****

**American International University-Bangladesh**

**FINALTERM Report**

**Project Title: Ice Cream Parlour Management System**

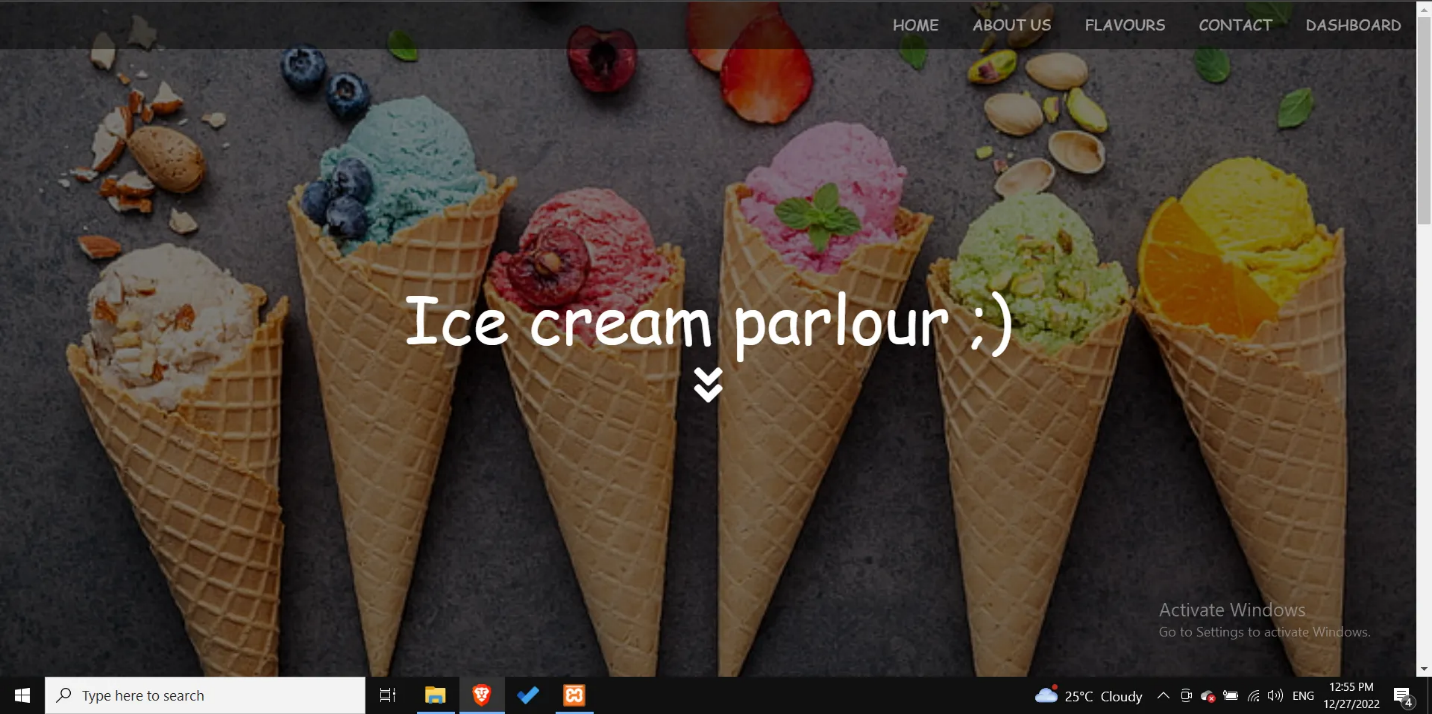
**Course Name: ADVANCE DATABASE MANAGEMENT SYSTEM**

