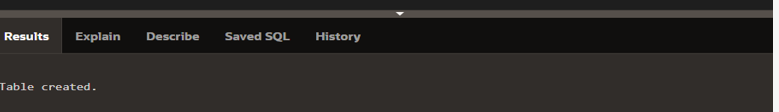
**Section11**

CREATE TABLE employees (

employee\_id NUMBER

);



BEGIN

INSERT INTO employees (employee\_id) VALUES (1);

INSERT INTO employees (employee\_id) VALUES (2);

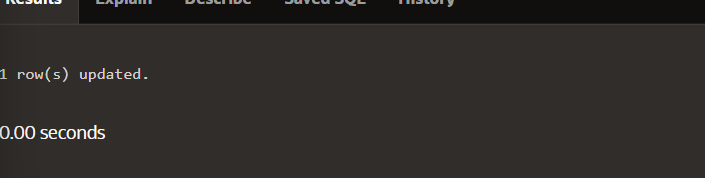
INSERT INTO employees (employee\_id) VALUES (3);

INSERT INTO employees (employee\_id) VALUES (4);

INSERT INTO employees (employee\_id) VALUES (5);

COMMIT;

END;



CREATE OR REPLACE PACKAGE curs\_pkg IS

CURSOR emp\_curs IS SELECT employee\_id FROM employees

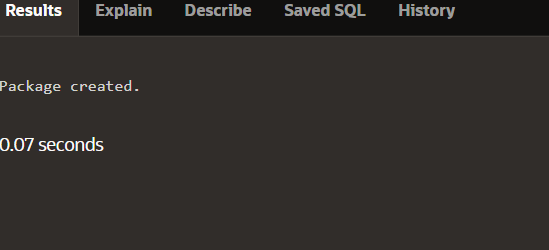
ORDER BY employee\_id;

PROCEDURE open\_curs;

FUNCTION fetch\_n\_rows(n NUMBER := 1) RETURN BOOLEAN;

PROCEDURE close\_curs;

END curs\_pkg;



CREATE OR REPLACE PACKAGE BODY curs\_pkg IS

PROCEDURE open\_curs IS

BEGIN

IF NOT emp\_curs%ISOPEN THEN OPEN emp\_curs; END IF;

END open\_curs;

FUNCTION fetch\_n\_rows(n NUMBER := 1) RETURN BOOLEAN IS

emp\_id employees.employee\_id%TYPE;

BEGIN

FOR count IN 1 .. n LOOP

FETCH emp\_curs INTO emp\_id;

EXIT WHEN emp\_curs%NOTFOUND;

DBMS\_OUTPUT.PUT\_LINE('Id: ' ||(emp\_id));

END LOOP;

RETURN emp\_curs%FOUND;

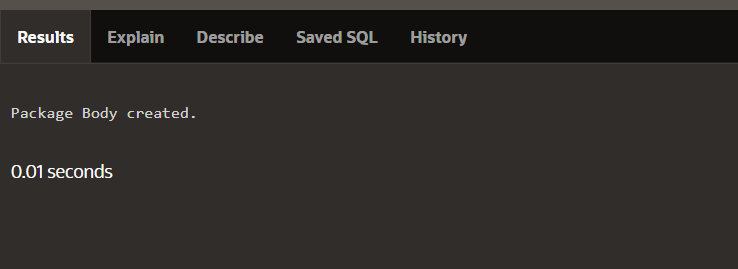
END fetch\_n\_rows;

PROCEDURE close\_curs IS BEGIN

IF emp\_curs%ISOPEN THEN CLOSE emp\_curs; END IF;

END close\_curs;

END curs\_pkg;



DECLARE

v\_more\_rows\_exist BOOLEAN := TRUE;

BEGIN

curs\_pkg.open\_curs; --1

LOOP

v\_more\_rows\_exist := curs\_pkg.fetch\_n\_rows(3); --2

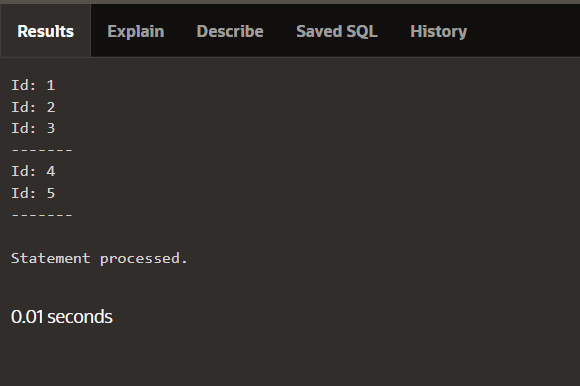
DBMS\_OUTPUT.PUT\_LINE('-------');

