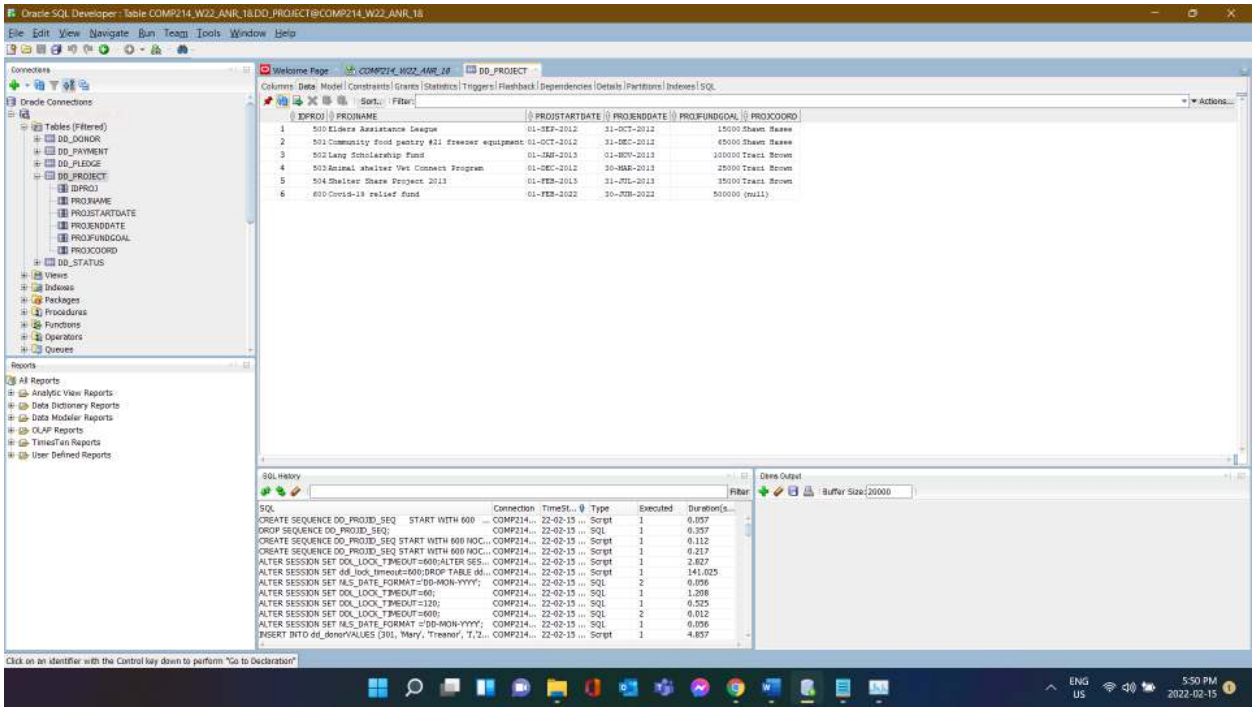
- PL/SQL procedure successfully completed (Execution)



This document is available free of charge on studocu

Downloaded by Vansh Chug (vanshchug1@gmail.com)

- Execution result (New Project: Covid-19 relief fund added to DD_project database)



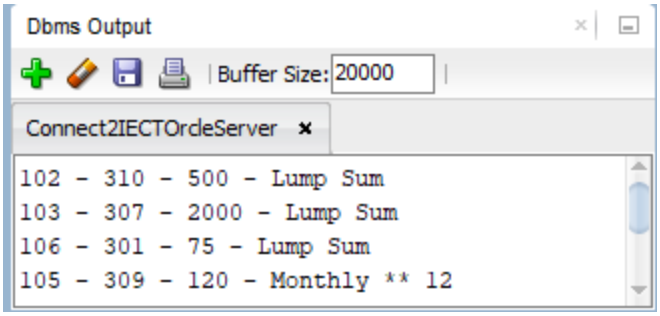
4. **[4 marks]** Create anonymous block to retrieve and display data for all pledges made in a specified month. One row of output should be displayed for each pledge. More specifically, each row include:

    a. Pledge ID, donor ID, and pledge amount

    b. If the pledge is being paid in a lump sum, display “Lump Sum”

    c. If the pledge is being paid in monthly, display “Monthly ** ” followed by number of months for payment

    d. The list should be sorted to display all lump sum pledges first



/*This is used to show the result in DBMS output if no output is currently showing in it*/

