TYPES OF PL/SQL BLOCK

OVERVIEW OF THE TYPES OF PL/SQL BLOCKS:

The P/L SQL blocks can be divided into two broad categories:

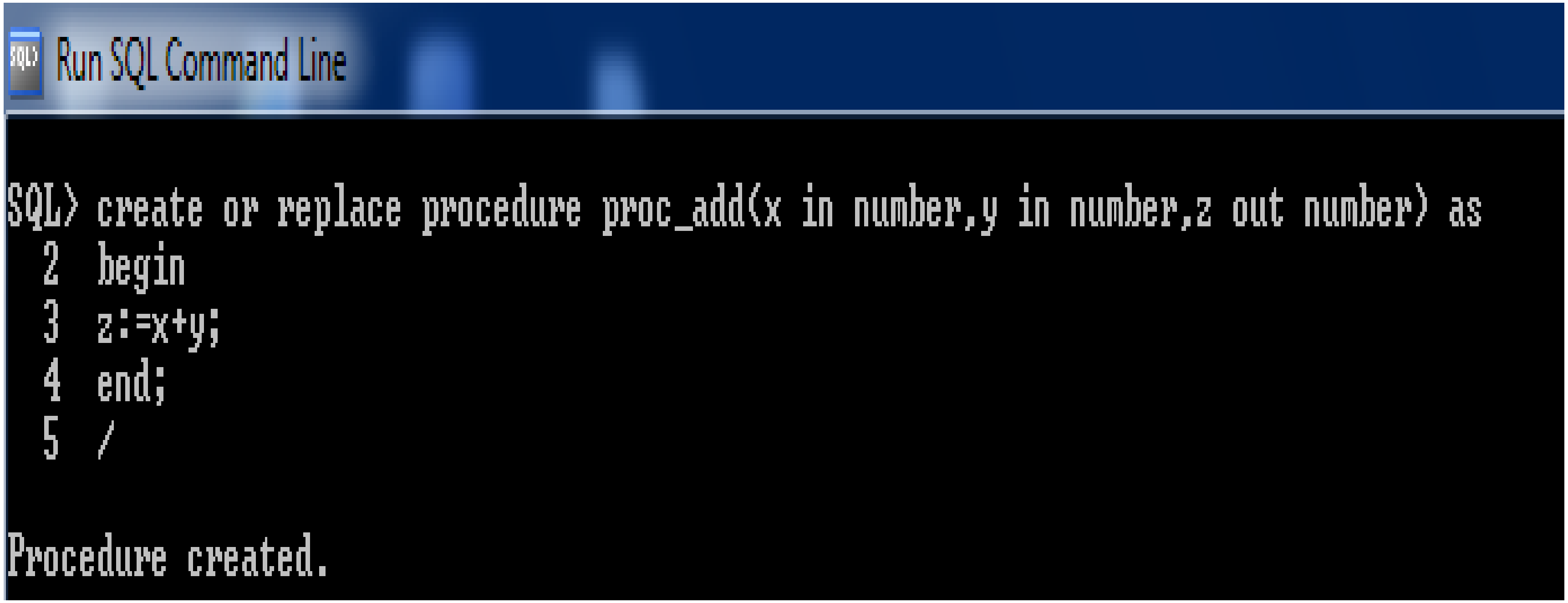
Anonymous Block:

The anonymous block is the simplest unit in PL/SQL. It is called anonymous block because it is not saved in the database.

