

Class Test 03

PL/SQL

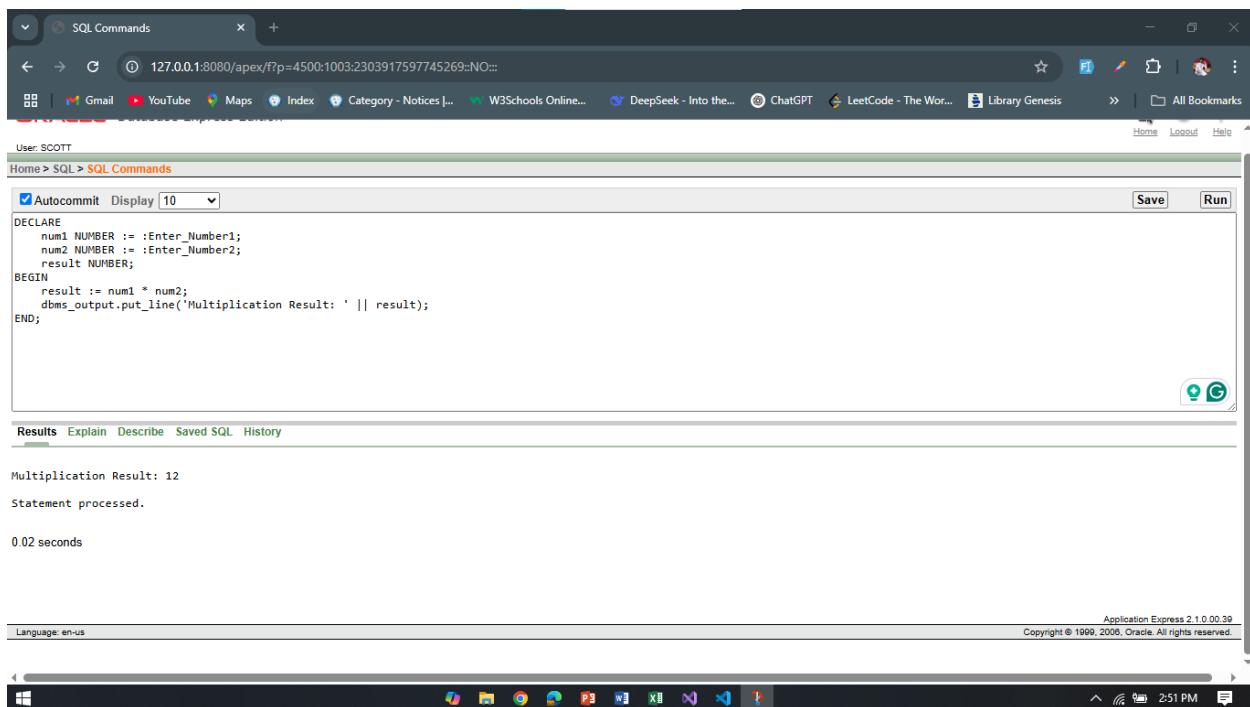
Part 01:

1. Write a query that can multiply two numbers taking input from user.

Answer:

```
DECLARE
    num1 NUMBER := :Enter_Number1;
    num2 NUMBER := :Enter_Number2;
    result NUMBER;
BEGIN
    result := num1 * num2;
    dbms_output.put_line('Multiplication Result: ' || result);
END;
```

Output:



2. Write a query that can add two numbers if the numbers are equal. Use CASE Statement.

Answer:

```
DECLARE
    num1 NUMBER := :Enter_Number1;
    num2 NUMBER := :Enter_Number2;
    result NUMBER;
```

```

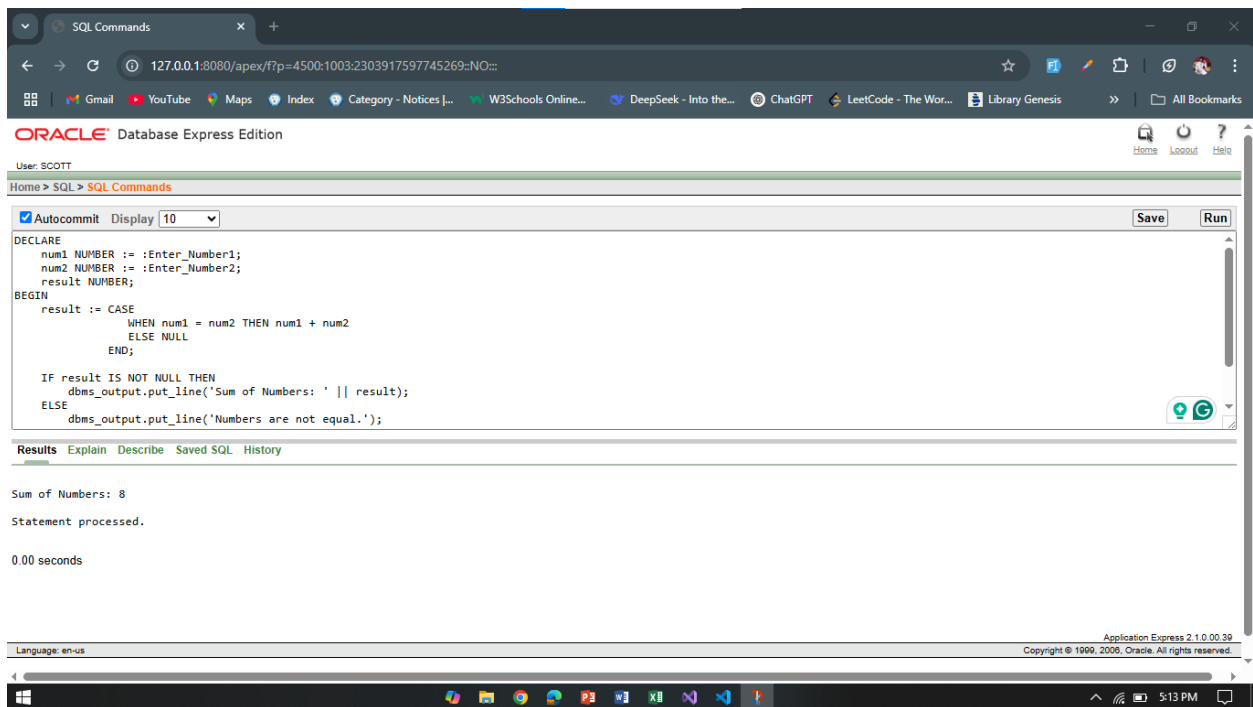
BEGIN
    result := CASE
        WHEN num1 = num2 THEN num1 + num2
        ELSE NULL
    END;

    IF result IS NOT NULL THEN
        dbms_output.put_line('Sum of Numbers: ' || result);
    ELSE
        dbms_output.put_line('Numbers are not equal.');
```

```

    END IF;
END;
```

Output:



- Write a query that can check if two strings are equal or not. Use IF-THEN-ELSIF Statement.

