Create the following tables with appropriate constraints using SQL command.

A) Table Name : Member

COLUMN NAME	DATA TYPE	DESCRIPTION
Member_Id	Number(5)	Unique Member ID
Member_Name	Varchar2(30)	Name of the Library member
Member_address	Varchar2(50)	Address of the member
Acc_Open_Date	Date	Date of membership
Membership_type	Varchar2(20)	Type of the membership such as ‘Lifetime’,’ Annual’, ‘Half Yearly’,’ Quarterly’
Fees_paid	Number(4)	Membership fees paid
Max_Books_Allowed	Number(2)	Total Number of books that can be issued to the member.
Penalty_Amount	Number(7,2)	Penalty amount due

CONSTRAINT:
a. Member_Id – Primary Key
b. Member_Name – NOT NULL
c. Membership_type - ‘Lifetime’,’ Annual’, ‘Half Yearly’,’ Quarterly’
d. Max_books_allowed <7
e. Penalty_amt maximum 1000

QUERY:

```
CREATE TABLE MEMBER(Member_Id NUMBER(5) PRIMARY KEY,

Member_Name VARCHAR2(30),

Member_address VARCHAR2(30),

Acc_Open_Date DATE,

Membership_type VARCHAR2(20),

Fees_paid NUMBER(4),

Max_Books_Allowed NUMBER(2),

Penalty_Amount NUMBER(7,2));

ALTER TABLE Member MODIFY( Member_Name NOT NULL);

ALTER TABLE Member ADD CONSTRAINT M1 CHECK (Membership_Type IN
('Lifetime','Annual','Half Yearly','Quarterly'));

ALTER TABLE Member ADD CONSTRAINT M2 CHECK (Max_Books_Allowed < 7);

ALTER TABLE Member ADD CONSTRAINT M3 CHECK (Penalty_Amount < 1000);
```

B) Table Name : BOOKS

COLUMN NAME	DATA TYPE	DESCRIPTION
Book_No	Number(6)	Book identification number
Book_Name	VarChar2(30)	Name of the book
Author_name	Varchar2(30)	Author of the book
Cost	Number(7,2)	Cost of the book
Category	Char(10)	Category like Science , Fiction etc.

CONSTRAINT:
a. Book_No – Primary Key
b. Book_Name – Not Null
c. Category – Science, Database, System, Others

QUERY:

```
CREATE TABLE BOOKS(Book_No NUMBER(6) PRIMARY KEY,

Book_Name VARCHAR(30) NOT NULL,

Author_name VARCHAR2(30),

Cost NUMBER(7,2),

Category CHAR(10));


ALTER TABLE BOOKS ADD CONSTRAINT B1 CHECK (Category IN
('Science','Database','System','Others'));
```

C) Table Name : ISSUE

COLUMN NAME	DATA TYPE	DESCRIPTION
Lib_Issue_Id	Number(10)	Library Book Issue No
Book_No	Number(6)	The ID of book, which is issued
Member_Id	Number(5)	Member that issued the book
Issue_Date	Date	Date of Issue
Return_date	Date	Return date

CONSTRAINT:

a. Lib_Issue_Id -Primary key
b. Book_No - foreign key
c. Member_id - foreign key

QUERY:

```
CREATE TABLE ISSUE(Lib_Issue_Id NUMBER(10) Primary Key,

Book_No NUMBER(6),

FOREIGN KEY(Book_No) REFERENCES BOOKS(Book_No),

Member_Id NUMBER(5),

FOREIGN KEY(Member_ID) REFERENCES Member(Member_Id),

Issue_Date DATE,

Return_Date DATE);
```

 Insert the following data to the appropriate table using SQL command.