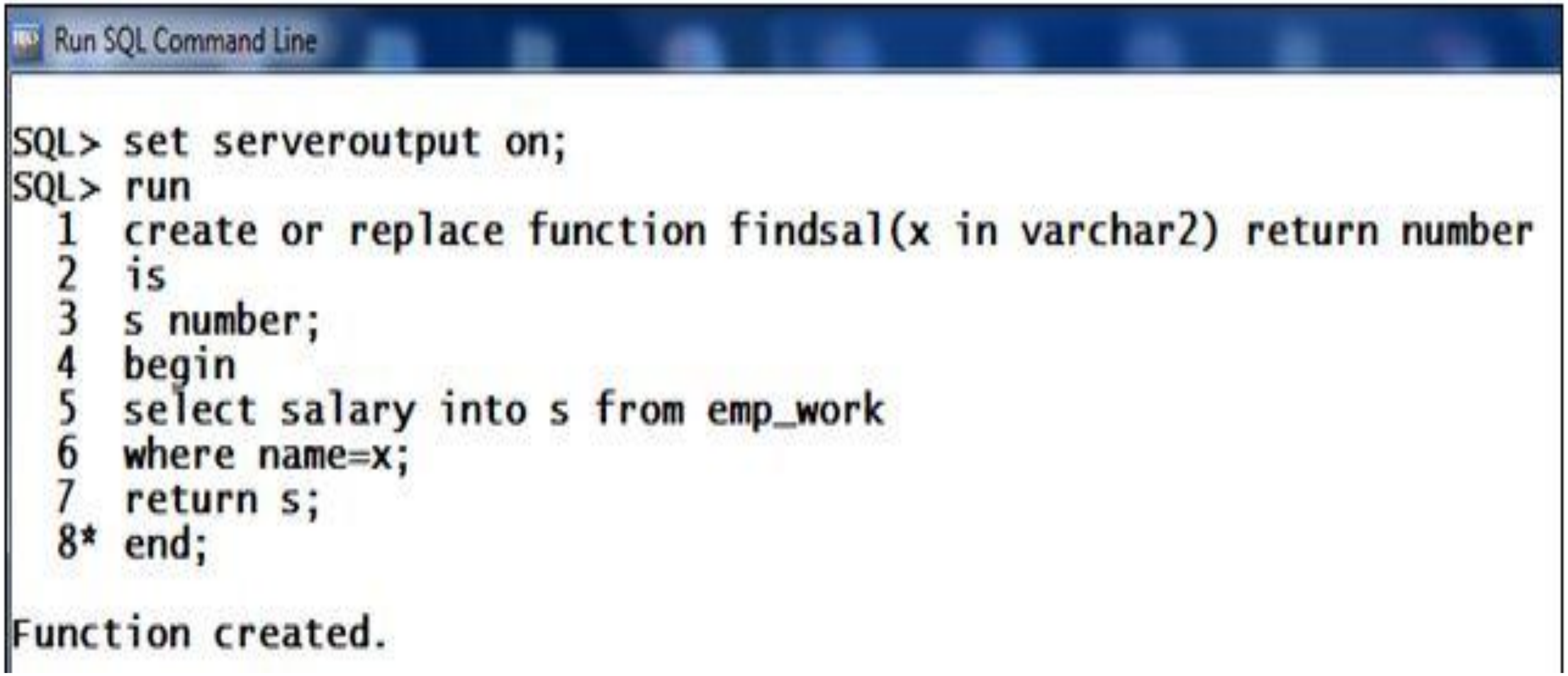
It is the P/L SQL Block without name. The Anonymous block is explained in detail in the next section.

Named Block: Named Block is a type of block which starts with the HEADER section which specifies the name and the type of the block. There are2 types of named blocks namely:-

- **Procedures:** It is a collection of statements which collectively perform a certain task. It passes variables through parameters and return one or more value.
- **Functions:** It is a series of statements performing a specific task and returning only one value.



```
SQL> create or replace procedure proc_add(x in number,y in number,z out number) as
  2  begin
  3  z:=x+y;
  4  end;
  5  /

Procedure created.
```



```
SQL> set serveroutput on;
SQL> run
  1  create or replace function findsal(x in varchar2) return number
  2  is
  3  s number;
  4  begin
  5  select salary into s from emp_work
  6  where name=x;
  7  return s;
  8* end;

Function created.
```

STRUCTURE OF A PL SQL BLOCK:

Each PL/SQL program consists of SQL and PL/ SQL statements which form a PL /SQL block. A PL/SQL Block consists of four sections:

The Header section.

The Declaration section.

The Execution section.

The Exception (or Error) Handling section.

HEADER

<Type and Name of block >

DECLARE

<All variables, Cursors are declared here>

BEGIN

<All programming logic, queries, program statements are written here>

EXCEPTION

<All Error Handling code is written here>

Header:

Relevant for named blocks only, the header determines the way that the named block or program must be called.