Answer:

```

DECLARE
    str1 VARCHAR2(50) := 'String1';
    str2 VARCHAR2(50) := 'String2';
BEGIN
    IF str1 = str2 THEN
        dbms_output.put_line('Strings are equal.');
```

```

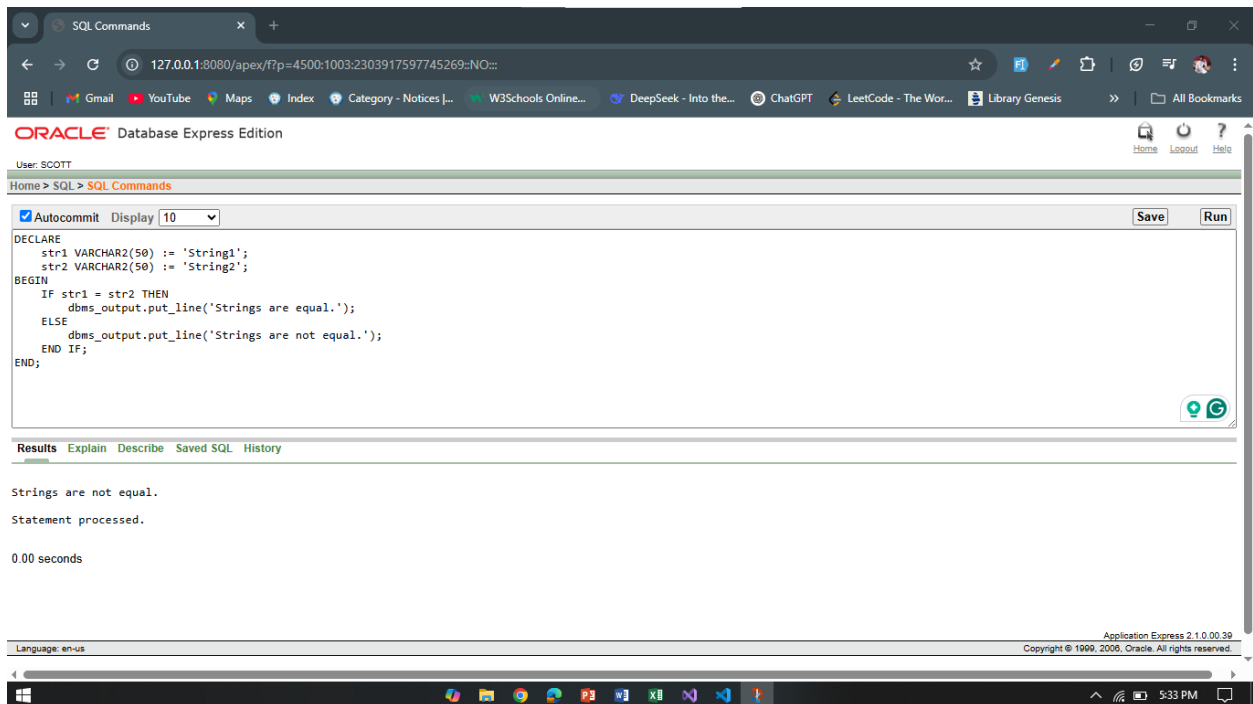
ELSE
    dbms_output.put_line('Strings are not equal.');
```

END IF;

```

END;
```

Output:



- Write a query that can multiply two numbers. If the result obtained is less than 100, **Hi** is displayed, if the result obtained is more than 100, **Bye** is displayed and if the result obtained is equal to 100, **ADBMS** is displayed. Use IF-THEN-ELSIF Statement

Answer:

```

DECLARE
    num1 NUMBER := :Enter_Number1;
    num2 NUMBER := :Enter_Number2;
    result NUMBER;

BEGIN
    result := num1 * num2;

    IF result < 100 THEN
        dbms_output.put_line('Hi');
```

ELSIF result > 100 THEN

```

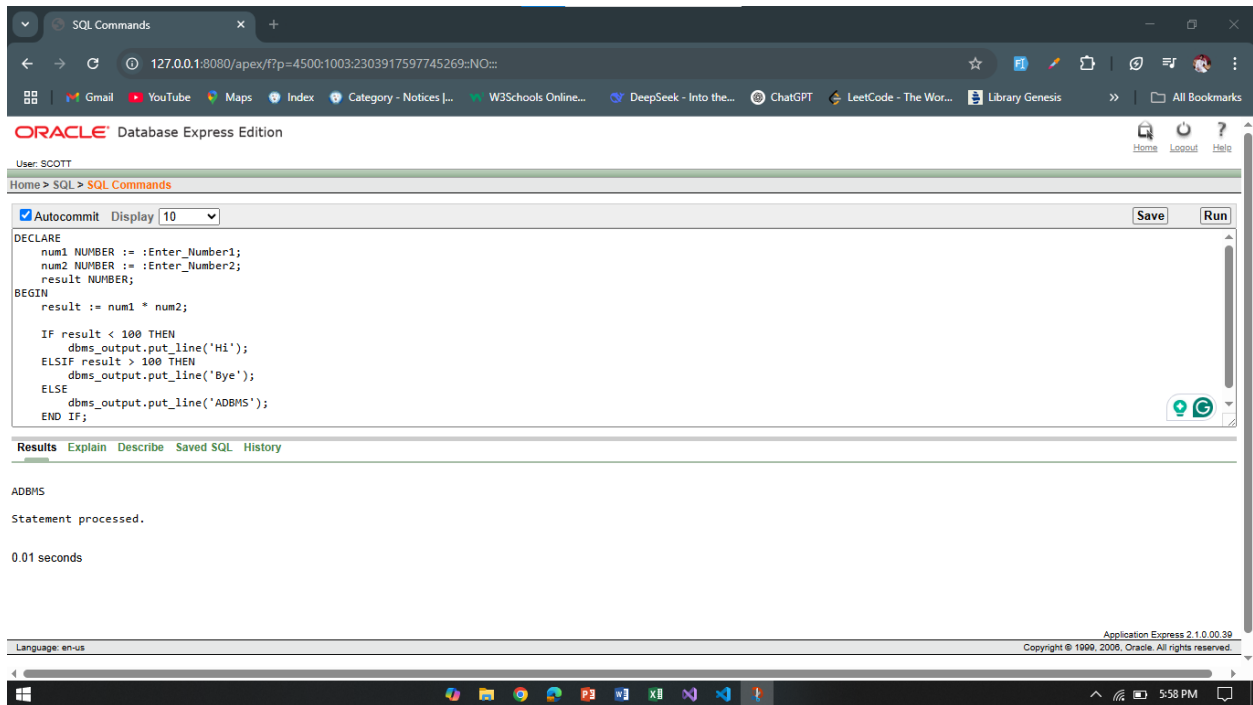
        dbms_output.put_line('Bye');
```

