

Name- Khushi Nitinkumar Patel

PRN- 2020BTECS00037

Batch- T2

Assignment No 1

28/01/2023

Objective: To study and implement PL-SQL and object-oriented database

Introduction:

PL/SQL is a procedural language that uses SQL as its database. Both PLSQL and SQL run within same server process. It is an application-oriented language. It supports object-oriented programming through object types.

Basic Algorithm:

PL/SQL Block

Block Header: Defines type Block.

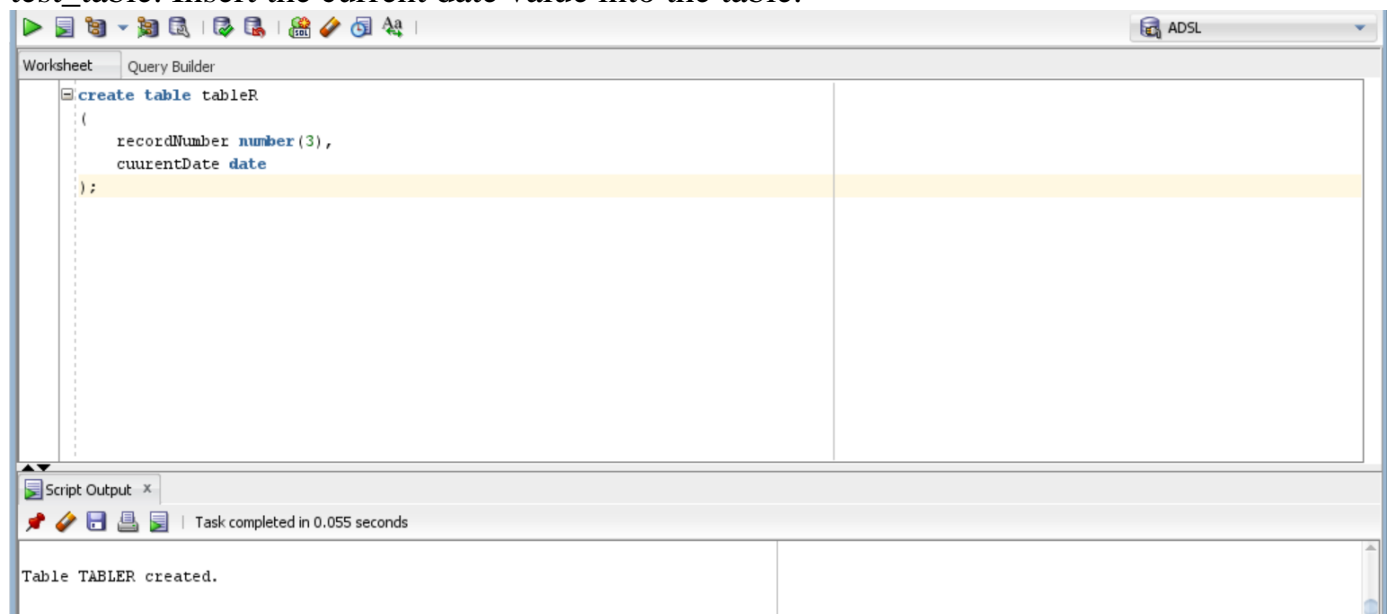
Declaration Header: Declare any variable used in the block.

Execution Selection: Use var and other PL/SQL object to perform action.

Experiment:

Q.1

a) Create a table called test_table with 2 columns RecordNumber (type : Number(3)) and CurrentDate (type : Date)). Write PL/SQL block which will insert 50 records into test_table. Insert the current date value into the table.



SQL Worksheet History

Worksheet Query Builder

```

declare
  n number := 1;
  d date;
begin
  select sysdate into d from dual;
  while n < 51
  loop
    insert into tableR values (n, d);
    n := n + 1;
  end loop;
end;

```

Script Output x

Task completed in 0.804 seconds

PL/SQL procedure successfully completed.