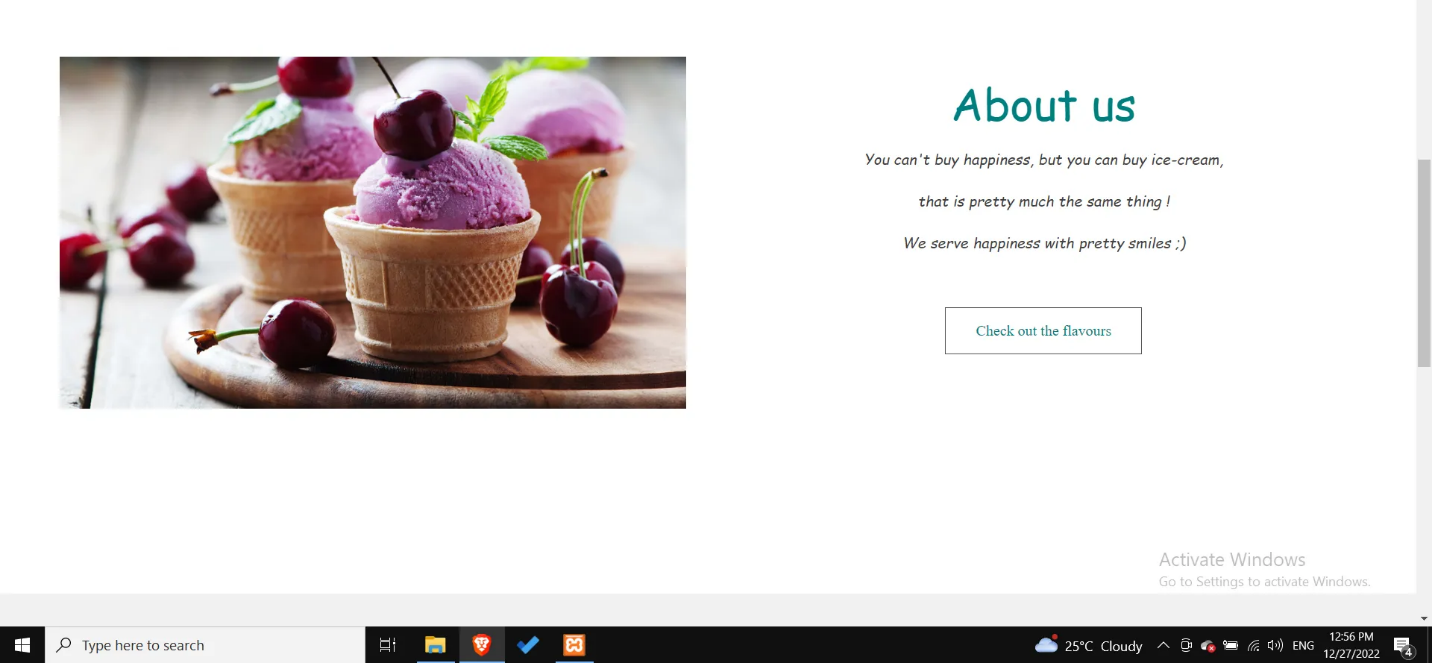
**Course Teacher: Rezwan Ahmed**

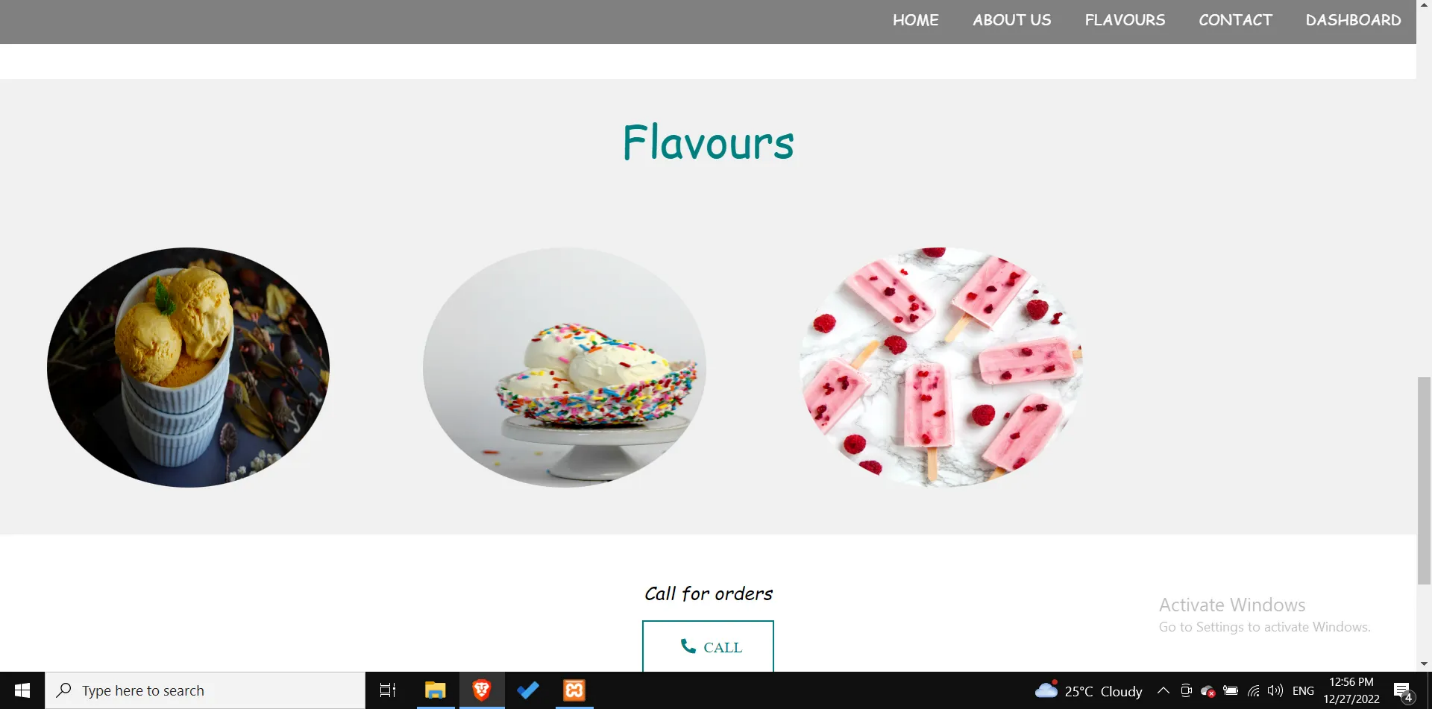
**Semester: Fall 22-23**

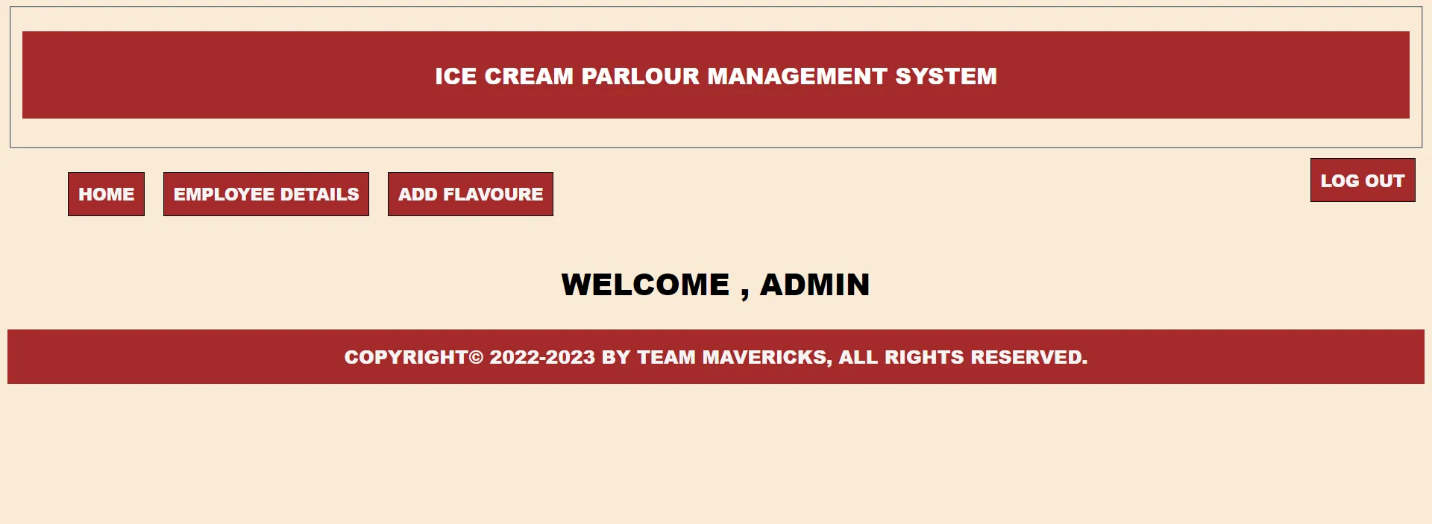
|  |  |  |
| --- | --- | --- |
| **Name** | **Id** | **Section** |
| **API ALAM** | **19-40880-2** | **A** |
| **MD. ABID HASAN MIAD** | **19-40890-2** | **A** |
| **SHANZIDA AHAMMAD SHARNA** | **19-41312-3** | **A** |
| **FARIHA TAHSEEN KARIM** | **20-43289-1** | **A** |