ELSE

```

        dbms_output.put_line('ADBMS');
    END IF;
END;

```



5. Write a query that can check if two numbers are equal or not. Use CASE Statement.

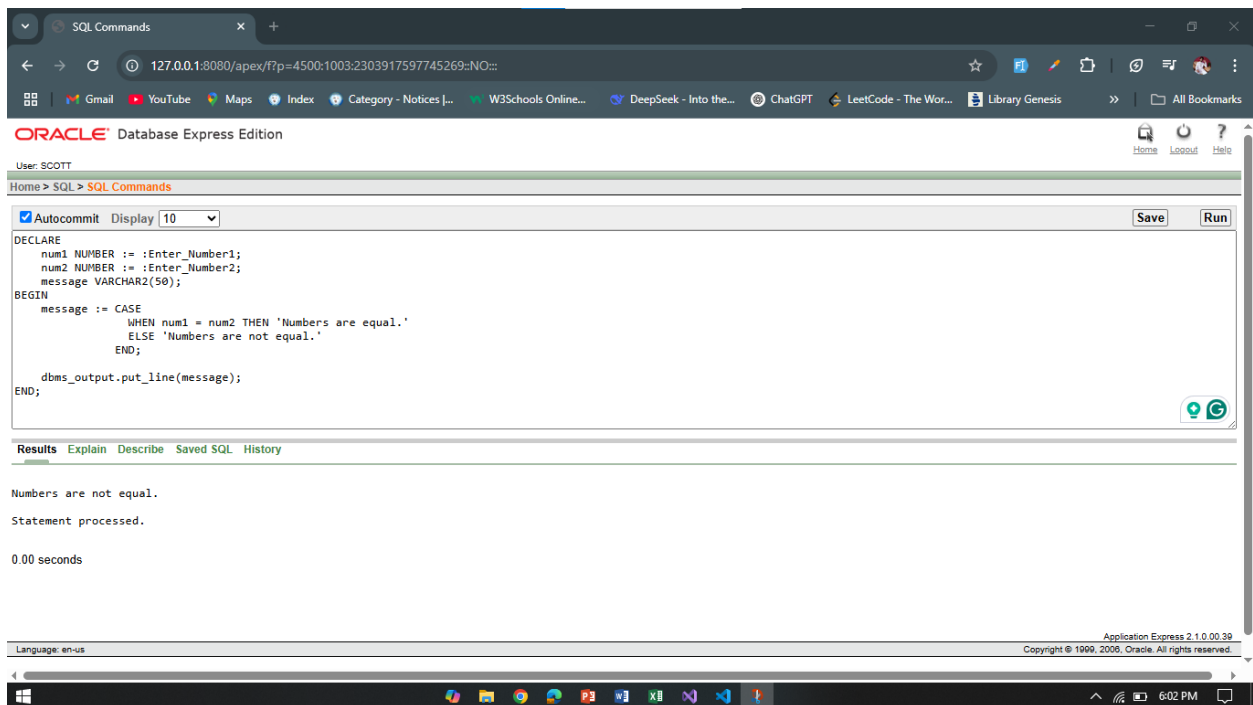
Answer:

```

DECLARE
    num1 NUMBER := :Enter_Number1;
    num2 NUMBER := :Enter_Number2;
    message VARCHAR2(50);
BEGIN
    message := CASE
        WHEN num1 = num2 THEN 'Numbers are equal.'
        ELSE 'Numbers are not equal.'
    END;

    dbms_output.put_line(message);
END;

```



Part 02:

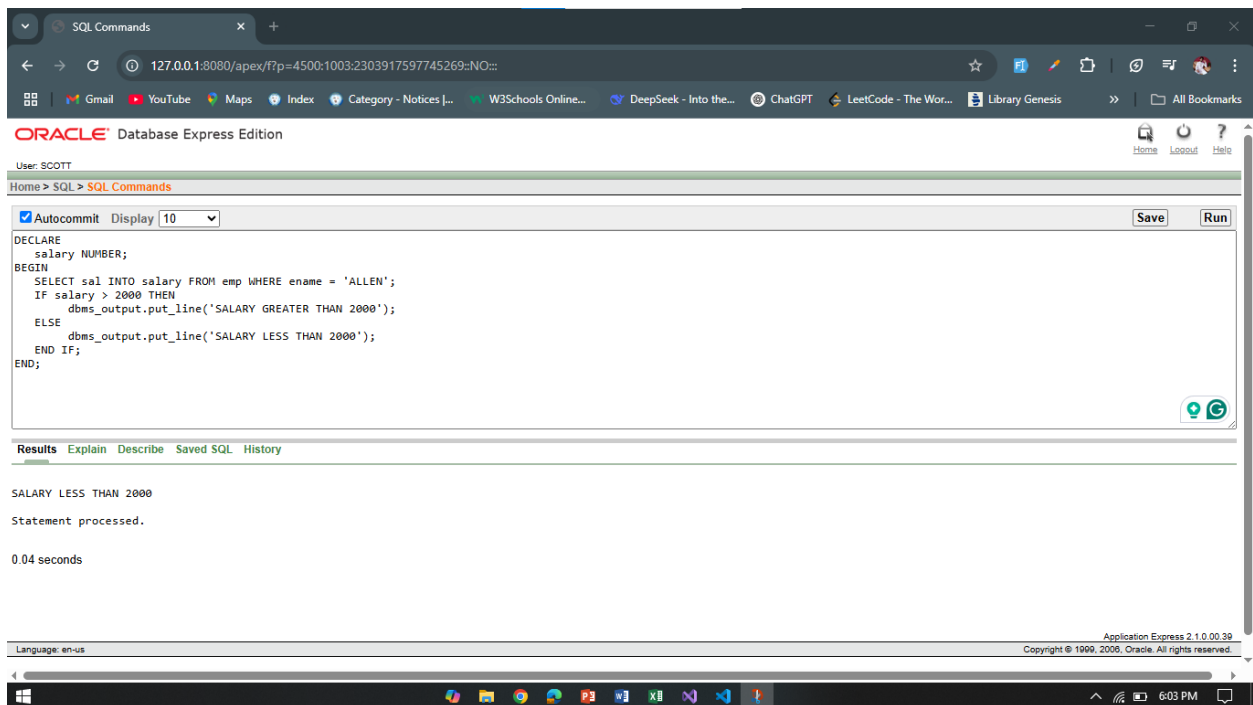
To solve the following use the scott schema

