**IF THEN ELSE Statement:**

DECLARE

v\_myage NUMBER:=10;

BEGIN

IF v\_myage < 11

THEN

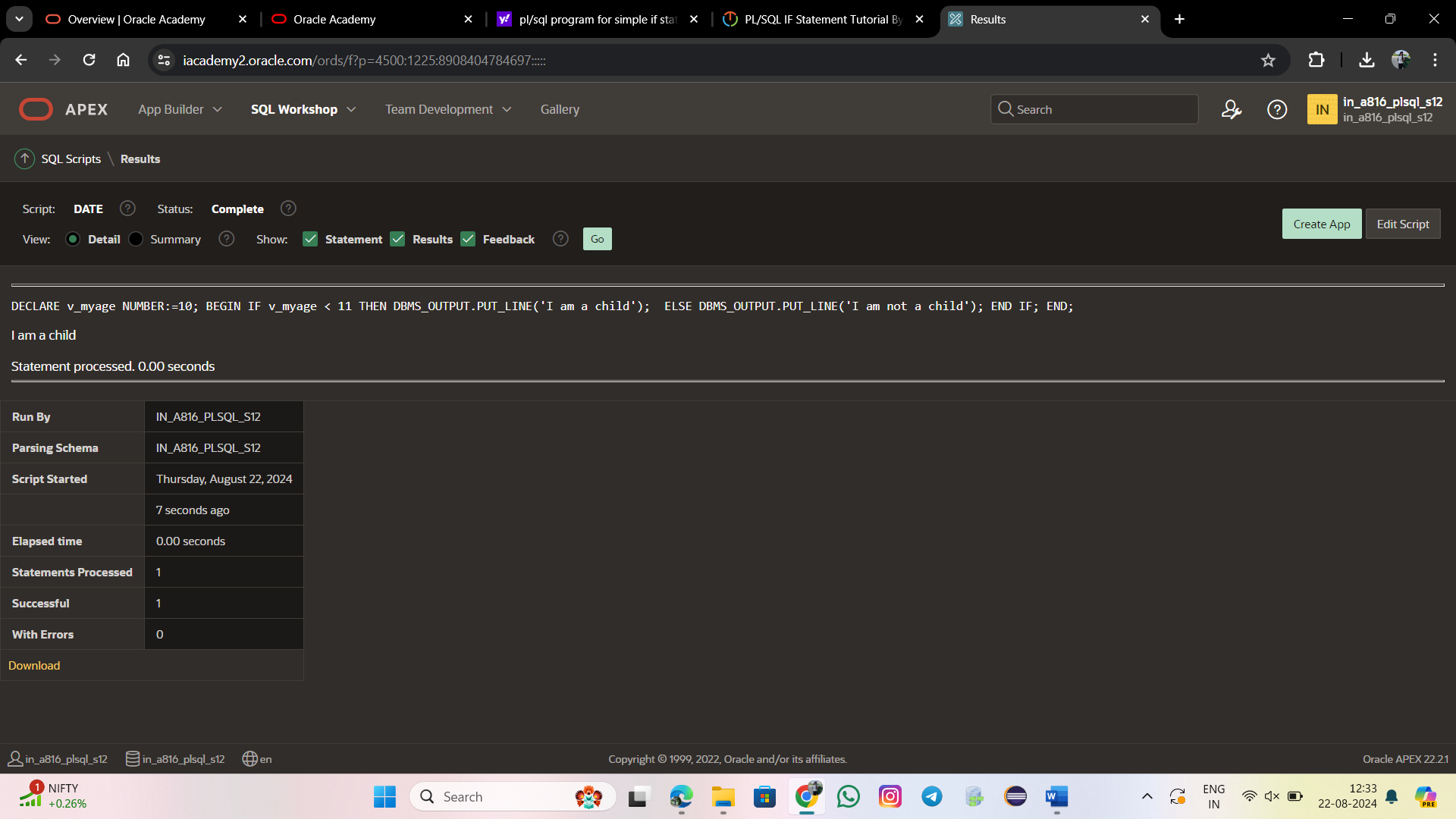
DBMS\_OUTPUT.PUT\_LINE('I am a child');

ELSE

DBMS\_OUTPUT.PUT\_LINE('I am not a child');

END IF;

END;



**IF ELSIF ELSE Clause:**

DECLARE

v\_myage NUMBER := 31;

BEGIN

IF v\_myage < 11

THEN

DBMS\_OUTPUT.PUT\_LINE('I am a child');

ELSIF v\_myage < 20

THEN

DBMS\_OUTPUT.PUT\_LINE('I am young');

ELSIF v\_myage < 30

THEN

DBMS\_OUTPUT.PUT\_LINE('I am in my twenties');

ELSIF v\_myage < 40

THEN

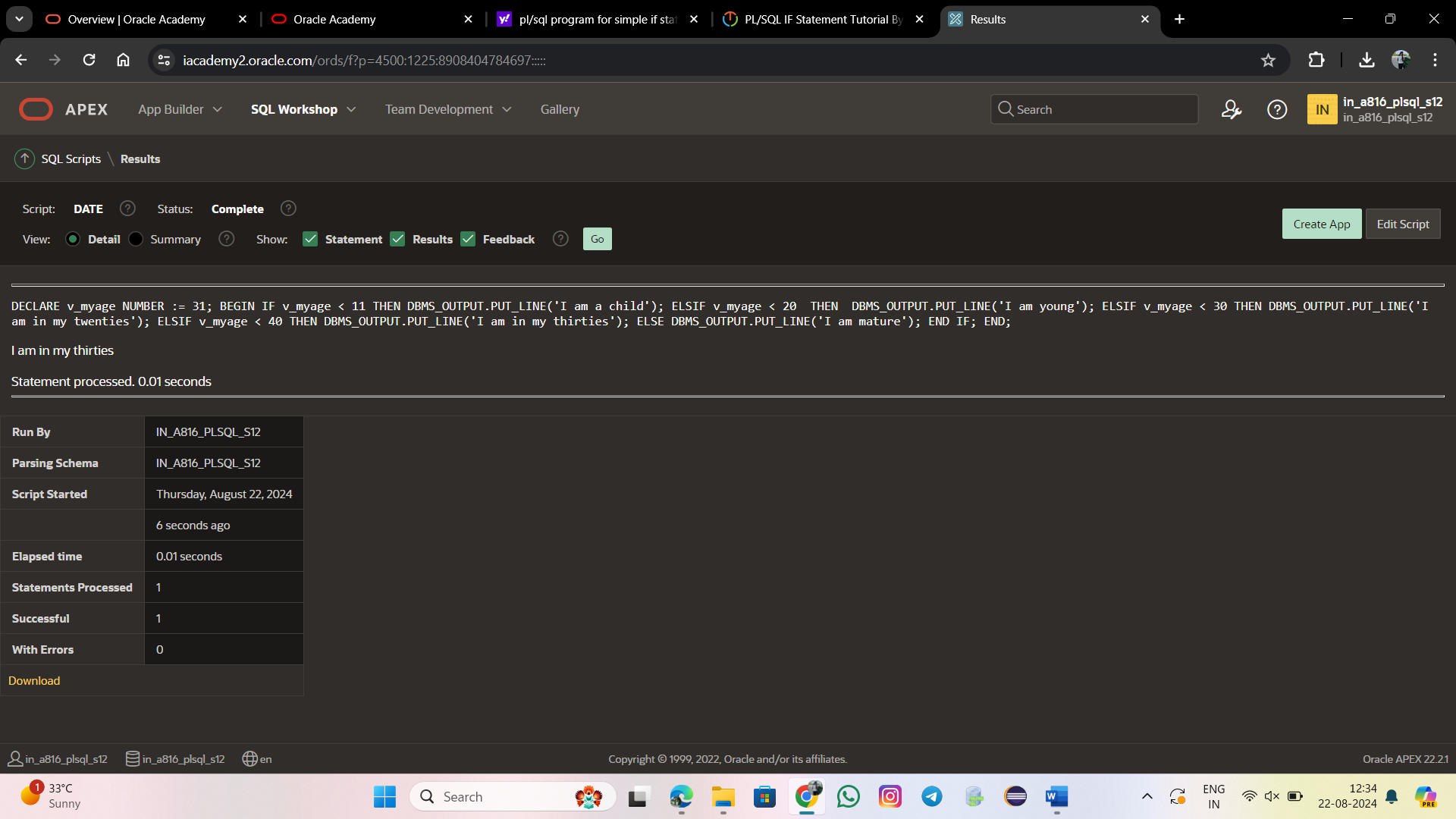
DBMS\_OUTPUT.PUT\_LINE('I am in my thirties');