EXIT WHEN NOT v\_more\_rows\_exist;

END LOOP;

curs\_pkg.close\_curs; --3

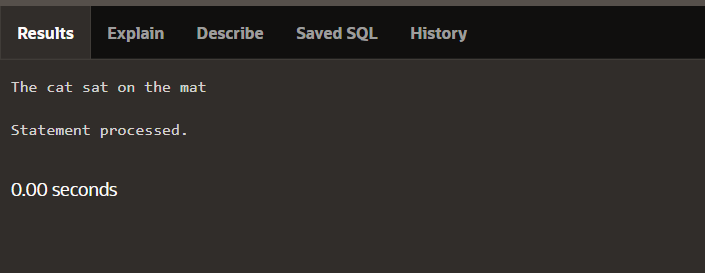
END;



BEGIN

DBMS\_OUTPUT.PUT\_LINE('The cat sat on the mat');

END;



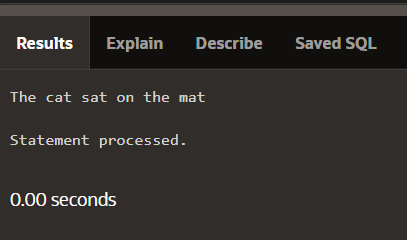
BEGIN

DBMS\_OUTPUT.PUT('The cat sat ');

DBMS\_OUTPUT.PUT('on the mat');

DBMS\_OUTPUT.NEW\_LINE;

END;



DECLARE

v\_bool1 BOOLEAN := true;

v\_bool2 BOOLEAN := false;

v\_number NUMBER;

BEGIN

IF v\_bool1 AND NOT v\_bool2 AND v\_number < 25 THEN

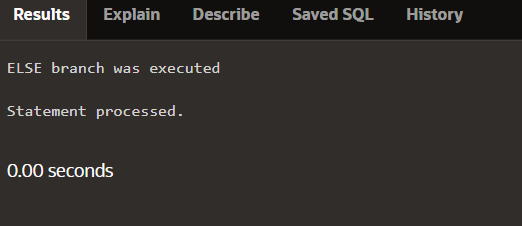
DBMS\_OUTPUT.PUT\_LINE('IF branch was executed');

ELSE

DBMS\_OUTPUT.PUT\_LINE('ELSE branch was executed');

END IF;

END;

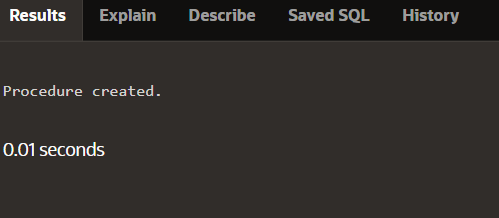


CREATE OR REPLACE PROCEDURE do\_some\_work IS

BEGIN

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

END;



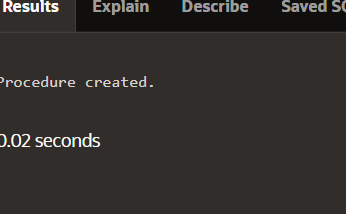
CREATE OR REPLACE PROCEDURE do\_some\_work

( p\_output OUT VARCHAR2)

IS BEGIN

p\_output := 'string';

END;



CREATE SYNONYM UTL\_FILE FOR SYS.UTL\_FILE;