- Write a query that can display the salary of employee ALLEN. If ALLEN's salary is greater than 2000 display 'SALARY GREATER THAN 2000' and If not then display 'SALARY LESS THAN 2000'.

Answer:

```
DECLARE
    salary NUMBER;
BEGIN
    SELECT sal INTO salary FROM emp WHERE ename = 'ALLEN';
    IF salary > 2000 THEN
        dbms_output.put_line('SALARY GREATER THAN 2000');
    ELSE
        dbms_output.put_line('SALARY LESS THAN 2000');
    END IF;
END;
```

Output:



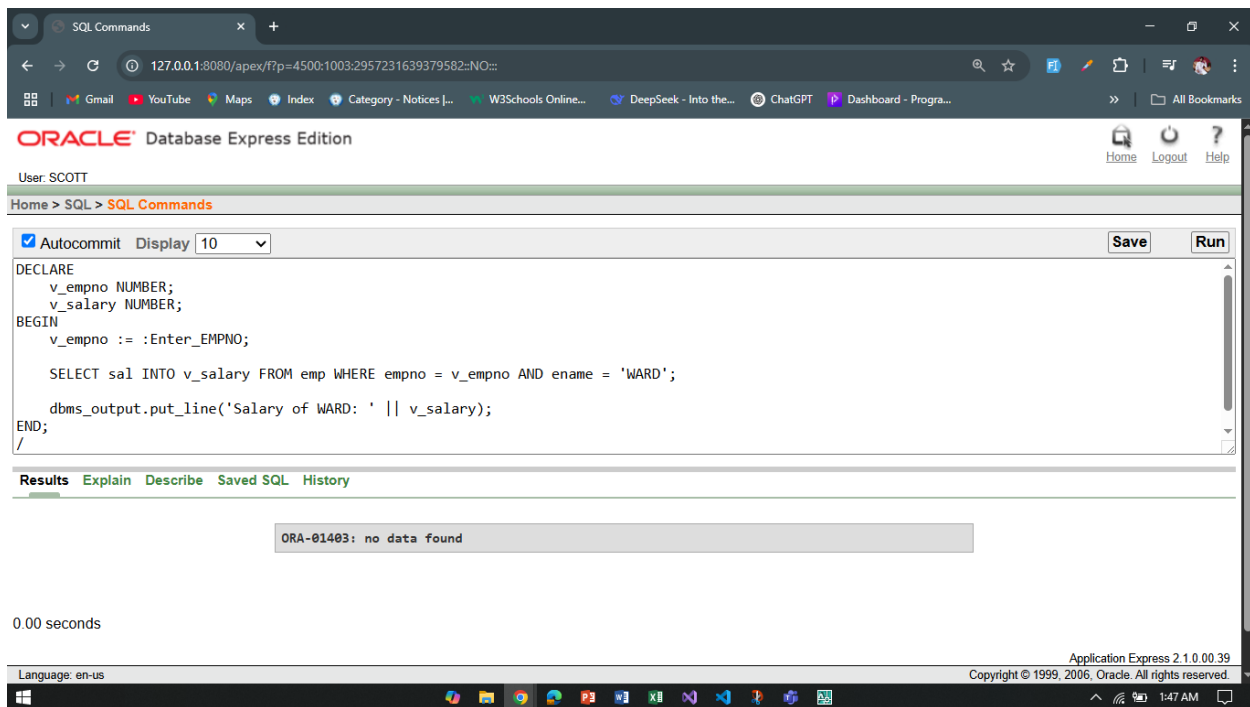
7. Write a query that can ask user to input the EMPNO of employee WARD and display his salary.

Answer:

```
DECLARE
    v_empno NUMBER;
    v_salary NUMBER;
BEGIN
    v_empno := :Enter_EMPNO;

    SELECT sal INTO v_salary FROM emp WHERE empno = v_empno AND ename =
'WARD';

    dbms_output.put_line('Salary of WARD: ' || v_salary);
END;
/
```



8. Write a query that can ask user to input the EMPNO of employee BLAKE, CLARK and TURNER and display their respective salary.

Answer:

DECLARE

 v_empno NUMBER;

 v_salary NUMBER;

 v_ename VARCHAR2(20);

BEGIN

 v_empno := :Enter_EMPNO;

 SELECT ename, sal INTO v_ename, v_salary FROM emp WHERE empno = v_empno AND
 ename IN ('BLAKE', 'CLARK', 'TURNER');

 dbms_output.put_line('Employee: ' || v_ename || ', Salary: ' || v_salary);