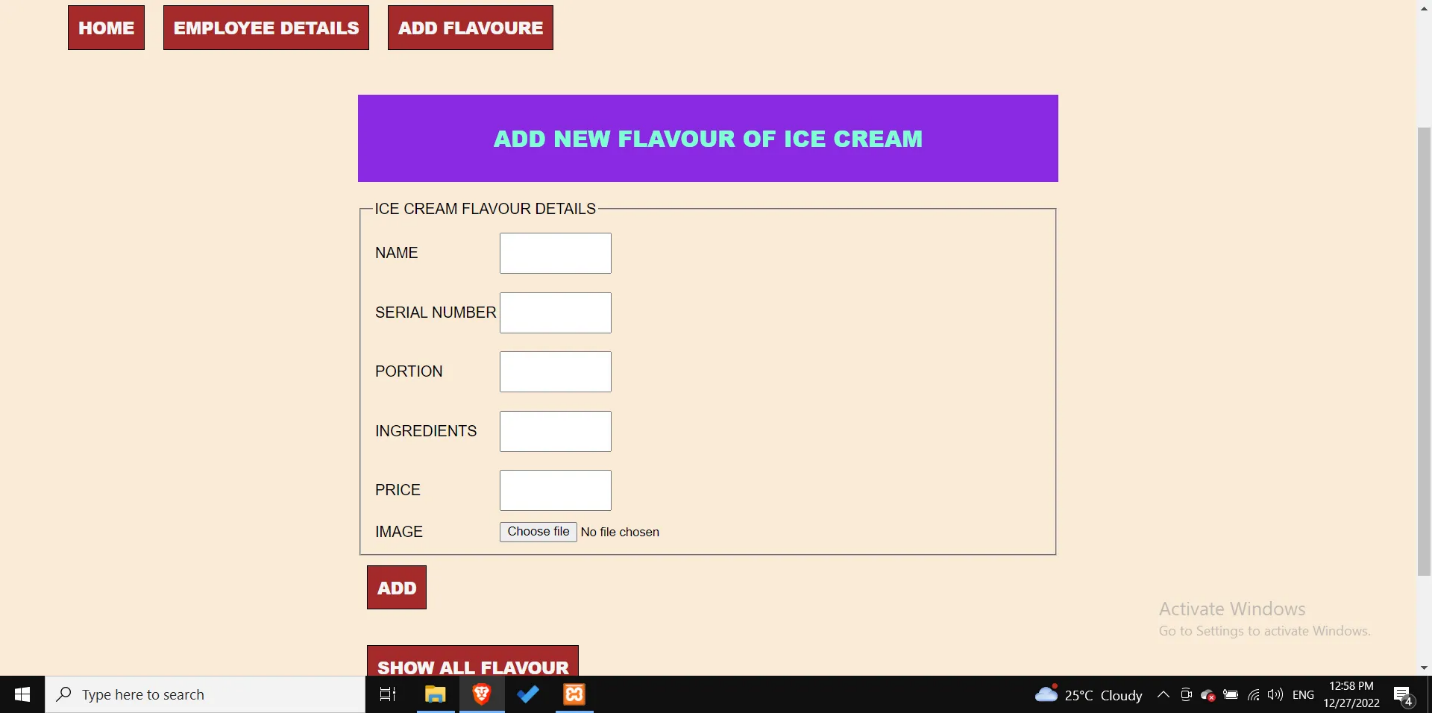
**Admin Panel**

****

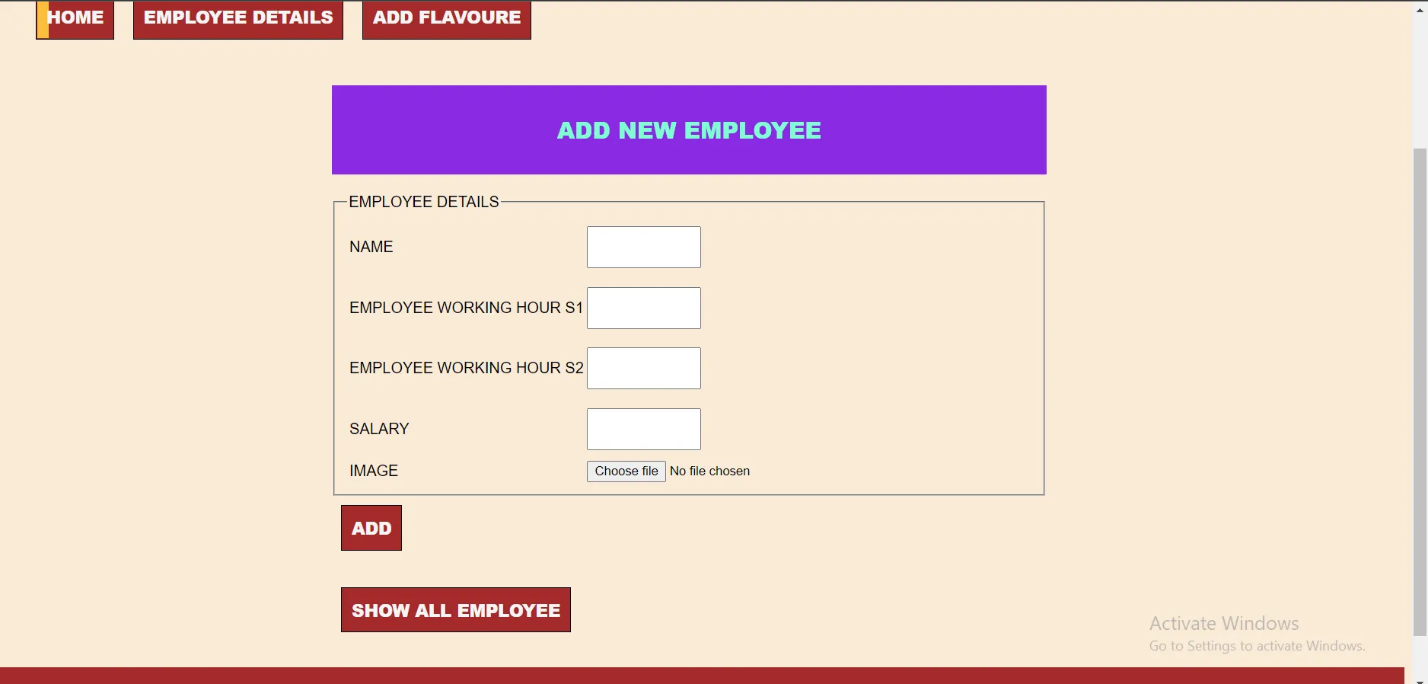
****

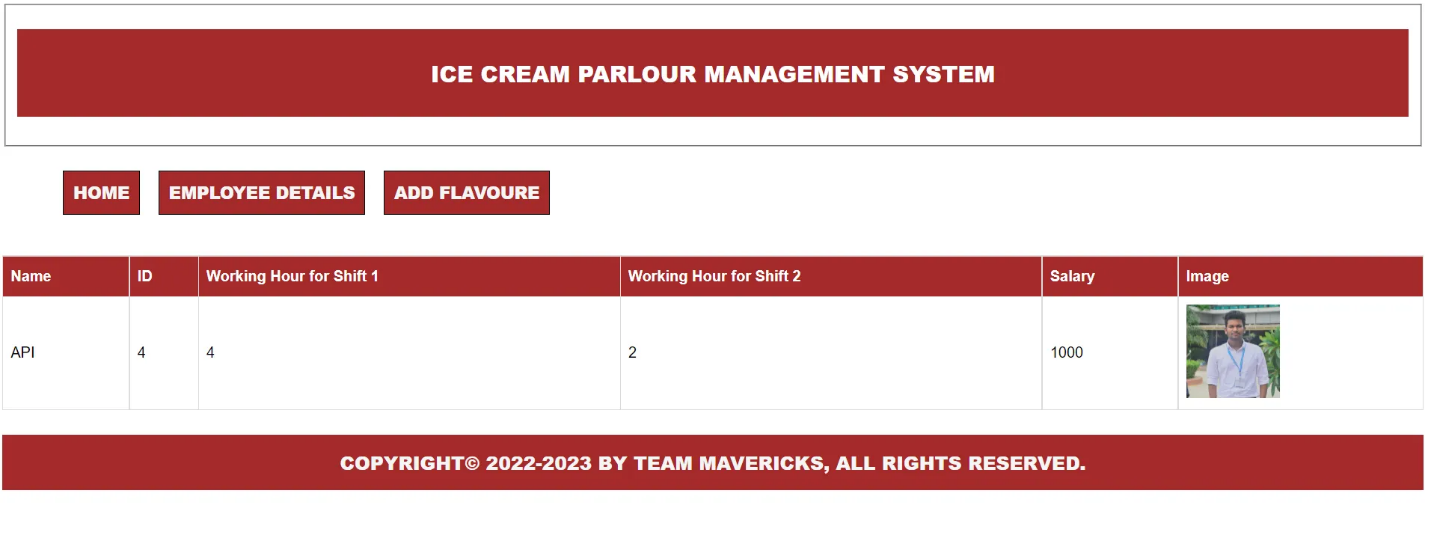
****

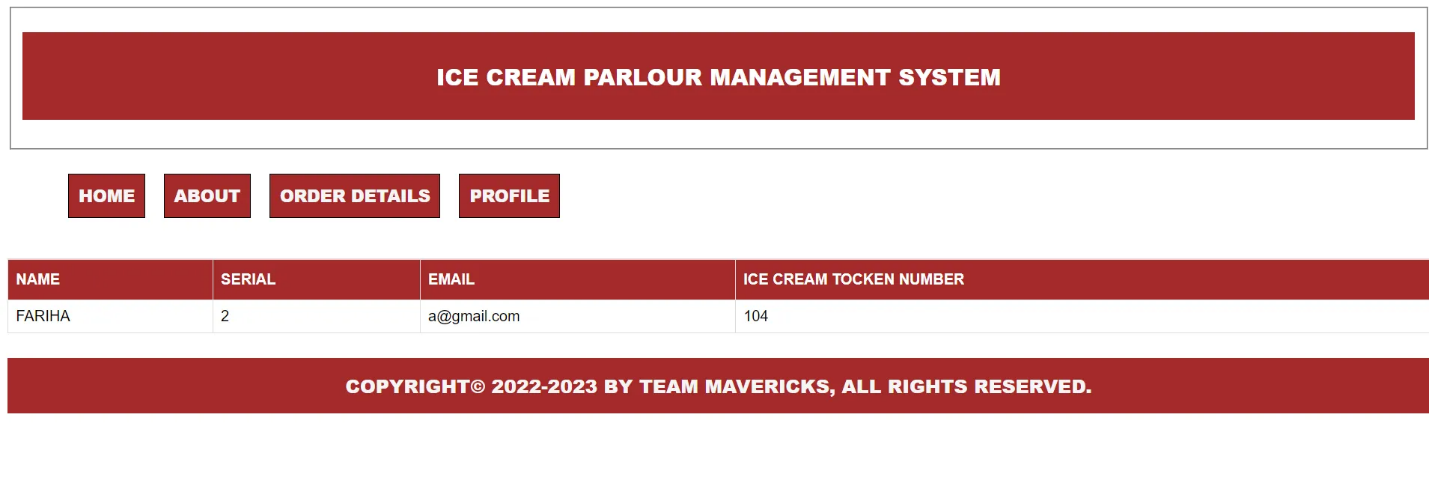
****

****

****

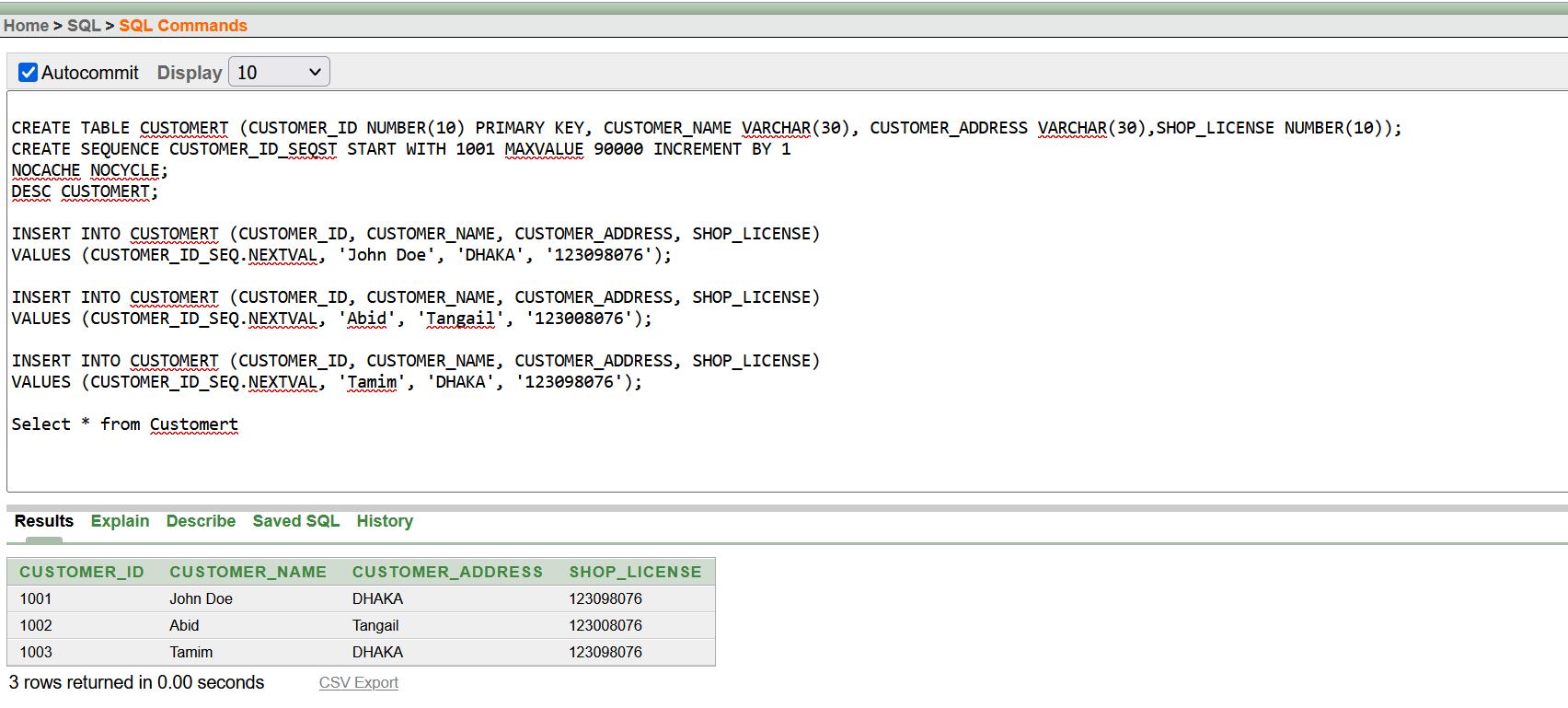
****

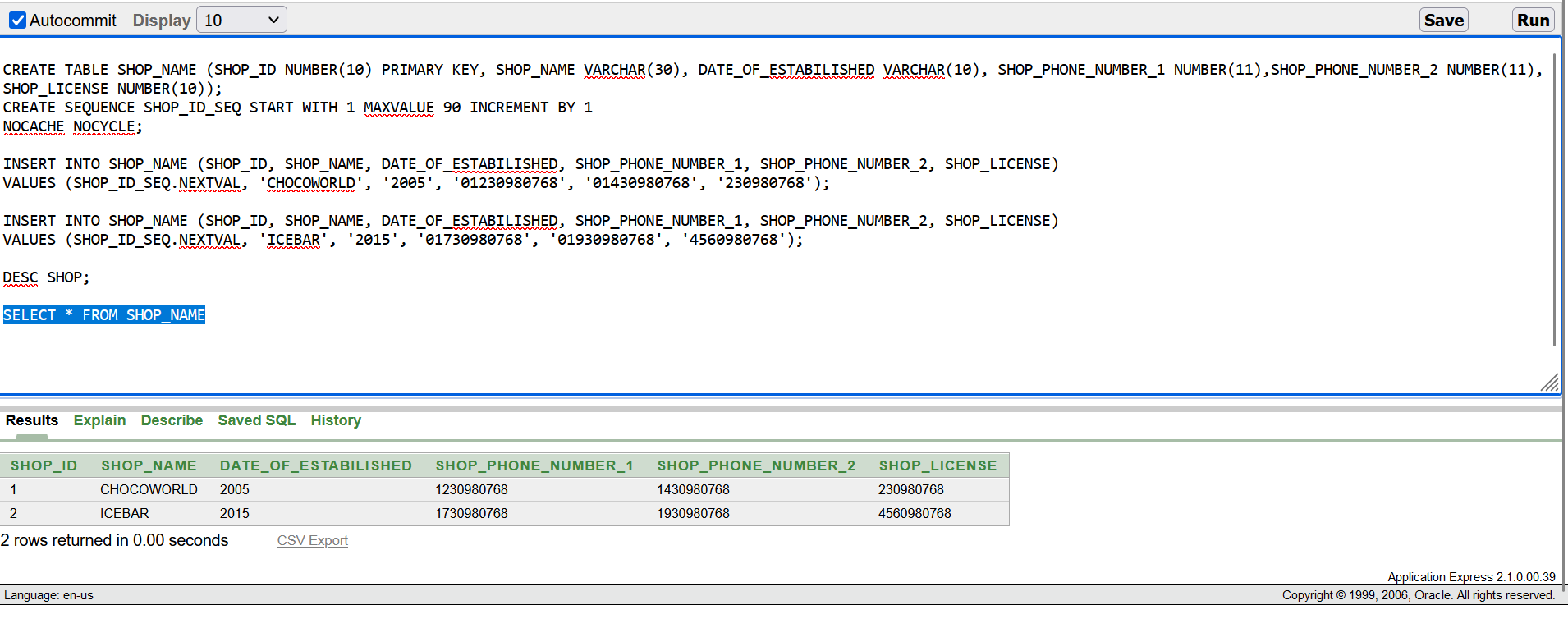
****

****

This is the all pages for our project. An admin can see the table which is on oracle. And also here admin can edit data for update and delete. Also the admin can search for data he/she can search by id.