The header includes the name, parameter list, and RETURN clause (for a function only).

Declaration Section:

The Declaration section of a PL/SQL Block starts with the reserved keyword DECLARE.

Any variables, constants, records and cursors, which are used to manipulate data in the execution section are declared here. Cursors are also declared in this section.

This section is optional.

Execution Section:

The Execution section of a PL SQL Block starts with the reserved keyword BEGIN and ends with END.

This is a mandatory section and it is the section where the execution statements are written to perform a specific task.

The programmatic constructs like loops, conditional statement and SQL statements form the part of execution section.

This section is a mandatory section of the PL SQL Block.

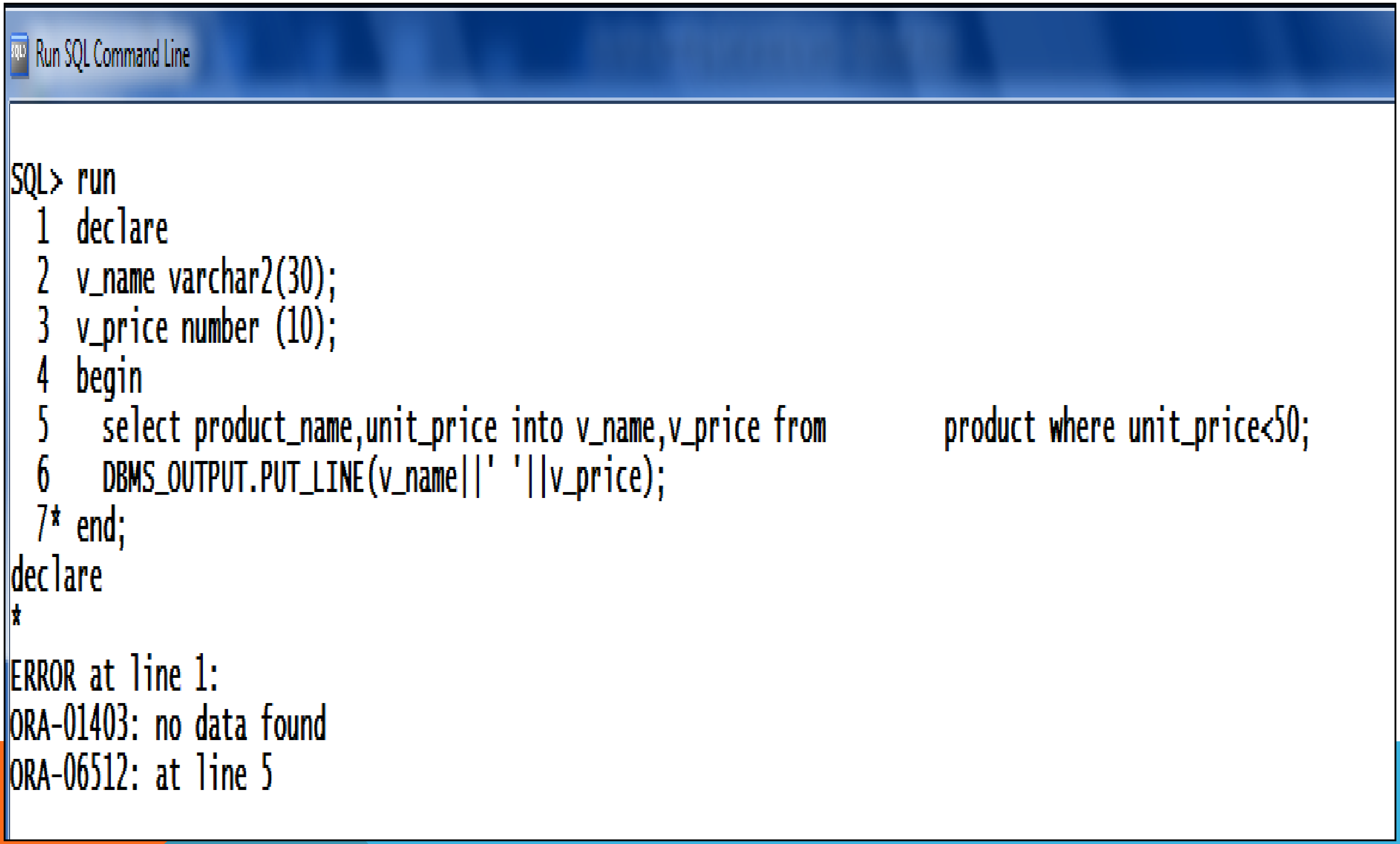
Exception Section:

The Exception section of a PL/SQL Block starts with the reserved keyword EXCEPTION.