Line 19 Column 1 | Insert | Modified | Windows: CI

Worksheet Query Builder

```

select * from tableR;

```

Script Output x Query Result x

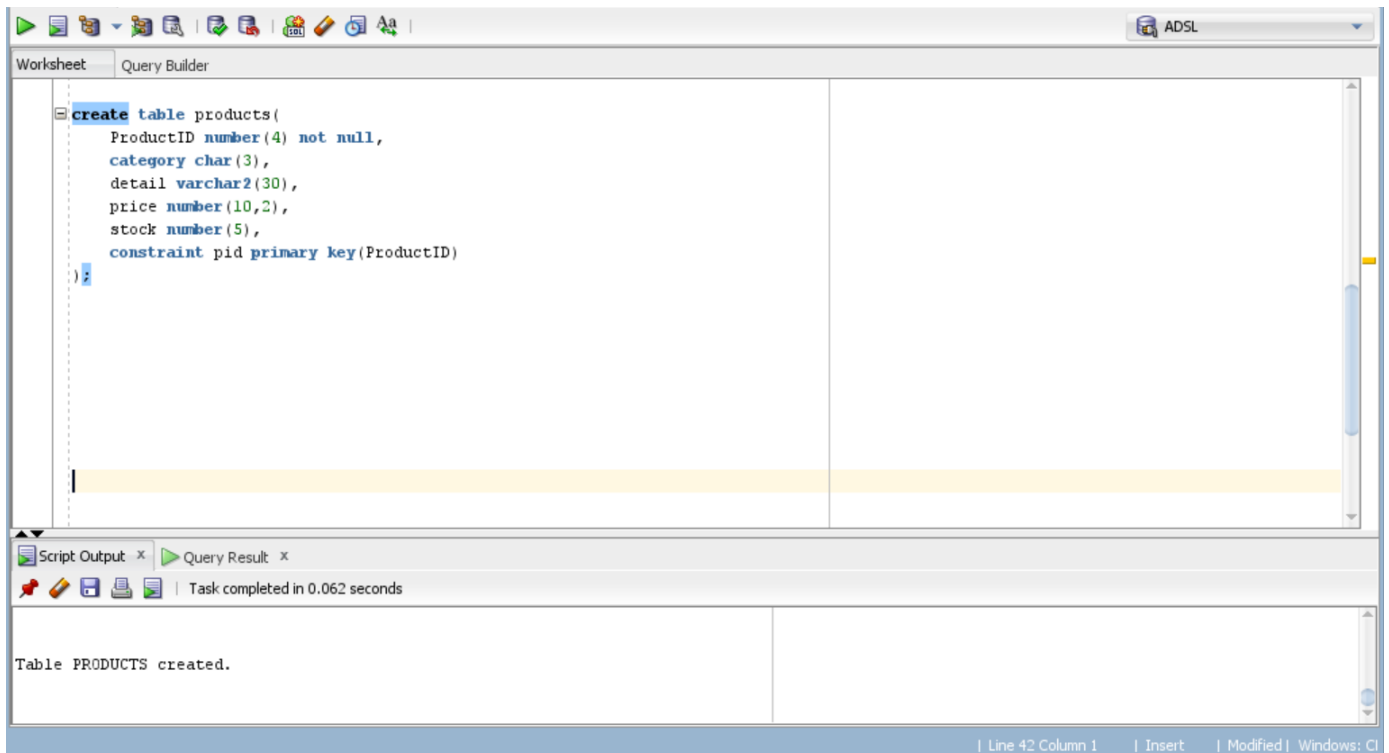
SQL | All Rows Fetched: 50 in 1.581 seconds

| RECORDNUMBER | CUURENDATE |
|--------------|-------------|
| 1 | 1 31-01-23 |
| 2 | 2 31-01-23 |
| 3 | 3 31-01-23 |
| 4 | 4 31-01-23 |
| 5 | 5 31-01-23 |
| 6 | 6 31-01-23 |
| 7 | 7 31-01-23 |
| 8 | 8 31-01-23 |
| 9 | 9 31-01-23 |
| 10 | 10 31-01-23 |
| 11 | 11 31-01-23 |
| 12 | 12 31-01-23 |
| 13 | 13 31-01-23 |
| 14 | 14 31-01-23 |
| 15 | 15 31-01-23 |
| 16 | 16 31-01-23 |
| 17 | 17 31-01-23 |
| 18 | 18 31-01-23 |
| 19 | 19 31-01-23 |

Line 21 Column 1 | Insert | Modified | Windows: CI

b) Create a products table products(ProductID number(4),category char(3),detail varchar2(30),price number(10,2),stock number(5)).
Insert the sample data.