A. Table Name : Member

MEMBER_ID	MEMBER_ NAME	MEMBER_ ADDRESS	ACC_ OPEN_DATE	MEMBER-SHIP_TYPE	FEES_ PAID	MAX_BOOK _ALLOWED	PENALTY_ AMOUNT
1	Sayantan Sinha	Pune	10-Dec-10	Lifetime	2000	6	50
2	Abhirup Sarkar	Kolkata	19-jan-11	Annual	1400	3	0
3	Ritesh Bhuniya	Gujarat	20-feb-11	Quarterly	350	2	100
4	Paresh sen	Tripura	21-mar-11	Half yearly	700	1	200
5	Sohini Haldar	Birbhum	11-apr-11	Lifetime	2000	6	10
6	Suparna Biswas	Kolkata	12-apr-11	Half Yearly	700	1	0
7	Suranjana Basu	Purulia	30-june-11	Annual	1400	3	50
8	Arpita Roy	Kolkata	31-july-11	Half yearly	700	1	0

```
INSERT INTO Member VALUES(1,'Sayantan Sinha','Pune','10-Dec-10','Lifetime',2000,6,50);
INSERT INTO Member VALUES(2,'Abhirup Sarkar','Kolkata','19-Jan-11','Annual',1400,3,0);
INSERT INTO Member VALUES(3,'Ritesh Bhuniya','Gujarat','20-Feb-11','Quarterly',350,2,100);
INSERT INTO Member VALUES(4,'Paresh Sen','Tripura','21-Mar-11','Half Yearly',700,1,200);
INSERT INTO Member VALUES(5,'Sohini Haldar','Birbhum','11-Apr-11','Lifetime',2000,6,10);
INSERT INTO Member VALUES(6,'Suparna Biswas','Kolkata','12-Apr-11','Half Yearly',700,1,0);
INSERT INTO Member VALUES(7,'Suranjana Basu','Kolkata','30-June-11','Annual',1400,3,50);
INSERT INTO Member VALUES(8,'Arpita Roy','Kolkata','31-July-11','Half Yearly',700,1,0);

select * from member;
```

B. Table Name : BOOKS

BOOK_NO	BOOK_NAME	AUTHOR_NAME	COST	CATEGORY
101	Let us C	Denis Ritchie	450	Others
102	Oracle – Complete Ref	Loni	550	Database
103	Visual Basic 10	BPB	700	Others
104	Mastering SQL	Loni	450	Database
105	PL SQL-Ref	Scott Urman	750	Database
106	UNIX	Sumitava Das	300	System
107	Optics	Ghatak	600	Science
108	Data Structure	G.S. Baluja	350	Others

```
INSERT INTO BOOKS VALUES(101,'Let us C','Denis Ritchie', 450,'Others');
INSERT INTO BOOKS VALUES(102,'Oracle-Complete Ref','Loni', 550,'Database');
INSERT INTO BOOKS VALUES(103,'Visual Basic 10','BPB', 700,'Others');
INSERT INTO BOOKS VALUES(104,'Mastering SQL','Loni', 450,'Database');
INSERT INTO BOOKS VALUES(105,'PL SQL-Ref','Scott Urman', 750,'Database');
INSERT INTO BOOKS VALUES(106,'UNIX','Sumitava Das', 300,'System');
INSERT INTO BOOKS VALUES(107,'Optics','Ghatak', 600,'Science');
INSERT INTO BOOKS VALUES(108,'Data Structure','G.S.Baluja', 350,'Others');

select * from books;
```

C. Table Name : ISSUE

LIB_ISSUE_ID	BOOK_NO	MEMBER_ID	ISSUE_DATE	RETURN_DATE
7001	101	1	10-jan-11	
7002	102	2	25-jan-11	
7003	104	1	1-Feb-11	
7004	104	2	15-Mar-11	
7005	101	4	04-Apr-11	
7006	108	5	12-apr-11	
7007	101	8	1-Aug-11	

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7001,101,1,'10-jan-11');

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7002,102,2,'25-jan-11');

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7003,104,1,'1-Feb-11');

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7004,104,2,'15-Mar-11');

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7005,101,4,'04-Apr-11');

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7006,108,5,'12-apr-11');

INSERT INTO ISSUE(Lib_Issue_Id,Book_No,Member_Id,Issue_Date) VALUES(7007,101,8,'1-Aug-11');

select * from issue;

commit;

SET SERVEROUTPUT ON; (Enables the server to give output)

1) Write a PL/SQL program where take input of two numbers and display the largest number.

```
DECLARE
        NUM1 NUMBER(10);
        NUM2 NUMBER(10);
BEGIN
        NUM1:=&NUM1;
        NUM2:=&NUM2;
        IF NUM1>NUM2 THEN
                DBMS_OUTPUT.PUT_LINE(NUM1 || ' IS GREATER THAN ' || NUM2);
        ELSIF NUM2>NUM1 THEN
    DBMS_OUTPUT.PUT_LINE(NUM2 || ' IS GREATER THAN ' || NUM1);
  ELSE
                DBMS_OUTPUT.PUT_LINE('BOTH NUMBERS ARE EQUAL');
        END IF;
END;
/
```

