CSE 4308
Database Management Systems Lab
Lab 10
Intro to PL/SQL
Anonymous Blocks, IF Statements, Functions

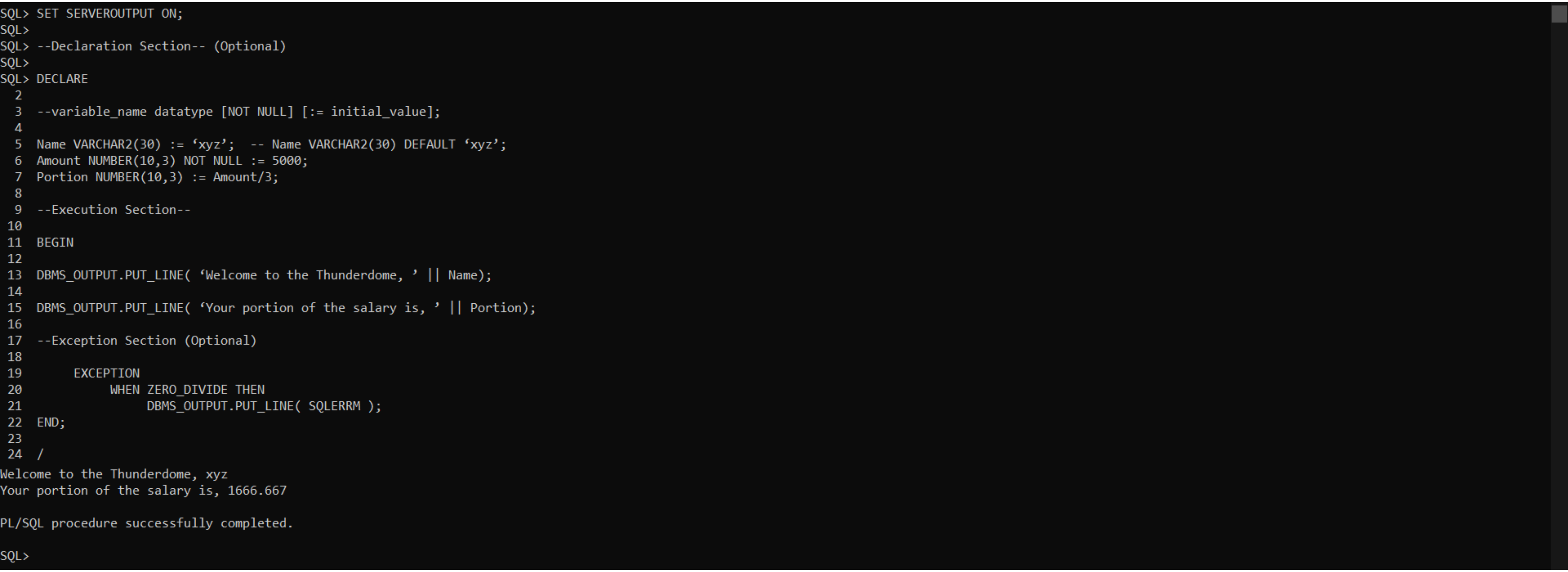
Department of Computer Science and Engineering
Islamic University of Technology, OIC