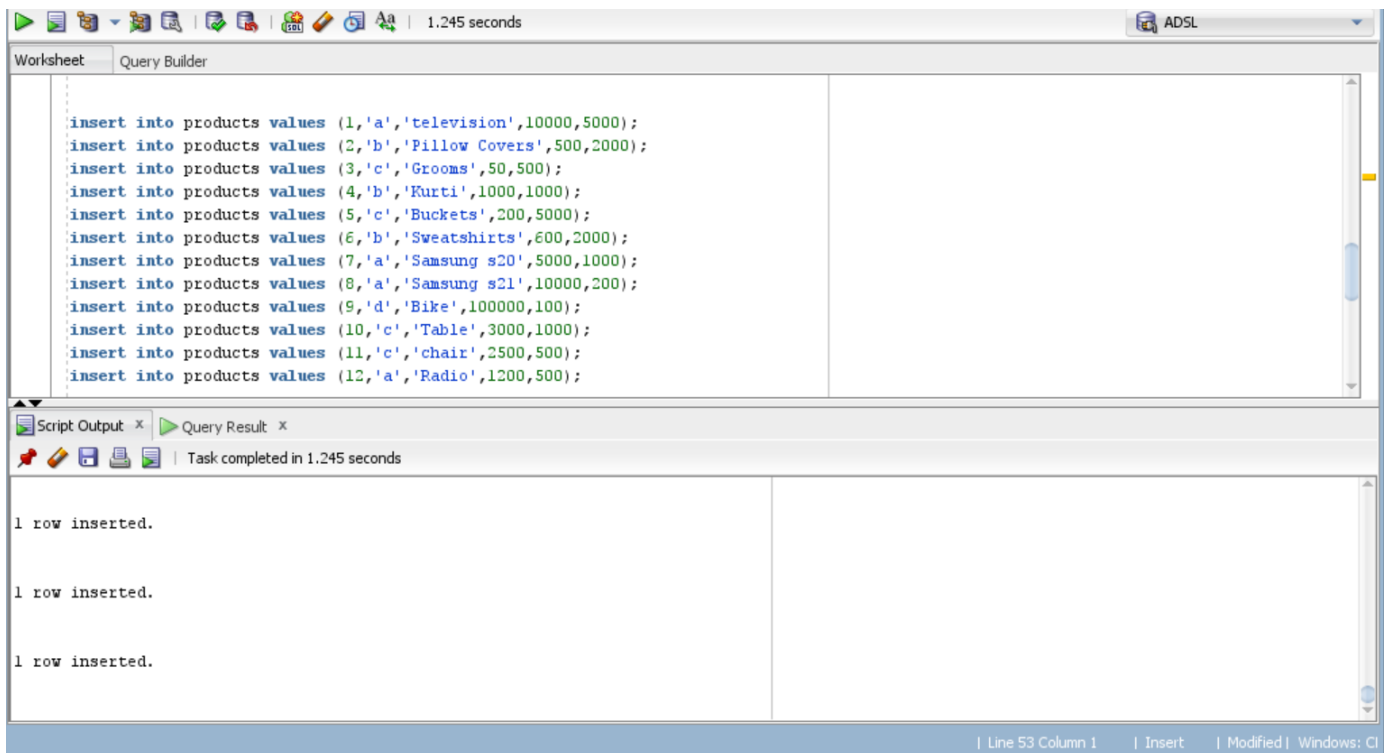
Write PL/SQL procedure with two arguments **X** & **Y** which will increase price by **X%** for all products in category **Y**. X and Y will be given by user.



The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the main editor is as follows:

```
create table products(  
  ProductID number(4) not null,  
  category char(3),  
  detail varchar2(30),  
  price number(10,2),  
  stock number(5),  
  constraint pid primary key(ProductID)  
);
```

The 'Script Output' pane at the bottom shows the message: 'Table PRODUCTS created.' The status bar at the bottom indicates 'Line 42 Column 1 | Insert | Modified | Windows: CI'.