ELSE

DBMS\_OUTPUT.PUT\_LINE('I am mature');

END IF;

END;



**IF Statement with Multiple Expressions:**

DECLARE

v\_myage NUMBER := 10;

v\_myfirstname VARCHAR2(11) := 'Christopher';

BEGIN

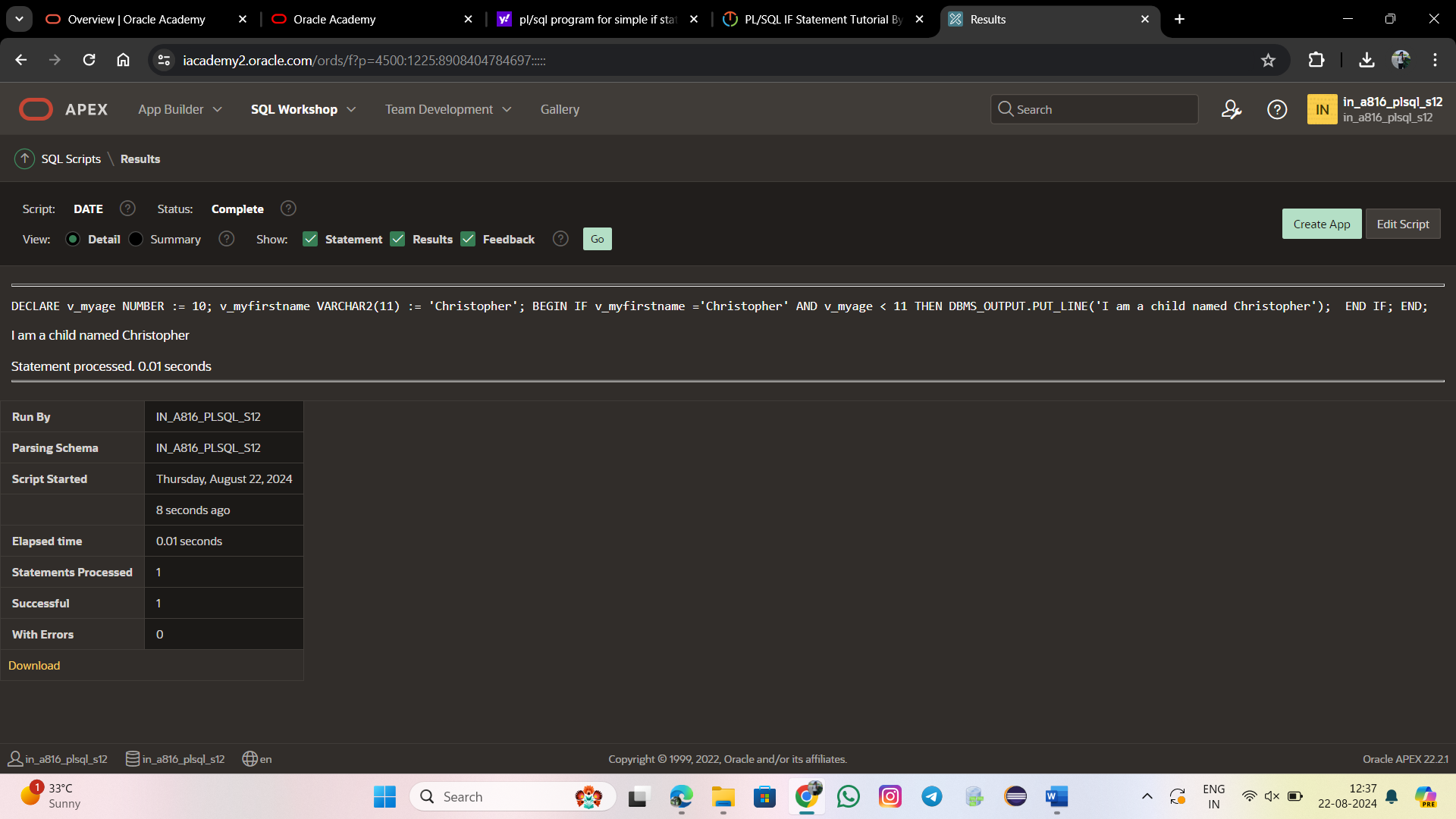
IF v\_myfirstname ='Christopher' AND v\_myage < 11

THEN

DBMS\_OUTPUT.PUT\_LINE('I am a child named Christopher');

END IF;

END;



**NULL Values in IF Statements:**

DECLARE

v\_myage NUMBER;