Mohammad Anas Jawad
Lecturer, IUT CSE

PL/SQL Anonymous Block & Syntax

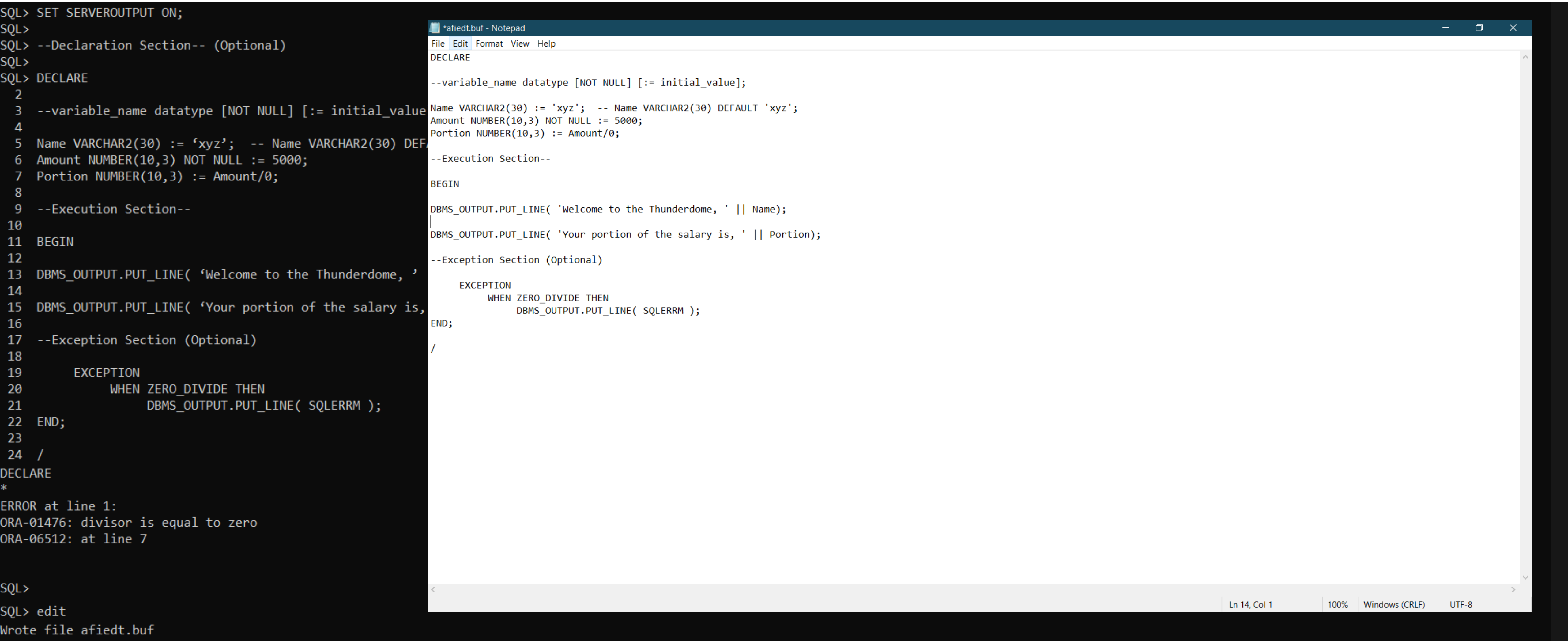
```
SET SERVEROUTPUT ON;

--Declaration Section-- (Optional)

DECLARE

--variable_name datatype [NOT NULL] [:= initial_value];

Name VARCHAR2(30) := ‘xyz’;
-- Name VARCHAR2(30) DEFAULT ‘xyz’;
Amount NUMBER(10,3) NOT NULL := 5000;
Portion NUMBER(10,3) := Amount/3;

--Execution Section--

BEGIN

DBMS_OUTPUT.PUT_LINE( ‘Welcome to the Thunderdome, ’ || Name);

DBMS_OUTPUT.PUT_LINE( ‘Your portion of the salary is, ’ || Portion);

--Exception Section (Optional)

    EXCEPTION
        WHEN ZERO_DIVIDE THEN
            DBMS_OUTPUT.PUT_LINE( SQLERRM );
END;
/
```

PL/SQL Anonymous Block Demonstration



```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> --Declaration Section-- (Optional)
SQL>
SQL> DECLARE

     --variable_name datatype [NOT NULL] [:= initial_value];

     Name VARCHAR2(30) := ‘xyz’;  -- Name VARCHAR2(30) DEFAULT ‘xyz’;
     Amount NUMBER(10,3) NOT NULL := 5000;
     Portion NUMBER(10,3) := Amount/3;

     --Execution Section--

     BEGIN

     DBMS_OUTPUT.PUT_LINE( ‘Welcome to the Thunderdome, ’ || Name);

     DBMS_OUTPUT.PUT_LINE( ‘Your portion of the salary is, ’ || Portion);

     --Exception Section (Optional)

          EXCEPTION
               WHEN ZERO_DIVIDE THEN
                    DBMS_OUTPUT.PUT_LINE( SQLERRM );
     END;

     /
Welcome to the Thunderdome, xyz
Your portion of the salary is, 1666.667

PL/SQL procedure successfully completed.

SQL>
```