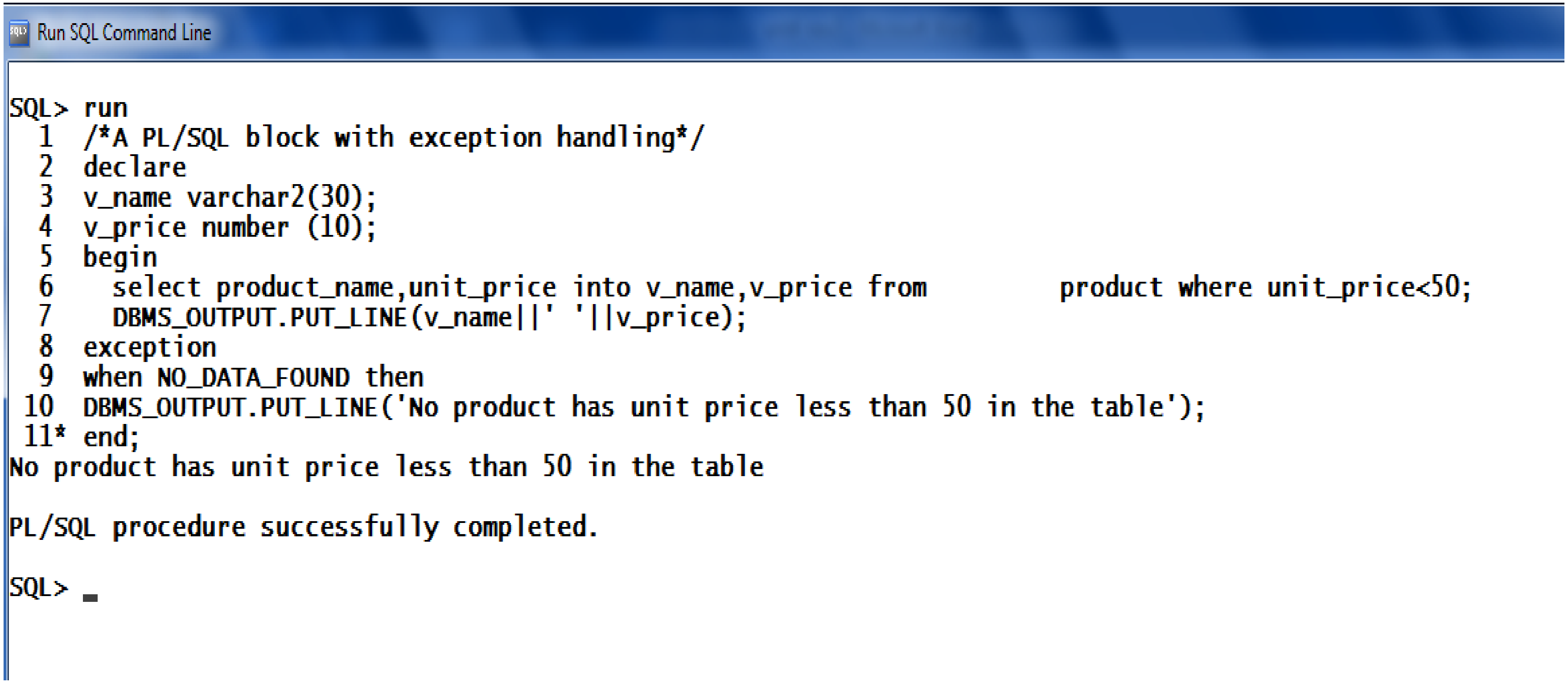
This section is optional.