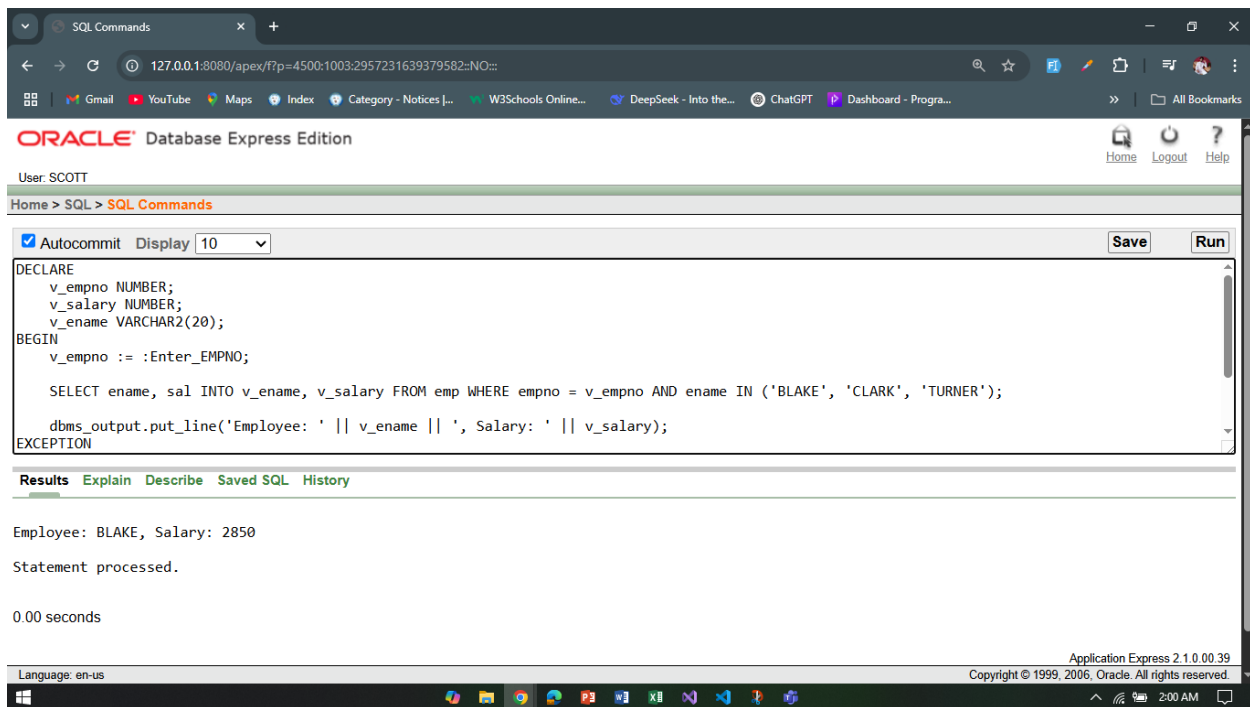
EXCEPTION

 WHEN NO_DATA_FOUND THEN

 dbms_output.put_line('Employee not found.');

END;

/



9. Write a query that can ask user to input the EMPNO of employee BLAKE, CLARK and TURNER and display their respective salary, add the salaries and display the total.

Answer:

DECLARE

 v_empno NUMBER;

 v_salary NUMBER;

 v_total_salary NUMBER := 0;

 v_name VARCHAR2(20);

BEGIN

 FOR i IN 1..3 LOOP

 v_empno := :Enter_EMPNO;

 SELECT ENAME, SAL INTO v_name, v_salary

 FROM EMP

 WHERE EMPNO = v_empno AND ENAME IN ('BLAKE', 'CLARK', 'TURNER');


```

        dbms_output.put_line('Employee: ' || v_ename || ', Salary: ' || v_salary);

        v_total_salary := v_total_salary + v_salary;

    END LOOP;

    dbms_output.put_line('Total Salary: ' || v_total_salary);

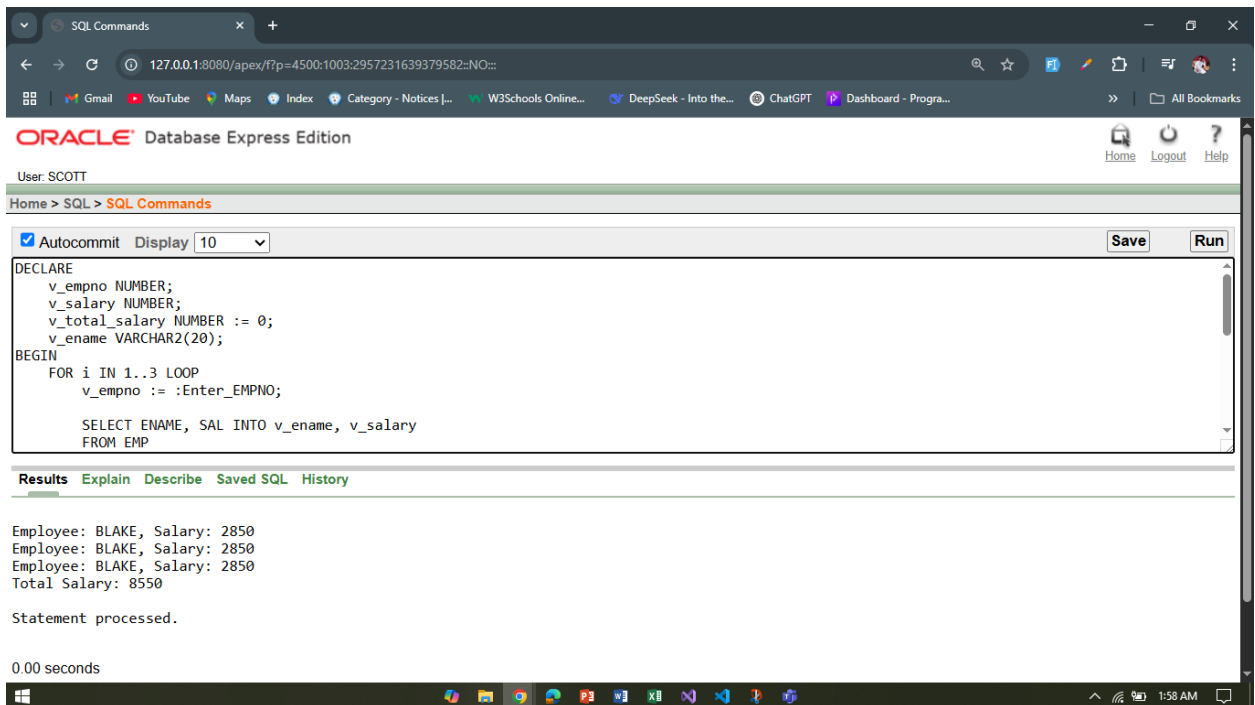
EXCEPTION

    WHEN NO_DATA_FOUND THEN

        dbms_output.put_line('Employee not found.');
```

END;

/



10. Write a query that displays the commission of employee SMITH. If SMITH's commission is NULL. Display 'NOT APPLICABLE FOR COMMISSION'

Answer:

```

DECLARE
```

```
    v_comm NUMBER;
```

```

BEGIN
```

```
    SELECT comm INTO v_comm FROM emp WHERE ename = 'SMITH';
```

