**PL SQL EXERCISES**

Exercise 1 – Control Structures

Control structures in PL/SQL are programming constructs that control the flow of execution in a PL/SQL block. They allow you to:

* Make decisions
* Perform repetitive tasks
* Control the sequence of execution

1. If Statement

DECLARE

emp\_salary NUMBER := 60000;

BEGIN

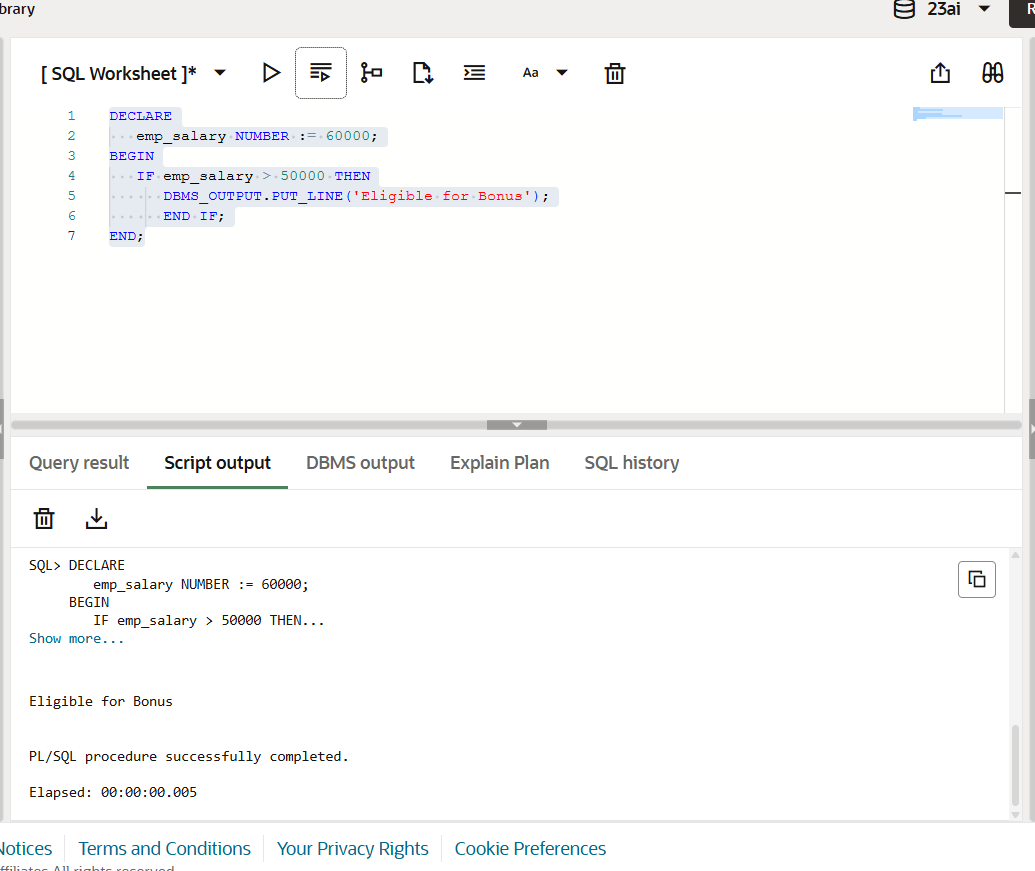
IF emp\_salary > 50000 THEN

DBMS\_OUTPUT.PUT\_LINE('Eligible for Bonus');

   END IF;

END;

Output:



1. If Else DECLARE

student\_score NUMBER := 45;

BEGIN

IF student\_score >= 50 THEN

DBMS\_OUTPUT.PUT\_LINE('Result: Passed');

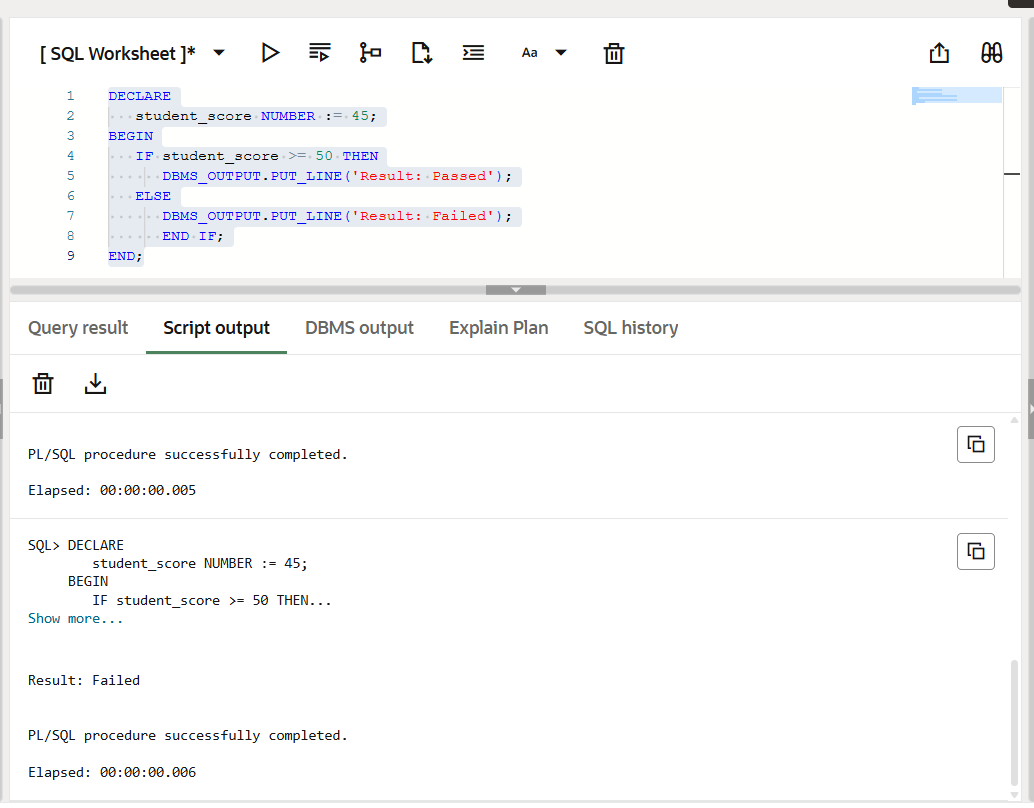
ELSE

DBMS\_OUTPUT.PUT\_LINE('Result: Failed');