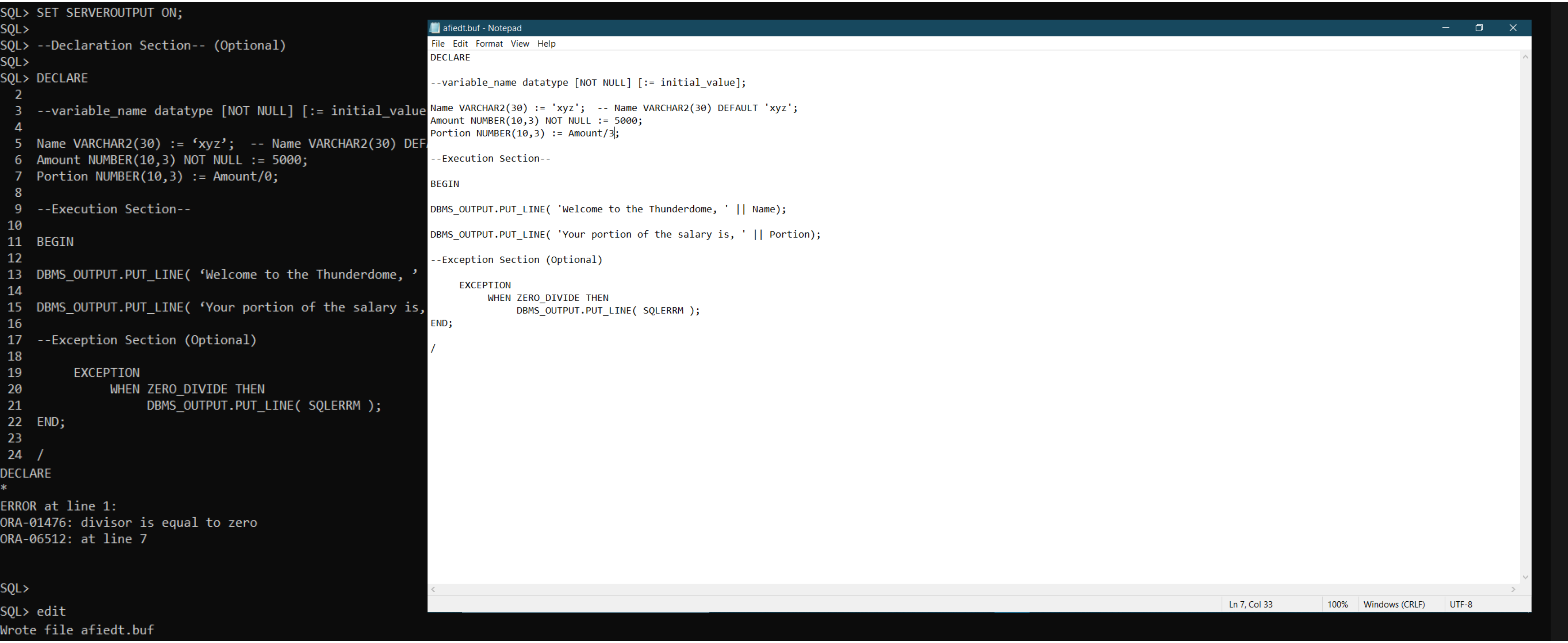
PL/SQL Anonymous Block Demonstration



```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> --Declaration Section-- (Optional)
SQL>
SQL> DECLARE
  2
  3  --variable_name datatype [NOT NULL] [:= initial_value
  4
  5  Name VARCHAR2(30) := ‘xyz’;  -- Name VARCHAR2(30) DEF
  6  Amount NUMBER(10,3) NOT NULL := 5000;
  7  Portion NUMBER(10,3) := Amount/0;
  8
  9  --Execution Section--
 10
 11  BEGIN
 12
 13  DBMS_OUTPUT.PUT_LINE( ‘Welcome to the Thunderdome, ’
 14
 15  DBMS_OUTPUT.PUT_LINE( ‘Your portion of the salary is,
 16
 17  --Exception Section (Optional)
 18
 19       EXCEPTION
 20            WHEN ZERO_DIVIDE THEN
 21                 DBMS_OUTPUT.PUT_LINE( SQLERRM );
 22  END;
 23
 24  /
DECLARE
*
ERROR at line 1:
ORA-01476: divisor is equal to zero
ORA-06512: at line 7


SQL>
SQL> edit
Wrote file afiedt.buf
```

```
DECLARE

--variable_name datatype [NOT NULL] [:= initial_value];

Name VARCHAR2(30) := 'xyz';  -- Name VARCHAR2(30) DEFAULT 'xyz';
Amount NUMBER(10,3) NOT NULL := 5000;
Portion NUMBER(10,3) := Amount/0;

--Execution Section--

BEGIN

DBMS_OUTPUT.PUT_LINE( 'Welcome to the Thunderdome, ' || Name);

DBMS_OUTPUT.PUT_LINE( 'Your portion of the salary is, ' || Portion);

--Exception Section (Optional)

     EXCEPTION
          WHEN ZERO_DIVIDE THEN
               DBMS_OUTPUT.PUT_LINE( SQLERRM );
END;

/
```