OUTPUT:

```
old:DECLARE
        NUM1 NUMBER(10);
        NUM2 NUMBER(10);
BEGIN
        NUM1:=&NUM1;
        NUM2:=&NUM2;
        IF NUM1>NUM2 THEN
                DBMS_OUTPUT.PUT_LINE(NUM1 || ' IS GREATER THAN ' || NUM2);
        ELSIF NUM2>NUM1 THEN
    DBMS_OUTPUT.PUT_LINE(NUM2 || ' IS GREATER THAN ' || NUM1);
  ELSE
                DBMS_OUTPUT.PUT_LINE('BOTH NUMBERS ARE EQUAL');
```

```
        END IF;
END;

new:DECLARE
        NUM1 NUMBER(10);
        NUM2 NUMBER(10);
BEGIN
        NUM1:=55;
        NUM2:=68;
        IF NUM1>NUM2 THEN
                DBMS_OUTPUT.PUT_LINE(NUM1 || ' IS GREATER THAN ' || NUM2);
        ELSIF NUM2>NUM1 THEN
      DBMS_OUTPUT.PUT_LINE(NUM2 || ' IS GREATER THAN ' || NUM1);
   ELSE
                DBMS_OUTPUT.PUT_LINE('BOTH NUMBERS ARE EQUAL');
        END IF;
END;
68 IS GREATER THAN 55


PL/SQL procedure successfully completed.
```

2) Write a PL/SQL program where take input of any number and display whether it is even or odd.

```
DECLARE
        NUM NUMBER(10);
        NUM2 NUMBER(10);
BEGIN
        NUM:=&NUM;
        NUM2:=55;
        IF MOD(NUM,2)=0 THEN
                DBMS_OUTPUT.PUT_LINE(NUM||'IS EVEN');
        ELSE
                DBMS_OUTPUT.PUT_LINE(NUM||'IS ODD');
        END IF;
END;
/
```

OUTPUT:

```
old:DECLARE
        NUM NUMBER(10);
        NUM2 NUMBER(10);
BEGIN
        NUM:=&NUM;
        NUM2:=55;
        IF MOD(NUM,2)=0 THEN
                DBMS_OUTPUT.PUT_LINE(NUM||'IS EVEN');
        ELSE
                DBMS_OUTPUT.PUT_LINE(NUM||'IS ODD');
        END IF;
END;

new:DECLARE
        NUM NUMBER(10);
```

```
        NUM2 NUMBER(10);
BEGIN
        NUM:=6;
        NUM2:=55;
        IF MOD(NUM,2)=0 THEN
                DBMS_OUTPUT.PUT_LINE(NUM||'IS EVEN');
        ELSE
                DBMS_OUTPUT.PUT_LINE(NUM||'IS ODD');
        END IF;
END;
6IS EVEN


PL/SQL procedure successfully completed.
```

3) Write a PL/SQL program where take input of any number and find factorial of the given number.

```
DECLARE
   NUM NUMBER(10);
   FACT NUMBER(10);
BEGIN
   NUM:=&NUM;
   FACT:=1;
   WHILE NUM>1
   LOOP
      FACT:=FACT*NUM;
      NUM:=NUM-1;
   END LOOP;
   DBMS_OUTPUT.PUT_LINE('THE FACTORIAL IS' || FACT);
END;
/
```

OUTPUT:

```
old:DECLARE
   NUM NUMBER(10);
   FACT NUMBER(10);
BEGIN
   NUM:=&NUM;
   FACT:=1;
   WHILE NUM>1
   LOOP
      FACT:=FACT*NUM;
      NUM:=NUM-1;
   END LOOP;
   DBMS_OUTPUT.PUT_LINE('THE FACTORIAL IS' || FACT);
END;
```

```
new:DECLARE
   NUM NUMBER(10);
   FACT NUMBER(10);
BEGIN
   NUM:=8;
   FACT:=1;
   WHILE NUM>1
   LOOP
      FACT:=FACT*NUM;
      NUM:=NUM-1;
   END LOOP;
   DBMS_OUTPUT.PUT_LINE('THE FACTORIAL IS' || FACT);
END;
THE FACTORIAL IS40320


PL/SQL procedure successfully completed.
```