The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the main editor is as follows:

```
insert into products values (1,'a','television',10000,5000);  
insert into products values (2,'b','Pillow Covers',500,2000);  
insert into products values (3,'c','Grooms',50,500);  
insert into products values (4,'b','Kurti',1000,1000);  
insert into products values (5,'c','Buckets',200,5000);  
insert into products values (6,'b','Sweatshirts',600,2000);  
insert into products values (7,'a','Samsung s20',5000,1000);  
insert into products values (8,'a','Samsung s21',10000,200);  
insert into products values (9,'d','Bike',100000,100);  
insert into products values (10,'c','Table',3000,1000);  
insert into products values (11,'c','Chair',2500,500);  
insert into products values (12,'a','Radio',1200,500);
```

The 'Script Output' pane at the bottom shows the message: '1 row inserted.' repeated three times. The status bar at the bottom indicates 'Line 53 Column 1 | Insert | Modified | Windows: CI'.

Worksheet Query Builder ADSL

```
select * from products;
```

Script Output x Query Result x

SQL All Rows Fetched: 12 in 0.019 seconds

| | PRODUCTID | CATEGORY | DETAIL | PRICE | STOCK |
|----|-----------|---------------|--------|-------|-------|
| 1 | 1 a | television | 10000 | 5000 | |
| 2 | 2 b | Pillow Covers | 500 | 2000 | |
| 3 | 3 c | Grooms | 50 | 500 | |
| 4 | 4 b | Kurti | 1000 | 1000 | |
| 5 | 5 c | Buckets | 200 | 5000 | |
| 6 | 6 b | Sweatshirts | 600 | 2000 | |
| 7 | 7 a | Samsung s20 | 5000 | 1000 | |
| 8 | 8 a | Samsung s21 | 10000 | 200 | |
| 9 | 9 d | Bike | 100000 | 100 | |
| 10 | 10 c | Table | 3000 | 1000 | |
| 11 | 11 c | chair | 2500 | 500 | |
| 12 | 12 a | Radio | 1200 | 500 | |

Line 61 Column 1 Insert Modified Windows: CI

Worksheet Query Builder ADSL 15.89700031 seconds

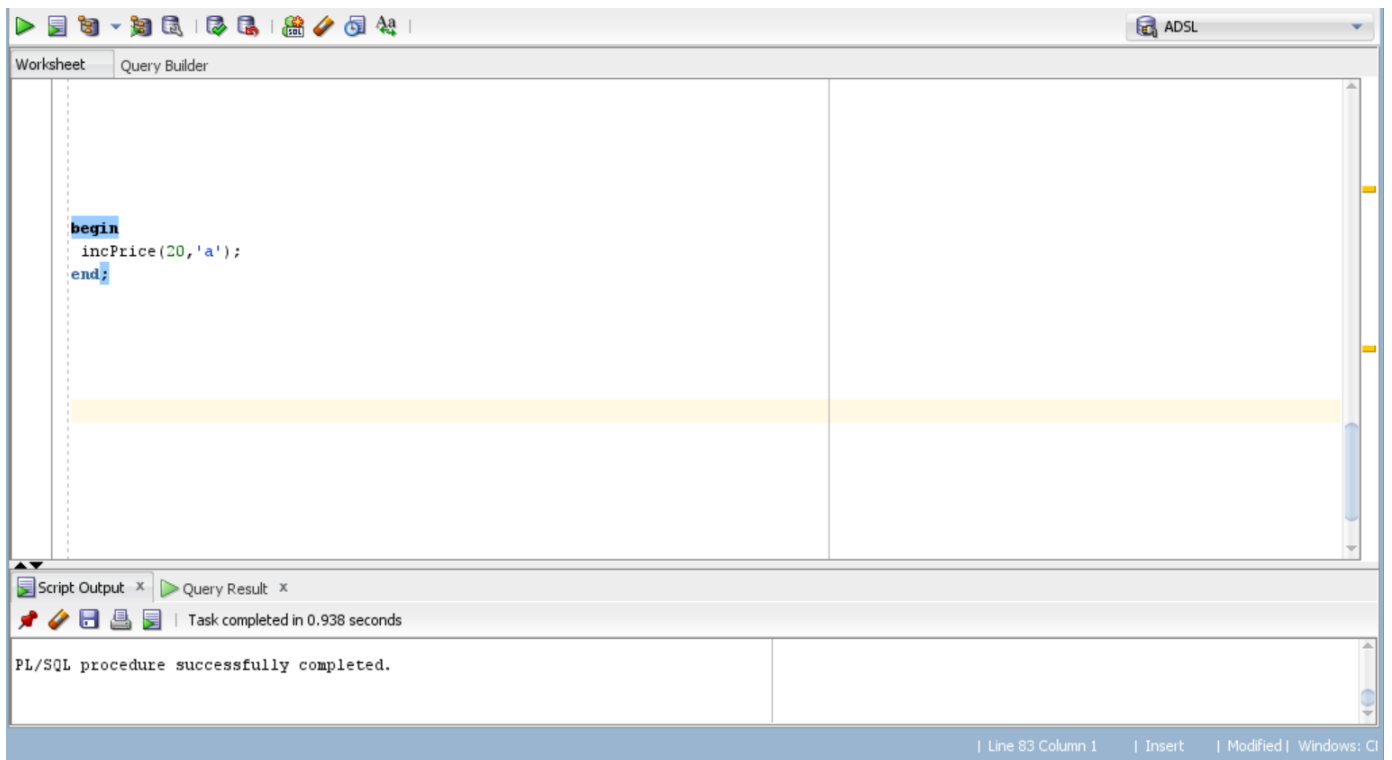
```
create or replace procedure incPrice(  
    x in products.price%type,  
    y in products.category%type)  
is  
begin  
    update products set price=price+price*x/100;  
    commit;  
end;
```