PL/SQL Anonymous Block Demonstration

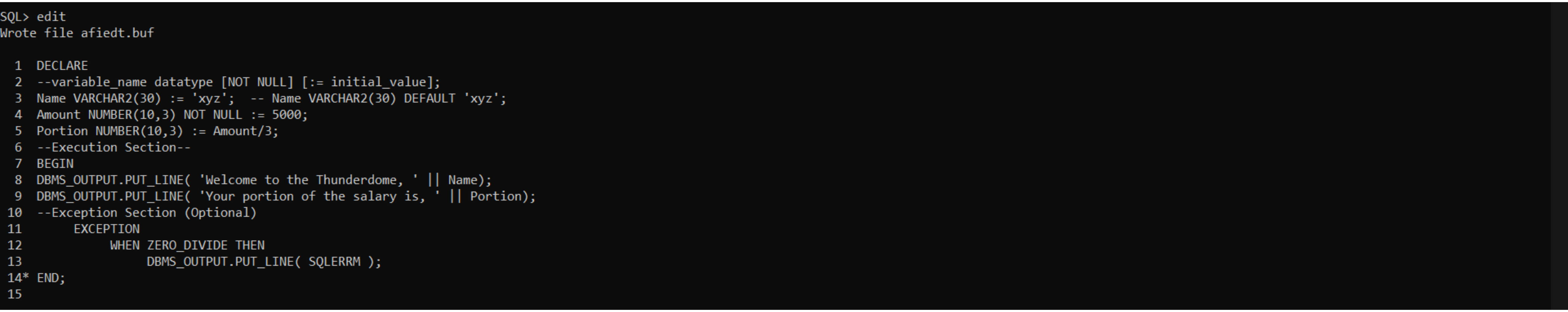


```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> --Declaration Section-- (Optional)
SQL>
SQL> DECLARE
  2
  3  --variable_name datatype [NOT NULL] [:= initial_value
  4
  5  Name VARCHAR2(30) := ‘xyz’;  -- Name VARCHAR2(30) DEF
  6  Amount NUMBER(10,3) NOT NULL := 5000;
  7  Portion NUMBER(10,3) := Amount/0;
  8
  9  --Execution Section--
 10
 11  BEGIN
 12
 13  DBMS_OUTPUT.PUT_LINE( ‘Welcome to the Thunderdome, ’
 14
 15  DBMS_OUTPUT.PUT_LINE( ‘Your portion of the salary is,
 16
 17  --Exception Section (Optional)
 18
 19       EXCEPTION
 20            WHEN ZERO_DIVIDE THEN
 21                 DBMS_OUTPUT.PUT_LINE( SQLERRM );
 22  END;
 23
 24  /
DECLARE
*
ERROR at line 1:
ORA-01476: divisor is equal to zero
ORA-06512: at line 7


SQL>
SQL> edit
Wrote file afiedt.buf
```