BEGIN

IF v\_myage < 11

THEN

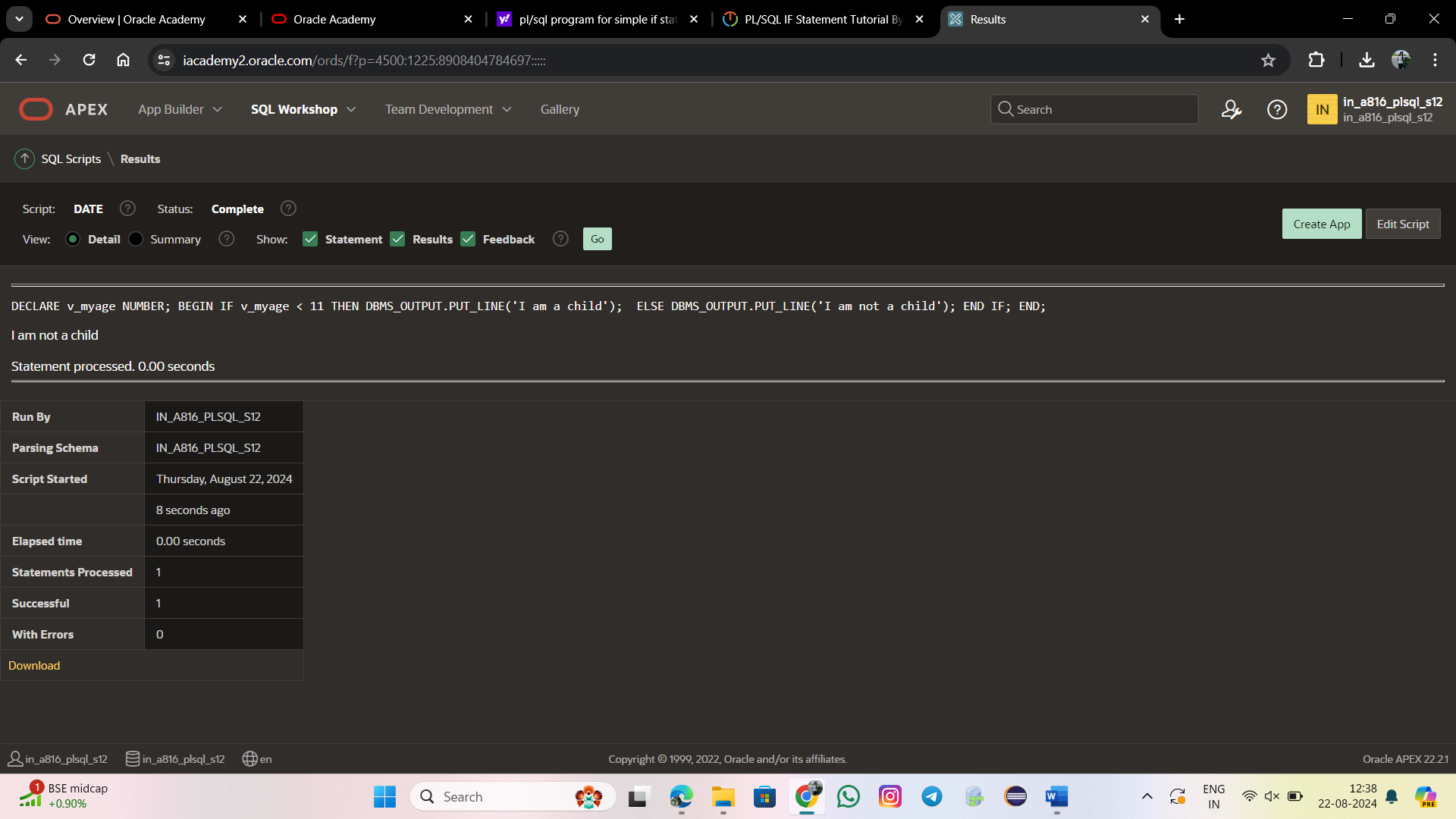
DBMS\_OUTPUT.PUT\_LINE('I am a child');

ELSE

DBMS\_OUTPUT.PUT\_LINE('I am not a child');

END IF;

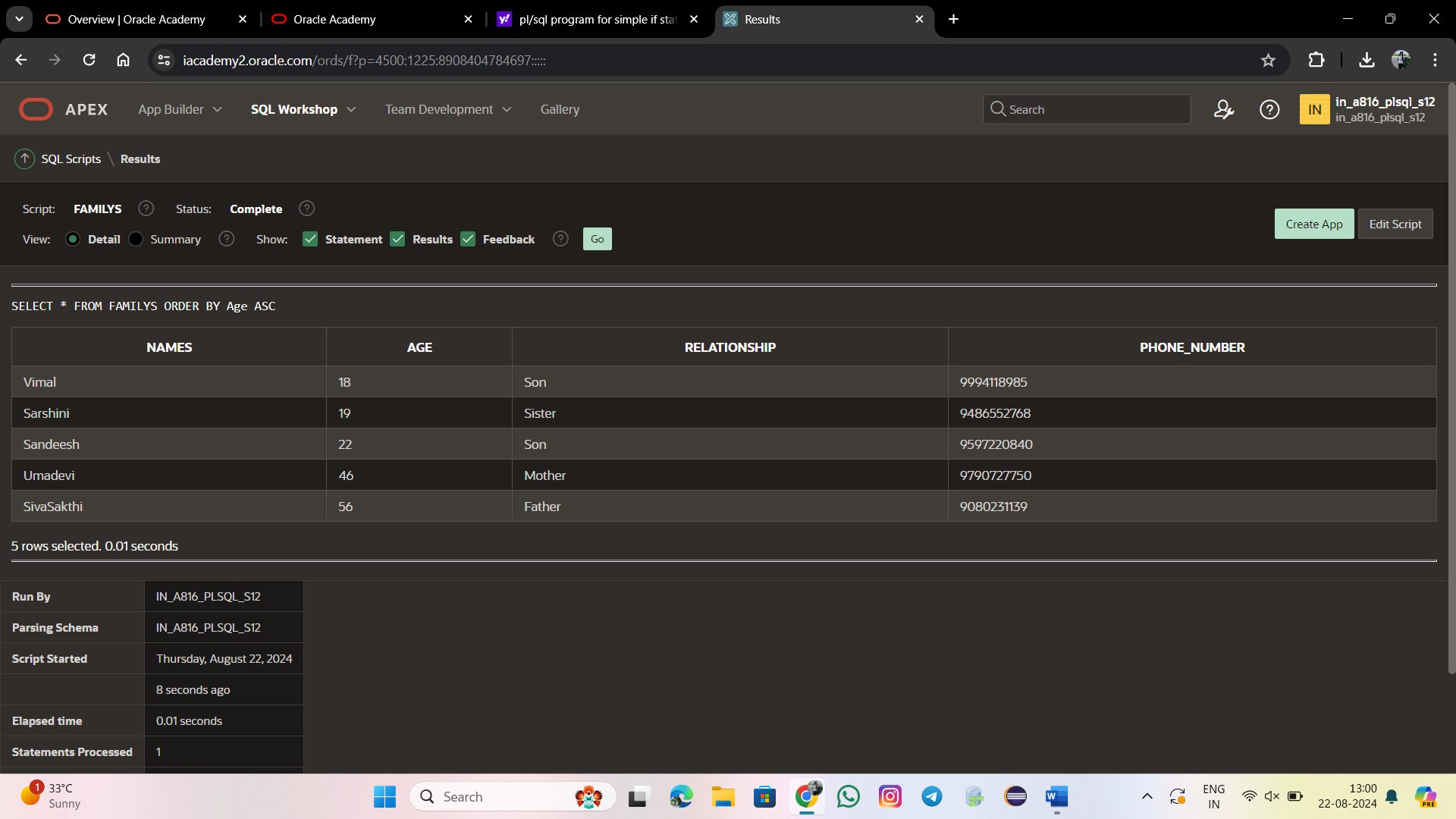
END;