   END IF;

END;

Output:



1. IF...ELSIF...ELSE Statement DECLARE

grade CHAR := 'B';

BEGIN

IF grade = 'A' THEN

DBMS\_OUTPUT.PUT\_LINE('Excellent');

ELSIF grade = 'B' THEN

DBMS\_OUTPUT.PUT\_LINE('Good');

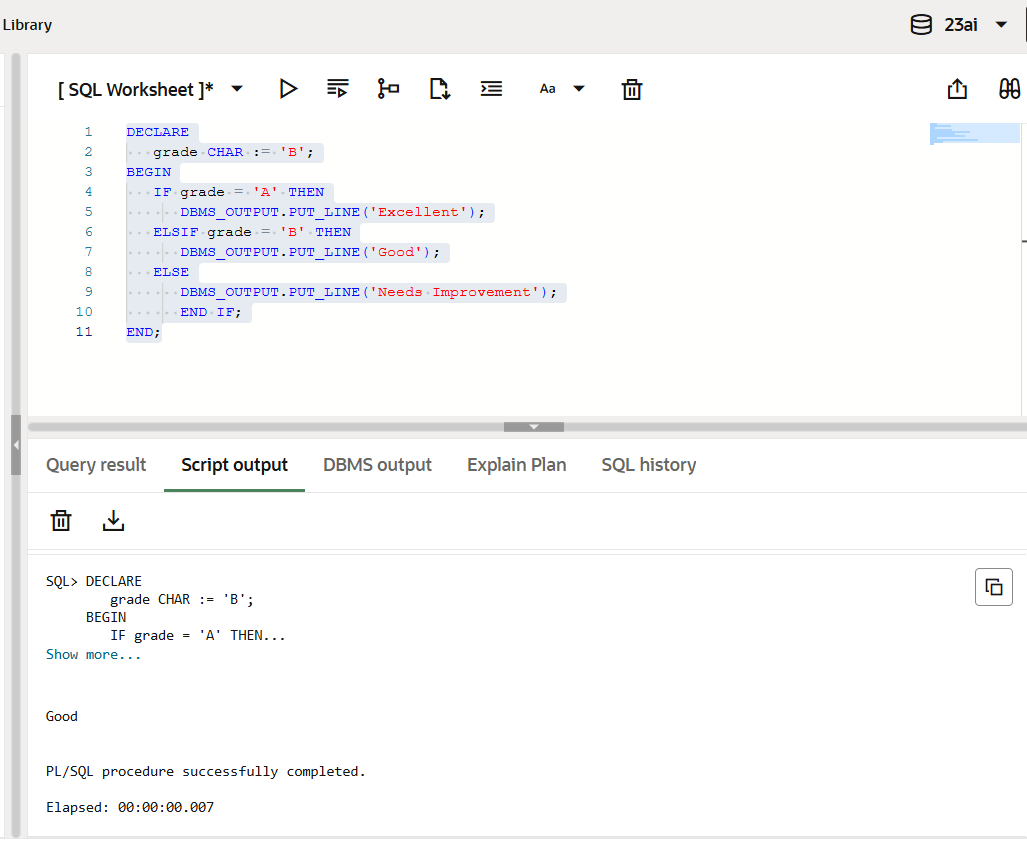
ELSE

DBMS\_OUTPUT.PUT\_LINE('Needs Improvement');

   END IF;

END;

Output:



1. For Loop Statement DECLARE

i NUMBER := 1;

BEGIN

LOOP

DBMS\_OUTPUT.PUT\_LINE('Count: ' || i);

i := i + 1;

EXIT WHEN i > 5;

  END LOOP;

END;

Output:

1. While Loop Statement DECLARE

i NUMBER := 2;

BEGIN

WHILE i <= 10 LOOP

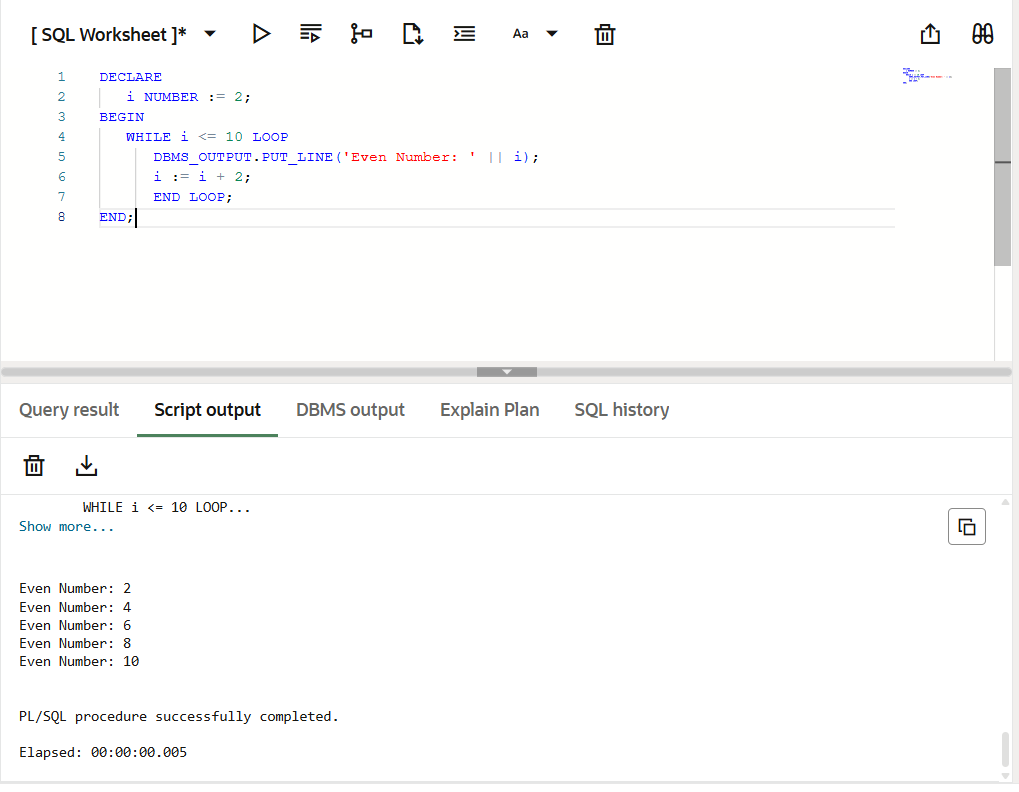
DBMS\_OUTPUT.PUT\_LINE('Even Number: ' || i);

i := i + 2;