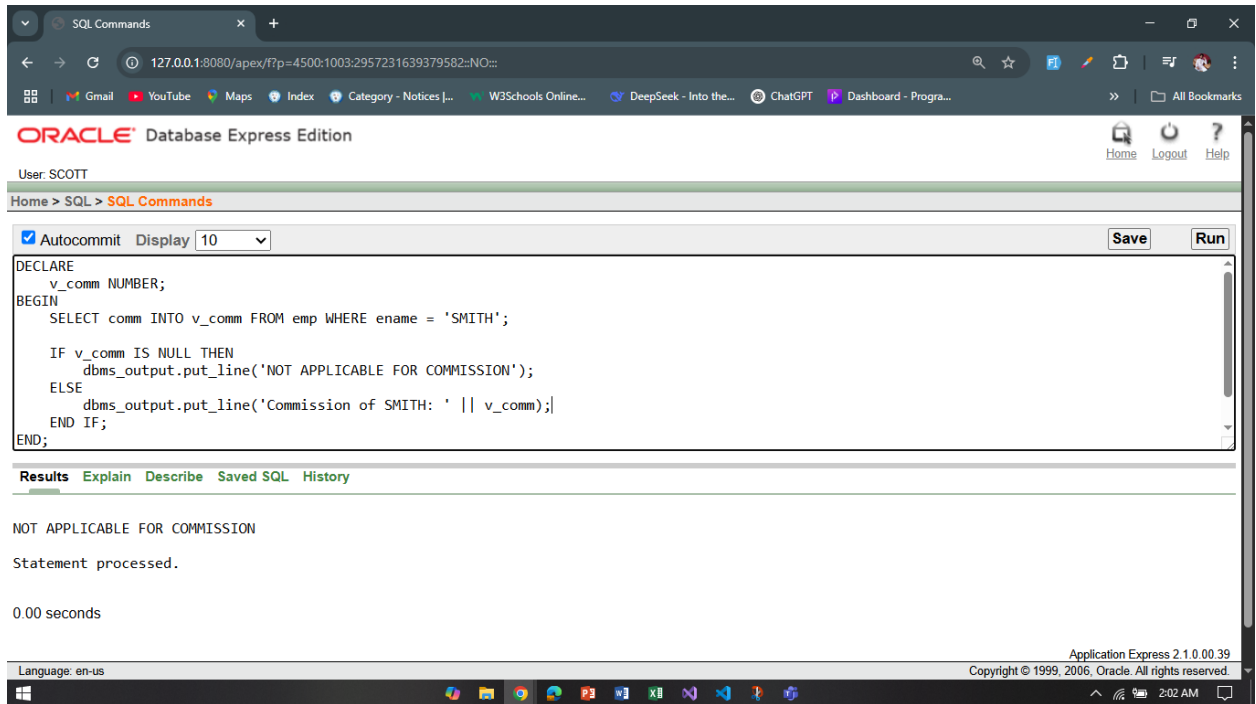
```

    IF v_comm IS NULL THEN
```

```

        dbms_output.put_line('NOT APPLICABLE FOR COMMISSION');
ELSE
        dbms_output.put_line('Commission of SMITH: ' || v_comm);
END IF;
END;
/

```



Part 03:

To solve the following use the scott schema

- Write a query that can display the salary of employee JONES three times using basic loop.

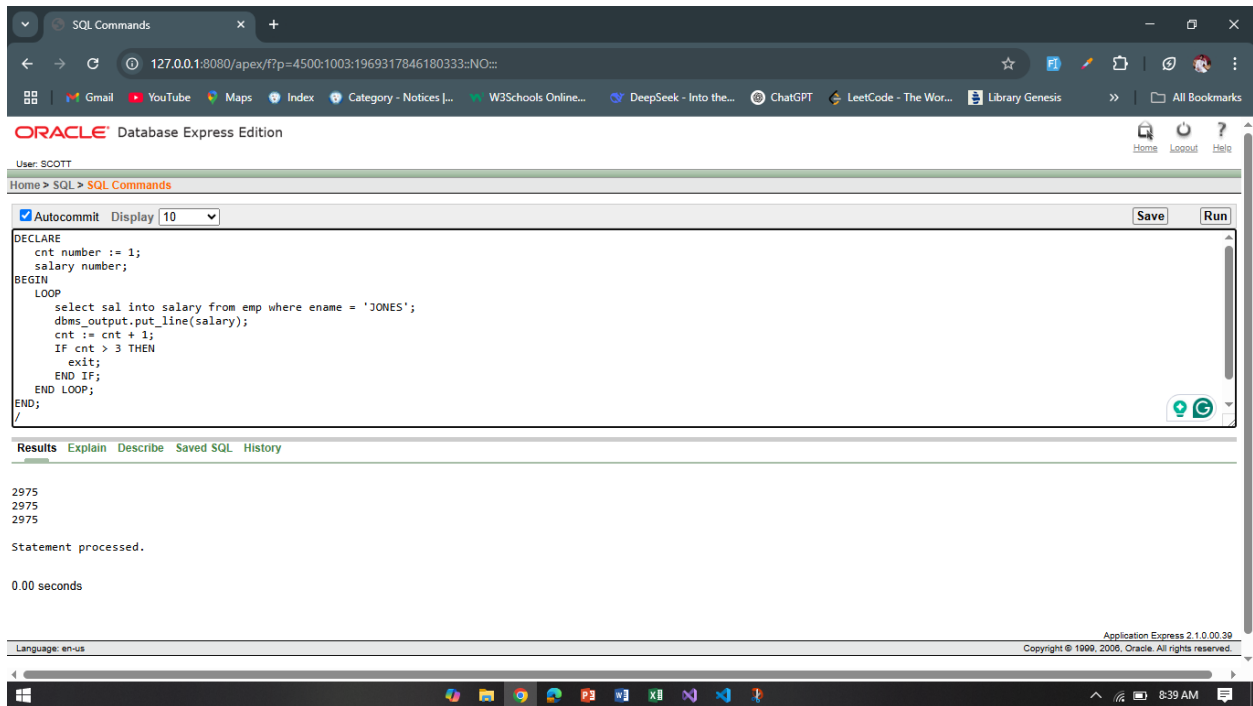
Answer:

```

DECLARE
  cnt number := 1;
  salary number;
BEGIN
  LOOP
    select sal into salary from emp where ename = 'JONES';
    dbms_output.put_line(salary);
    cnt := cnt + 1;
    IF cnt > 3 THEN
      exit;
    END IF;
  END LOOP;
END;
/

```

Output:



12. Write a query that can display the salary of employee JONES three times using while loop.

Answer:

DECLARE

 v_salary NUMBER ;

 counter NUMBER := 1;

