

Date:31.01.2022

Third Year B. Tech., Sem VI 2021-22

Advanced Database System Lab

Assignment submission

PRN No: 2019BTECS00064

Full name: Kunal Santosh Kadam

Batch: T2

Assignment: 2

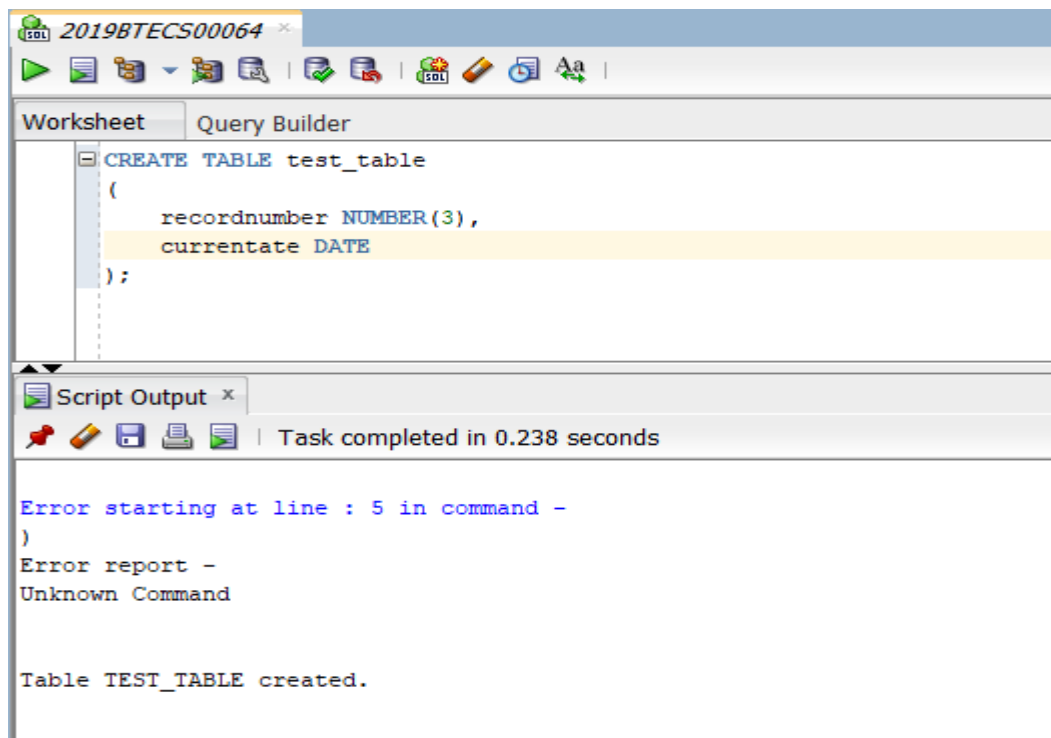
**Title of assignment: PL/SQL Review And Object Relational
Database**

I. PL/SQL Review:

- 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.

Ans

Create table test_table:



The screenshot shows the SQL Developer interface with a worksheet titled '2019BTECS00064'. The 'Query Builder' tab is active, displaying the following SQL statement:

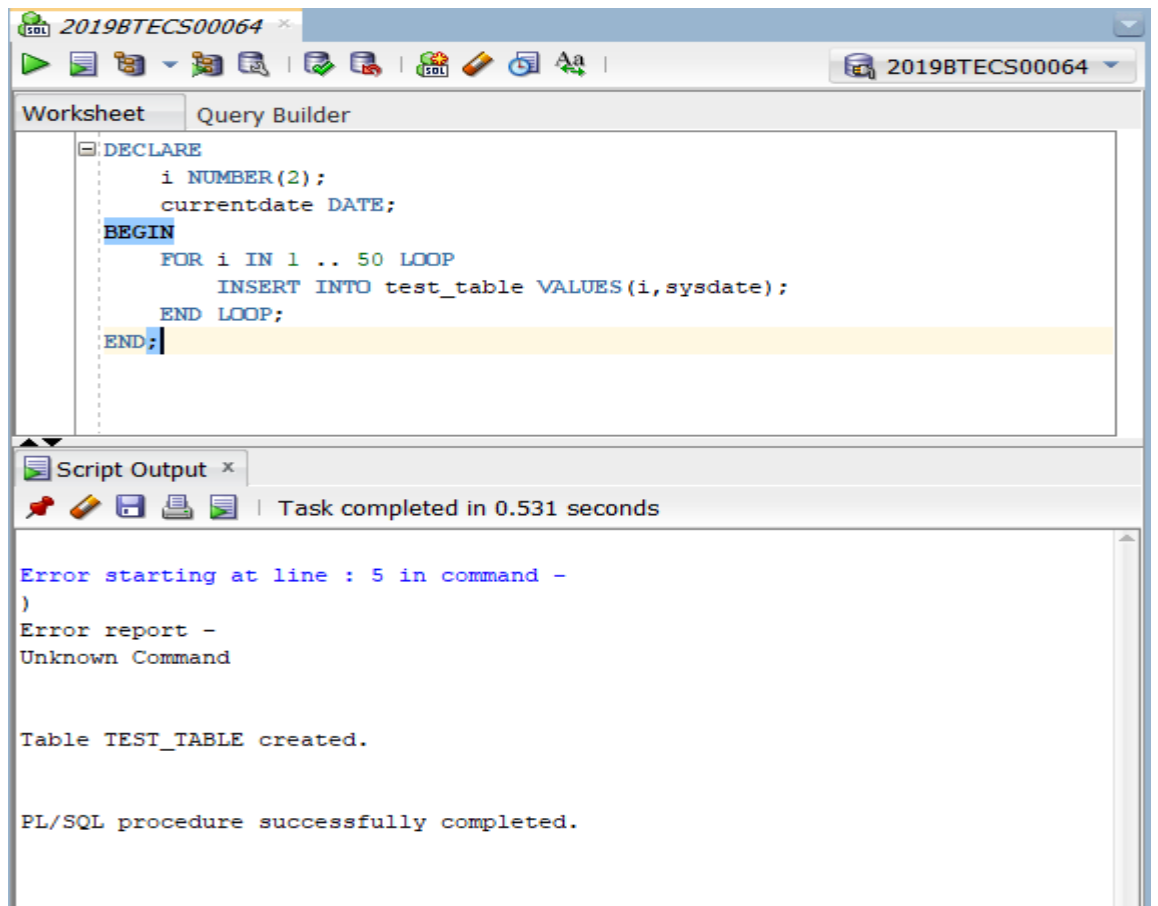
```
CREATE TABLE test_table
(
  recordnumber NUMBER(3),
  currentdate DATE
);
```