4) Write a PL/SQL program where check a year leap year or not. Take a year as user input and check it is leap year or not.

```
DECLARE
   YEAR NUMBER(4);
BEGIN
   YEAR:=&YEAR;
   IF MOD(YEAR,4)=0 THEN
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS A LEAP YEAR');
   ELSIF MOD(YEAR,100)=0 AND MOD(YEAR,4)!=0 THEN
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS A LEAP YEAR');
   ELSE
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS NOT A LEAP YEAR');
   END IF;
END;
/
```

OUTPUT:

```
old:DECLARE
   YEAR NUMBER(4);
BEGIN
   YEAR:=&YEAR;
   IF MOD(YEAR,4)=0 THEN
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS A LEAP YEAR');
   ELSIF MOD(YEAR,100)=0 AND MOD(YEAR,4)!=0 THEN
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS A LEAP YEAR');
   ELSE
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS NOT A LEAP YEAR');
   END IF;
END;

new:DECLARE
   YEAR NUMBER(4);
```

```
BEGIN
   YEAR:=2202;
   IF MOD(YEAR,4)=0 THEN
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS A LEAP YEAR');
   ELSIF MOD(YEAR,100)=0 AND MOD(YEAR,4)!=0 THEN
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS A LEAP YEAR');
   ELSE
      DBMS_OUTPUT.PUT_LINE(YEAR || 'IS NOT A LEAP YEAR');
   END IF;
END;
2202IS NOT A LEAP YEAR


PL/SQL procedure successfully completed.
```

5) Write a PL/SQL program where take a string as input and print the reverse of it.

```
DECLARE
   STR VARCHAR(20);
   LEN NUMBER;
   STR1 VARCHAR(20);
BEGIN
   STR:=:STR;
   LEN := LENGTH(STR);
   FOR I IN REVERSE 1.. LEN
   LOOP
      STR1 := STR1 || SUBSTR(STR, I, 1);
   END LOOP;
   DBMS_OUTPUT.PUT_LINE('REVERSE OF STRING IS ' || STR1);
END;
/
```

OUTPUT:

REVERSE OF STRING IS

yob a si MAHBUHS

PL/SQL procedure successfully completed.

6) Create a table named CIRCLE with two attributes RADIUS number (3) and AREA number(10,3), then write a PL/SQL program which can calculate area for every radius up to 10 and insert into the table. (Use while and for loop individually).

```
CREATE TABLE CIRCLE(RADIUS NUMBER(5),AREA NUMBER(10,3));
INSERT INTO CIRCLE(RADIUS) VALUES(6);
INSERT INTO CIRCLE(RADIUS) VALUES(4);
INSERT INTO CIRCLE(RADIUS) VALUES(3);
INSERT INTO CIRCLE(RADIUS) VALUES(8);
```

PROGRAM IN FOR LOOP:

```
BEGIN
   FOR RECORD IN (SELECT * FROM CIRCLE)
   LOOP
      UPDATE CIRCLE SET AREA=(3.14*RECORD.RADIUS*RECORD.RADIUS) WHERE
RADIUS=RECORD.RADIUS;
   END LOOP;
END;
/

SELECT * FROM CIRCLE;
```

PROGRAM IN WHILE LOOP:

```
DECLARE
   CURSOR CIRCLE_CURSOR IS SELECT RADIUS FROM CIRCLE;
   ROWRECORD CIRCLE_CURSOR%ROWTYPE;
   RAD NUMBER(3);
BEGIN
   OPEN CIRCLE_CURSOR;
   FETCH CIRCLE_CURSOR INTO RAD;
```

```
    WHILE CIRCLE_CURSOR%FOUND
    LOOP
        UPDATE CIRCLE SET AREA=(3.14*RAD*RAD) WHERE RADIUS=RAD;
        FETCH CIRCLE_CURSOR INTO RAD;
    END LOOP;
END;
/

SELECT * FROM CIRCLE;
```

OUTPUT:

	RADIUS	AREA
1	6	113.04
2	4	50.24
3	3	28.26
4	8	200.96

7) Write a PL/SQL program which can update cost value of corresponding book number of the BOOKS_COPY