BEGIN

 SELECT sal INTO v_salary FROM emp WHERE ename = 'JONES';

 WHILE counter <= 3 LOOP

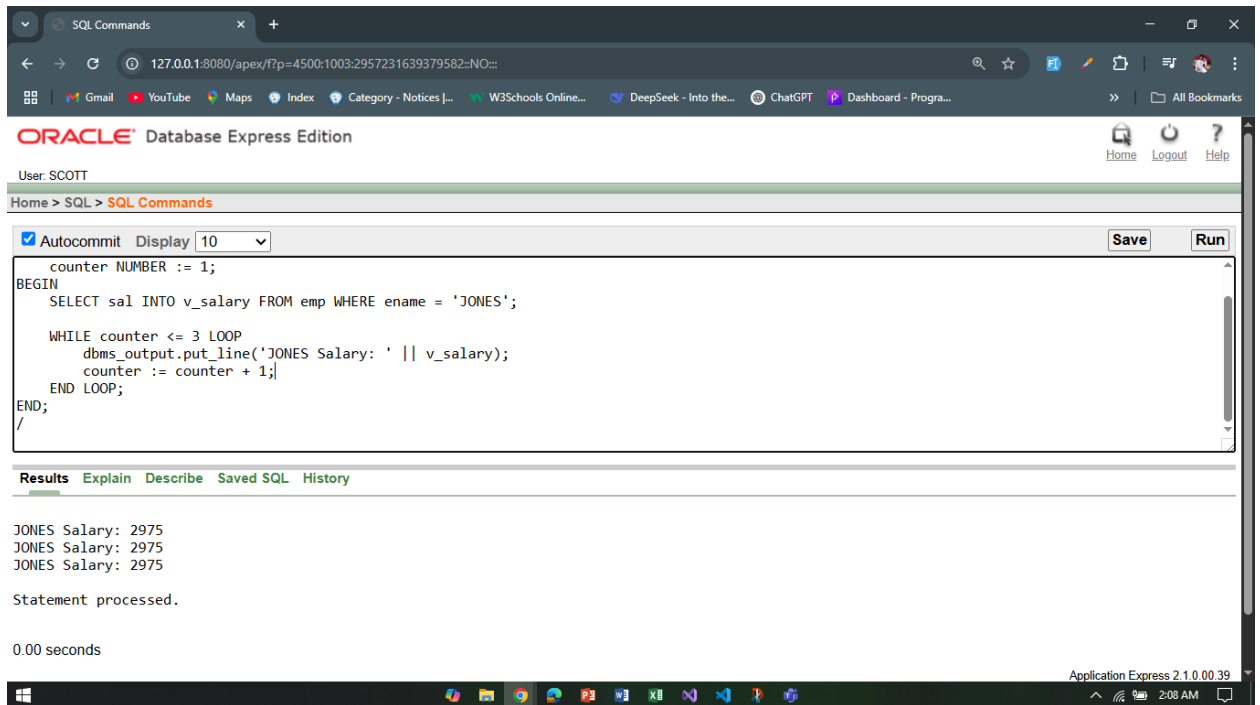
 dbms_output.put_line('JONES Salary: ' || v_salary);

 counter := counter + 1;

 END LOOP;

END;

/



13. Write a query that can display the salary of employee JONES three times using for loop.

Answer:

DECLARE

 v_salary number;

BEGIN

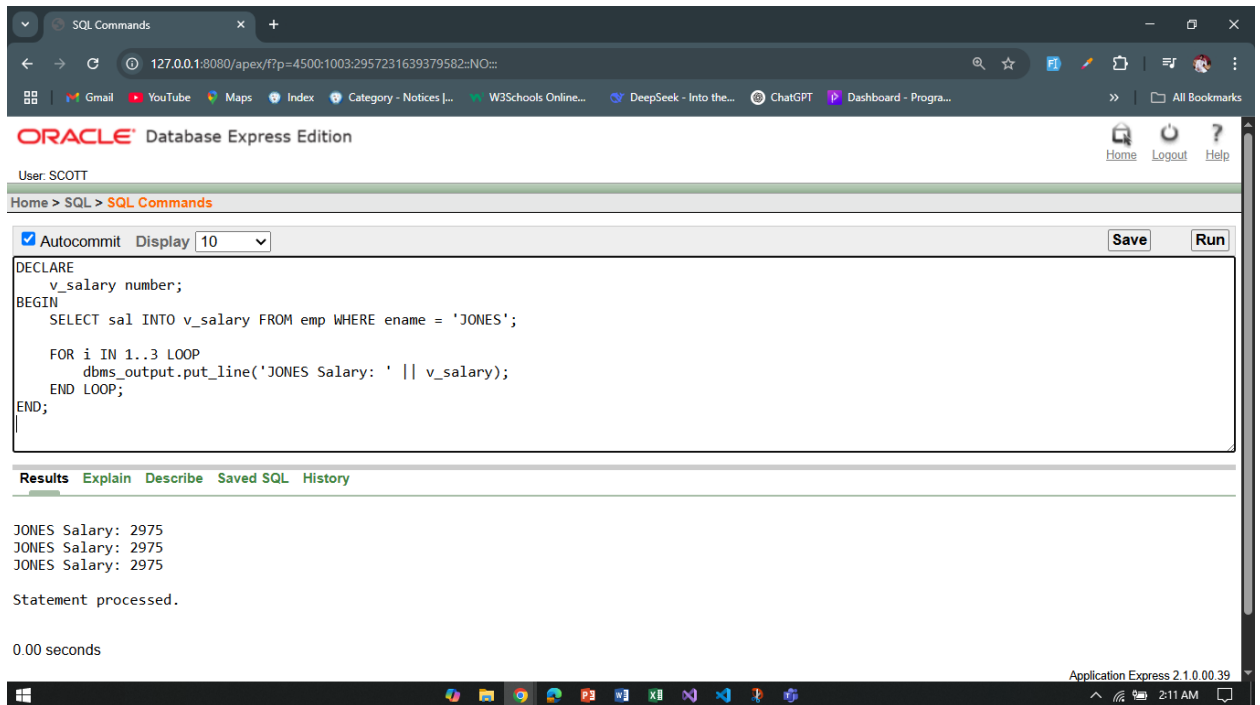
 SELECT sal INTO v_salary FROM emp WHERE ename = 'JONES';

 FOR i IN 1..3 LOOP

 dbms_output.put_line('JONES Salary: ' || v_salary);

 END LOOP;

END;



14. Create a function that returns the total number of departments.

Answer:

```
CREATE OR REPLACE FUNCTION get_total_departments
```

```
RETURN NUMBER
```

```
IS
```

```
    v_total NUMBER;
```

```
BEGIN
```

```
    SELECT COUNT(*) INTO v_total FROM dept;
```

```
    RETURN v_total;
```

```
END;
```

```
/
```

```
DECLARE
```

```
    dept_count NUMBER;
```

```
BEGIN
```

```
    dept_count := get_total_departments;
```

```
    dbms_output.put_line('Total Departments: ' || dept_count);
```

```
END;
```

```
/
```

The screenshot shows the Oracle Database Express Edition interface. The user is SCOTT. The SQL Commands window contains the following code:

```
CREATE OR REPLACE FUNCTION get_total_departments
RETURN NUMBER
IS
    v_total NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_total FROM dept;
    RETURN v_total;
END;
/
```

The code is executed, and the results show:

Function created.