  END LOOP;

END;

Output:



1. Nested Loops BEGIN

FOR i IN 1..3 LOOP

FOR j IN 1..3 LOOP

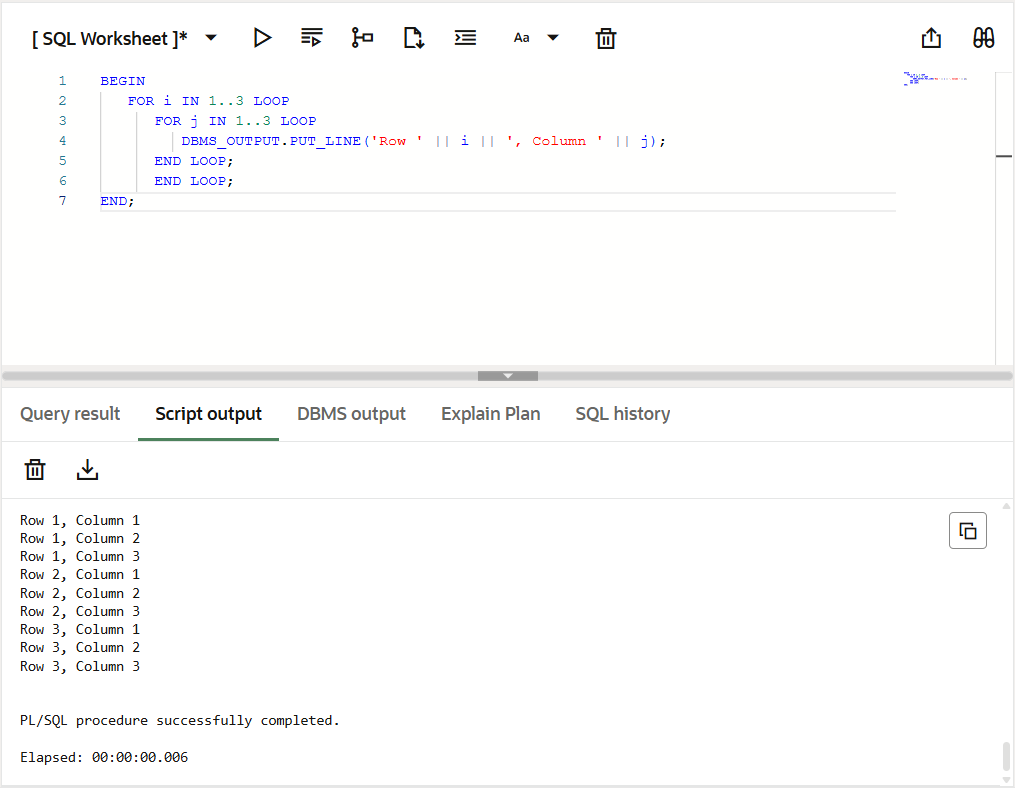
DBMS\_OUTPUT.PUT\_LINE('Row ' || i || ', Column ' || j);

END LOOP;

  END LOOP;

END;

Output:



**STORED PROCEDURES**

A stored procedure is a named block of PL/SQL code that performs a specific task and is stored in the database. It can be executed (called) multiple times with different parameters.

**Key Features of Stored Procedures:**

* Encapsulate business logic
* Improve code reusability
* Increase performance by reducing network traffic
* Accept input, return output, or just perform actions
* Can be called from applications, triggers, or other procedures

Syntax:

CREATE OR REPLACE PROCEDURE procedure\_name (

parameter1 IN datatype,

parameter2 OUT datatype

) IS

BEGIN

-- statements

END;

1. Stored Procedure without Parameters

CREATE OR REPLACE PROCEDURE welcome\_message IS

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Welcome to CTS Assessment!');

END;