SELECT \*

FROM FAMILYS

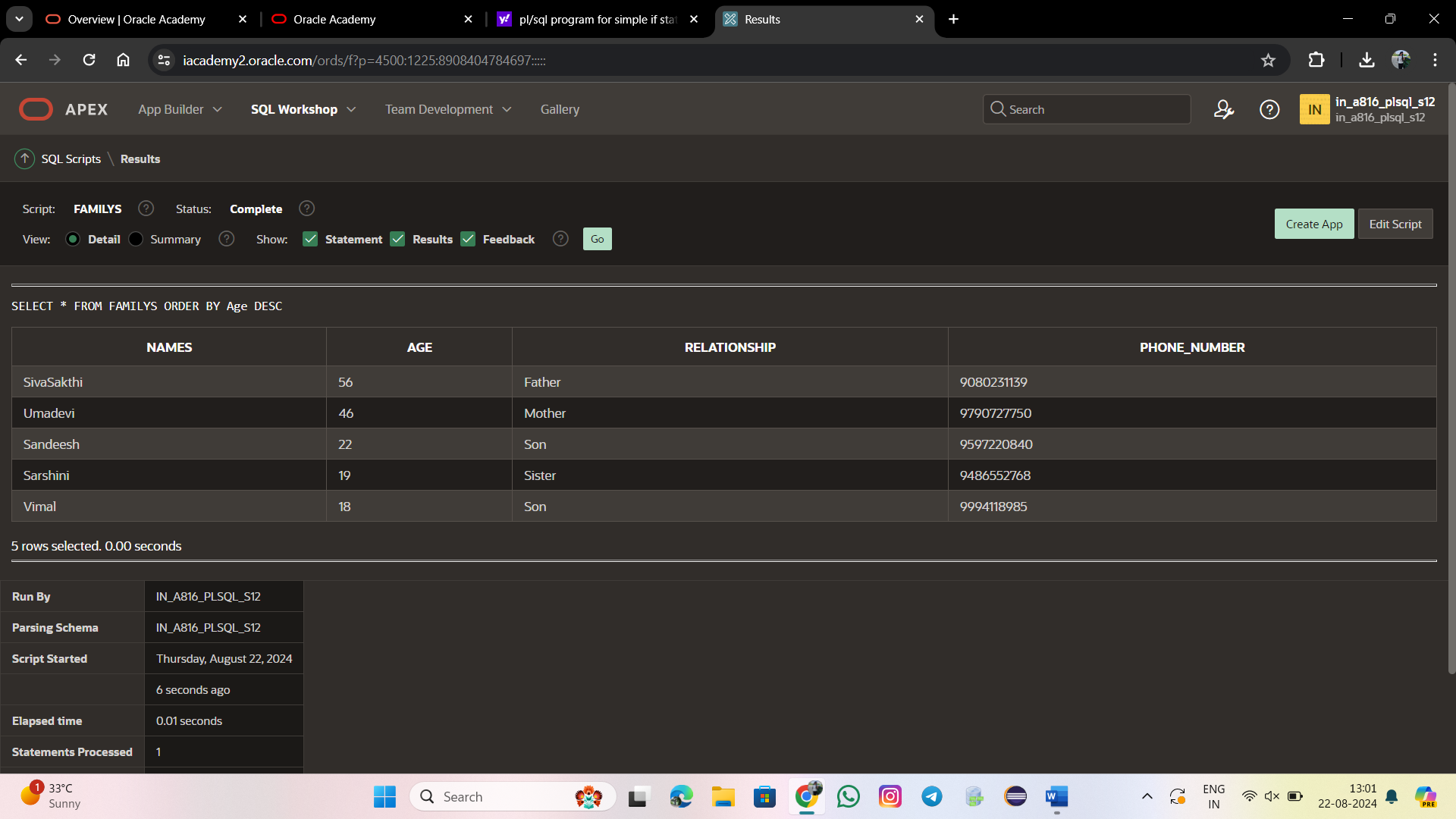
ORDER BY Age ASC;



SELECT \*

FROM FAMILYS

ORDER BY Age DESC;



DECLARE

CURSOR my\_cursor IS

SELECT Names,Age,Phone\_Number

FROM FAMILYS

ORDER BY Age ASC;

my\_record my\_cursor%ROWTYPE;

BEGIN

OPEN my\_cursor;

LOOP

FETCH my\_cursor INTO my\_record;

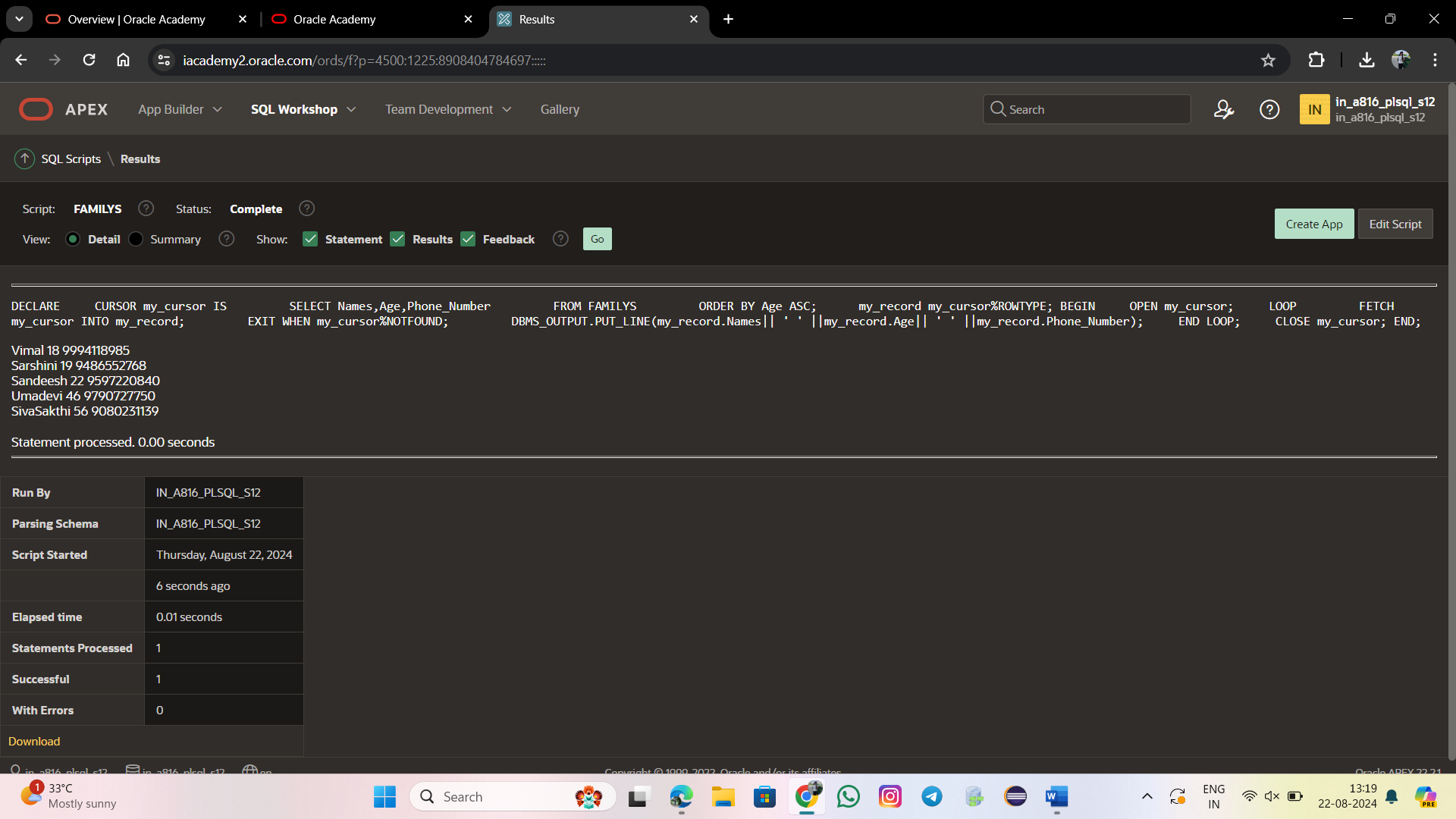
EXIT WHEN my\_cursor%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE(my\_record.Names|| ' ' ||my\_record.Age|| ' ' ||my\_record.Phone\_Number);