Script Output x Query Result x

Task completed in 15.897 seconds

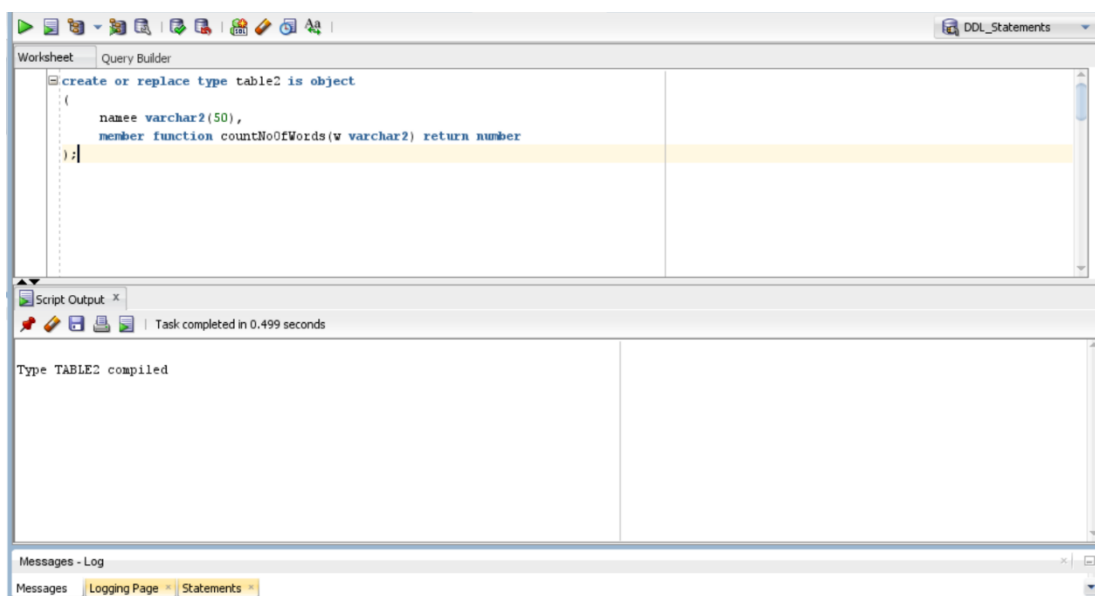
Procedure INCPRICE compiled

Line 68 Column 1 Insert Modified Windows: CI



Q.2

a) Create Object Table containing field “name” of size 50 characters and member function “countNoOfWords” which returns the no. of words in “name” field. Demonstrate the working by entering different data.



DDL_Statements

Worksheet Query Builder

```
create or replace type body table2 as
member function countNoOfWords(v varchar2) return number is
n number(2) := 1;
s char;
begin
for j in 1..length(v)
loop
s := substr(v, j, 1);
if s = ' ' then
n := n + 1;
end if;
end loop;
return n;
end countNoOfWords;
end;
```

Script Output

Task completed in 0.195 seconds

Type TABLE2 compiled

Type Body TABLE2 compiled

Messages - Log

Messages Logging Page Statements

DDL_Statements

Worksheet Query Builder

```
end countNoOfWords;
end;
```

```
declare
str table2;
begin
str := table2('walchand college of engineering');
dbms_output.put_line(str.countNoOfWords(str.name));
end;
```

Script Output

Task completed in 1.058 seconds

PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.