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

```
Error starting at line : 5 in command -
)
Error report -
Unknown Command

Table TEST_TABLE created.
```

PL/SQL statement to add 50 entries:



The screenshot shows the SQL Developer interface with a worksheet titled '2019BTECS00064'. The 'Query Builder' tab is active, displaying the following PL/SQL procedure:

```
DECLARE
  i NUMBER(2);
  currentdate DATE;
BEGIN
  FOR i IN 1 .. 50 LOOP
    INSERT INTO test_table VALUES(i,sysdate);
  END LOOP;
END;
```

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

```
Error starting at line : 5 in command -
)
Error report -
Unknown Command

Table TEST_TABLE created.





PL/SQL procedure successfully completed.
```

Table Values:

The screenshot shows a SQL query execution window. The title bar indicates the database is '2019BTECS00064'. The 'Query Builder' tab is active, showing the query: `SELECT * FROM test_table;`. Below the query, the 'Script Output' and 'Query Result' tabs are visible. The 'Query Result' tab shows the execution status: 'All Rows Fetched: 50 in 0.191 seconds'. The query result is displayed as a table with two columns: 'RECORDNUMBER' and 'CURRENTATE'. The table contains 33 rows of data, with record numbers ranging from 1 to 33 and current dates all being '05-02-22'.

RECORDNUMBER	CURRENTATE
1	05-02-22
2	05-02-22
3	05-02-22
4	05-02-22
5	05-02-22
6	05-02-22
7	05-02-22
8	05-02-22
9	05-02-22
10	05-02-22
11	05-02-22
12	05-02-22
13	05-02-22
14	05-02-22
15	05-02-22
16	05-02-22
17	05-02-22
18	05-02-22
19	05-02-22
20	05-02-22
21	05-02-22
22	05-02-22
23	05-02-22
24	05-02-22
25	05-02-22
26	05-02-22
27	05-02-22
28	05-02-22
29	05-02-22
30	05-02-22
31	05-02-22
32	05-02-22
33	05-02-22

Script Output x Query Result x

    SQL | All Rows Fetched: 50 in 0.191 seconds

	RECORDNUMBER	CURRENTATE
31	31	05-02-22
32	32	05-02-22
33	33	05-02-22
34	34	05-02-22
35	35	05-02-22
36	36	05-02-22
37	37	05-02-22
38	38	05-02-22
39	39	05-02-22
40	40	05-02-22
41	41	05-02-22
42	42	05-02-22
43	43	05-02-22
44	44	05-02-22
45	45	05-02-22
46	46	05-02-22
47	47	05-02-22
48	48	05-02-22
49	49	05-02-22
50	50	05-02-22

- 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.

Ans

Create Products table:

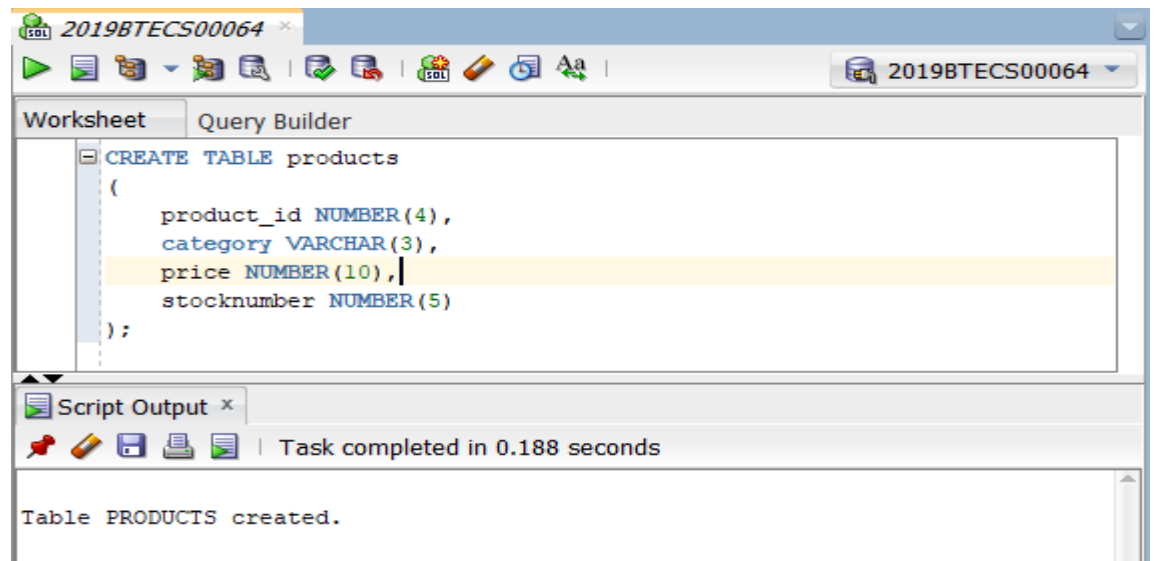


Table Data:

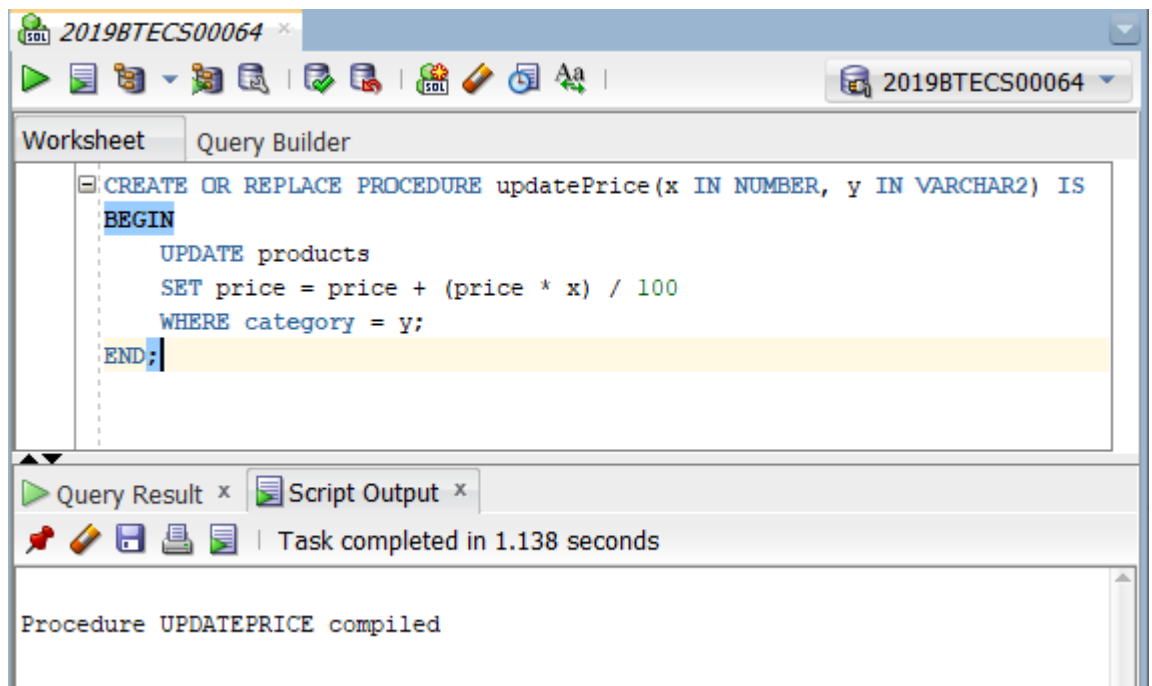
The screenshot shows the SQL Developer interface with the 'Query Result' tab selected. The SQL statement in the main editor is:

```
SELECT * FROM products;
```

The 'Query Result' tab at the bottom shows the data fetched from the table. The status bar indicates 'All Rows Fetched: 15 in 0.009 seconds'.

	PRODUCT_ID	CATEGORY	PRICE	STOCKNUMBER
1	1 A		123	12
2	2 B		125	10
3	3 C		134	45
4	4 D		160	23
5	5 A		20	23
6	6 D		200	43
7	7 B		344	56
8	8 C		56	6
9	9 A		344	2
10	10 B		790	45
11	11 C		34	56
12	12 A		23	67
13	13 D		345	34
14	14 B		156	23
15	15 D		134	56

SQL Procedure to update price:

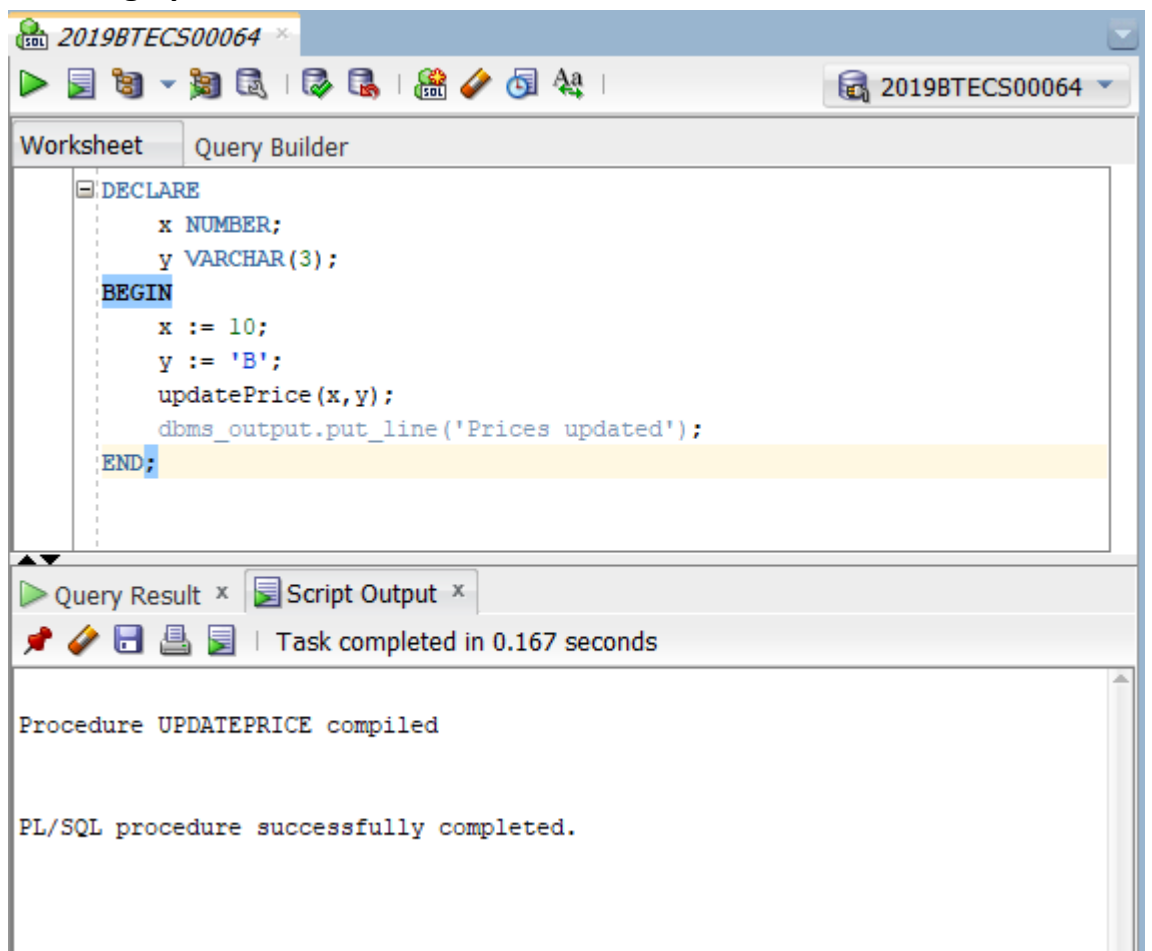


The screenshot shows the SQL Developer interface with a window titled '2019BTECS00064'. The 'Query Builder' tab is active, displaying the following SQL code:

```
CREATE OR REPLACE PROCEDURE updatePrice(x IN NUMBER, y IN VARCHAR2) IS
BEGIN
    UPDATE products
    SET price = price + (price * x) / 100
    WHERE category = y;
END;
```

Below the code editor, the 'Query Result' and 'Script Output' tabs are visible. The 'Script Output' tab shows the message: 'Task completed in 1.138 seconds'. The main output area displays: 'Procedure UPDATEPRICE compiled'.