/

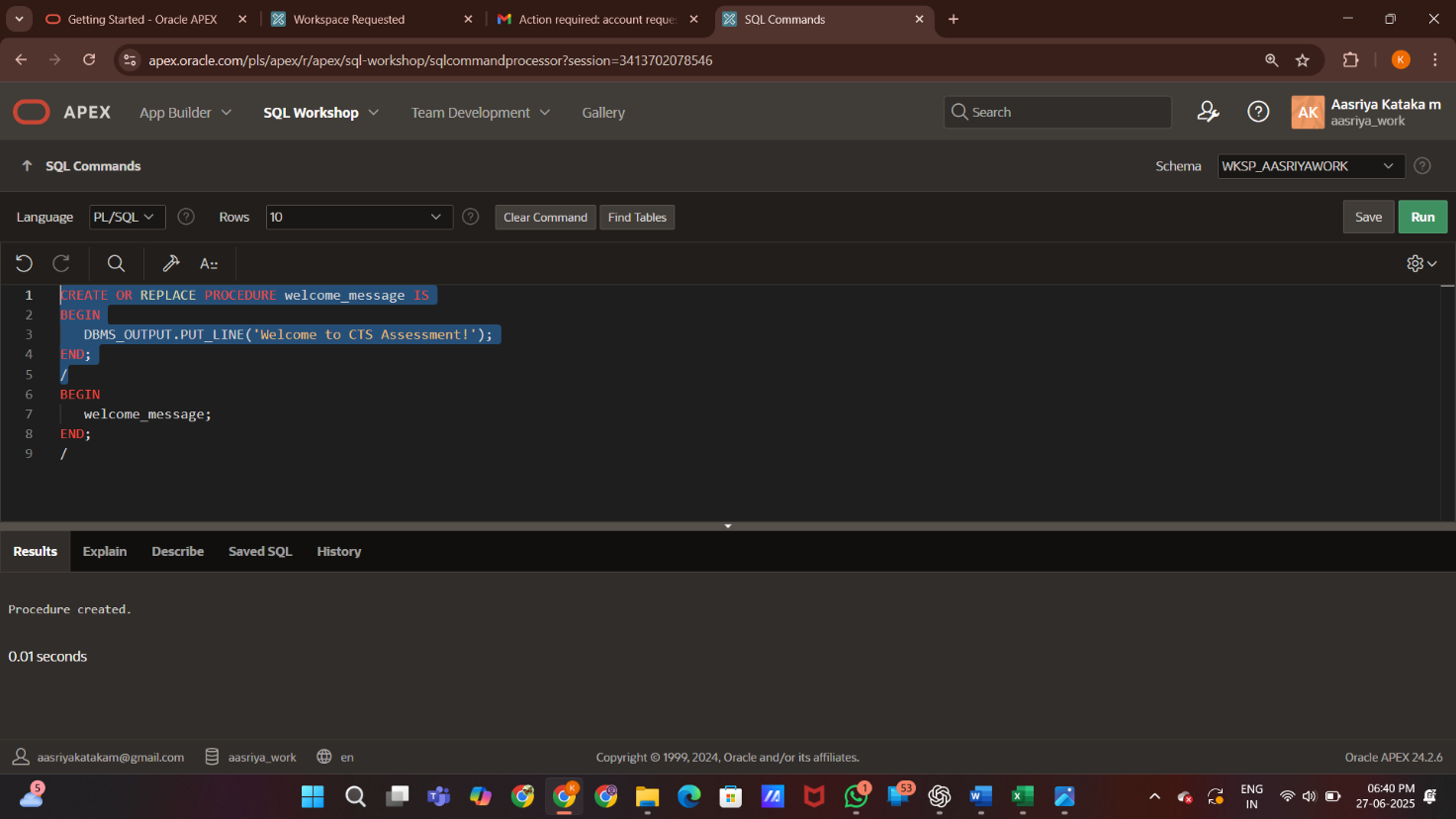
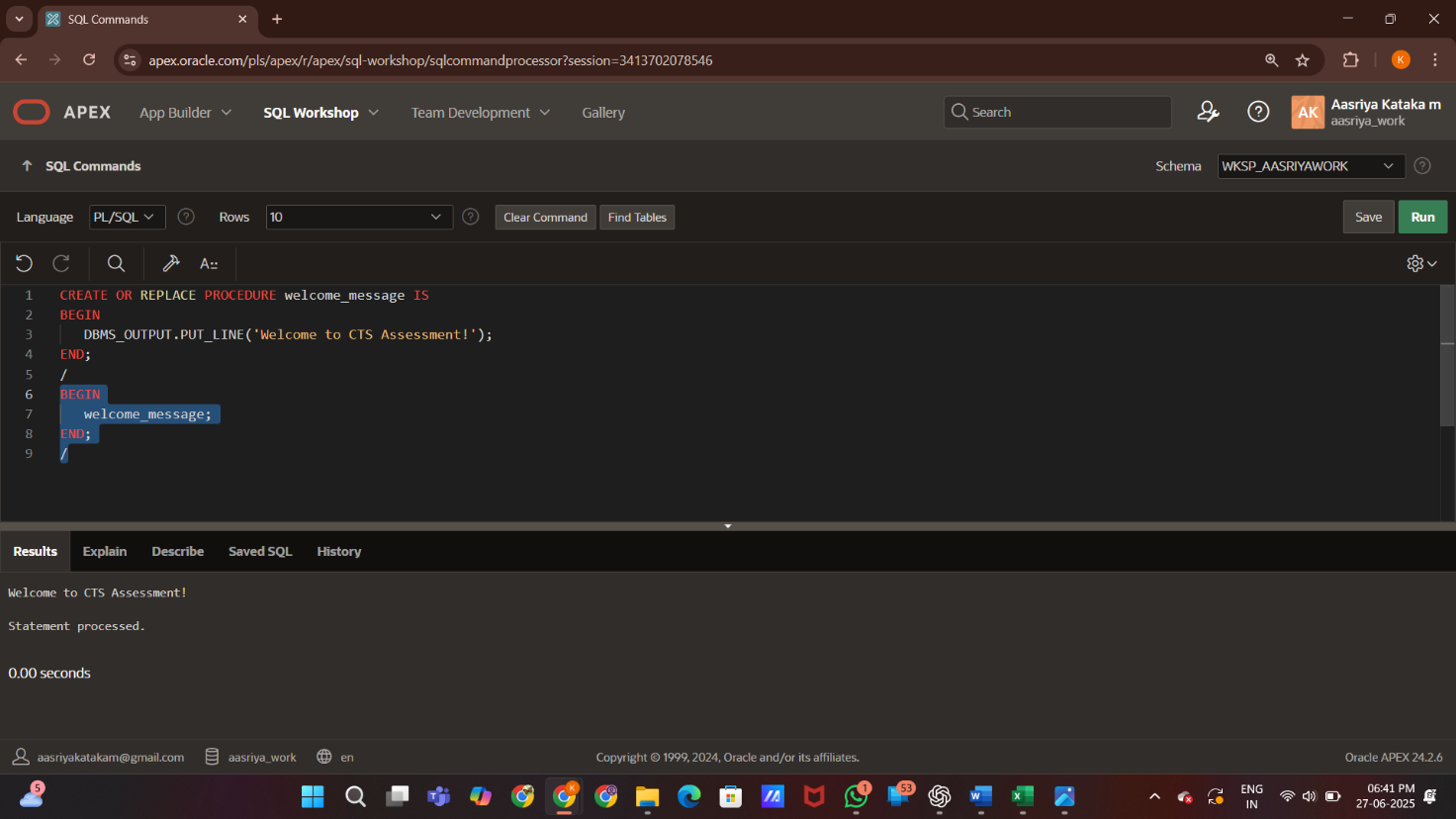
BEGIN

welcome\_message;

