--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

--Name:Gozde Arslan

--ID: 150320190

--Date: 19/02/2021

--Purpose: Lab 5 DBS311

--\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

SET SERVEROUTPUT ON

-- Q1. Write a store procedure that get an integer number and prints The number is even.

-- If a number is divisible by 2. Otherwise, it prints The number is odd.

CREATE

OR replace PROCEDURE findevenodd(x IN number) IS

BEGIN

IF MOD(x, 2) = 0

THEN

dbms\_output.put\_line( ' The number is even ');

ELSE

dbms\_output.put\_line('The number is odd ');

END

IF;

END

;

/

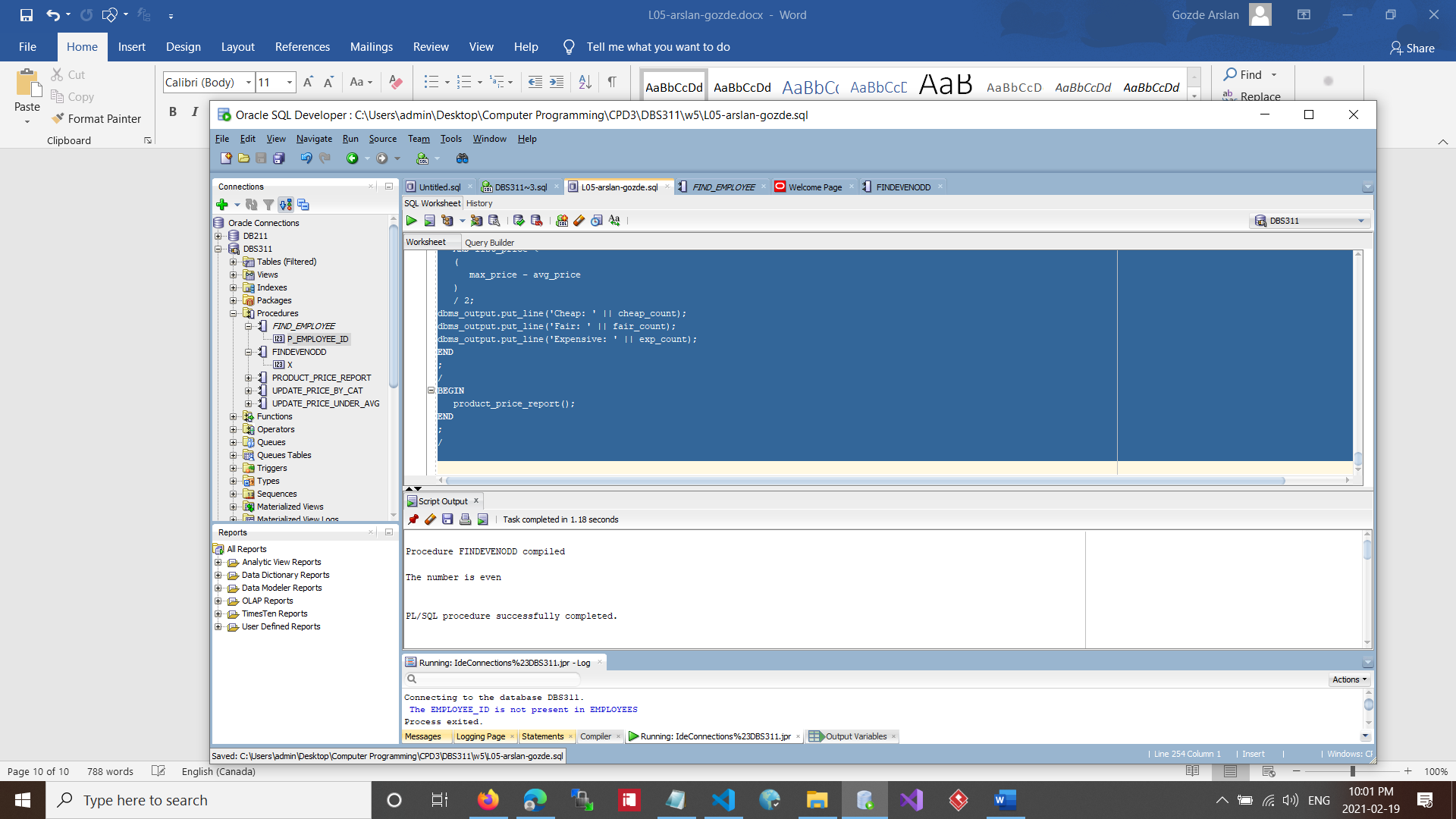
BEGIN

findevenodd(10) ;