**Sequence to maintain primary key in 2 tables**

****

****

**Searching**

Search by ID

SELECT \* FROM LOG where USERNAME='$USERNAME'

OR

SELECT SHOP\_ID, SHOP\_NAME, DATE\_OF\_ESTABILISHED, SHOP\_PHONE\_NUMBER\_1, SHOP\_PHONE\_NUMBER\_2, SHOP\_LICENSE FROM Shop WHERE SHOP\_ID=$cid

**Advance Search**

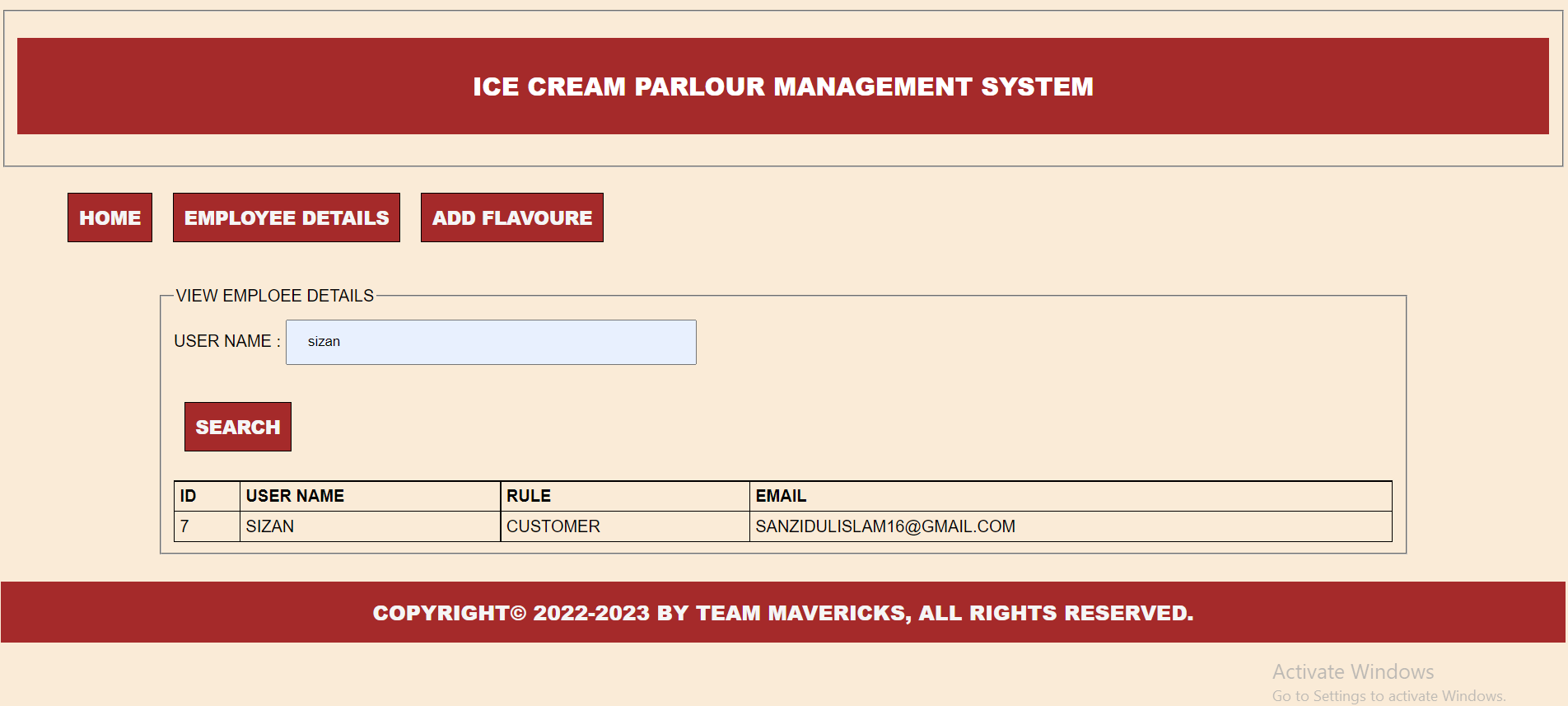
Search for ice cream products by name, price, or flavor

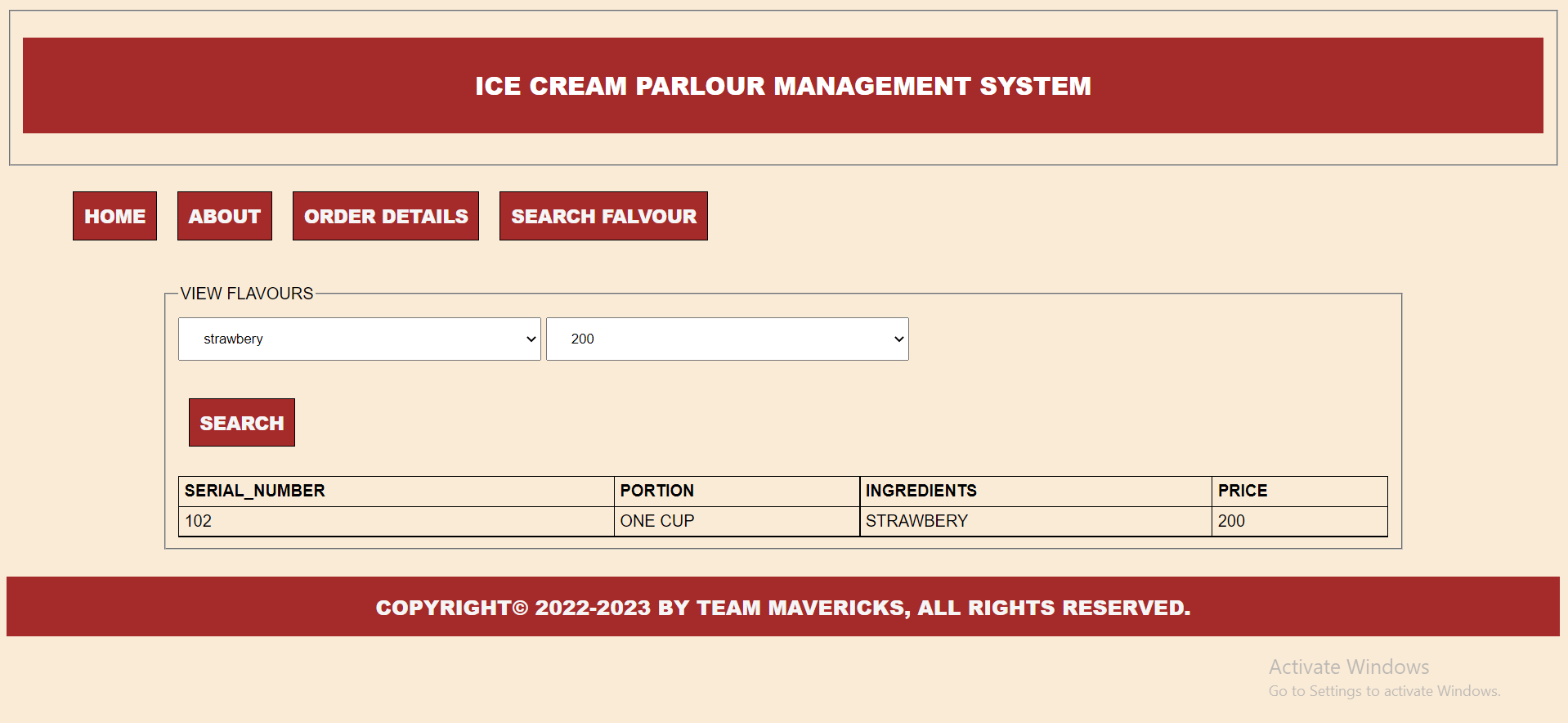
SELECT \* FROM MENUCARD WHERE NAME LIKE '%$NAME%' OR PRICE LIKE '%$PRICE%' OR FLAVOR LIKE '%$FLAVOR%'

OR,

SELECT \* FROM MENUCARD

WHERE NAME LIKE '%SEARCH STRING%' OR PRICE BETWEEN 100 AND 200 OR FLAVOR = 'STRAWBERRY';