afiedt.buf - Notepad

```
DECLARE

--variable_name datatype [NOT NULL] [:= initial_value];

Name VARCHAR2(30) := 'xyz';  -- Name VARCHAR2(30) DEFAULT 'xyz';
Amount NUMBER(10,3) NOT NULL := 5000;
Portion NUMBER(10,3) := Amount/3;

--Execution Section--

BEGIN

DBMS_OUTPUT.PUT_LINE( 'Welcome to the Thunderdome, ' || Name);

DBMS_OUTPUT.PUT_LINE( 'Your portion of the salary is, ' || Portion);

--Exception Section (Optional)

    EXCEPTION
        WHEN ZERO_DIVIDE THEN
            DBMS_OUTPUT.PUT_LINE( SQLERRM );
END;

/
```

PL/SQL Anonymous Block Demonstration



```
SQL> edit
Wrote file afiedt.buf

DECLARE
--variable_name datatype [NOT NULL] [:= initial_value];
Name VARCHAR2(30) := 'xyz';  -- Name VARCHAR2(30) DEFAULT 'xyz';
Amount NUMBER(10,3) NOT NULL := 5000;
Portion NUMBER(10,3) := Amount/3;
--Execution Section--
BEGIN
DBMS_OUTPUT.PUT_LINE( 'Welcome to the Thunderdome, ' || Name);
DBMS_OUTPUT.PUT_LINE( 'Your portion of the salary is, ' || Portion);
--Exception Section (Optional)
     EXCEPTION
          WHEN ZERO_DIVIDE THEN
               DBMS_OUTPUT.PUT_LINE( SQLERRM );
END;
```

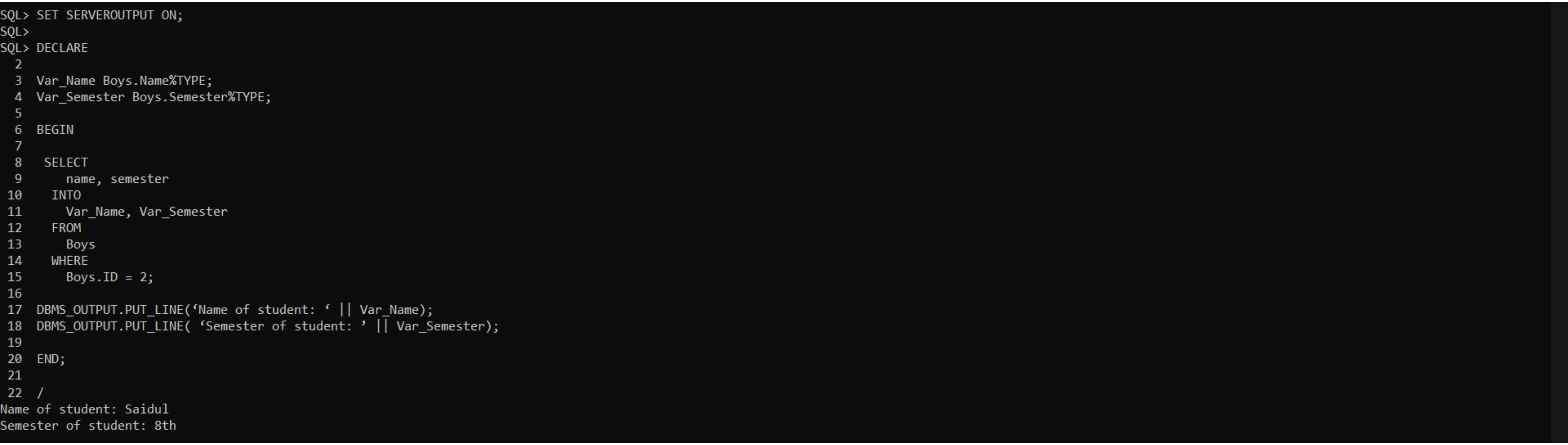
PL/SQL Anchored Declarations

```
DECLARE

--variable_name table_name.column_name%TYPE
Var_Name Boys.Name%TYPE;
Var_Semester Boys.Semester%TYPE;

BEGIN

 SELECT
    name, semester
  INTO
    Var_Name, Var_Semester
  FROM
    Boys
  WHERE
    Boys.ID = 2;

DBMS_OUTPUT.PUT_LINE('Name of student: ' || Var_Name);
DBMS_OUTPUT.PUT_LINE( 'Semester of student: ' || Var_Semester);

END;

/
```