END

;

/



-- Q2. Create a stored procedure named find\_employee. This procedure gets an employee number and prints the following

-- employee information: First name Last name Email Phone Hire date Job title

CREATE

OR replace PROCEDURE find\_employee(p\_employee\_id IN number) AS v\_count number;

v\_first\_name employees.first\_name % type;

v\_last\_name employees.last\_name % type;

v\_email employees.email % type;

v\_phone employees.phone % type;

v\_hire\_date employees.hire\_date % type;

v\_job\_title employees.job\_title % type;

BEGIN

SELECT

COUNT(employee\_id) INTO v\_count

FROM

employees

WHERE

employee\_id = p\_employee\_id;

IF v\_count = 0

THEN

dbms\_output.put\_line( ' The EMPLOYEE\_ID is not present in EMPLOYEES ');

ELSE

SELECT

first\_name,

last\_name,

email,

phone,

hire\_date,

job\_title INTO v\_first\_name,

v\_last\_name,

v\_email,

v\_phone,

v\_hire\_date,

v\_job\_title

FROM

employees

WHERE

employee\_id = p\_employee\_id;

dbms\_output.put\_line('FIRST\_NAME : ' || v\_first\_name);

dbms\_output.put\_line('LAST\_NAMENAME: ' || v\_last\_name);

dbms\_output.put\_line('EMAIL: ' || v\_email);

dbms\_output.put\_line('PHONE: ' || v\_phone);

dbms\_output.put\_line('HIRE\_DATE: ' || v\_hire\_date);

dbms\_output.put\_line('JOB\_TITLE: ' || v\_job\_title);

END

IF;

exception

WHEN

no\_data\_found

THEN

dbms\_output.put\_line('Employee not found.');

WHEN

others

THEN

dbms\_output.put\_line('Stored PROCEDURE has errors. Please take a look');

END

;