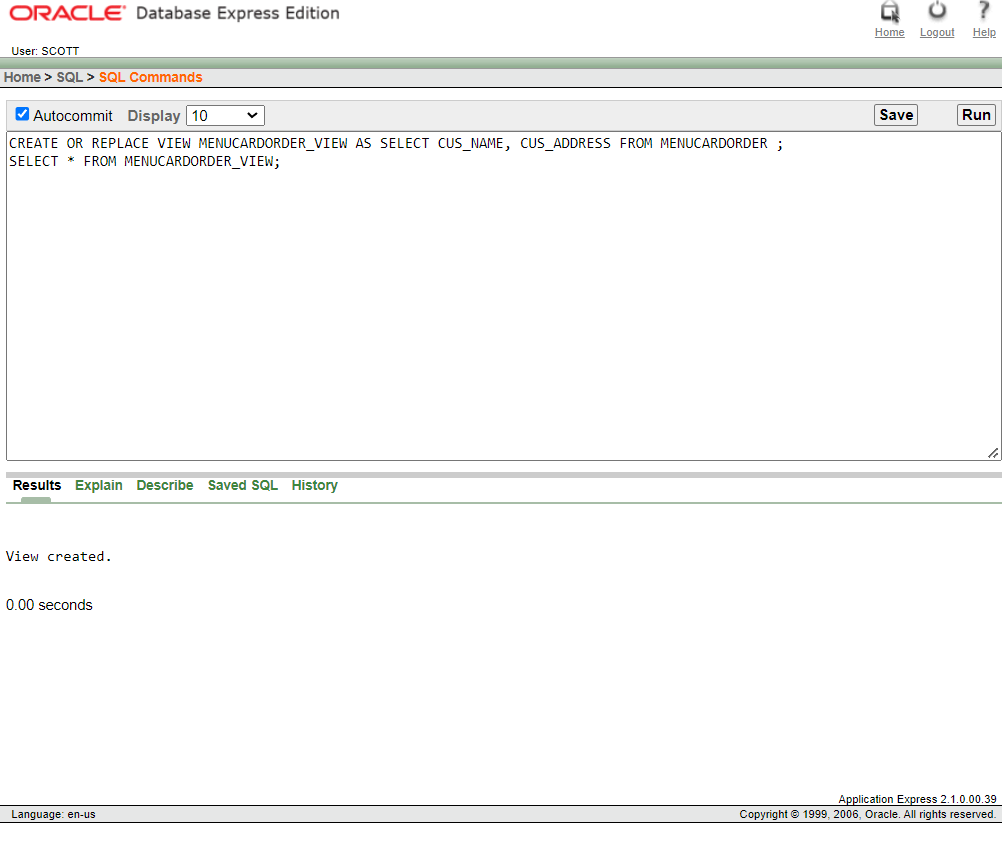


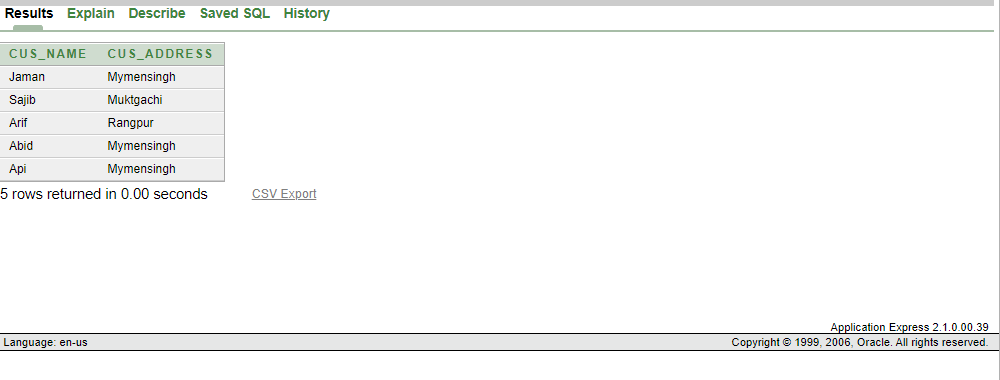
**VIEW**

**01. Create a view called MENUCARDORDER\_VIEW based on CUS\_NAME and CUS\_ADDRESS from the MENUCARDORDER Table**

CREATE OR REPLACE VIEW MENUCARDORDER\_VIEW AS SELECT CUS\_NAME, CUS\_ADDRESS FROM MENUCARDORDER ;

SELECT \* FROM MENUCARDORDER\_VIEW;

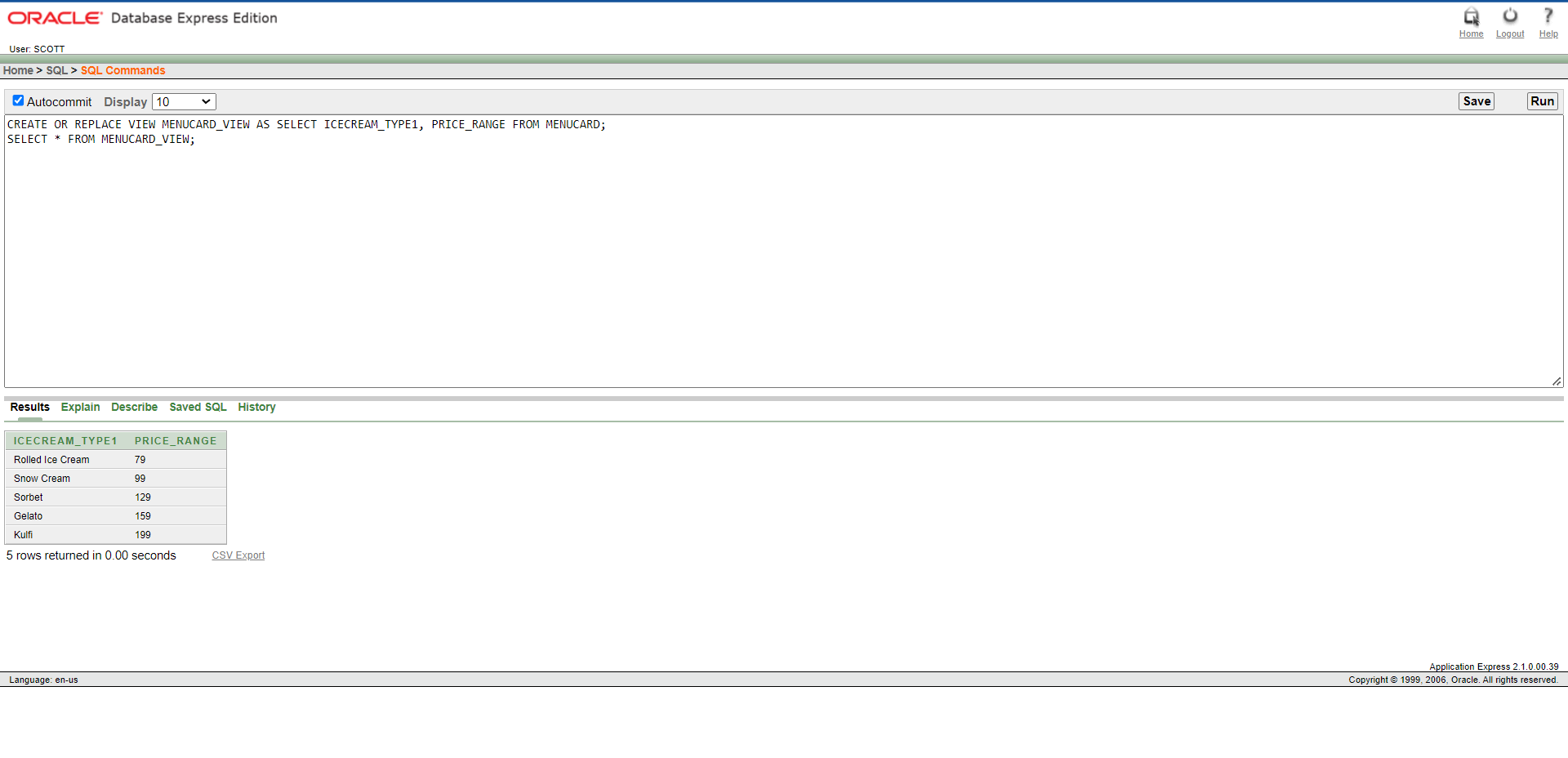




**02. Create a view called MENUCARD\_VIEW based on ICECREAM\_TYPE1and PRICE\_RANGE from the MenuCard Table**

CREATE OR REPLACE VIEW MENUCARD\_VIEW AS SELECT ICECREAM\_TYPE1, PRICE\_RANGE FROM MENUCARD;

SELECT \* FROM MENUCARD\_VIEW;