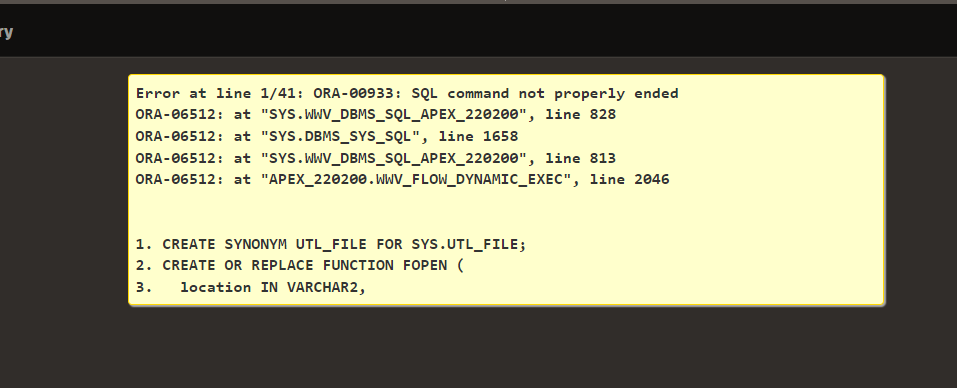
CREATE OR REPLACE FUNCTION FOPEN (

location IN VARCHAR2,

filename IN VARCHAR2,

open\_mode IN VARCHAR2

) RETURN UTL\_FILE.FILE\_TYPE;



PROCEDURE read(dir VARCHAR2, filename VARCHAR2) IS

file UTL\_FILE.FILE\_TYPE;

BEGIN

IF NOT UTL\_FILE.IS\_OPEN(file) THEN

file := UTL\_FILE.FOPEN (dir, filename, 'r');

END IF; ...

END read;



CREATE OR REPLACE PROCEDURE send\_mail\_with\_text

(p\_text\_attachment IN VARCHAR2)