Dbms Output

Buffer Size: 20000

DDL_Statements

4

Messages - Log

Messages Logging Page Statements

- b) Create an address type with the following attributes : address, city, state & pincode. Include the following methods
- i. to extract the addresses based on given keyword.
 - j. to return the no. of words in each given field (method should accept the name of attribute/field)

The screenshot shows the SQL Developer interface with a worksheet containing the following SQL code:

```
CREATE OR REPLACE TYPE addresss_type_1 AS OBJECT
(
  address_1 VARCHAR2(255),
  city_1 VARCHAR2(50),
  state_1 VARCHAR2(50),
  pincode_1 NUMBER,
  MEMBER FUNCTION extract_addresses1 (keyword VARCHAR2)
  RETURN VARCHAR2,
  MEMBER FUNCTION count_words1 (field VARCHAR2)
  RETURN NUMBER
);
```

The Script Output pane at the bottom shows the message: "Type ADDRESS_TYPE_1 compiled".

The screenshot shows the SQL Developer interface with a worksheet containing the following SQL code:

```
);
CREATE OR REPLACE TYPE BODY addresss_type_1 AS
MEMBER FUNCTION extract_addresses1 (keyword VARCHAR2)
  RETURN VARCHAR2
IS
  extract_adrs VARCHAR2(1000);
BEGIN
  IF INSTR(address_1, keyword) > 0 THEN
    extract_adrs := address_1;
  END IF;
  RETURN extract_adrs;
END extract_addresses1;

MEMBER FUNCTION count_words1 (field VARCHAR2)
  RETURN NUMBER
IS
  word_count NUMBER;
BEGIN
  IF field = 'address_1' THEN
```

The Script Output pane at the bottom shows the message: "Type Body ADDRESS_TYPE_1 compiled".




Worksheet

Query Builder

```
insert into add_table values(address_type1('vishram bag','Sangli','Mh',416415));
insert into add_table values(address_type1('Tarabai park','kop','Mh',416003));
insert into add_table values(address_type1('Vijay nagar','sangli','Mh',416415));
insert into add_table values(address_type1('Gaon bag','Sangli','Mh',416415));
insert into add_table values(address_type1('Tilak wadi','Belgaum','Mh',416414));
insert into add_table values(address_type1('Shivaji Nagar','Gadhinglaj','Mh',416502));
insert into add_table values(address_type1('Jaysing pur','kop','Mh',416400));
```

Script Output x

Query Result x



Task completed in 0.103 seconds

```
1 row inserted.

1 row inserted.

1 row inserted.
```

ntifiers

| Line 140 Column 1 | Insert | Modified | Windows: C

- c) Create a user defined data type `course_Type` with 2 attributes `course_id`, `description`:
- i. Create an object table based on the type created.
 - j. Insert rows into the table
- Demonstrate the working with different data sets

The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the editor is as follows:

```
create or replace type course_type as object(  
    course_id varchar(10),  
    course_description varchar(250)  
);  
  
create table courses of course_type  
  
INSERT INTO courses  
VALUES (course_type ('5CS101','Programming for Problem Solving'));
```

The 'Script Output' tab is also visible, showing the execution results:

```
Type COURSE_TYPE compiled  
  
Table COURSES created.  
  
1 row inserted.
```

The screenshot shows the SQL Developer interface with the 'Query Builder' tab active. The SQL script in the editor is as follows:

```
create or replace type course_type as object(  
    course_id varchar(10),  
    course_description varchar(250)  
);  
  
create table courses of course_type  
  
INSERT INTO courses  
VALUES (course_type ('5CS101','Programming for Problem Solving'));  
  
SELECT* from courses;
```

The 'Query Result' tab is also visible, showing the execution results:

| COURSE_ID | COURSE_DESCRIPTION |
|-----------|---------------------------------|
| 1 5CS101 | Programming for Problem Solving |

Result/Observation:

Created a table with contains user defined data type and executed the query on table.

Created a table with basic data types and executed query.

Created an object table and member function and then executed the function for required answer.

Created the user defined data type and methods. Executed the methods.

Conclusion:

RDBMS fails handling non-primitive data and it cannot handle content-based information on non-primitive data. So, the Universal solution for the above problem is Object-Oriented paradigm.