END LOOP;

CLOSE my\_cursor;

END;



DECLARE

TYPE num\_list\_type IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

v\_nums num\_list\_type;

BEGIN

v\_nums(1) := 1;

v\_nums(2) := 3;

v\_nums(3) := 5;

v\_nums(4) := 7;

v\_nums(5) := 11;

FORALL i IN v\_nums.FIRST .. v\_nums.LAST

INSERT INTO ARRAY (n) VALUES (v\_nums(i));

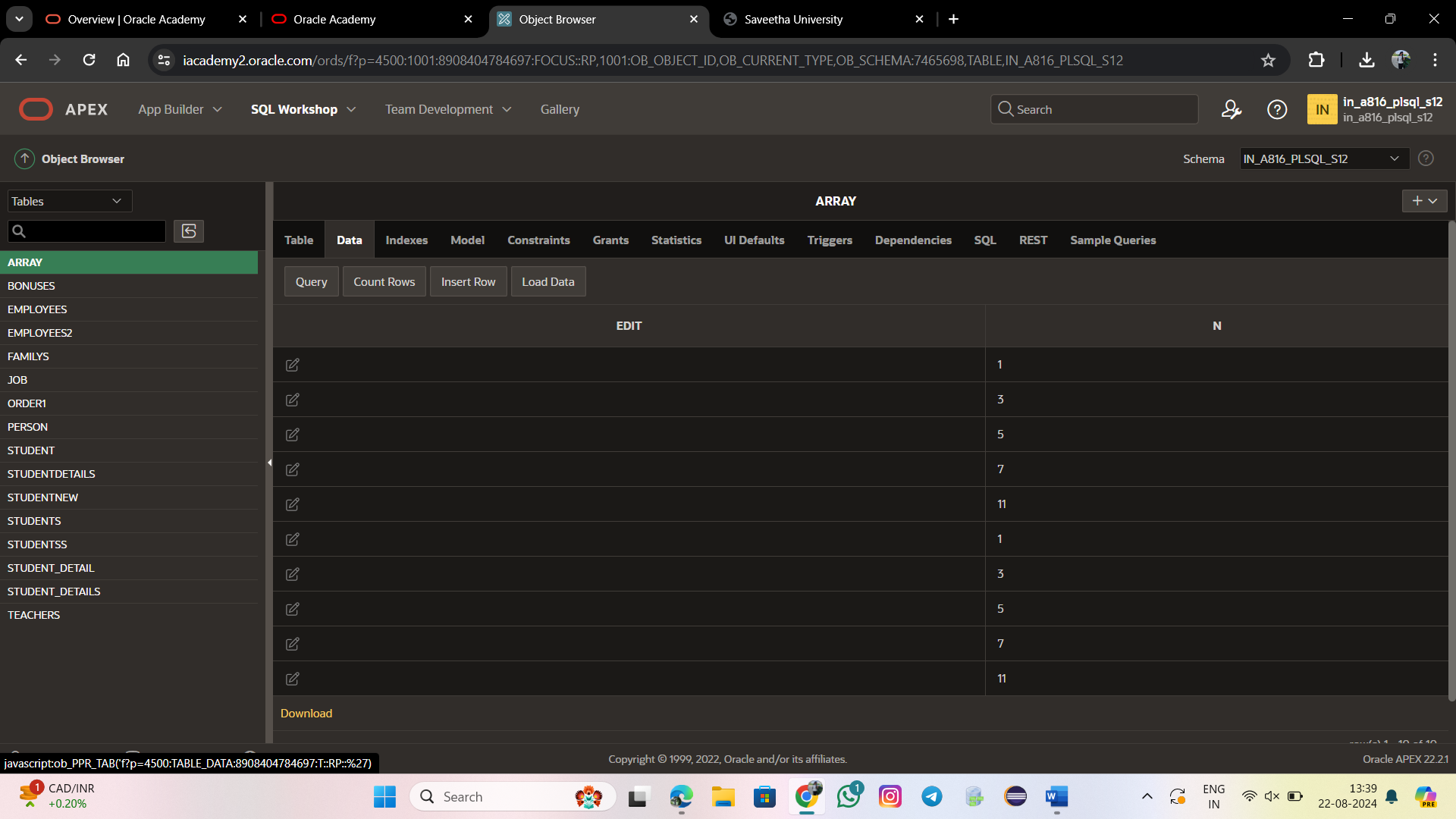
FOR i IN v\_nums.FIRST .. v\_nums.LAST

LOOP

DBMS\_OUTPUT.PUT\_LINE('Inserted ' || SQL%BULK\_ROWCOUNT(i) || ' row(s) on iteration ' || i);

END LOOP;

END;



DECLARE

TYPE num\_list\_type IS TABLE OF NUMBER INDEX BY BINARY\_INTEGER;

v\_nums num\_list\_type;

v\_nums1 num\_list\_type;

BEGIN

v\_nums(1) := 1;

v\_nums(2) := 3;

v\_nums(3) := 5;

v\_nums(4) := 7;

v\_nums(5) := 11;

v\_nums1(1) := 1;

v\_nums1(2) := 3;

v\_nums1(3) := 5;

v\_nums1(4) := 7;

v\_nums1(5) := 11;

FOR i IN v\_nums.FIRST .. v\_nums.LAST LOOP

FOR i IN v\_nums1.FIRST .. v\_nums1.LAST LOOP

INSERT INTO NUMS\_ADDS (n, m, SUM\_N\_M)

VALUES (v\_nums(i),v\_nums1(i), v\_nums(i) + v\_nums1(i));

END LOOP;

END LOOP;