Any errors in the program can be handled in this section, so that the PL /SQL Blocks terminate gracefully.

If the PL /SQL Block contain exceptions that cannot be handled, the Block terminates abruptly with errors.



```
SQL> run
  1  declare
  2  v_name varchar2(30);
  3  v_price number (10);
  4  begin
  5    select product_name,unit_price into v_name,v_price from         product where unit_price<50;
  6    DBMS_OUTPUT.PUT_LINE(v_name||' '||v_price);
  7* end;
declare
*
ERROR at line 1:
ORA-01403: no data found
ORA-06512: at line 5
```



```
SQL> run
/*A PL/SQL block with exception handling*/
declare
v_name varchar2(30);
v_price number (10);
begin
  select product_name,unit_price into v_name,v_price from         product where unit_price<50;
  DBMS_OUTPUT.PUT_LINE(v_name||' '||v_price);
exception
when NO_DATA_FOUND then
DBMS_OUTPUT.PUT_LINE('No product has unit price less than 50 in the table');
end;
No product has unit price less than 50 in the table

PL/SQL procedure successfully completed.

SQL> _
```

RULES OF WRITING A PL/ SQL BLOCK:

Every statement in the sections of a PL/SQL block must end with a semicolon.

PL/SQL blocks can be nested within other PL/SQL blocks.

Comments can be used to document code.

Data definition statements like CREATE, DROP, or ALTER are not allowed in a PL SQL block.

CREATION OF AN ANONYMOUS PL/ SQL BLOCK:

The anonymous block is a type of PL SQL block which has no name associated with it. In fact, the anonymous block is missing the header section altogether.

Instead it simply uses the DECLARE reserved word to mark the beginning of its optional declaration section.

For Example,

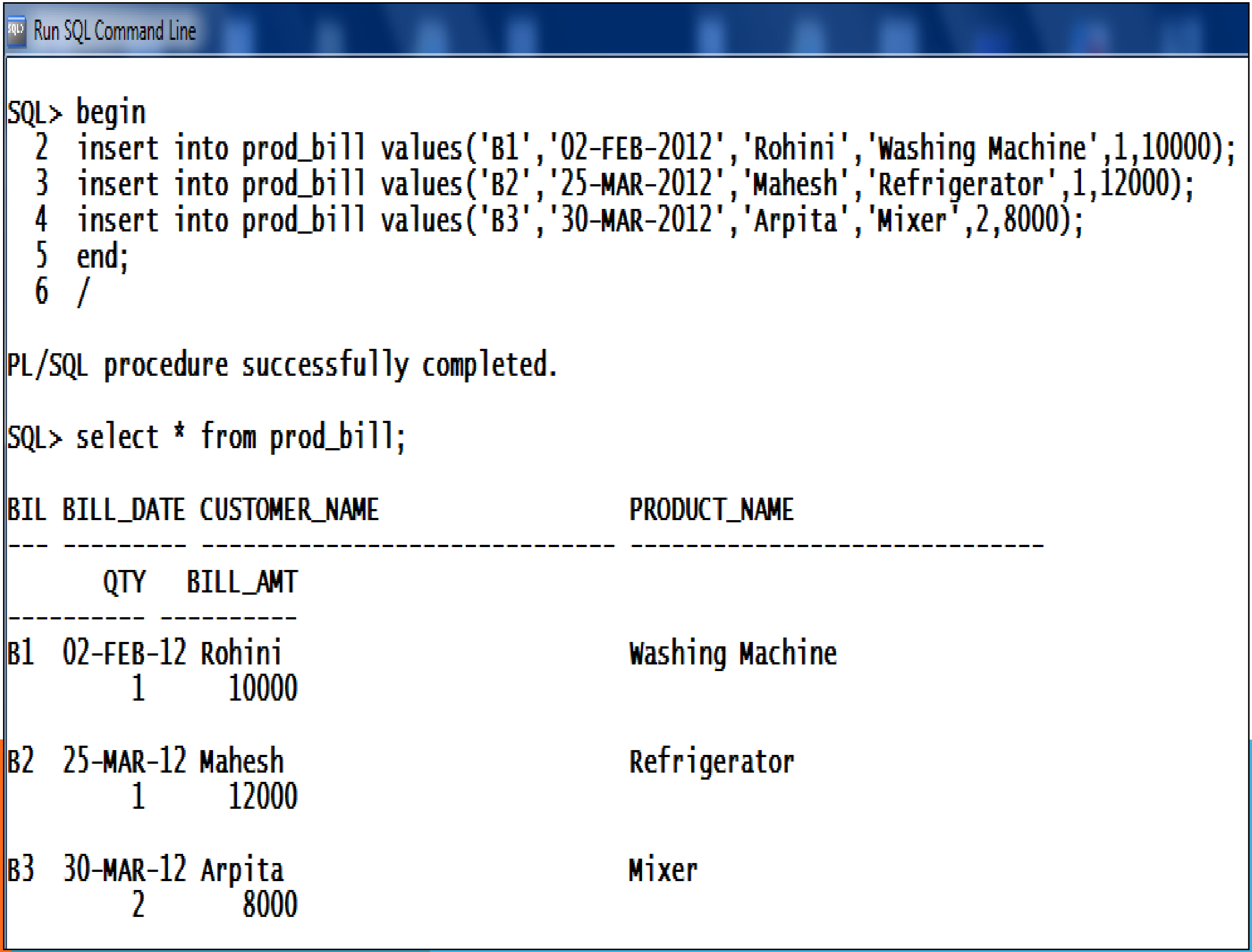
To create a P/L SQL Block which inserts the following 3 records into the table “Prod_bill”.