Invoking updatePrice:

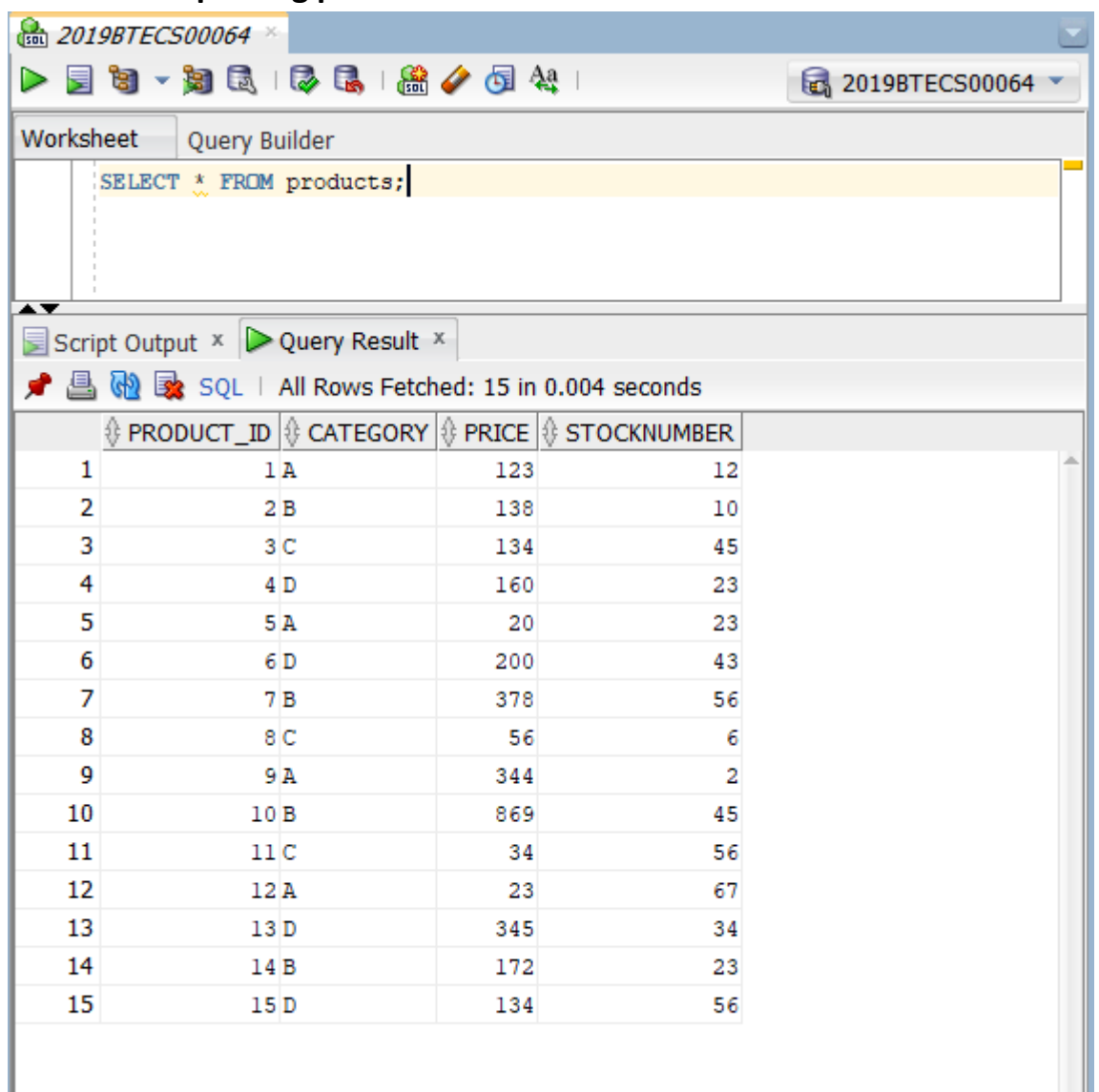


The screenshot shows the SQL Developer interface with the same window '2019BTECS00064'. The 'Query Builder' tab is active, displaying the following SQL code:

```
DECLARE
    x NUMBER;
    y VARCHAR(3);
BEGIN
    x := 10;
    y := 'B';
    updatePrice(x,y);
    dbms_output.put_line('Prices updated');
END;
```

Below the code editor, the 'Query Result' and 'Script Output' tabs are visible. The 'Script Output' tab shows the message: 'Task completed in 0.167 seconds'. The main output area displays two messages: 'Procedure UPDATEPRICE compiled' and 'PL/SQL procedure successfully completed.'

Table after updating prices:



The screenshot shows a database application window titled "2019BTECS00064". The interface includes a toolbar with various icons, a "Worksheet" tab, and a "Query Builder" tab. The "Query Builder" tab displays the SQL query: `SELECT * FROM products;`. Below the query, there is a "Script Output" tab and a "Query Result" tab. The "Query Result" tab shows the results of the query, indicating that all 15 rows were fetched in 0.004 seconds. The results are displayed in a table with the following columns: PRODUCT_ID, CATEGORY, PRICE, and STOCKNUMBER.

	PRODUCT_ID	CATEGORY	PRICE	STOCKNUMBER
1	1	A	123	12
2	2	B	138	10
3	3	C	134	45
4	4	D	160	23
5	5	A	20	23
6	6	D	200	43
7	7	B	378	56
8	8	C	56	6
9	9	A	344	2
10	10	B	869	45
11	11	C	34	56
12	12	A	23	67
13	13	D	345	34
14	14	B	172	23
15	15	D	134	56