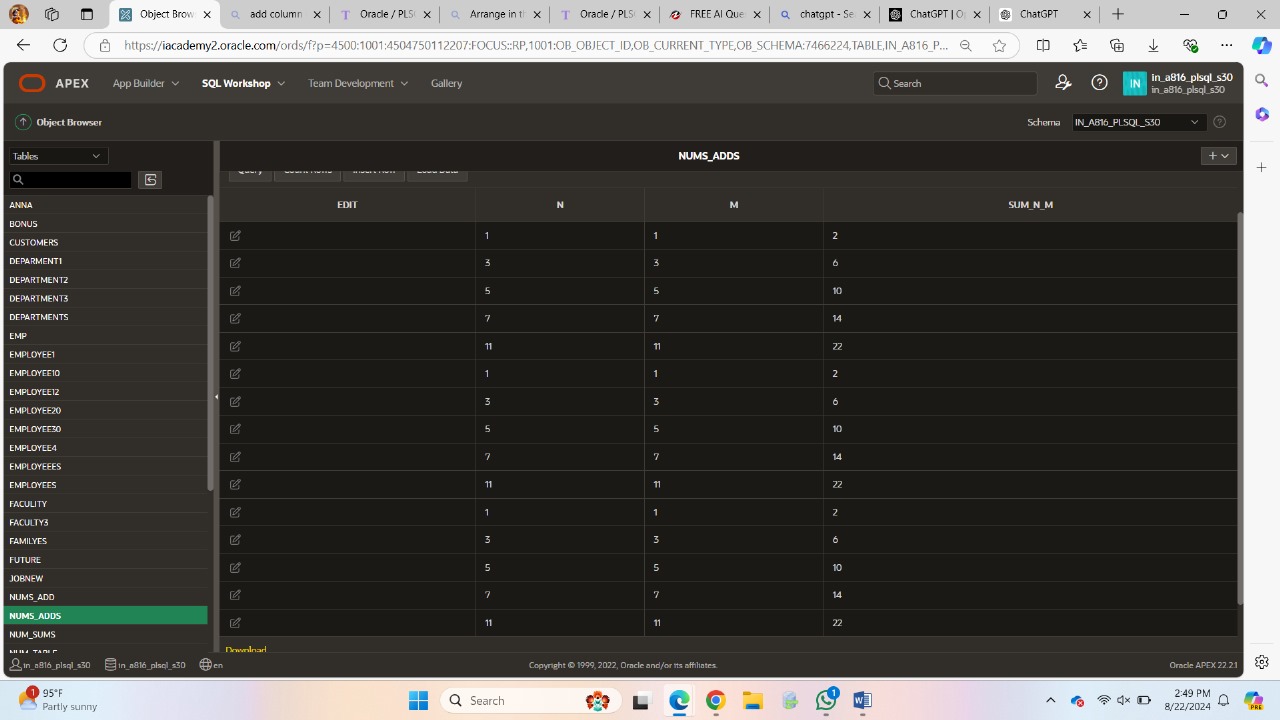
COMMIT;

FOR i IN v\_nums.FIRST .. v\_nums.LAST LOOP

DBMS\_OUTPUT.PUT\_LINE('Inserted ' || SQL%ROWCOUNT || ' row(s) in total.');

END LOOP;

END;



DECLARE

a number(2) := 30;

BEGIN

<<loopstart>>

-- while loop execution

WHILE a < 50 LOOP

dbms\_output.put\_line ('value of a: ' || a);

a := a + 1;

IF a = 35 THEN

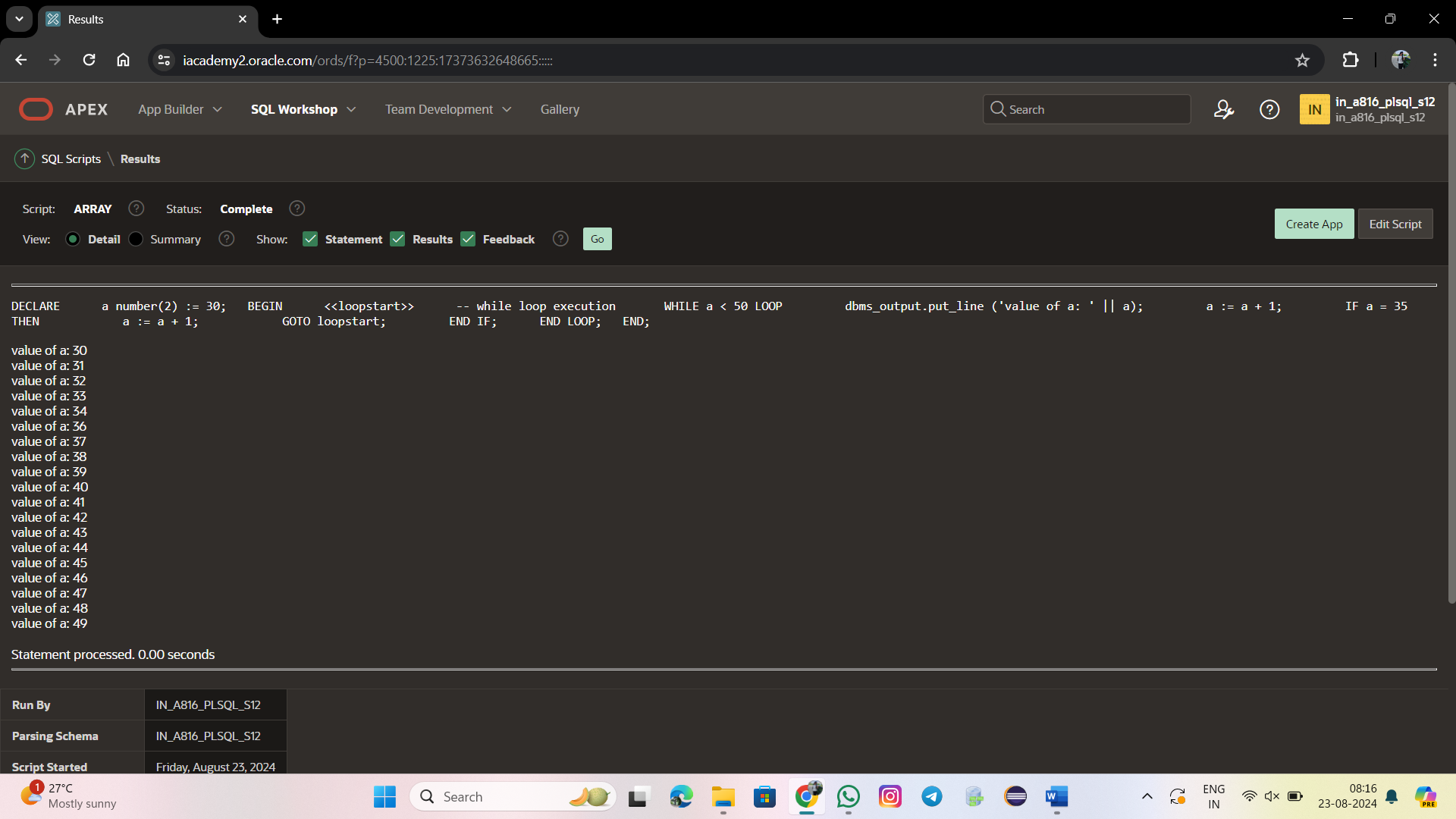
a := a + 1;

GOTO loopstart;