---------------------

/

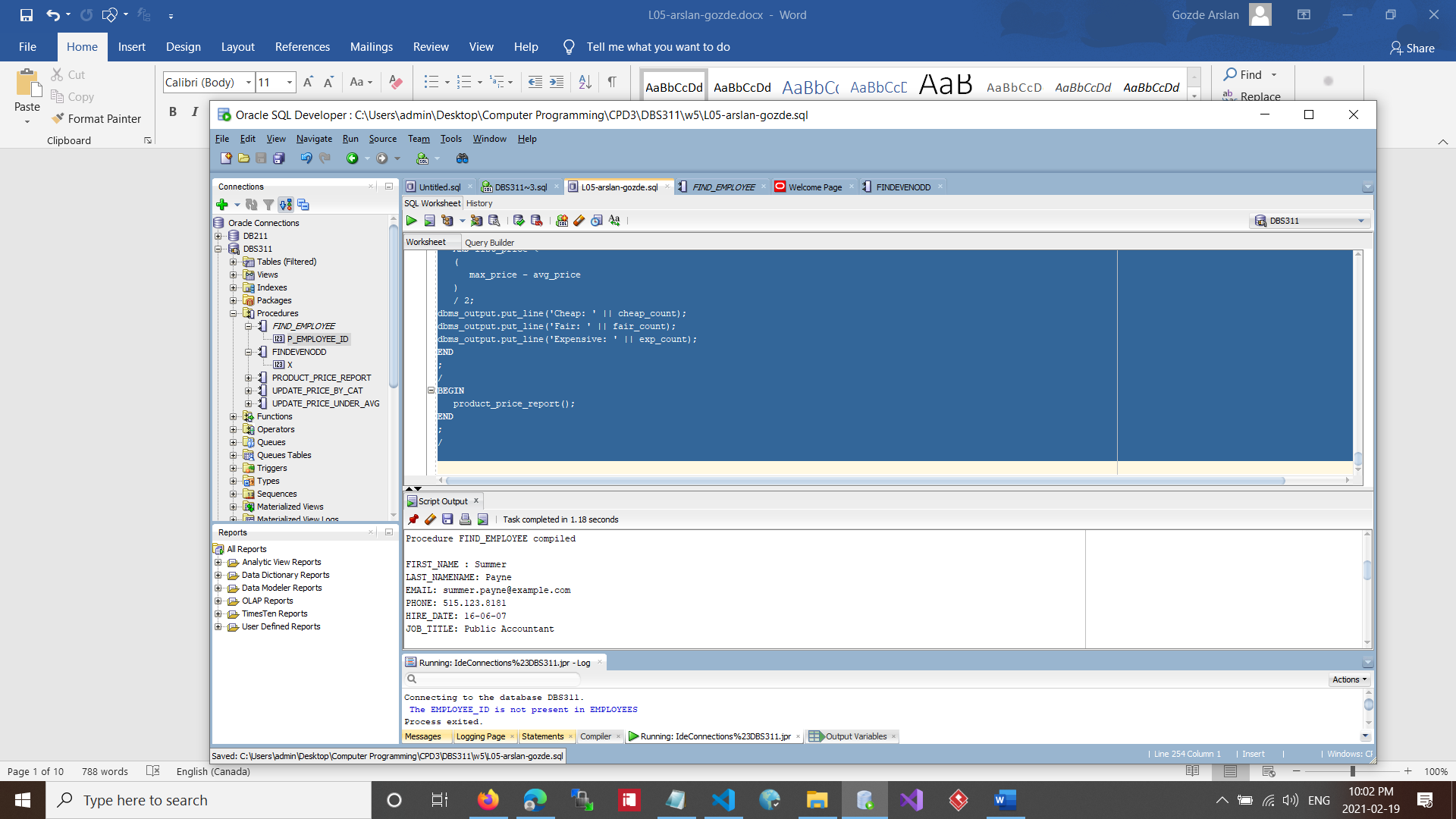
BEGIN

find\_employee(107);

END

;

/



-- Q3.Every year, the company increasesthe price of all products in one category. For example,

--the company wants to increase the price(list\_price)of products in category 1 by $5.

--Write a procedure named update\_price\_by\_catto update the price of all products ina given category and the given amount to be added to the current priceif the price is greater than 0.

--The procedure shows the number of updated rows if the update is successful--

CREATE

OR replace PROCEDURE update\_price\_by\_cat(p\_category\_id IN products.category\_id % type, p\_amount IN products.list\_price % type) AS v\_count number;

BEGIN

SELECT

COUNT(category\_id) INTO v\_count

FROM

products

WHERE

category\_id = p\_category\_id;

IF (p\_amount > 0

AND v\_count > 0)

THEN

UPDATE

products

SET

list\_price = list\_price + p\_amount

WHERE

category\_id = p\_category\_id;

dbms\_output.put\_line('Rows Updated =' || SQL % rowcount);

ELSE

dbms\_output.put\_line('Either there are no CATEGORY matching or the input price is lesser than zero');

END

IF;

exception

WHEN

no\_data\_found

THEN

dbms\_output.put\_line('PRODUCTS not found.');

WHEN

others

THEN

dbms\_output.put\_line('Stored PROCEDURE has errors.');

END

;

----------

DECLARE p\_category\_id number : = 1;