END;

/

Output:



1. Stored Procedure with IN Parameter

CREATE OR REPLACE PROCEDURE greet\_user(p\_name IN VARCHAR2) IS

BEGIN

DBMS\_OUTPUT.PUT\_LINE('Hello, ' || p\_name || '! Welcome!');

END;

/

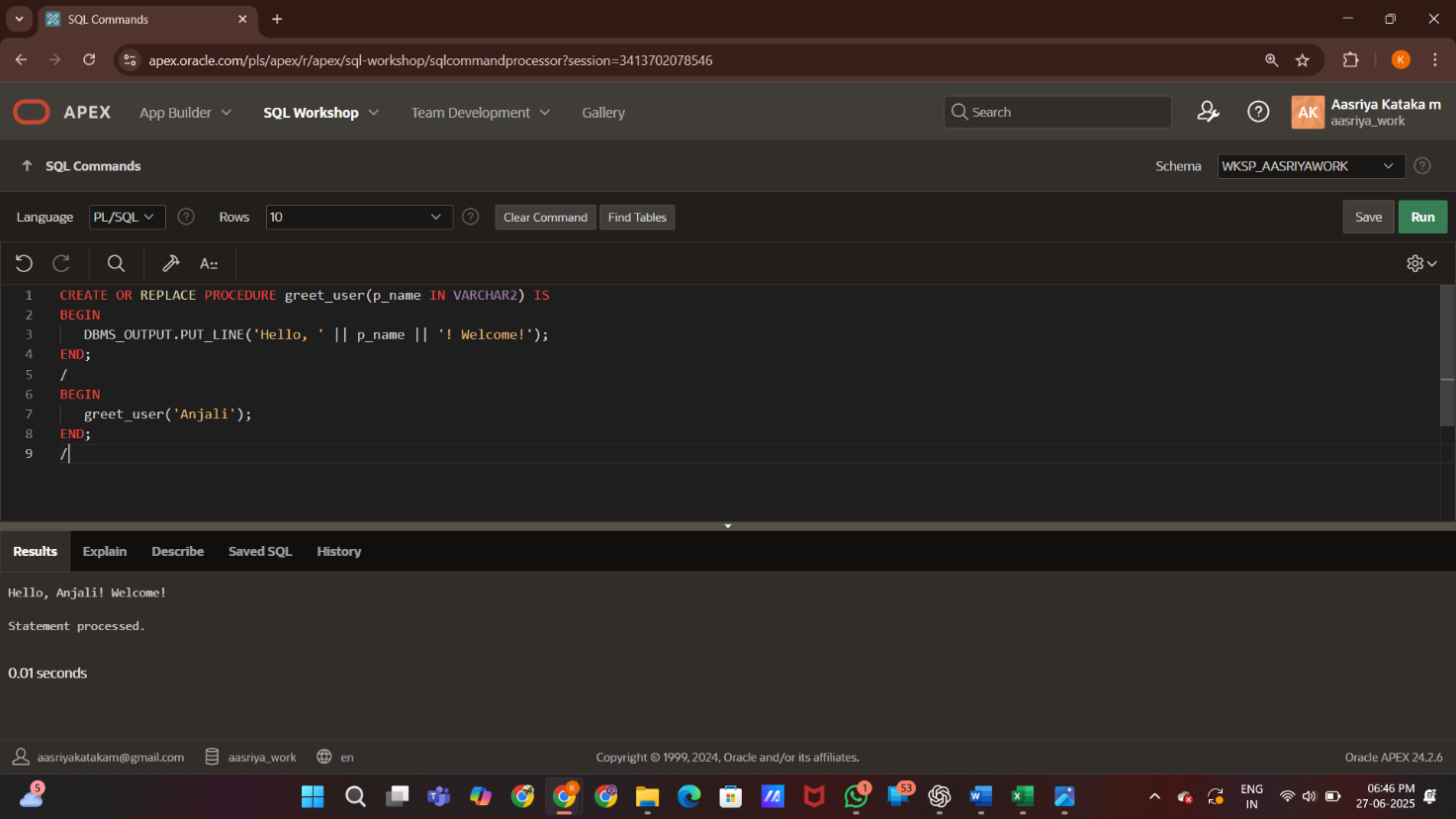
BEGIN

greet\_user('Anjali');

END;

/

Output:

1. Stored procedure with OUT parameter

CREATE OR REPLACE PROCEDURE square\_number(

p\_input IN NUMBER,

p\_result OUT NUMBER

) IS

BEGIN

p\_result := p\_input \* p\_input;

END;