END IF;

END LOOP;

END;



DECLARE

emp\_name VARCHAR2(100);

emp\_id NUMBER := 225;

BEGIN

BEGIN

SELECT first\_name INTO emp\_name FROM EMPLOYEES WHERE Employees\_id = emp\_id;

DBMS\_OUTPUT.PUT\_LINE('Employee Name: ' || emp\_name);

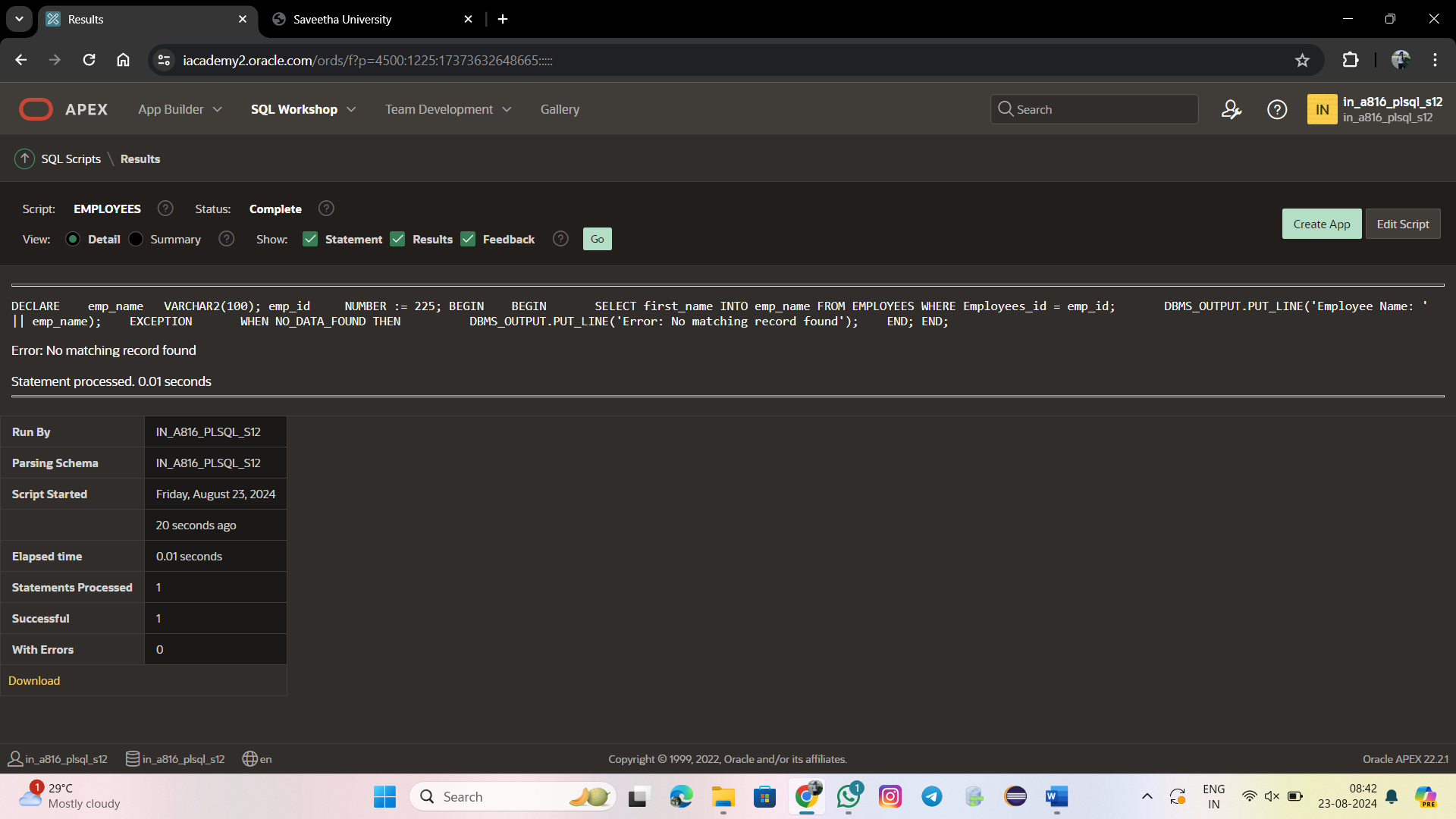
EXCEPTION

WHEN NO\_DATA\_FOUND THEN

DBMS\_OUTPUT.PUT\_LINE('Error: No matching record found');

END;

END;



DECLARE

v\_dep\_id employees.department\_id%TYPE := 90;

v\_emp\_id employees.employees\_id%TYPE;

v\_emp\_name employees.first\_name%TYPE;

v\_exception\_msg VARCHAR2(200);

BEGIN

SELECT first\_name, department\_id

INTO v\_emp\_name,v\_dep\_id

FROM employees

WHERE department\_id = v\_dep\_id;

EXCEPTION

WHEN TOO\_MANY\_ROWS THEN

v\_exception\_msg := 'Multiple rows found for the given department ID: ' || v\_dep\_id;

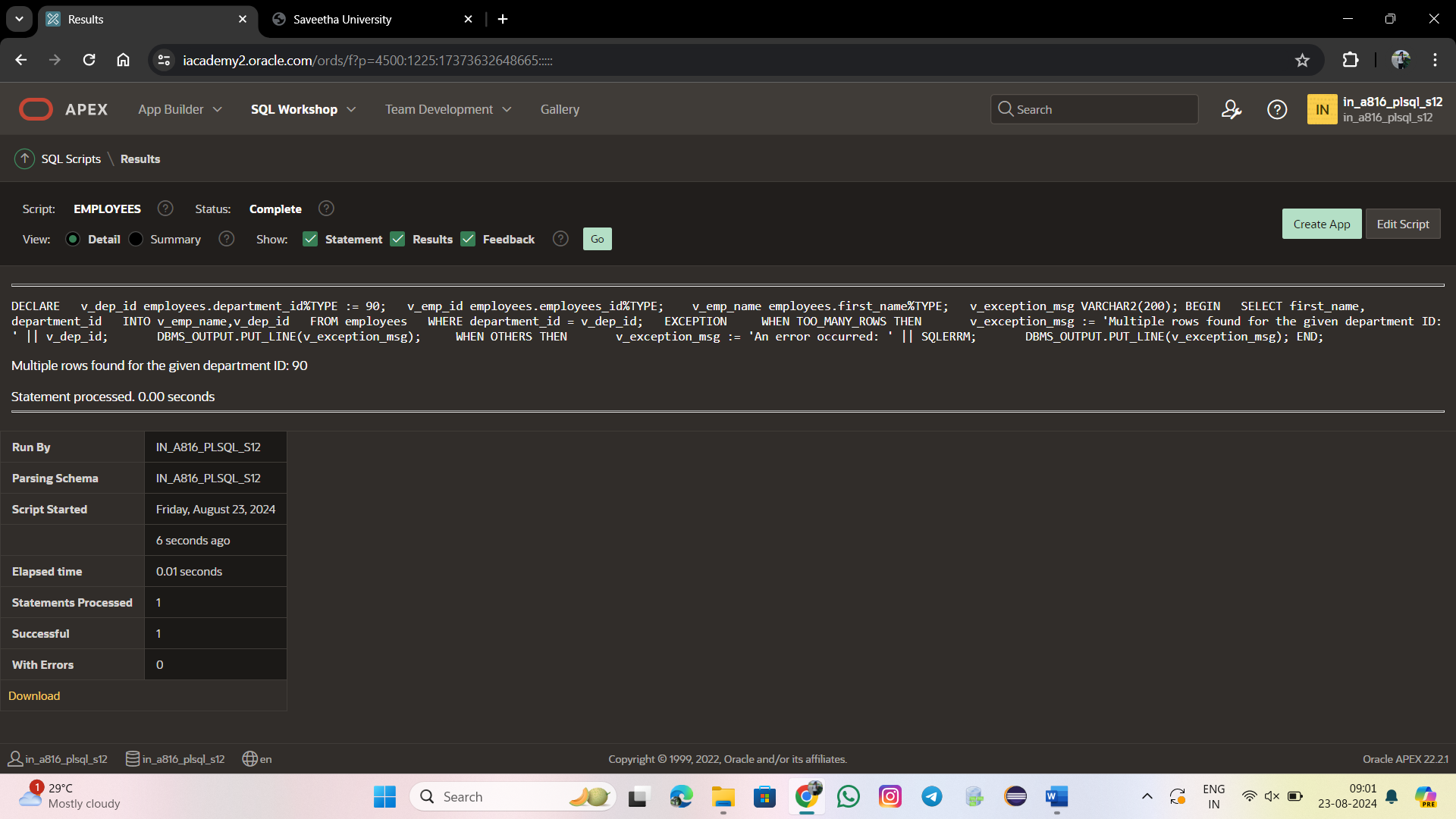
DBMS\_OUTPUT.PUT\_LINE(v\_exception\_msg);