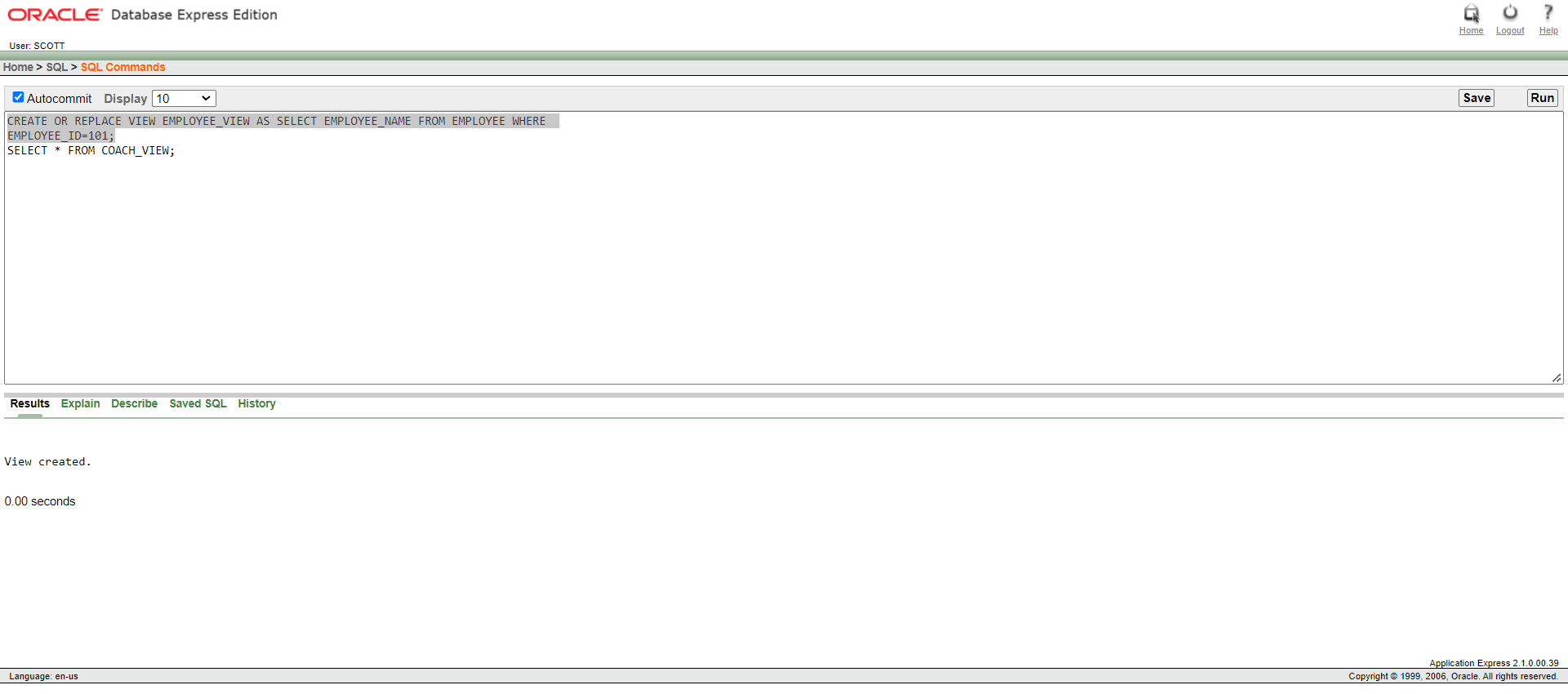


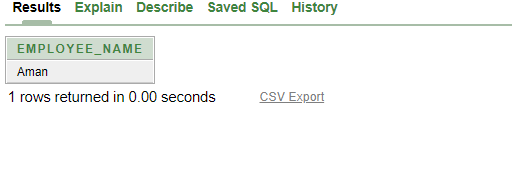
**03. Create a view to show the employee name ,employee id 101 from Employee Table.**

CREATE OR REPLACE VIEW EMPLOYEE\_VIEW AS SELECT EMPLOYEE\_NAME FROM EMPLOYEE WHERE

EMPLOYEE\_ID=101;

SELECT \* FROM EMPLOYEE\_VIEW;

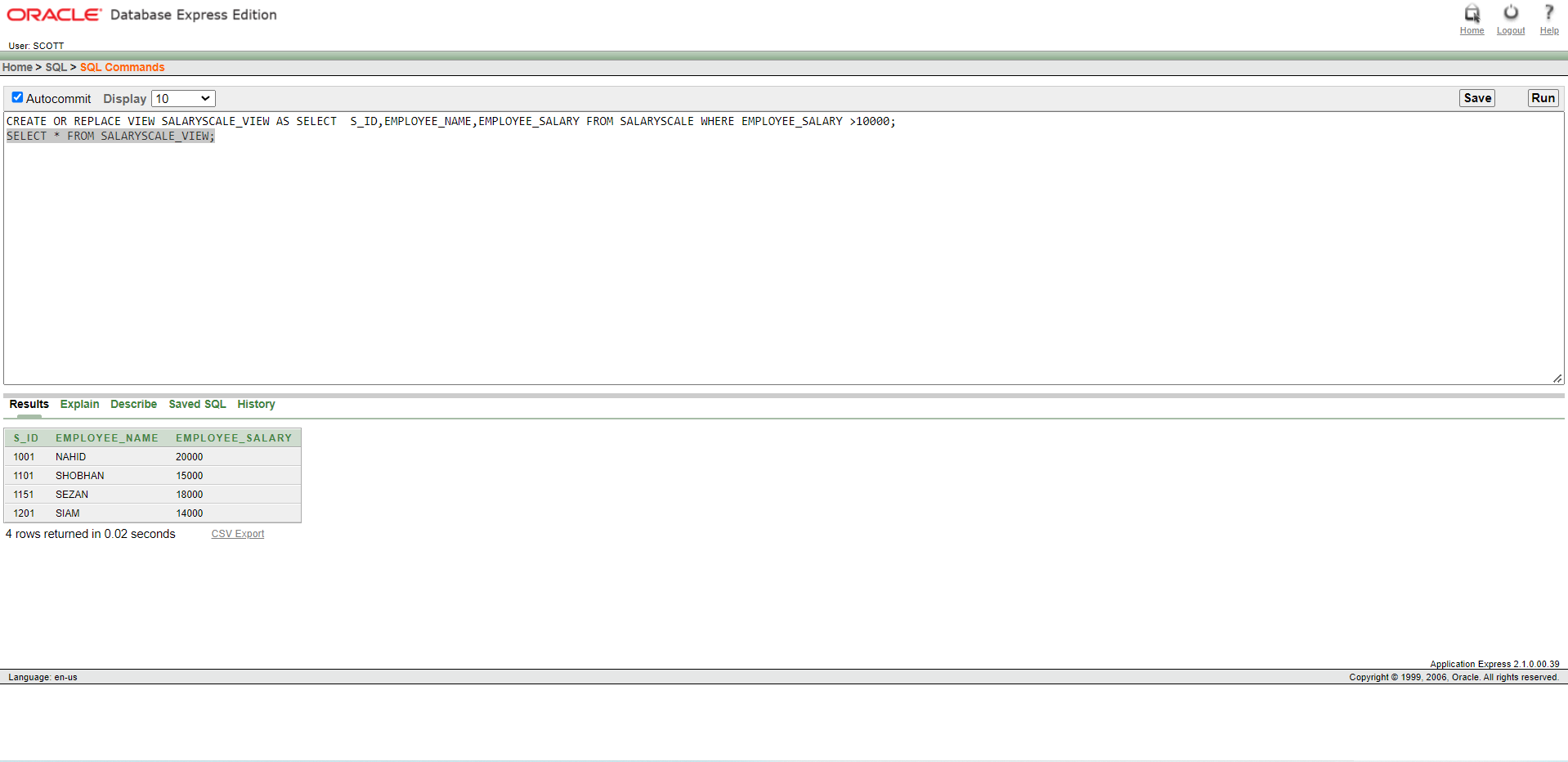




**04. Create a view to show the Name, ID and Salary of Salary Scale table who got salary more than 10000.**

CREATE OR REPLACE VIEW SALARYSCALE\_VIEW AS SELECT S\_ID,EMPLOYEE\_NAME,EMPLOYEE\_SALARY FROM SALARYSCALE WHERE EMPLOYEE\_SALARY >10000;

SELECT \* FROM SALARYSCALE\_VIEW;



**05. Create a View for Display the Shop all information along with their respective EMAIL.**

CREATE OR REPLACE VIEW OWNER\_SHOP\_VIEW AS SELECT C.\*,O.EMAIL

FROM SHOP C,OWNER O WHERE C.SHOP\_ID=O.SHOP\_ID;

SELECT \* FROM OWNER\_SHOP\_VIEW;



**PROCEDURE**

1. **Create a procedure to update the salary of EMPLOYEE API to 25000.**

DECLARE

Employee\_Salary1 number;

PROCEDURE UpadateSal(x IN OUT number)

IS

BEGIN x :=25000;

UPDATE SALARAYSCALE

SET Employee\_Salary=x

WHERE ENAME='API';

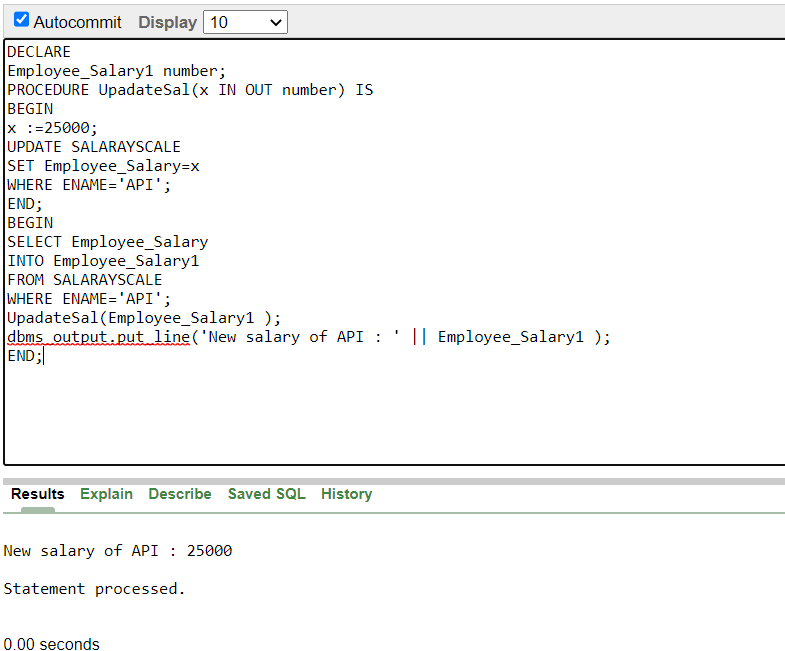
END;

BEGIN

SELECT Employee\_Salary INTO Employee\_Salary1 FROM SALARAYSCALE WHERE ENAME='API'; UpadateSal(Employee\_Salary1 );

dbms\_output.put\_line('New salary of API : ' || Employee\_Salary1 );

END;



1. **Create a procedure that takes two numbers using the IN mode(salary of API and AMAN) and returns their maximum salary using the OUT parameters FROM SALARYSCALE TABLE.**

DECLARE

a number;

b number;

c number;

PROCEDURE findMax(x IN number, y IN number, z OUT number)

IS

BEGIN IF x > y THEN z:= x;

ELSE z:= y;

END IF;

END;

BEGIN SELECT EMPLOYEE\_SALARY INTO a FROM SALARAYSCALE

WHERE ENAME='API';

SELECT EMPLOYEE\_SALARY INTO b FROM SALARAYSCALE WHERE ENAME='AMAN';