Record #1: B1,02-FEB-2012, Rohini,Washing Machine,1,10000

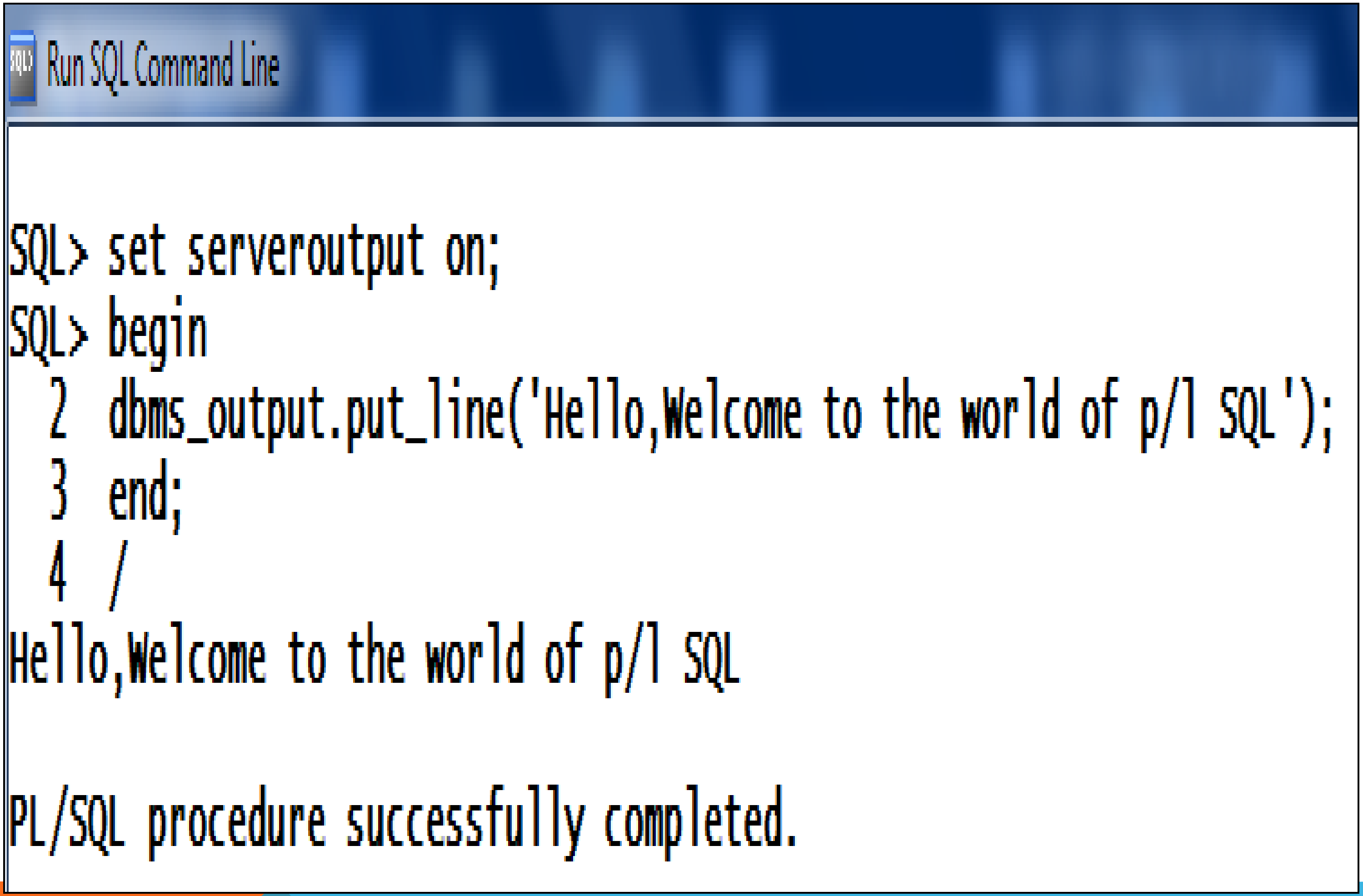
Record #2:B2,25-MAR-2012,Mahesh,Refrigerator,1,12000