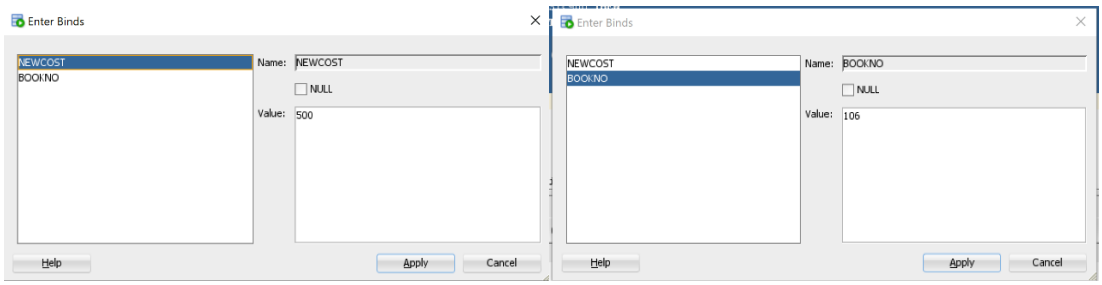
Table.

- *INPUT: BOOK_NO, NEW COST,*

- *CONDITION: Old cost value will less than 450 and new cost value will less than 900 otherwise provide an error massage.*

```
DECLARE
   OLDCOST NUMBER(10);
   NEWCOST NUMBER(10);
   BOOKNO NUMBER(10);
BEGIN
   NEWCOST:=:NEWCOST;
   BOOKNO:=:BOOKNO;
   SELECT COST INTO OLDCOST FROM BOOKS WHERE BOOK_NO=BOOKNO;
   IF OLDCOST<450 AND NEWCOST<900 THEN
      UPDATE BOOKS SET COST=NEWCOST WHERE COST=OLDCOST;
   ELSE
      DBMS_OUTPUT.PUT_LINE('ERROR: COST IS NOT VALID.');
   END IF;
END;
/
SELECT * FROM BOOKS;
```

OUTPUT:



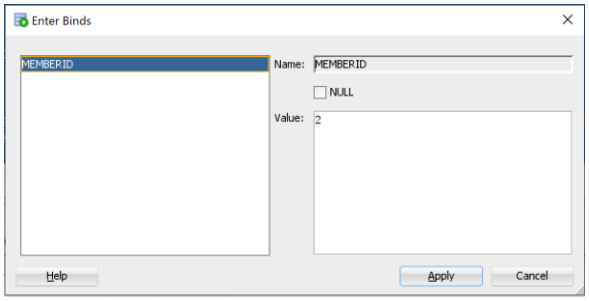
PL/SQL procedure successfully completed.

	BOOK_NO	BOOK_NAME	AUTHOR_NAME	COST	CATEGORY
1	101	Let us C	Denis Ritchie	450	Others
2	102	Oracle-Complete Ref	Loni	550	Database
3	103	Visual Basic 10	BPB	700	Others
4	104	Mastering SQL	Loni	450	Database
5	105	PL SQL-Ref	Scott Urman	750	Database
6	106	UNIX	Sumitava Das	500	System
7	107	Optics	Ghatak	600	Science
8	108	Data Structure	G.S.Baluja	350	Others

8) Write a PL/SQL Program which take MEMBER_ID as input and provide the corresponding MEMBER_NAME, MEMBER_ADDRESS AND FEES PAID.

```
DECLARE
   MEMBERID NUMBER(10);
   MEMBERNAME VARCHAR2(30);
   MEMBERADDRESS VARCHAR2(50);
   FEES NUMBER(10);
BEGIN
   MEMBERID:=:MEMBERID;
   SELECT MEMBER_NAME,MEMBER_ADDRESS,FEES_PAID INTO MEMBERNAME,MEMBERADDRESS,FEES FROM MEMBER WHERE MEMBER_ID=MEMBERID;
   DBMS_OUTPUT.PUT_LINE('MEMBER NAME: ' || MEMBERNAME || ', MEMBER ADDRESS: ' || MEMBERADDRESS || ', FEES PAID: RS. ' || FEES);
END;
/
```

OUTPUT:



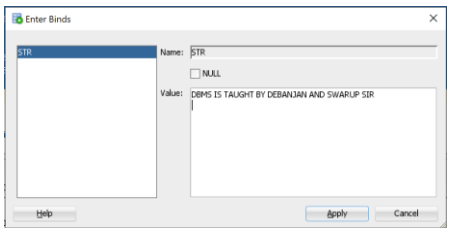
PL/SQL procedure successfully completed.