IS BEGIN

UTL\_MAIL.SEND\_ATTACH\_VARCHAR2(

sender => 'me@here.com',

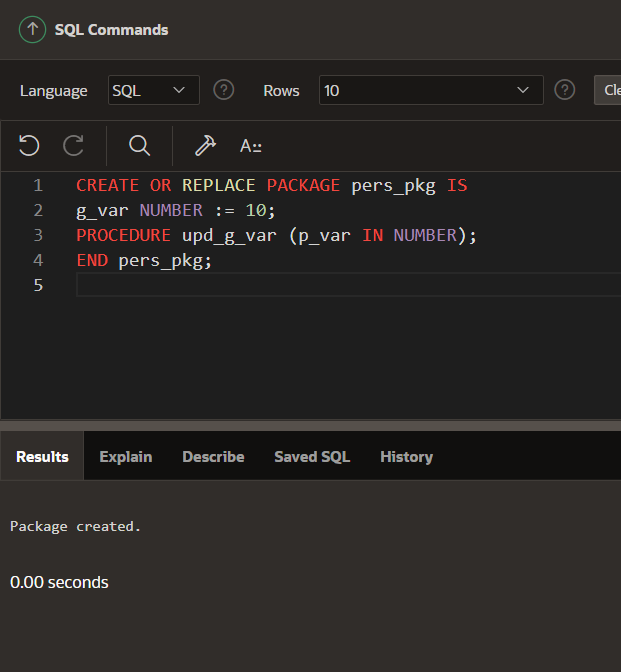
recipients => 'you@somewhere.net',

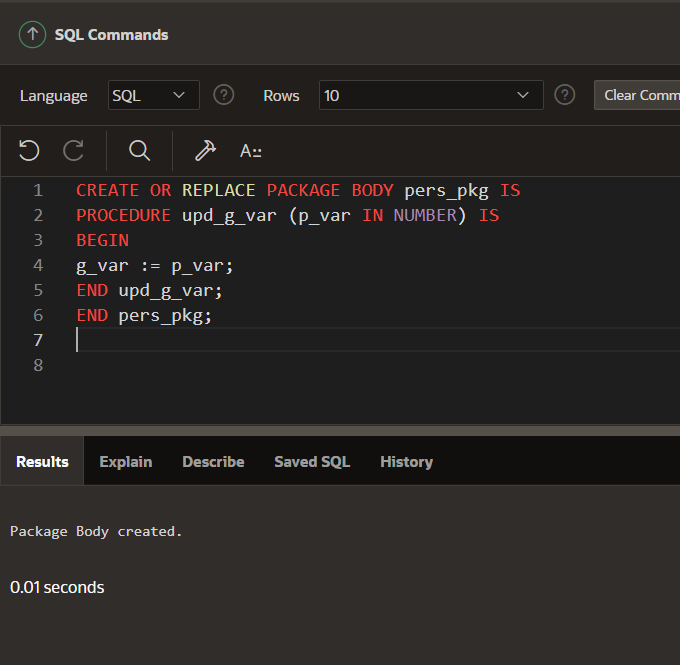
message => 'See attachment',

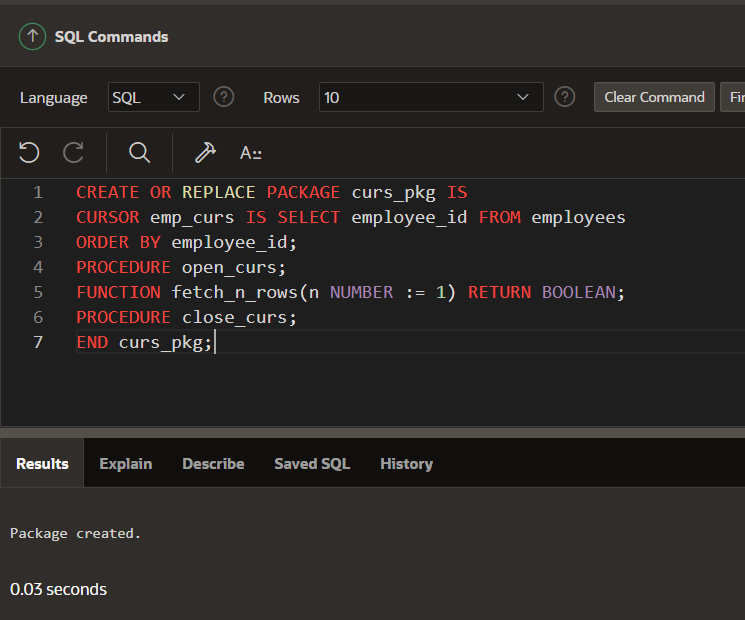
subject => 'Useful document for our project',