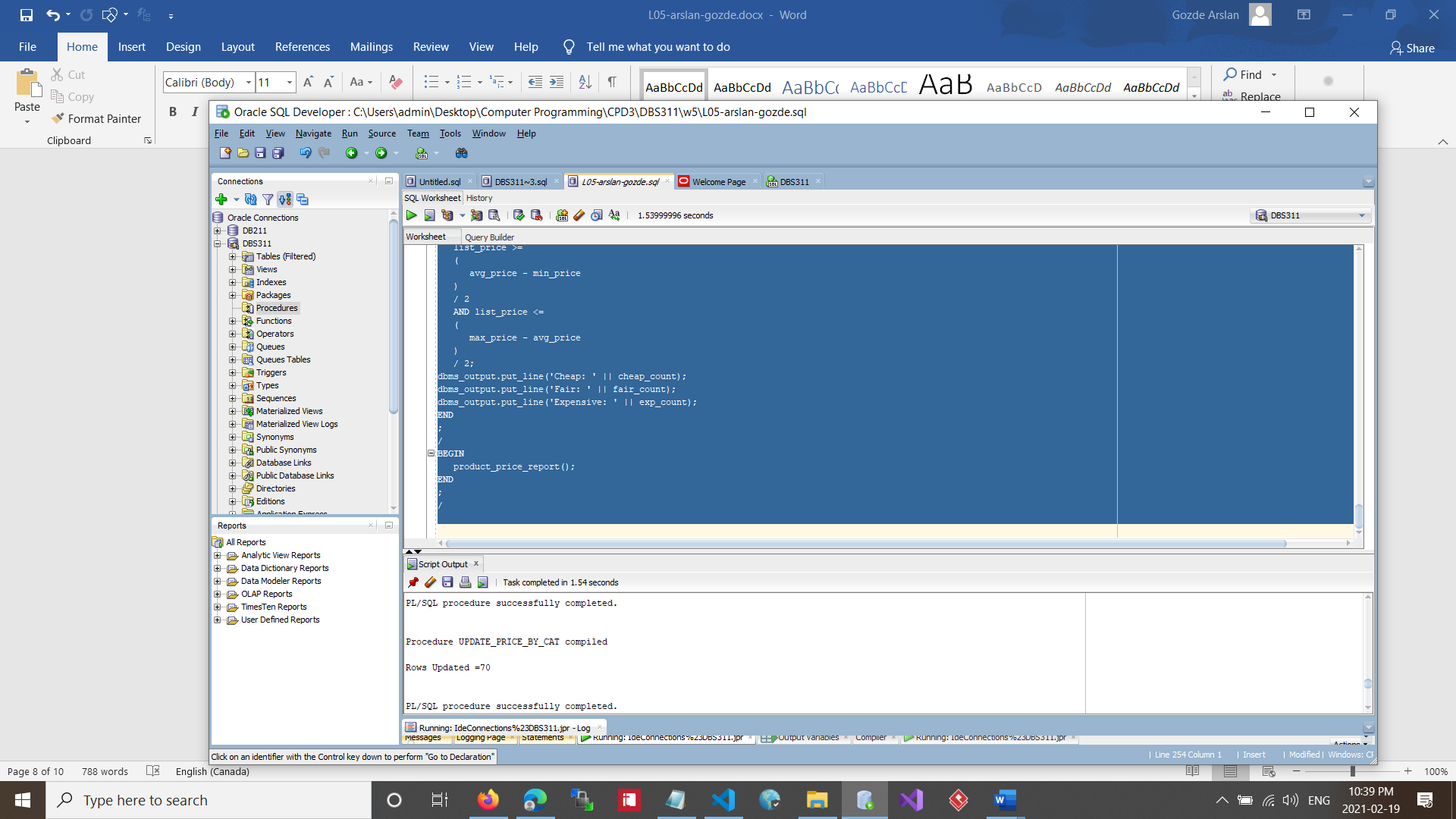
p\_amount number : = 11.11;

BEGIN

update\_price\_by\_cat(p\_category\_id , p\_amount );

END

;



-- Q4. Every year, the company increase the price of products whose price is less than the average price of all products by 1%.

--(list\_price \* 1.01). Write a stored procedure named update\_price\_under\_avg. This procedure do not have any parameters.

--You need to find the average price of all products and store it into a variable of the same type.

--If the average price is less than or equal to $1000, update products’ priceby 2% if the price of the product is less than the calculated average.

--If the average price is greater than $1000, update products’ price by 1% if the price of the product is less than the calculatedaverage.

--The query displays an error message if any error occurs. Otherwise, it displays the number of updated rows.--

CREATE

or replace PROCEDURE update\_price\_under\_avg AS

v\_avg products.list\_price%TYPE;

v\_rate number;

BEGIN

select avg(LIST\_PRICE) into v\_avg from products ;

if v\_avg >= 1000 THEN

v\_rate := 1.02;

ELSE

v\_rate :=1.01;

end if;

update products set LIST\_PRICE = LIST\_PRICE \* v\_rate where LIST\_PRICE <= v\_avg;

DBMS\_OUTPUT.PUT\_LINE('Rows Updated =' || SQL%ROWCOUNT);

EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('PRODUCTS not found.');

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Stored PROCEDURE has errors. Please take a look');

end;

--------------

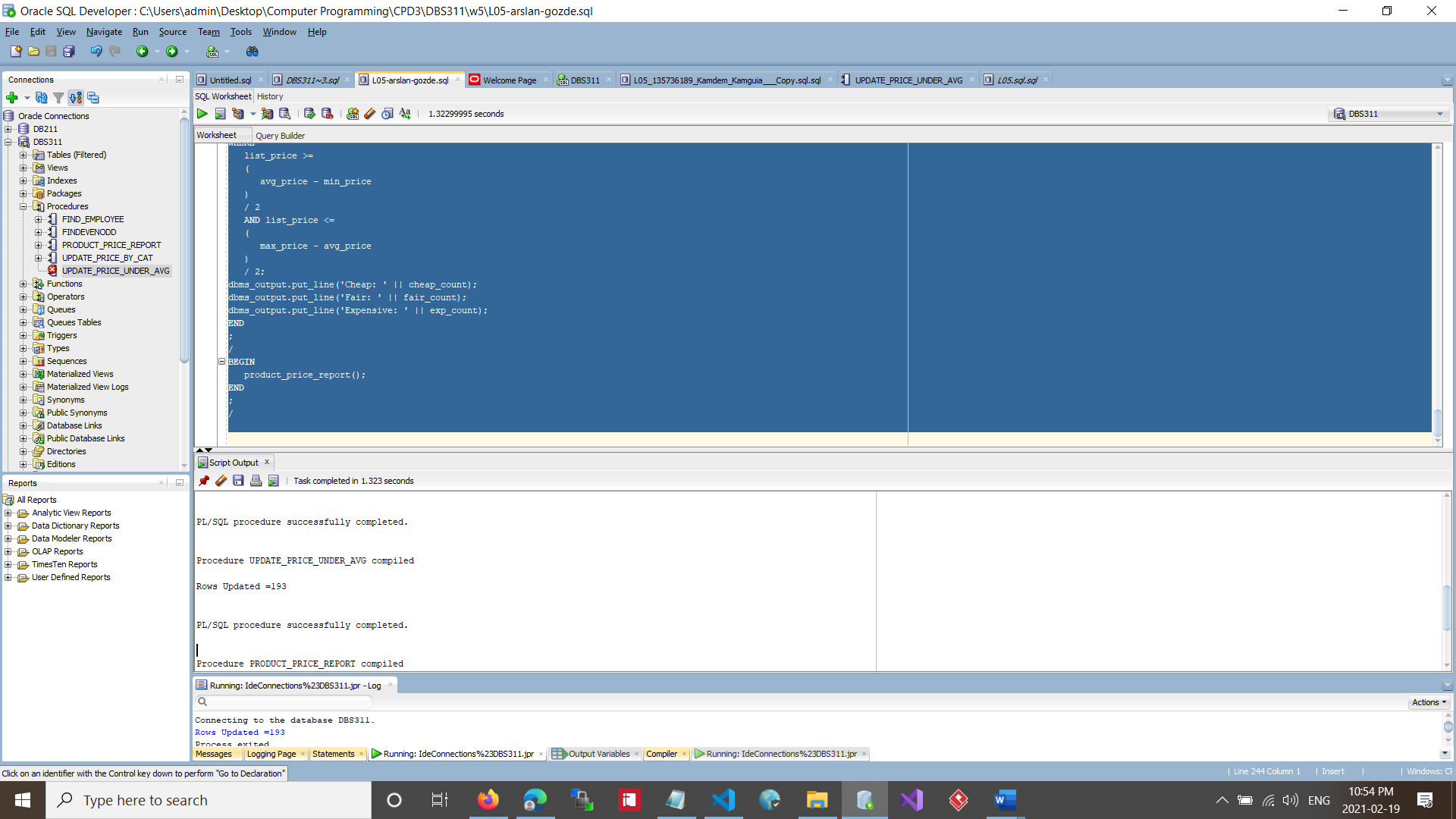
/

BEGIN

update\_price\_under\_avg;

END;

/



-- Q5. The company needs a report that shows three category of products basedtheir prices. The company needsto know if the product price is cheap, fair, or expensive.

CREATE

OR replace PROCEDURE product\_price\_report AS avg\_price number;

min\_price number;

max\_price number;

cheap\_count number;

fair\_count number;

exp\_count number;

BEGIN

SELECT

AVG(list\_price),

MAX(list\_price),

MIN(list\_price) INTO avg\_price,

max\_price,

min\_price

FROM

products;

SELECT

COUNT(list\_price) INTO cheap\_count

FROM

products

WHERE

list\_price < (avg\_price - min\_price) / 2;

SELECT

COUNT(list\_price) INTO exp\_count

FROM

products

WHERE

list\_price > (max\_price - avg\_price) / 2;

SELECT

COUNT(list\_price) INTO fair\_count

FROM

products

WHERE

list\_price >=

(

avg\_price - min\_price

)

/ 2

AND list\_price <=

(

max\_price - avg\_price

)

/ 2;

dbms\_output.put\_line('Cheap: ' || cheap\_count);

dbms\_output.put\_line('Fair: ' || fair\_count);

dbms\_output.put\_line('Expensive: ' || exp\_count);

END

;

/

BEGIN

product\_price\_report();

END

;

/