- SET SERVEROUTPUT ON

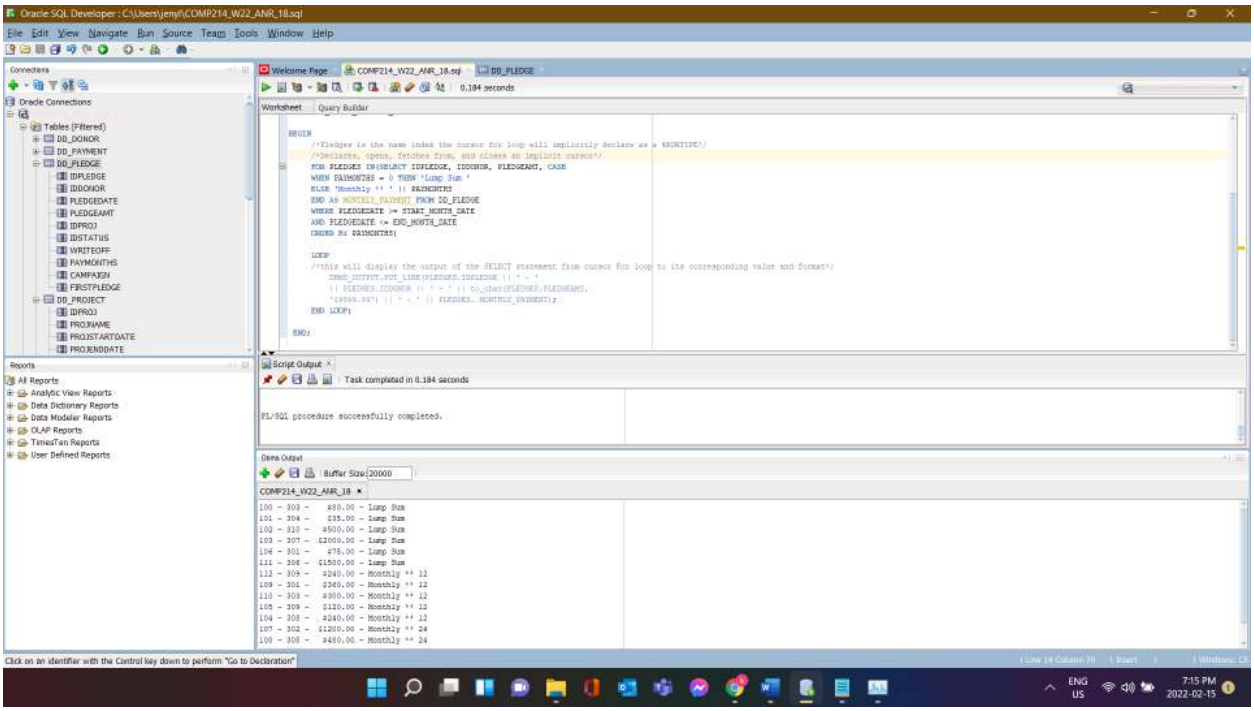
Downloaded by Vansh Chug (vanshchug1@gmail.com)

```
DECLARE
    PLEDGES DD_PLEDGE%ROWTYPE;
    START_MONTH_DATE DD_PLEDGE.PLEDGEDATE%TYPE := '01-OCT-12';
    END_MONTH_DATE DD_PLEDGE.PLEDGEDATE%TYPE := '31-OCT-12';

BEGIN
/*Pledges is the name index the cursor for loop will implicitly declare as a %ROWTYPE*/
/*Declares, opens, fetches from, and closes an implicit cursor*/
    FOR PLEDGES IN
    (SELECT IDPLEDGE, IDDONOR, PLEDGEAMT, CASE
    WHEN PAYMONTHS = 0 THEN 'Lump Sum.'
    ELSE 'Monthly - ' || PAYMONTHS
    END AS MONTHLY_PAYMENT
    FROM DD_PLEDGE WHERE PLEDGEDATE >= START_MONTH_DATE
    AND PLEDGEDATE <= END_MONTH_DATE ORDER BY PAYMONTHS)

    LOOP
/*this will display the output of the SELECT statement from cursor For loop to its corresponding value and format*/
    DBMS_OUTPUT.PUT_LINE('Pledge ID: ' || PLEDGES.IDPLEDGE || ', Donor
    ID: ' || PLEDGES.IDDONOR || ', Pledge Amount: ' ||
    to_char(PLEDGES.PLEDGEAMT, '$9999.99') || ', Monthly Payments: ' ||
    PLEDGES. MONTHLY_PAYMENT);
    END LOOP;
END;
```

OUTPUT FOR NUMBER 4:

This document is available free of charge on studocu

Downloaded by Vansh Chug (vanshchug1@gmail.com)



Downloaded by Vansh Chug (vanshchug1@gmail.com)