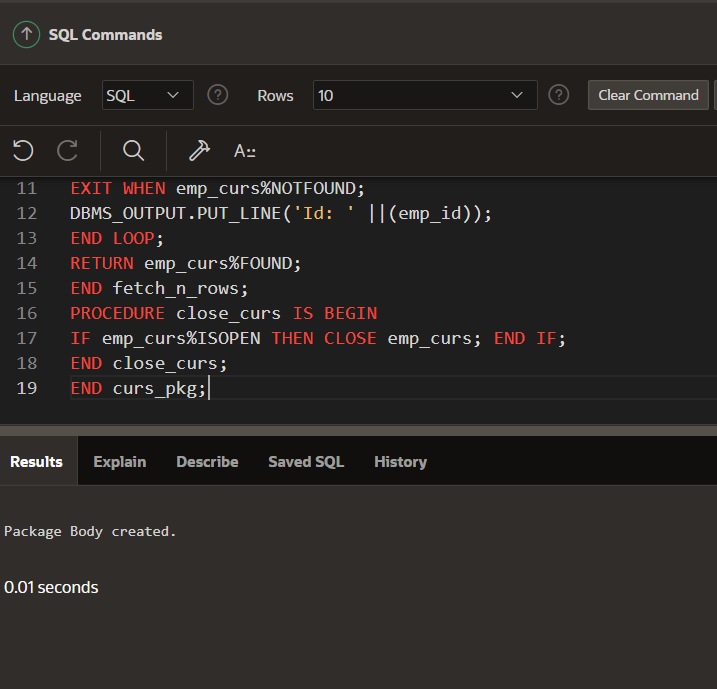
attachment => p\_text\_attachment,

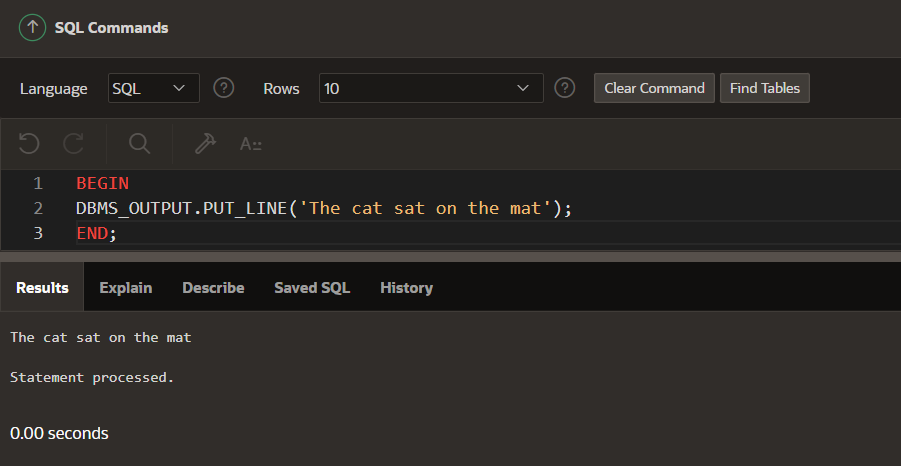
END send\_mail\_with\_text;

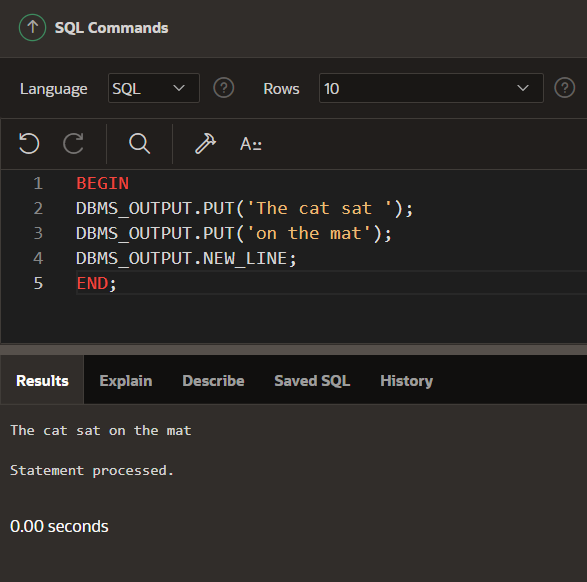
****

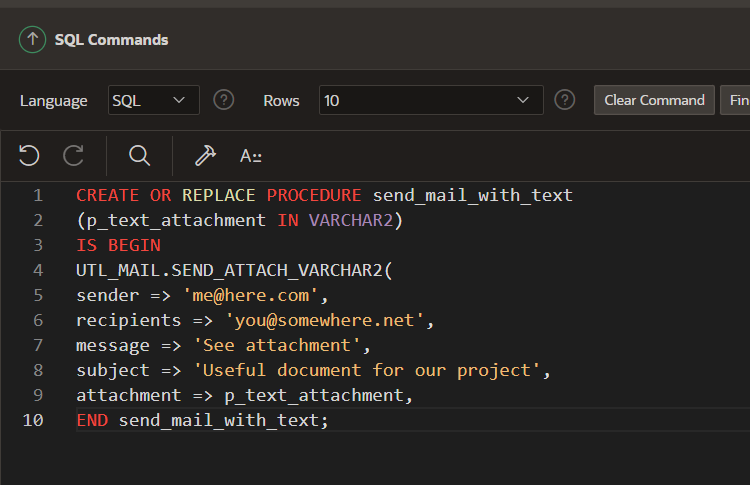
****

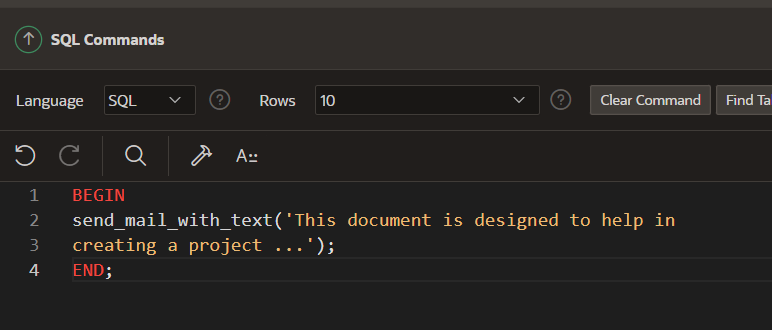
****

****

****

****

****

****