MEMBER NAME: Abhirup Sarkar, MEMBER ADDRESS: Kolkata, FEES PAID: RS. 1400

9) Write a PL/SQL program which can take a String as an input and display it without any space and also count the no of space available in the input string.

```
DECLARE
   LEN NUMBER(10);
   COUNTSPACE NUMBER(10);
   STR VARCHAR(50);
   STR1 VARCHAR(50);
BEGIN
   STR:=:STR;
   LEN:=LENGTH(STR);
   COUNTSPACE:=0;
   FOR I IN 1..LEN
   LOOP
      IF SUBSTR(STR, I, 1)=' ' THEN
         COUNTSPACE:=COUNTSPACE+1;
      ELSE
         STR1 := STR1 || SUBSTR(STR, I, 1);
      END IF;
   END LOOP;
   DBMS_OUTPUT.PUT_LINE('THE STRING WITHOUT ANY SPACE IS ' || STR1);
   DBMS_OUTPUT.PUT_LINE('THE NUMBER OF SPACE(S) IN THE STRING WAS ' ||
COUNTSPACE);
END;
/
```

OUTPUT:



PL/SQL procedure successfully completed.

THE STRING WITHOUT ANY SPACE IS DBMSISTAUGHTBYDEBANJANANDSWARUPSIR

THE NUMBER OF SPACE(S) IN THE STRING WAS 7

PL/SQL procedure successfully completed.

10) Take an input of any string and display each word in a separate line.

```
DECLARE
   LEN NUMBER(10);
   STR VARCHAR(50);
BEGIN
   STR:=:STR;
   LEN:=LENGTH(STR);
   FOR I IN 1..LEN
   LOOP
      DBMS_OUTPUT.PUT_LINE(SUBSTR(STR,I,1));
   END LOOP;
END;
/
```

OUTPUT:

PL/SQL procedure successfully completed.

S

H

U

B

H

A

M

T

R

I

B

E

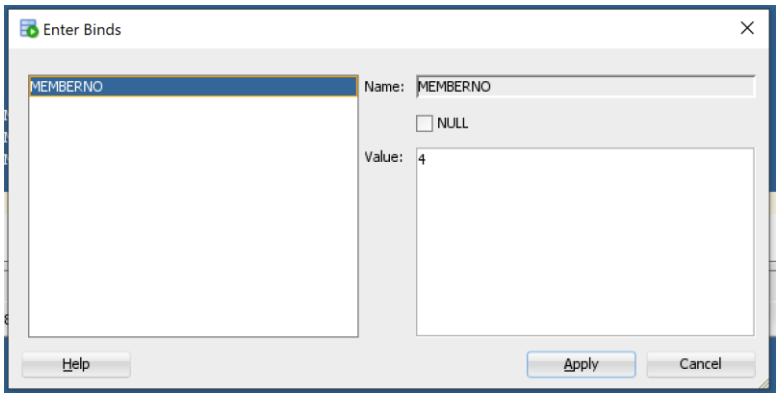
D

I

11) Take an input of any Member Number and display the Member Name in upper case and lower case.

```
DECLARE
   MEMBERNO NUMBER(10);
   MEMBERNAME VARCHAR(50);
BEGIN
   MEMBERNO:=:MEMBERNO;
   SELECT MEMBER_NAME INTO MEMBERNAME FROM MEMBER WHERE MEMBER_ID=MEMBERNO;
   DBMS_OUTPUT.PUT_LINE('MEMBER NAME IN UPPERCASE IS ' || UPPER(MEMBERNAME));
   DBMS_OUTPUT.PUT_LINE('MEMBER NAME IN LOWERCASE IS ' || LOWER(MEMBERNAME));
END;
/
COMMIT;
```

OUTPUT:



PL/SQL procedure successfully completed.

MEMBER NAME IN UPPERCASE IS PARESH SEN

MEMBER NAME IN LOWERCASE IS paresh sen

PL/SQL procedure successfully completed.

Commit complete.