Boys

ID	Name	Semester
1	Aflan	7th
2	Saidul	8th
3	Anas	8th

Anchored Declaration Demonstration



```
SQL> SET SERVEROUTPUT ON;
SQL>
SQL> DECLARE

Var_Name Boys.Name%TYPE;
Var_Semester Boys.Semester%TYPE;

BEGIN

 SELECT
    name, semester
  INTO
    Var_Name, Var_Semester
  FROM
    Boys
  WHERE
    Boys.ID = 2;

DBMS_OUTPUT.PUT_LINE(‘Name of student: ‘ || Var_Name);
DBMS_OUTPUT.PUT_LINE( ‘Semester of student: ’ || Var_Semester);

END;

/
Name of student: Saidul
Semester of student: 8th
```

PL/SQL IF Statements

```
--General Syntax for IF statements

IF condition1 THEN
    statement1;
ELSEIF condition2 THEN
    statement2;
.
.
ELSE
    else_statement;
END IF;
```

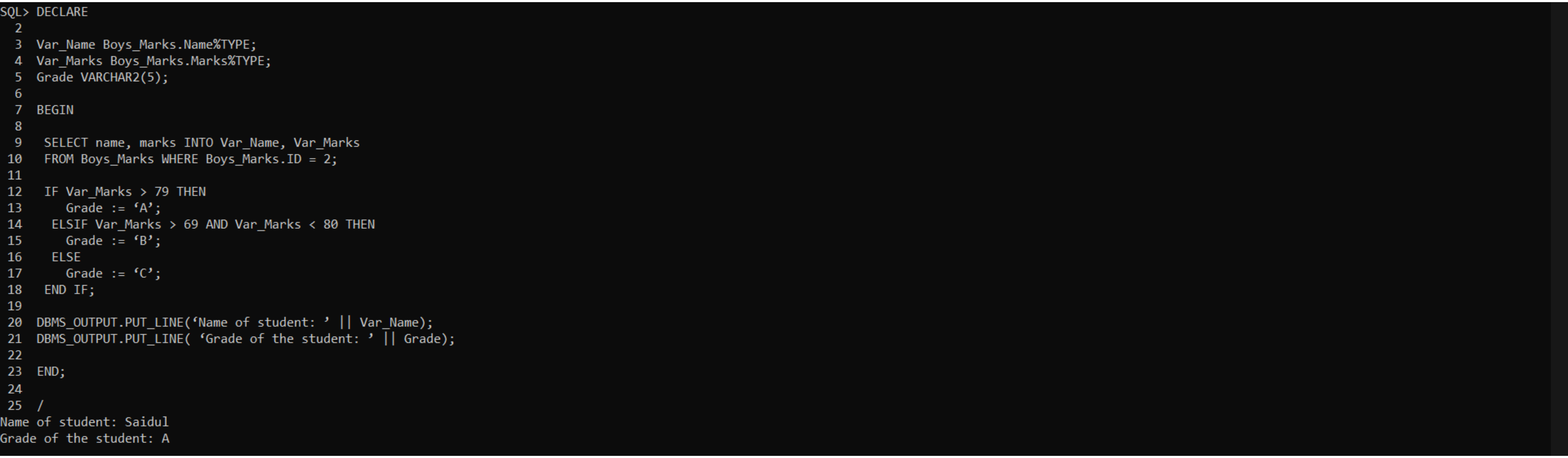
PL/SQL IF Statements

```
DECLARE

Var_Name Boys_Marks.Name%TYPE;
Var_Marks Boys_Marks.Marks%TYPE;
Grade VARCHAR2(5);

BEGIN

 SELECT name, marks INTO Var_Name, Var_Marks
 FROM Boys_Marks WHERE Boys_Marks.ID = 2;

 IF Var_Marks > 79 THEN
   Grade := ‘A’;
  ELSIF Var_Marks > 69 AND Var_Marks < 80 THEN
   Grade := ‘B’;
  ELSE
   Grade := ‘C’;
 END IF;

DBMS_OUTPUT.PUT_LINE(‘Name of student: ’ || Var_Name);
DBMS_OUTPUT.PUT_LINE( ‘Grade of the student: ’ || Grade);

END;

/
```