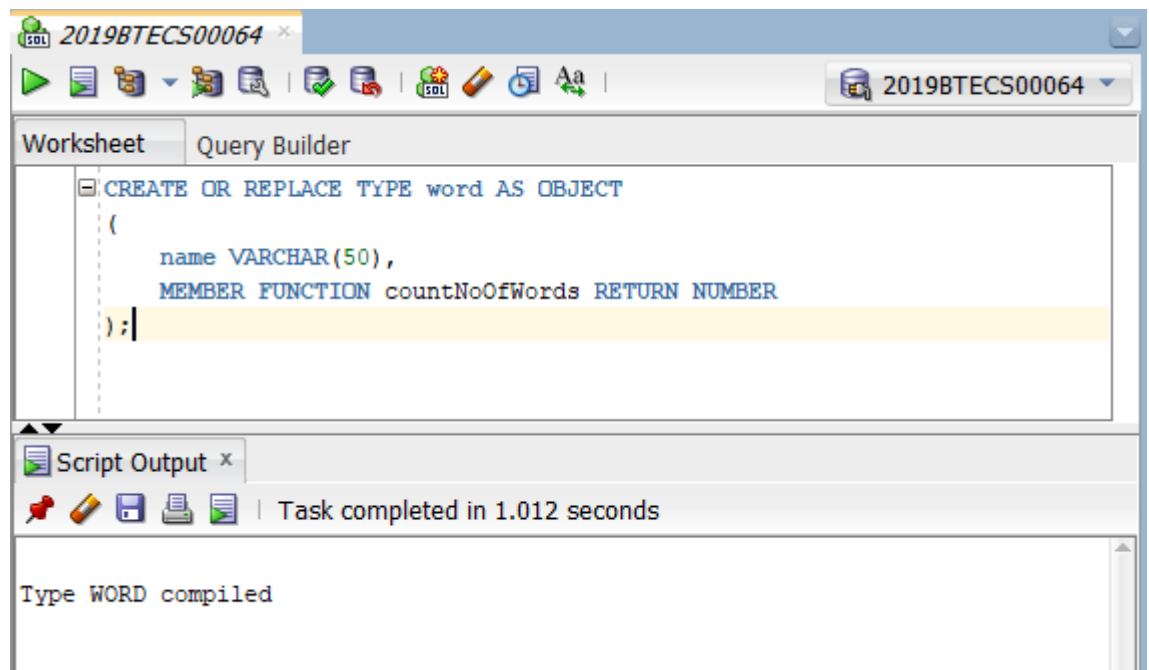
II. Object Relational Databases:

- 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.

Ans:

Create the type “word”:



Create the body of the type “word”:

The screenshot shows the SQL Developer interface with a script titled '2019BTECS00064'. The script is being edited in the 'Query Builder' tab. The script defines a type body for 'word' with a member function 'countNoOfWords' that counts the number of words in a string by splitting it on spaces, commas, and periods. The script is as follows:

```
CREATE OR REPLACE TYPE BODY word AS
  MEMBER FUNCTION countNoOfWords
  RETURN NUMBER IS
    wordcount NUMBER(20) := 1;
    ch char;
  BEGIN
    FOR i IN 1 .. Length(name) LOOP
      ch := SUBSTR(name, i, 1);
      IF ch = ' ' or ch = '.' or ch = ',' then
        wordcount := wordcount + 1;
      END IF;
    END LOOP;
    RETURN wordcount;
  END countNoOfWords;
END;
```

The 'Script Output' tab shows the compilation results. It indicates that the 'Type WORD' and 'Type Body WORD' were compiled successfully. However, there are two errors listed:

LINE/COL	ERROR
7/9	PL/SQL: Statement ignored
7/23	PLS-00201: identifier 'L' must be declared

The errors suggest that there is a syntax issue in the script, likely related to the use of the variable 'L' in the loop condition, which has not been declared. The output also includes the message 'Errors: check compiler log'.

Create table of word type:

The screenshot displays the Oracle SQL Developer environment. The top toolbar includes icons for running, saving, and other database operations. The main window is titled '2019BTECS00064' and contains a 'Query Builder' tab. The SQL script in the query builder is as follows:

```
CREATE TABLE word_t of word;  
INSERT INTO word_t VALUES(('Kunal Kadam'));  
INSERT INTO word_t VALUES(('2019BTECS00064'));  
INSERT INTO word_t VALUES(('I am studing at Third Year'));  
INSERT INTO word_t VALUES(('Course is Advanced Database System'));  
INSERT INTO word_t VALUES(('Example of creating object table'));
```

Below the query builder is the 'Script Output' window, which shows the execution results. The output indicates that the table 'WORD_T' was created successfully and that five rows were inserted. There is also a warning message about a statement being ignored and an identifier 'L' not being declared.

Task completed in 0.443 seconds

Type WORD compiled

Type Body WORD compiled

LINE/COL ERROR

7/9 PL/SQL: Statement ignored

7/23 PLS-00201: identifier 'L' must be declared

Errors: check compiler log

Type Body WORD compiled

Table WORD_T created.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

Query to get count of words:

The screenshot displays the SQL Server Enterprise Manager interface. The top toolbar includes icons for running queries, saving, and other database operations. The 'Query Builder' tab is active, showing a SQL query that selects the name and word count from a table named 'wordtable'. The query is as follows:

```
SELECT  
VALUE(wordtable) .name AS sentence,  
VALUE(wordtable) .countNoOfWords() as wordcount  
FROM word_t wordtable;
```

Below the query editor, the 'Query Results' tab is active, showing the output of the query. The results are displayed in a grid with two columns: 'SENTENCE' and 'WORDCOUNT'. The data consists of five rows, each representing a sentence and its corresponding word count.

	SENTENCE	WORDCOUNT
1	Kunal Kadam	2
2	2019BTECS00064	1
3	I am studing at Third Year	6
4	Course is Advanced Database System	5
5	Example of creating object table	5

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

Ans

Create Type "address":

The screenshot shows the Oracle SQL Developer interface. The top toolbar includes icons for running, saving, and other database operations. The main window is titled 'Worksheet' and contains the following SQL code:

```
CREATE TYPE address AS OBJECT
(
    short_address VARCHAR2(100),
    city VARCHAR2(20),
    state VARCHAR2(20),
    pincode NUMBER(7),

    MEMBER FUNCTION getAddress(key_val VARCHAR) RETURN NUMBER,
    MEMBER FUNCTION no_of_words(type VARCHAR) RETURN NUMBER
);

CREATE OR REPLACE TYPE BODY address AS
MEMBER FUNCTION getAddress(key_val in VARCHAR)
RETURN NUMBER IS
no_word NUMBER(20) := 0;
s CHAR := ' ';
str VARCHAR(20) := ' ';
flg NUMBER(1) := 0;
j NUMBER(10) := 0;
```