WHEN OTHERS THEN

v\_exception\_msg := 'An error occurred: ' || SQLERRM;

DBMS\_OUTPUT.PUT\_LINE(v\_exception\_msg);

END;



DECLARE

v\_input VARCHAR2(10) := 'abc';

v\_number NUMBER;

BEGIN

BEGIN

v\_number := TO\_NUMBER(v\_input);

DBMS\_OUTPUT.PUT\_LINE('Conversion successful. Number: ' || v\_number);

EXCEPTION

WHEN INVALID\_NUMBER THEN

DBMS\_OUTPUT.PUT\_LINE('Error: Invalid number');

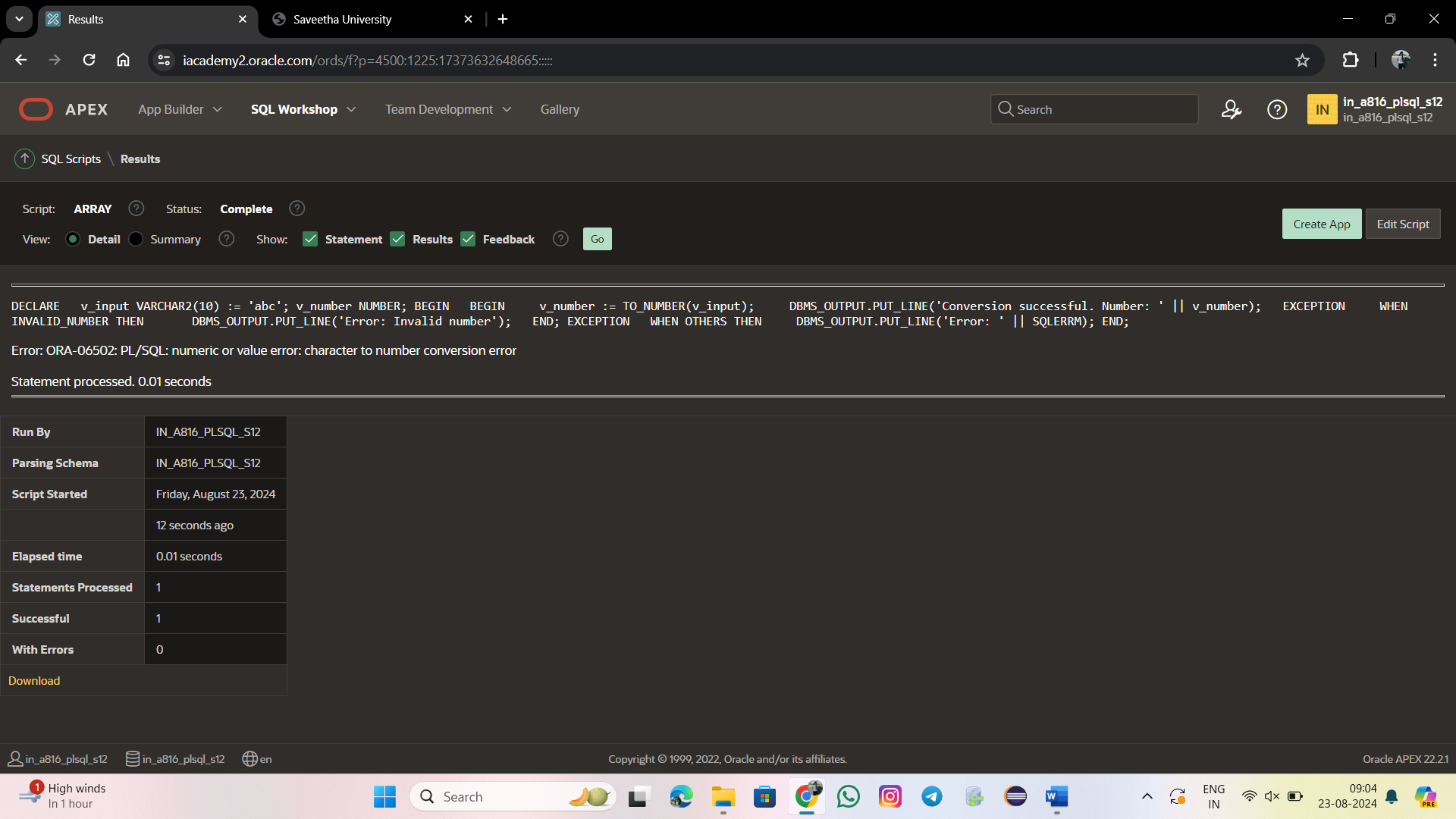
END;

EXCEPTION

WHEN OTHERS THEN

DBMS\_OUTPUT.PUT\_LINE('Error: ' || SQLERRM);

END;



DECLARE

v\_duplicate\_value NUMBER := 125;

BEGIN

BEGIN

INSERT INTO EMPLOYEES (employees\_id) VALUES (v\_duplicate\_value);

DBMS\_OUTPUT.PUT\_LINE('Insertion successful.');

EXCEPTION

WHEN DUP\_VAL\_ON\_INDEX THEN

-- Handle the exception

DBMS\_OUTPUT.PUT\_LINE('Error: Duplicate value already exists.');

-- You can perform additional error handling or logging here

END;

END;