0.00 seconds

Application Express 2.1.0.00.39
Copyright © 1999, 2006, Oracle. All rights reserved.

The screenshot shows the Oracle Database Express Edition interface. The user is SCOTT. The SQL Commands window contains the following code:

```
CREATE OR REPLACE FUNCTION get_total_departments
RETURN NUMBER
IS
    v_total NUMBER;
BEGIN
    SELECT COUNT(*) INTO v_total FROM dept;
    RETURN v_total;
END;
/
```

The code is executed, and the results show:

Total Departments: 4

Statement processed.

0.00 seconds

Application Express 2.1.0.00.39
Copyright © 1999, 2006, Oracle. All rights reserved.

15. Create a procedure to update the salary of employee Allen to 100.

Answer:

```
CREATE OR REPLACE PROCEDURE update_allen_salary IS
BEGIN
    UPDATE emp SET sal = 100 WHERE ename = 'ALLEN';
    COMMIT;
```

```
        dbms_output.put_line('Salary of ALLEN updated to 100.');
```

```
END;
```

```
/
```



```
BEGIN
```

```
    update_allen_salary;
```

```
END;
```

```
/
```

The screenshot displays the Oracle Database Express Edition web interface. The browser address bar shows the URL `127.0.0.1:8080/apex/f?p=4500:1003:2957231639379582::NO::`. The page title is "ORACLE Database Express Edition". The user is logged in as "SCOTT". The breadcrumb navigation shows "Home > SQL > SQL Commands".

In the "SQL Commands" section, the "Autocommit" checkbox is checked, and the "Display" dropdown is set to "10". The SQL command entered is:

```
CREATE OR REPLACE PROCEDURE update_allen_salary IS
BEGIN
    UPDATE emp SET sal = 100 WHERE ename = 'ALLEN';
    COMMIT;
    dbms_output.put_line('Salary of ALLEN updated to 100.');
```

```
END;
```

```
/
```

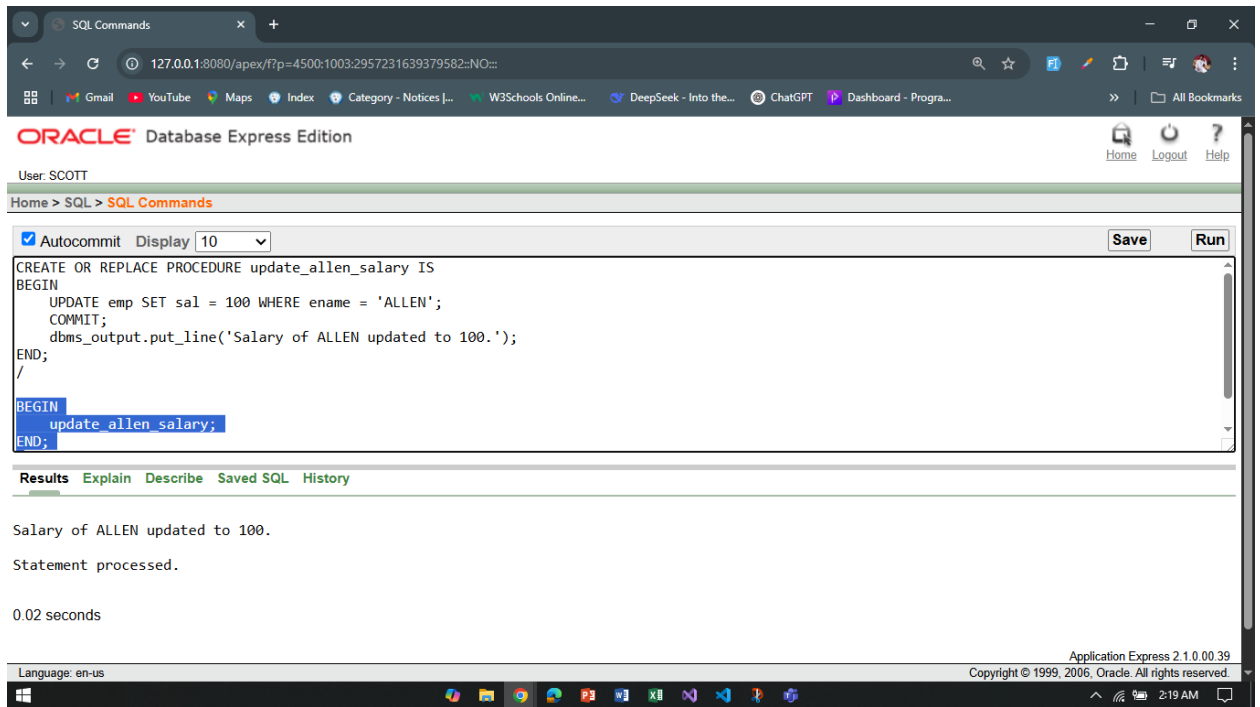
```
BEGIN
```

```
    update_allen_salary;
```

```
END;
```

Buttons for "Save" and "Run" are visible. Below the command editor, the "Results" tab is selected, showing the message "Procedure created." and a duration of "0.00 seconds".

The footer of the interface includes "Language: en-us", "Application Express 2.1.0.00.39", and "Copyright © 1999, 2006, Oracle. All rights reserved." The system clock at the bottom right indicates "2:20 AM".



****After solving the above questions using Oracle 10g, write the PL/SQLs in a MS Word document (Write down the answer and give screenshot of the result of the query. The name of the document MUST be your ID and the PL/SQLs MUST be numbered accordingly) and upload it in the provided link in your VUES account**