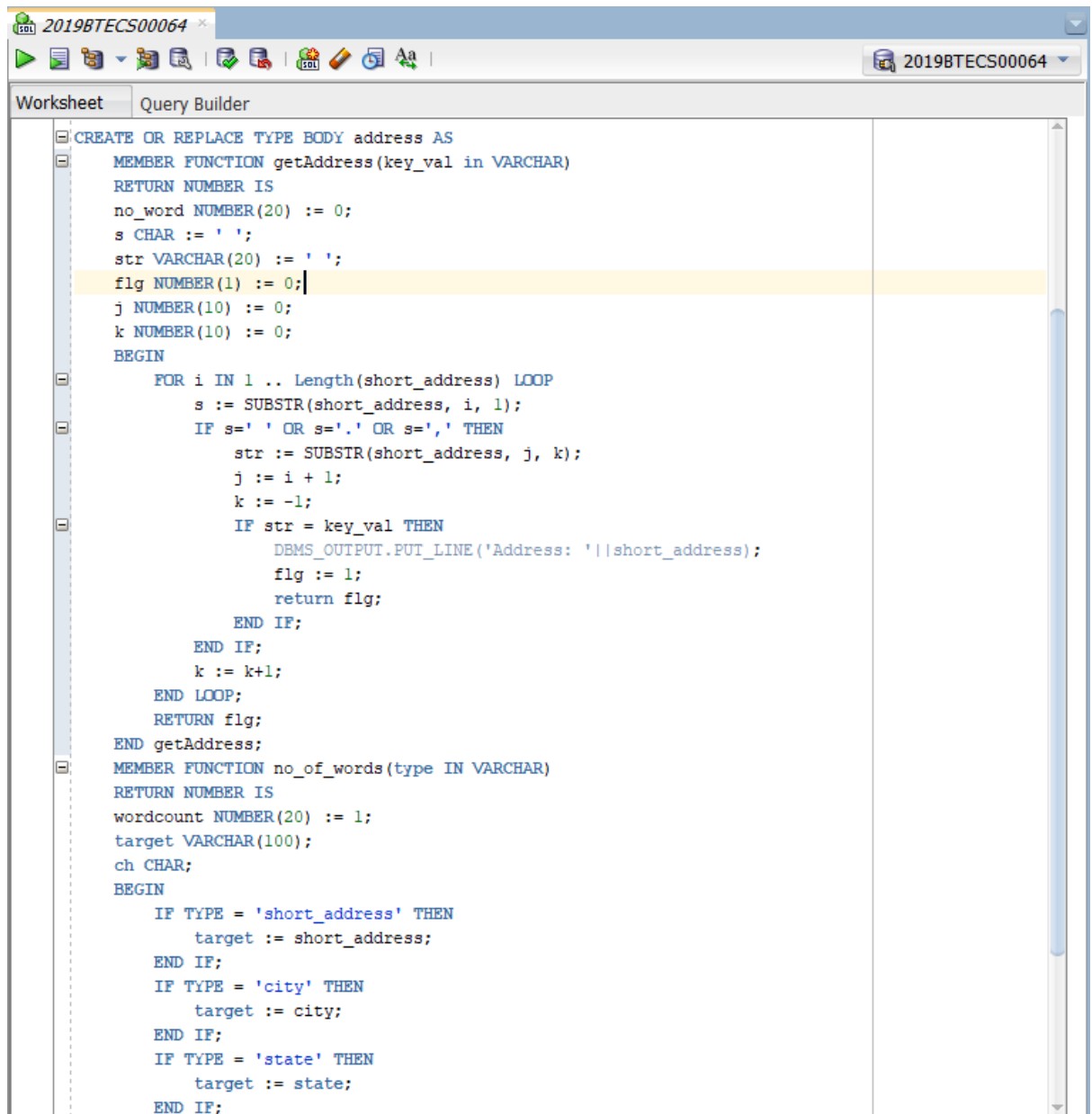
Below the code editor, the 'Script Output' window shows the execution results:

```
Task completed in 0.119 seconds

Type ADDRESS compiled

Type Body ADDRESS compiled
```

Create body for the “address”:



The screenshot shows a SQL IDE window with the title '2019BTECS00064'. The interface includes a toolbar with icons for running, saving, and other database operations. The main workspace is divided into two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab is active, displaying the following PL/SQL code:

```
CREATE OR REPLACE TYPE BODY address AS
  MEMBER FUNCTION getAddress(key_val IN VARCHAR)
  RETURN NUMBER IS
    no_word NUMBER(20) := 0;
    s CHAR := ' ';
    str VARCHAR(20) := ' ';
    flg NUMBER(1) := 0;
    j NUMBER(10) := 0;
    k NUMBER(10) := 0;
  BEGIN
    FOR i IN 1 .. Length(short_address) LOOP
      s := SUBSTR(short_address, i, 1);
      IF s = ' ' OR s = '.' OR s = ',' THEN
        str := SUBSTR(short_address, j, k);
        j := i + 1;
        k := -1;
        IF str = key_val THEN
          DBMS_OUTPUT.PUT_LINE('Address: ' || short_address);
          flg := 1;
          return flg;
        END IF;
      END IF;
      k := k + 1;
    END LOOP;
    RETURN flg;
  END getAddress;
  MEMBER FUNCTION no_of_words(type IN VARCHAR)
  RETURN NUMBER IS
    wordcount NUMBER(20) := 1;
    target VARCHAR(100);
    ch CHAR;
  BEGIN
    IF TYPE = 'short_address' THEN
      target := short_address;
    END IF;
    IF TYPE = 'city' THEN
      target := city;
    END IF;
    IF TYPE = 'state' THEN
      target := state;
    END IF;
  END IF;
```