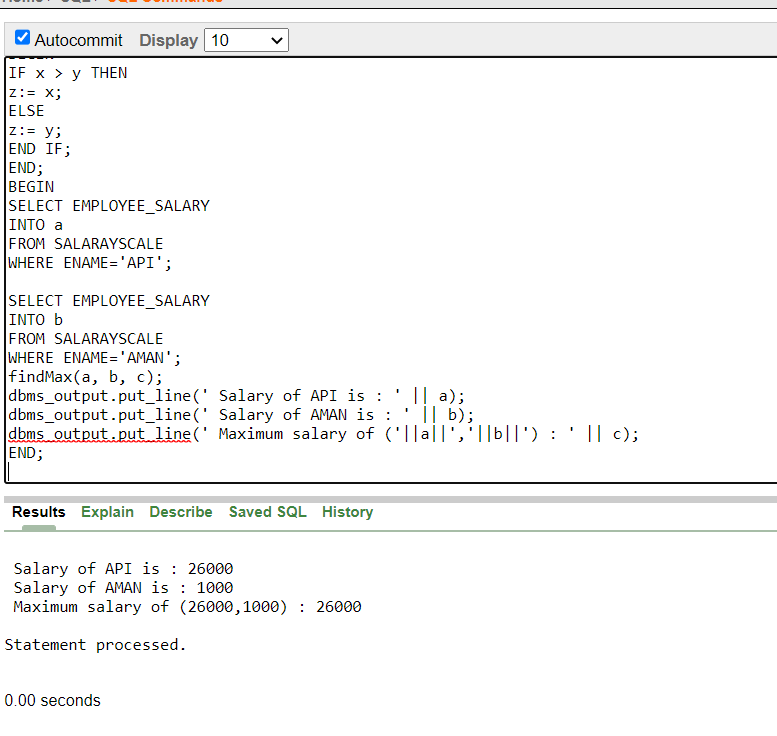
findMax(a, b, c);

dbms\_output.put\_line(' Salary of API is : ' || a);

dbms\_output.put\_line(' Salary of AMAN is : ' || b);

dbms\_output.put\_line(' Maximum salary of ('||a||','||b||') : ' || c);

END;



1. **Create a procedure to update the price to tk400 of vanilla flavored ice-cream in the menuecard**

CREATE OR REPLACE PROCEDURE Up(p IN MENUCARD.INAME%TYPE)

IS

BEGIN UPDATE MENUCARD SET PRICE=400

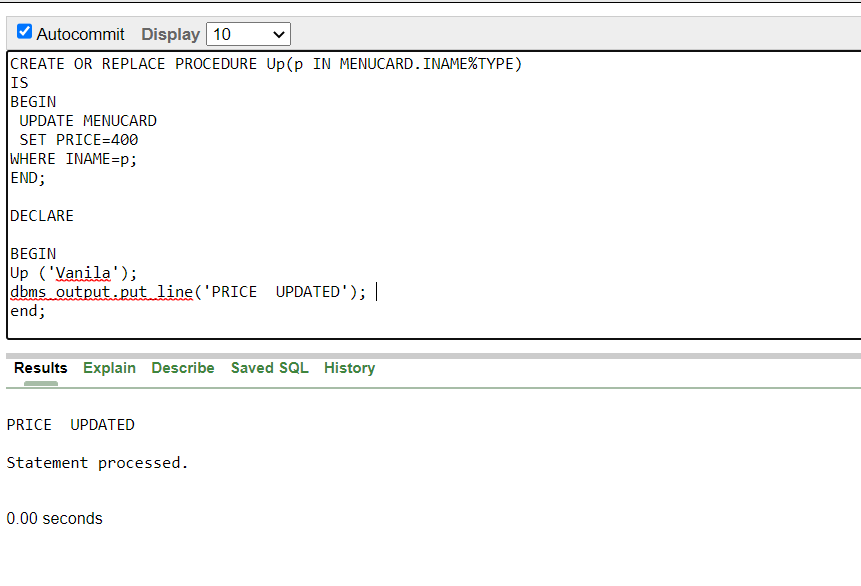
WHERE INAME=p;

END;

DECLARE BEGIN Up ('Vanila');

dbms\_output.put\_line('PRICE UPDATED');

end;



**Function**

1. **Find total number of EMPLOYEE**

CREATE OR REPLACE FUNCTION totalEmployee

RETURN number AS total number(2) := 0;

BEGIN SELECT count(\*) into total FROM SALARAYSCALE;

RETURN total;

END;

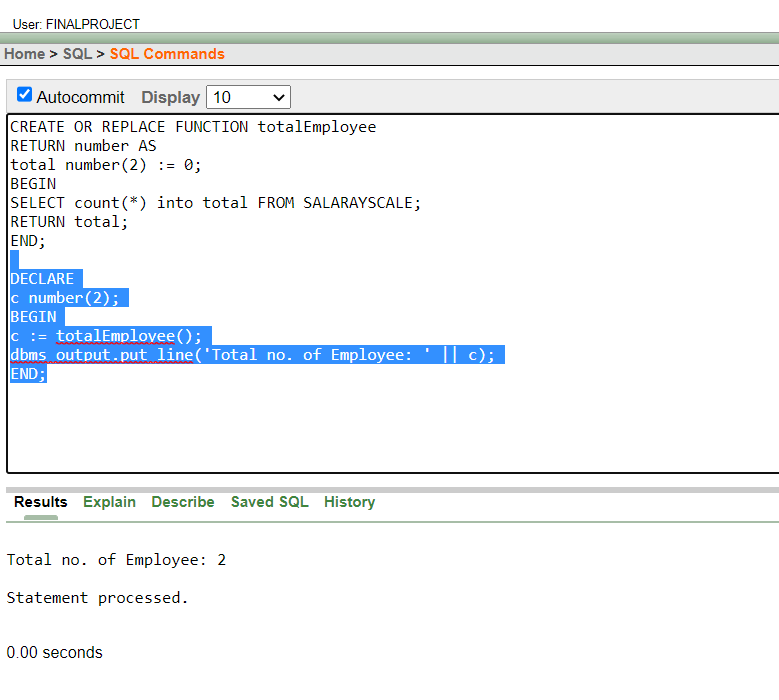
DECLARE

c number(2);

BEGIN c := totalEmployee();

dbms\_output.put\_line('Total no. of Employee: ' || c);

END;



1. **Display minimum salary FROM SALARYSCALE**

CREATE OR REPLACE FUNCTION Minimumsalary

RETURN number

AS

minimum number(20) := 0;

BEGIN

SELECT MIN(EMPLOYEE\_SALARY) into minimum FROM SALARAYSCALE;

RETURN minimum;

END;

DECLARE

c number(20);

BEGIN

c := Minimumsalary ();

dbms\_output.put\_line('Minimum Salary: ' || c);

END;



1. **Display the owner name whose owner\_id is 1151**

CREATE OR REPLACE FUNCTION FindOwner

RETURN OWNER.OWNER\_name%TYPE AS

Mini OWNER.OWNER\_name%TYPE;

BEGIN

SELECT OWNER\_name into mini FROM OWNER where Owner\_id= 1151;

RETURN mini;

END;

DECLARE

c varchar(20);

BEGIN

c := FindOwner();

dbms\_output.put\_line('Owner Name: ' || c);

END;



**TRIGGER**

1. **Create a trigger if insert in manager will print added.**

CREATE OR REPLACE TRIGGER MANAGER\_T

AFTER INSERT ON MANAGER

FOR EACH ROW

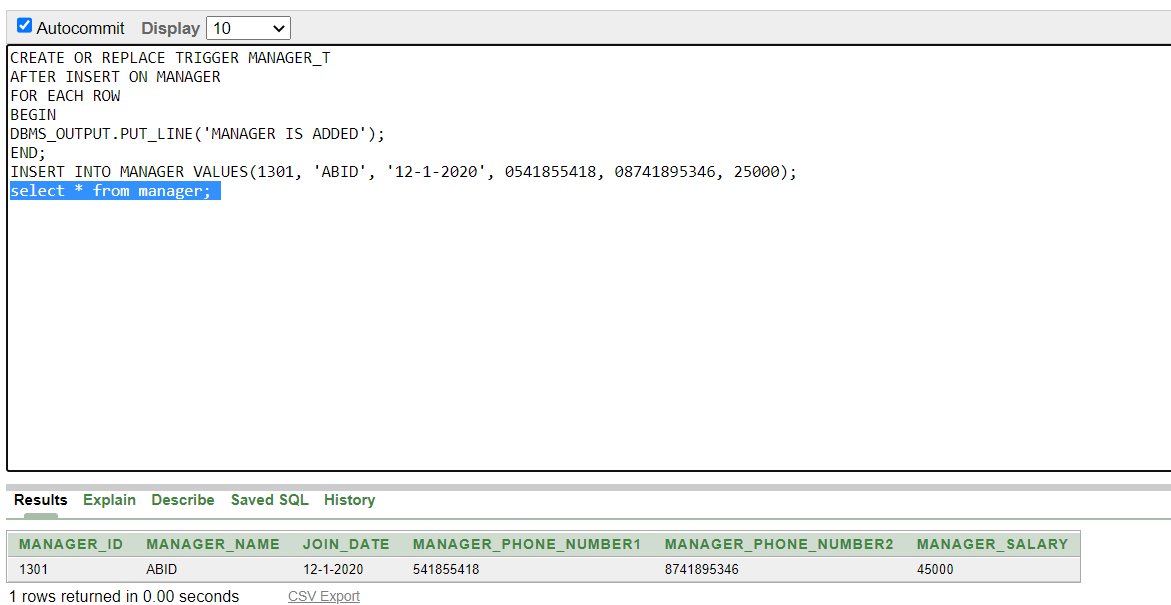
BEGIN

DBMS\_OUTPUT.PUT\_LINE('MANAGER IS ADDED');

END;

INSERT INTO MANAGER VALUES(1301, 'ABID', '12-1-2020', 0541855418, 08741895346, 25000);

select \* from manager;