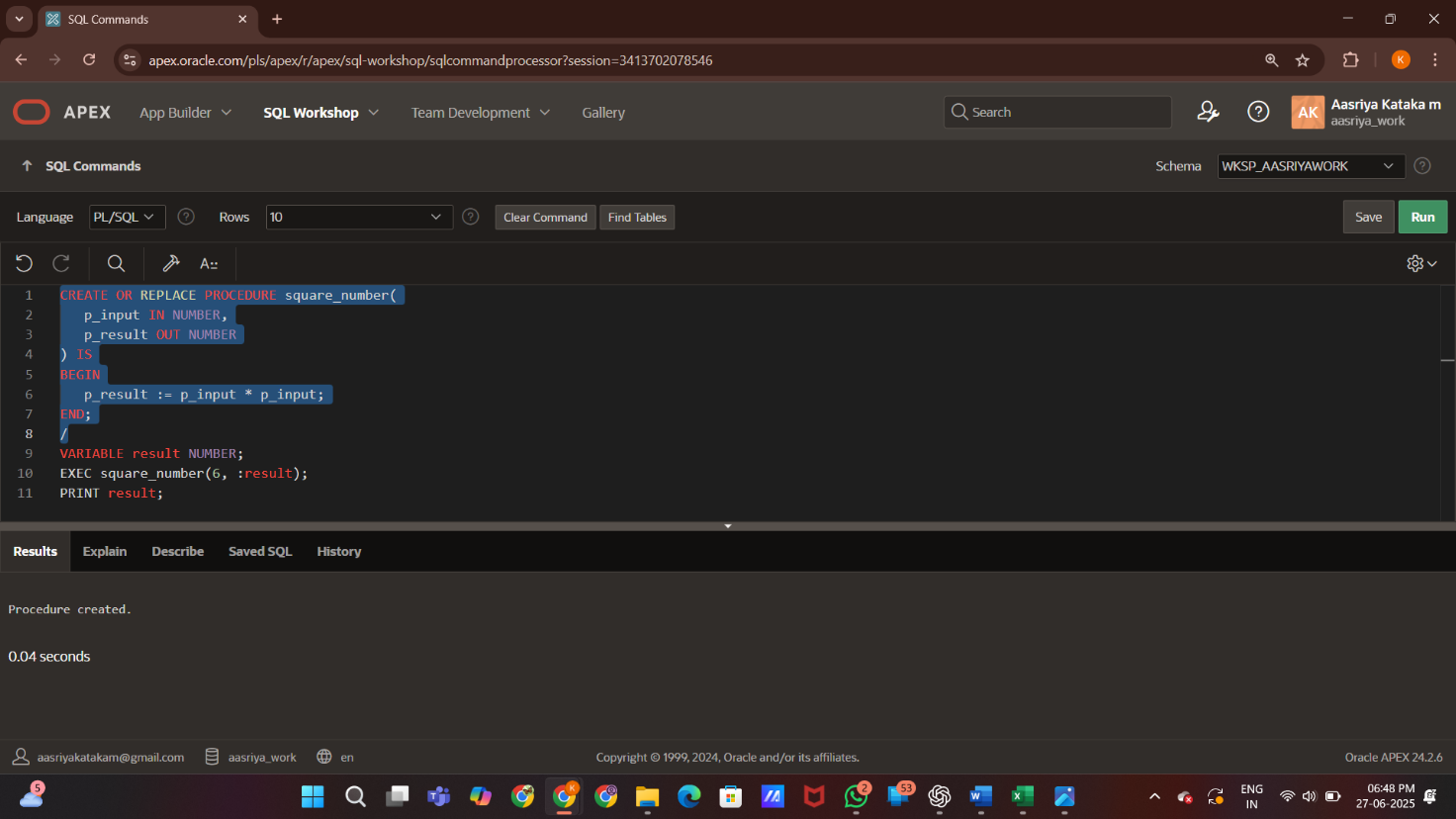
/

VARIABLE result NUMBER;

EXEC square\_number(6, :result);

PRINT result;

Output:

1. Procedure to Insert into a Table

CREATE TABLE students (

student\_id NUMBER,

student\_name VARCHAR2(50)

);

CREATE OR REPLACE PROCEDURE add\_student(p\_id IN NUMBER, p\_name IN VARCHAR2) IS

BEGIN

INSERT INTO students (student\_id, student\_name)

VALUES (p\_id, p\_name);

COMMIT;

END;

/

BEGIN

add\_student(1, 'Aarav');

add\_student(2,'Aarya');

add\_student(3,'Sashi');

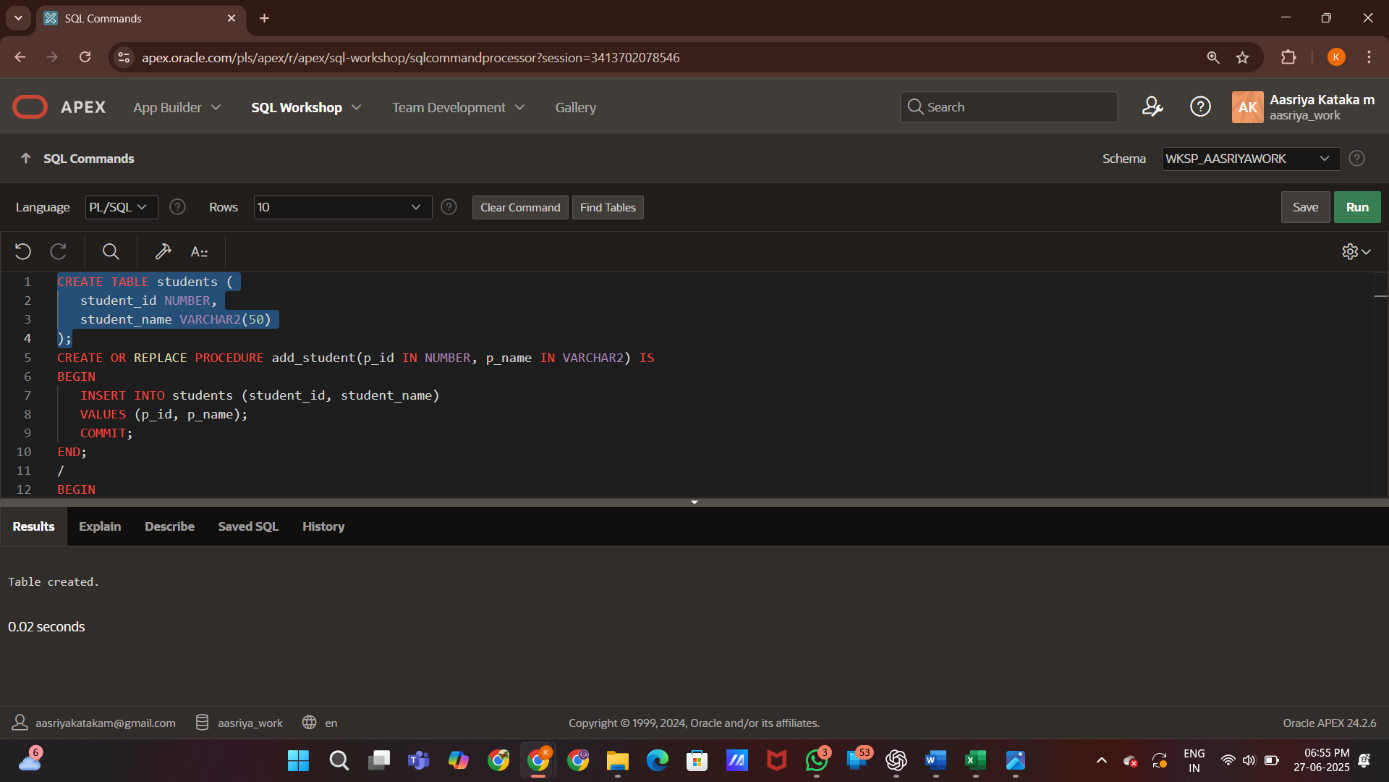
add\_student(4,'Bunny');