2019BTECS00064

Worksheet Query Builder

```
BEGIN
  FOR i IN 1 .. Length(short_address) LOOP
    s := SUBSTR(short_address, i, 1);
    IF s=' ' OR s='.' OR s=',' THEN
      str := SUBSTR(short_address, j, k);
      j := i + 1;
      k := -1;
      IF str = key_val THEN
        DEMS_OUTPUT.PUT_LINE('Address: '||short_address);
        flg := 1;
        return flg;
      END IF;
    END IF;
    k := k+1;
  END LOOP;
  RETURN flg;
END getAddress;

MEMBER FUNCTION no_of_words(type IN VARCHAR)
RETURN NUMBER IS
wordcount NUMBER(20) := 1;
target VARCHAR(100);
ch CHAR;
BEGIN
  IF TYPE = 'short_address' THEN
    target := short_address;
  END IF;
  IF TYPE = 'city' THEN
    target := city;
  END IF;
  IF TYPE = 'state' THEN
    target := state;
  END IF;
  FOR i in 1 .. Length(target) LOOP
    ch := SUBSTR(target, i, 1);
    IF ch = ' ' OR ch = '.' OR ch = ',' THEN
      wordcount := wordcount + 1;
    END IF;
  END LOOP;
  RETURN wordcount;
END no_of_words;
END;
```

Table Entries:

The screenshot shows a database management tool interface with two main panes. The top pane, titled 'Worksheet' and 'Query Builder', contains the following SQL script:

```
CREATE TABLE address_table of address;  
DESC address_table;  
  
INSERT INTO address_table VALUES('Ankush Nagar','Nagpur','Maharashtra', 415002);  
INSERT INTO address_table VALUES('Friends Colony','Beed','Maharashtra', 415102);  
INSERT INTO address_table VALUES('Sayaji Nagar','Kolhapur','Maharashtra', 415012);  
INSERT INTO address_table VALUES('Mahada Colony','Nagpur','Maharashtra', 415104);  
INSERT INTO address_table VALUES('Shivaj Nagar','Kolhapur','Maharashtra', 415013);
```

The bottom pane, titled 'Script Output', shows the execution results:

Table ADDRESS_TABLE created.

Name	Null?	Type
SHORT_ADDRESS		VARCHAR2(100)
CITY		VARCHAR2(20)
STATE		VARCHAR2(20)
PINCODE		NUMBER(7)

1 row inserted.

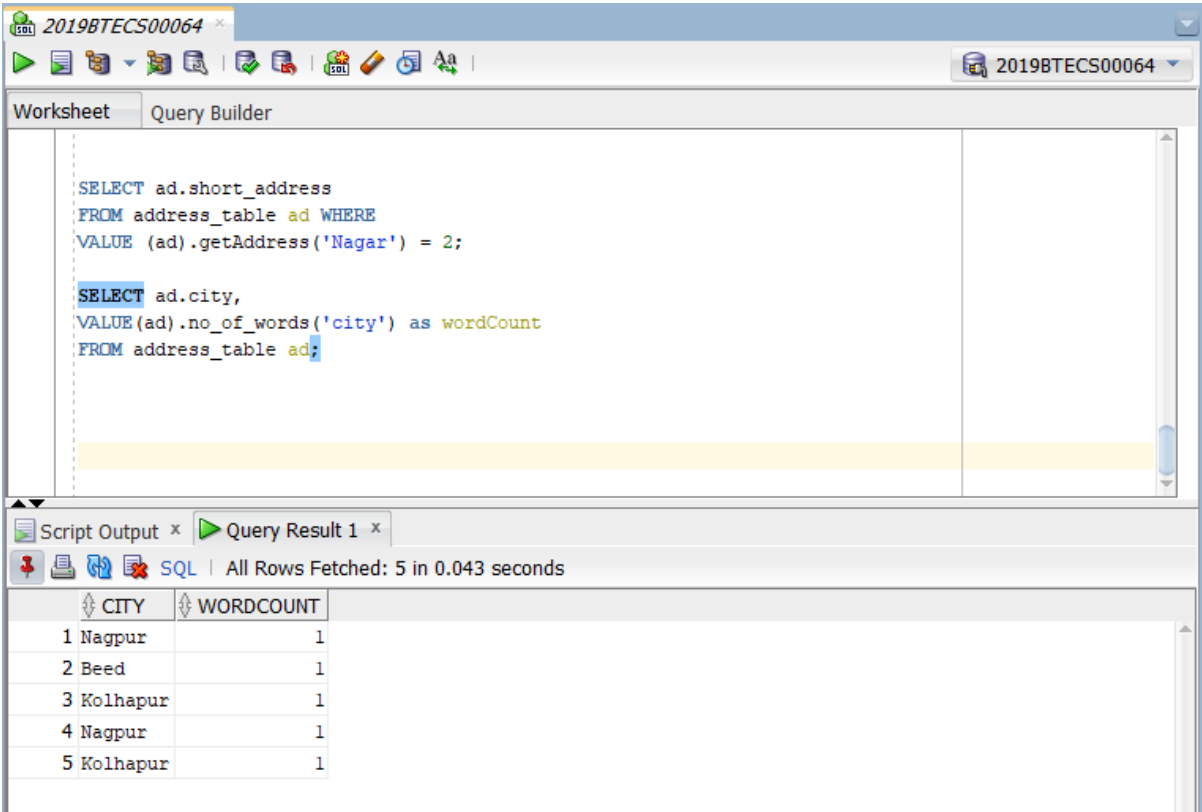
1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

Word count based on attribute name:



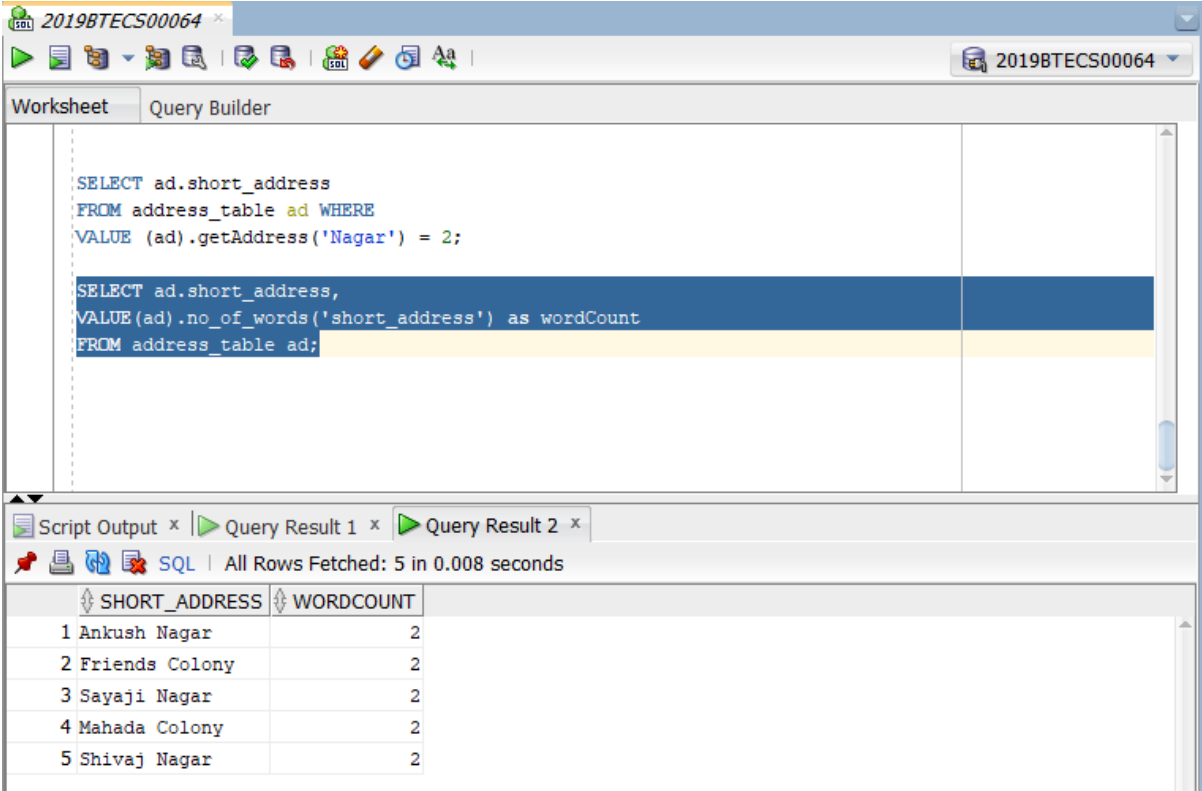
The screenshot shows a database query editor with a toolbar and a query window. The query window contains two SQL queries. The first query filters for 'Nagar' in the 'short_address' field. The second query counts the number of words in the 'city' attribute for all records in the 'address_table'.

```
SELECT ad.short_address
FROM address_table ad WHERE
VALUE (ad).getAddress('Nagar') = 2;

SELECT ad.city,
VALUE(ad).no_of_words('city') as wordCount
FROM address_table ad;
```

Below the query window, the 'Query Result 1' tab is active, showing a table with 5 rows and 2 columns: CITY and WORDCOUNT.

	CITY	WORDCOUNT
1	Nagpur	1
2	Beed	1
3	Kolhapur	1
4	Nagpur	1
5	Kolhapur	1



The screenshot shows the same database query editor. The query window now contains a query that counts the number of words in the 'short_address' attribute for all records in the 'address_table'.

```
SELECT ad.short_address
FROM address_table ad WHERE
VALUE (ad).getAddress('Nagar') = 2;

SELECT ad.short_address,
VALUE(ad).no_of_words('short_address') as wordCount
FROM address_table ad;
```

Below the query window, the 'Query Result 2' tab is active, showing a table with 5 rows and 2 columns: SHORT_ADDRESS and WORDCOUNT.

	SHORT_ADDRESS	WORDCOUNT
1	Ankush Nagar	2
2	Friends Colony	2
3	Sayaji Nagar	2
4	Mahada Colony	2
5	Shivaj Nagar	2

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

Demonstrate the working with different data sets

Ans

Course Type:

The screenshot displays the Oracle SQL Developer environment. The top toolbar includes icons for running, saving, and other database operations. The main window is titled '2019BTECS00064' and contains a 'Worksheet' tab. The SQL script in the worksheet is as follows:

```
CREATE OR REPLACE TYPE course AS OBJECT
(
    courseid VARCHAR(10),
    description VARCHAR2(100)
);
```

Below the script, the 'Script Output' tab is visible, showing the execution results. The output indicates that the type was compiled successfully, but there is a syntax error on line 3/25:

```
Task completed in 0.178 seconds

Type COURSE compiled

LINE/COL ERROR
-----
3/25      PLS-00103: Encountered the symbol ";" when expecting one of the following:      := ) , not null <
Errors: check compiler log

Type COURSE compiled
```

Creating Table of Type Course:

The screenshot displays a database management interface with two main panes. The top pane, titled 'Worksheet' and 'Query Builder', contains a SQL query: `CREATE TABLE coursetable OF course;`. The bottom pane, titled 'Script Output', shows the execution results. It indicates that the 'Type COURSE' was compiled successfully. However, an error occurred at line 3, column 25, with the message: 'PLS-00103: Encountered the symbol ";" when expecting one of the following: :=) , not null'. The error suggests a syntax issue with the semicolon at the end of the query. The output also states that the 'Table COURSETABLE' was created.

Worksheet Query Builder

```
CREATE TABLE coursetable OF course;
```

Script Output x

Task completed in 0.099 seconds

Type COURSE compiled

LINE/COL ERROR

3/25 PLS-00103: Encountered the symbol ";" when expecting one of the following: :=) , not null (

Errors: check compiler log

Type COURSE compiled

Table COURSETABLE created.

Insert Values into Table:

The screenshot displays a database management interface with two panels. The top panel shows a SQL script with four INSERT statements. The bottom panel shows the script output, which includes a compiler error and successful execution messages. A second screenshot below shows the same interface with a SELECT query and its results displayed in a table.

Worksheet: Query Builder

```
INSERT INTO coursetable VALUES('4CS301','Computer Networks');
INSERT INTO coursetable VALUES('4CS302','Data Structures');
INSERT INTO coursetable VALUES('4CS303','Design and Analysis of Algorithms');
INSERT INTO coursetable VALUES('4CS304','Advanced Database Engineering');
```

Script Output

Task completed in 0.07 seconds

Type COURSE compiled

LINE/COL ERROR

3/25 PLS-00103: Encountered the symbol ";" when expecting one of the following: :=) , not null (

Errors: check compiler log

Type COURSE compiled

Table COURSETABLE created.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

Worksheet: Query Builder

```
SELECT * from coursetable;
```

Script Output | **Query Result**

All Rows Fetched: 4 in 0.006 seconds

COURSEID	DESCRIPTION
1 4CS301	Computer Networks
2 4CS302	Data Structures
3 4CS303	Design and Analysis of Algorithms
4 4CS304	Advanced Database Engineering