1. **Create a trigger if update in manager will print updated.**

CREATE OR REPLACE TRIGGER MANAGER\_T

AFTER UPDATE ON MANAGER

FOR EACH ROW

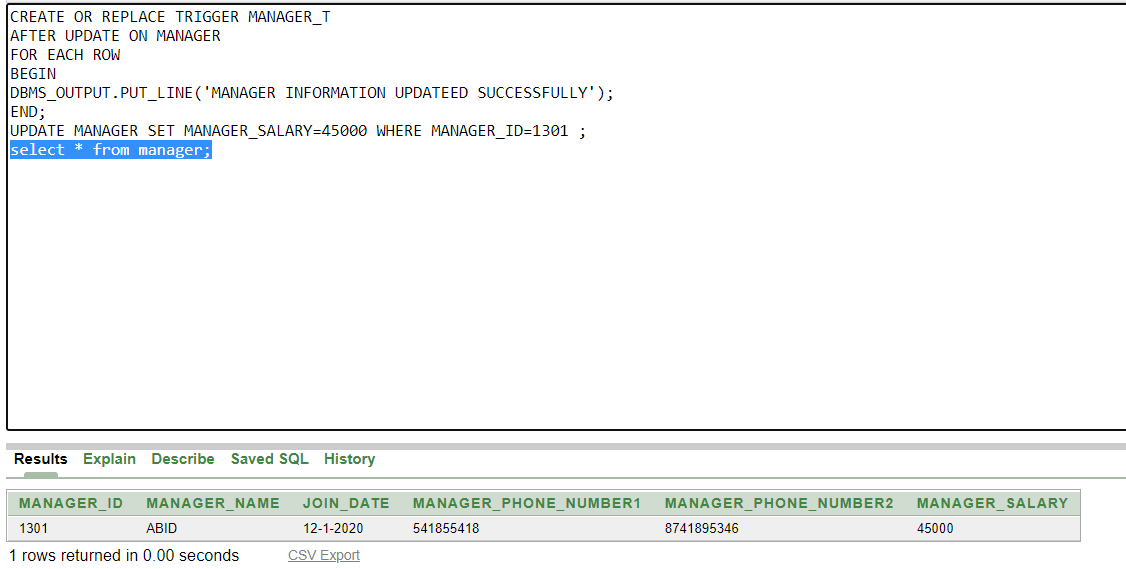
BEGIN

DBMS\_OUTPUT.PUT\_LINE('MANAGER INFORMATION UPDATEED SUCCESSFULLY');

END;

UPDATE MANAGER SET MANAGER\_SALARY=45000 WHERE MANAGER\_ID=1301 ;

select \* from manager;



1. **Create a trigger which shows difference of salary when update new salary in manager table**

CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE UPDATE ON manager

FOR EACH ROW

WHEN (NEW.MANAGER\_ID > 0)

DECLARE

MANAGER\_SALARY\_diff number;

BEGIN

MANAGER\_SALARY\_diff := :NEW.MANAGER\_SALARY - :OLD.MANAGER\_SALARY;

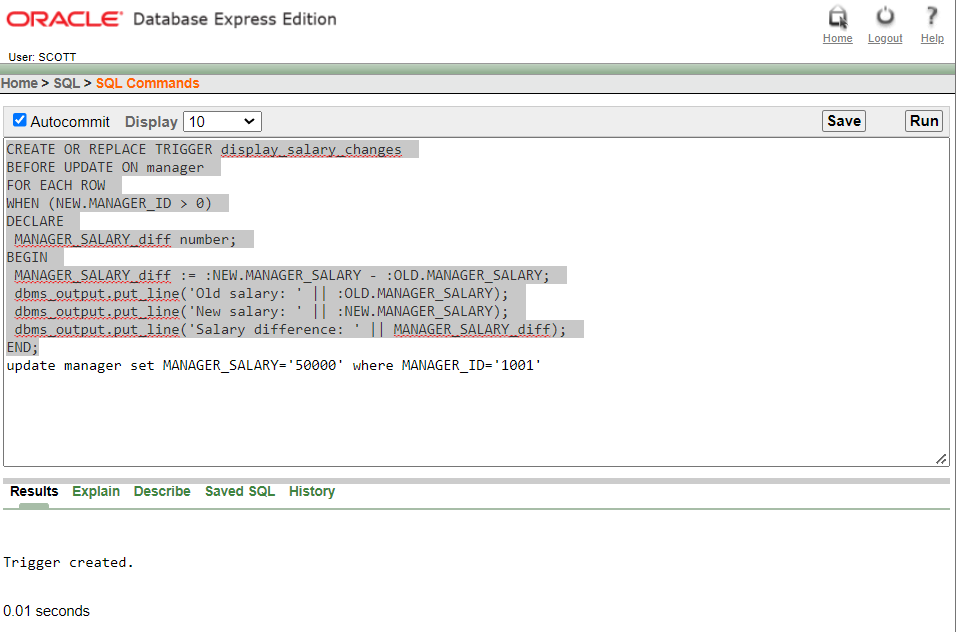
dbms\_output.put\_line('Old salary: ' || :OLD.MANAGER\_SALARY);

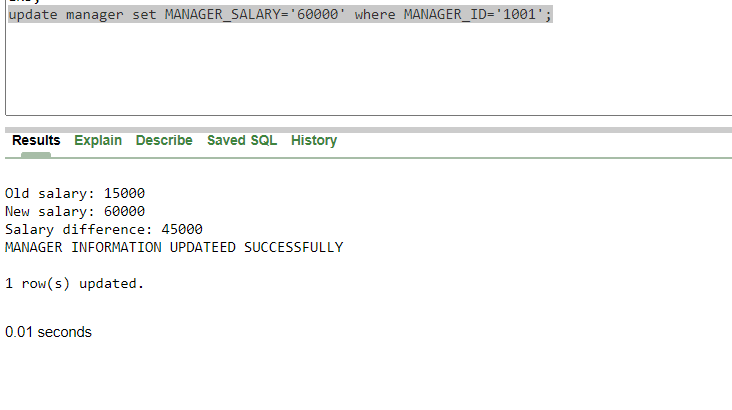
dbms\_output.put\_line('New salary: ' || :NEW.MANAGER\_SALARY);

dbms\_output.put\_line('Salary difference: ' || MANAGER\_SALARY\_diff);

END;

update manager set MANAGER\_SALARY='60000' where MANAGER\_ID='1001';





1. **Delete display\_salary\_changes.**

CREATE OR REPLACE TRIGGER display\_salary\_changes

BEFORE UPDATE OR DELETE ON manager

FOR EACH ROW

WHEN (NEW.MANAGER\_ID > 0)

DECLARE

MANAGER\_SALARY\_diff number;

BEGIN

MANAGER\_SALARY\_diff := :NEW.MANAGER\_SALARY - :OLD.MANAGER\_SALARY;

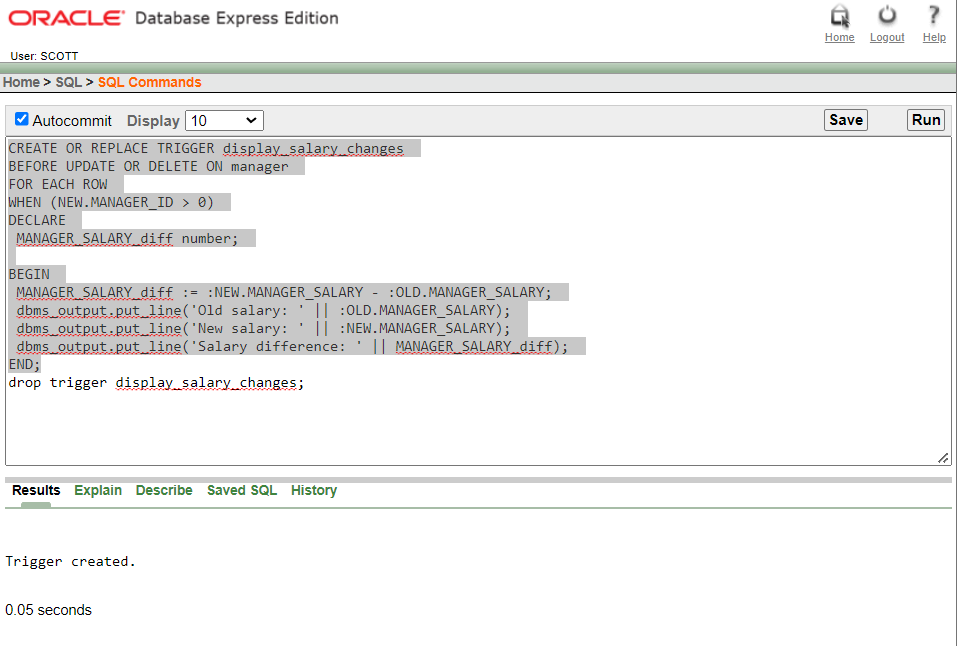
dbms\_output.put\_line('Old salary: ' || :OLD.MANAGER\_SALARY);

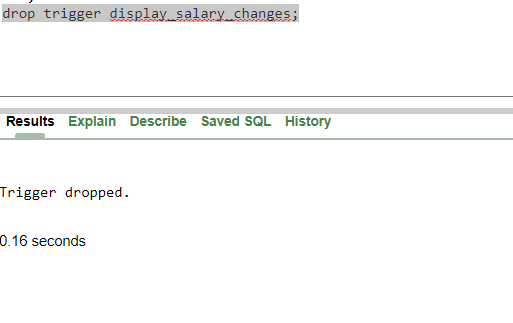
dbms\_output.put\_line('New salary: ' || :NEW.MANAGER\_SALARY);

dbms\_output.put\_line('Salary difference: ' || MANAGER\_SALARY\_diff);

END;

drop trigger display\_salary\_changes;





**PACKAGE**

1. **Create a package that contains a procedure which can display the Manager name whose id is passed as its parameter:**

CREATE OR REPLACE PACKAGE Managername\_pack AS

PROCEDURE display\_name(bid MANAGER.manager\_id%type);

END Managername\_pack;

CREATE OR REPLACE PACKAGE BODY Managername\_pack AS

PROCEDURE display\_name(bid MANAGER.manager\_id%type) IS

bname MANAGER.manager\_name%type;

BEGIN

SELECT manager\_name INTO bname

FROM MANAGER

WHERE manager\_id = bid;

DBMS\_OUTPUT.PUT\_LINE('Manager name is: '|| bname);

END display\_name;

END Managername\_pack ;

BEGIN

Managername\_pack.display\_name('1001');

END;