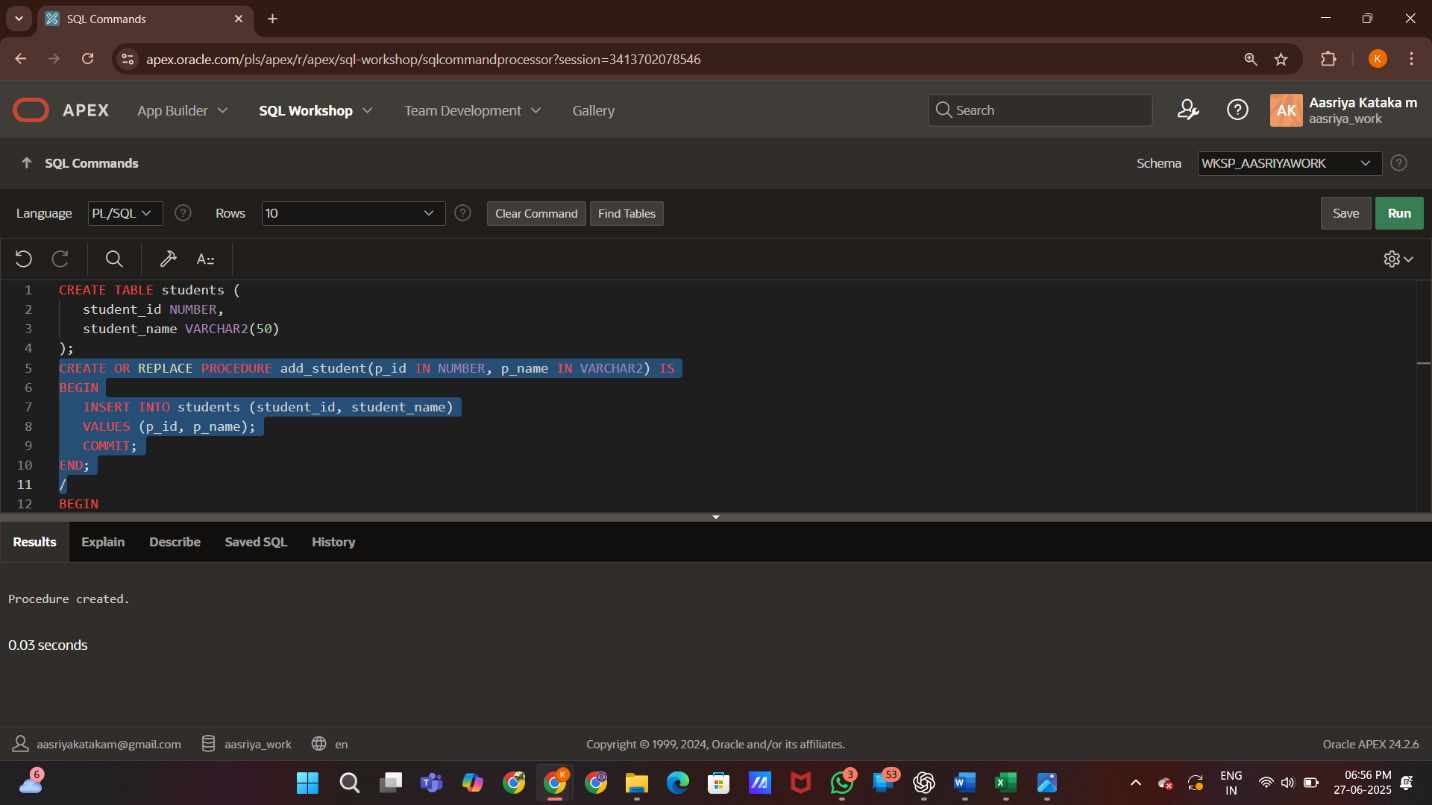
END;