Boys_Marks

ID	Name	Marks
1	Aflan	70
2	Saidul	85
3	Anas	60

IF Statement Demonstration



```
SQL> DECLARE

Var_Name Boys_Marks.Name%TYPE;
Var_Marks Boys_Marks.Marks%TYPE;
Grade VARCHAR2(5);

BEGIN

 SELECT name, marks INTO Var_Name, Var_Marks
 FROM Boys_Marks WHERE Boys_Marks.ID = 2;

 IF Var_Marks > 79 THEN
    Grade := ‘A’;
  ELSIF Var_Marks > 69 AND Var_Marks < 80 THEN
    Grade := ‘B’;
  ELSE
    Grade := ‘C’;
 END IF;

DBMS_OUTPUT.PUT_LINE(‘Name of student: ’ || Var_Name);
DBMS_OUTPUT.PUT_LINE( ‘Grade of the student: ’ || Grade);

END;

/
Name of student: Saidul
Grade of the student: A
```

PL/SQL Functions

```
--General Syntax for writing functions

CREATE [OR REPLACE] FUNCTION function_name (parameter_list)
   RETURN return_type
IS

/*Declaration Section*/

BEGIN

/*Execution Section (Function body)*/

--Exception Section (Optional)

END

/

--General Syntax for deleting or dropping functions

DROP FUNCTION function_name;
```

PL/SQL Functions

```
CREATE OR REPLACE FUNCTION get_total_sales (in_date DATE)
RETURN NUMBER
IS

total_sales NUMBER(10,3) := 0;

BEGIN

   -- get total sales
   SELECT SUM(Quantity * PPU)
   INTO total_sales
   FROM Orders, Order_Items
   WHERE Orders.Order_ID = Order_Items.Order_ID AND Order_Date = in_date;

   -- return the total sales
   RETURN total_sales;

END;
/

--Calling the function from an anonymous block
BEGIN
DBMS_OUTPUT.PUT_LINE(get_total_sales(to_date('15_07_2020','dd__mm_yyyy')));
END;
/
--Calling the function in an SQL statement
SELECT get_total_sales(to_date('15_07_2020','dd__mm_yyyy')) FROM dual;
```

Orders

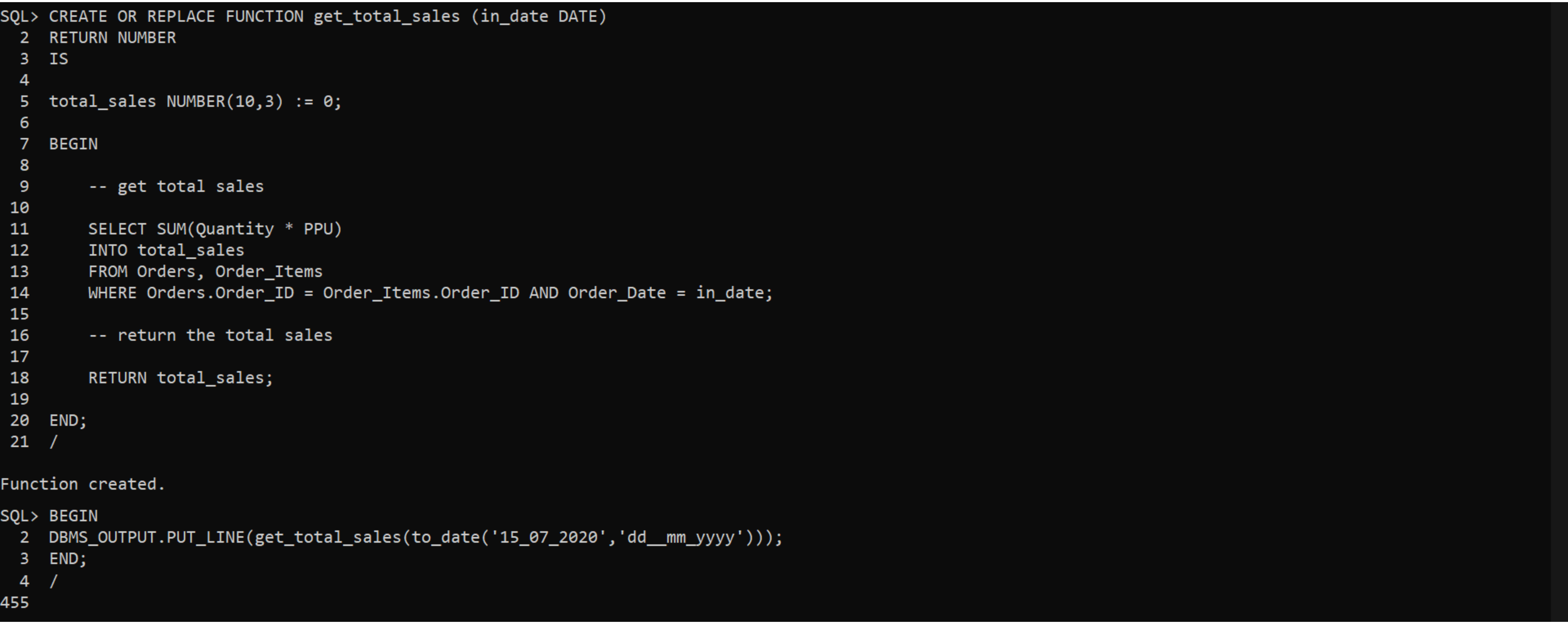
Order_ID	Client_ID	Order_Date
1	23	22-JUN-20
2	32	15-JUL-20
3	43	15-JUL-20

Order_Items

Order_ID	Name	Quantity	PPU
1	Bananas	6	9
2	Apples	10	5.5
2	Oranges	15	5
3	Coffee	50	6.5

<u>Obj:</u> Write a function that takes a date as parameter and returns the total money earned in sales on that date.
(10*5.5 + 15*5 + 50*6.5 = 455)

Function Demonstration



```
SQL> CREATE OR REPLACE FUNCTION get_total_sales (in_date DATE)
     RETURN NUMBER
     IS

     total_sales NUMBER(10,3) := 0;

     BEGIN

         -- get total sales

         SELECT SUM(Quantity * PPU)
         INTO total_sales
         FROM Orders, Order_Items
         WHERE Orders.Order_ID = Order_Items.Order_ID AND Order_Date = in_date;

         -- return the total sales

         RETURN total_sales;

     END;
     /

Function created.

SQL> BEGIN
     DBMS_OUTPUT.PUT_LINE(get_total_sales(to_date('15_07_2020','dd__mm_yyyy')));
     END;
     /
455
```

Thank You!