Record #3:B3,30-MAR-2012,Arpita,Mixer,2,8000

```
BEGIN
INSERT INTO prod_bill VALUES(‘B1’, ‘02-FEB-
    2012’,’Rohini’,’Washing Machine,1,10000);
INSERT INTO prod_bill VALUES(’B2’, ‘25-MAR-
    2012’,’Mahesh’,’Refrigerator’,1,12000);
INSERT INTO prod_bill VALUES(’B3’,’ 30-MAR-
    2012’,’Arpita’,’Mixer’,2,8000);
END;
```



```
SQL> begin
  2  insert into prod_bill values('B1','02-FEB-2012','Rohini','Washing Machine',1,10000);
  3  insert into prod_bill values('B2','25-MAR-2012','Mahesh','Refrigerator',1,12000);
  4  insert into prod_bill values('B3','30-MAR-2012','Arpita','Mixer',2,8000);
  5  end;
  6  /

PL/SQL procedure successfully completed.

SQL> select * from prod_bill;

BIL BILL_DATE CUSTOMER_NAME                  PRODUCT_NAME
--- --------- ------------------------------ ------------------------------
       QTY   BILL_AMT
---------- ----------
B1  02-FEB-12 Rohini                         Washing Machine
         1      10000

B2  25-MAR-12 Mahesh                         Refrigerator
         1      12000

B3  30-MAR-12 Arpita                         Mixer
         2       8000
```



```
SQL> set serveroutput on;
SQL> begin
  2  dbms_output.put_line('Hello,Welcome to the world of p/l SQL');
  3  end;
  4  /
Hello,Welcome to the world of p/l SQL

PL/SQL procedure successfully completed.
```

GENERATE OUTPUT FROM A PL/SQL BLOCK:

DBMS_OUTPUT is a Package which helps in displaying information from our PL/SQL program on our screen.

Each user has a DBMS_OUTPUT buffer size of up to 1,000,000 bytes.

Information to this buffer can be written using DBMS_OUTPUT.PUT and DBMS_OUTPUT.PUT_LINE procedures. The buffer stores three different types of data -- VARCHAR2, NUMBER, and DATE .It does not support printing of BOOLEAN data types.

To Enable output in SQL*Plus

To enable displaying messages with DBMS_OUTPUT, The SET SERVEROUTPUT command is used. The following commands call the DBMS_OUTPUT.ENABLE procedure and hence enable printing information on the screen.

```
SET SERVEROUTPUT ON SIZE 1000000;
```

Or

```
SET SERVEROUTPUT ON;
```