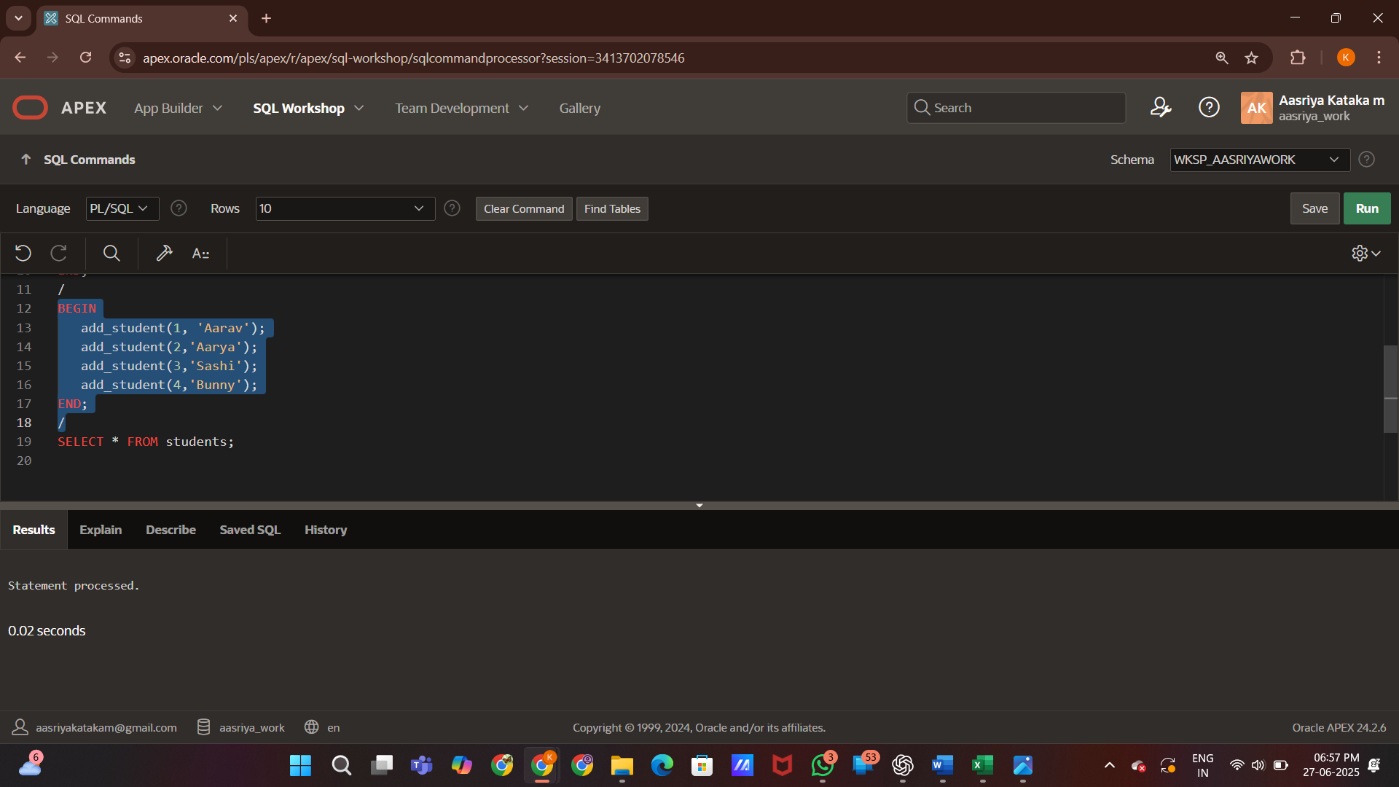
/

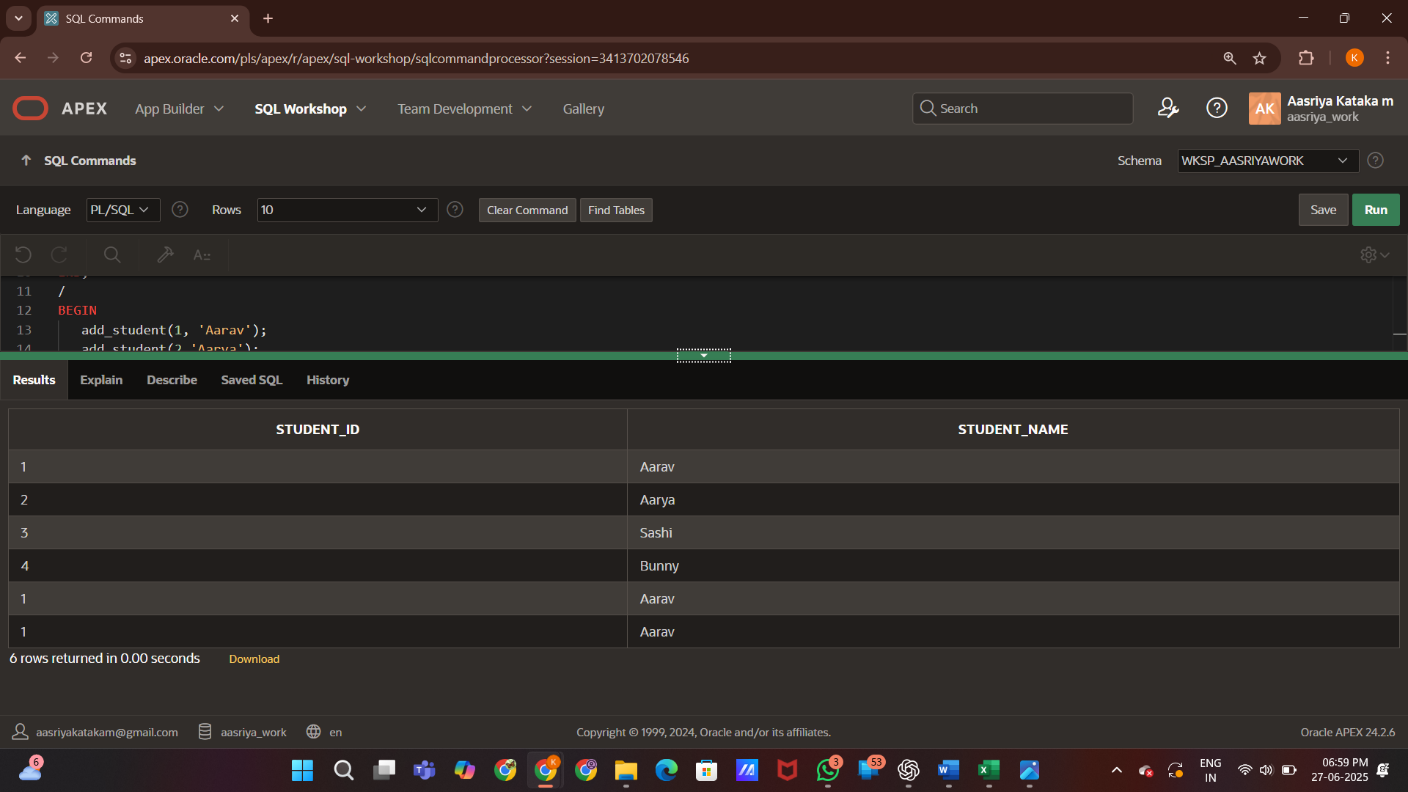
SELECT \* FROM students;

Output:









1. Procedure with IF/ELSE Logic

CREATE OR REPLACE PROCEDURE check\_pass(p\_score IN NUMBER) IS

BEGIN

IF p\_score >= 50 THEN

DBMS\_OUTPUT.PUT\_LINE('Status: Passed');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Status: Failed');

END IF;

END;

/

BEGIN

check\_pass(45);

END;

/

Output:

