**Section12**

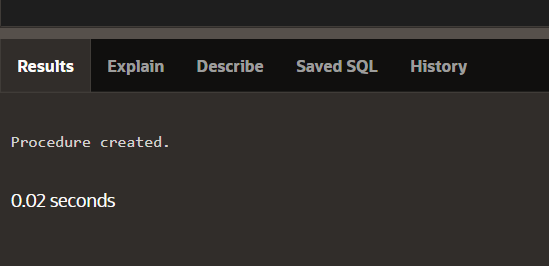
CREATE OR REPLACE PROCEDURE drop\_any\_table(p\_table\_name IN VARCHAR2)

IS

BEGIN

EXECUTE IMMEDIATE 'DROP TABLE ' || p\_table\_name;

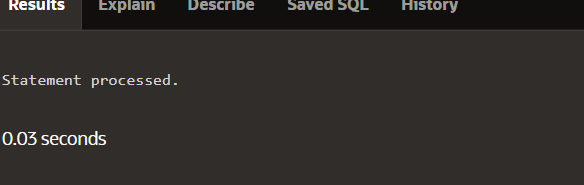
END;



BEGIN

drop\_any\_table('EMPLOYEES');

END;



CREATE FUNCTION del\_rows(p\_table\_name VARCHAR2)

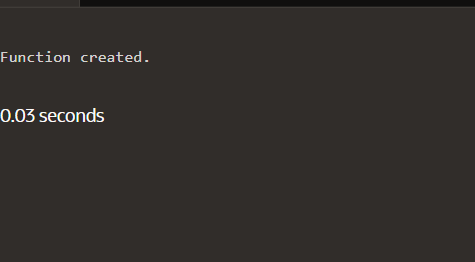
RETURN NUMBER IS

BEGIN

EXECUTE IMMEDIATE 'DELETE FROM ' || p\_table\_name;

RETURN SQL%ROWCOUNT;

END;



DECLARE

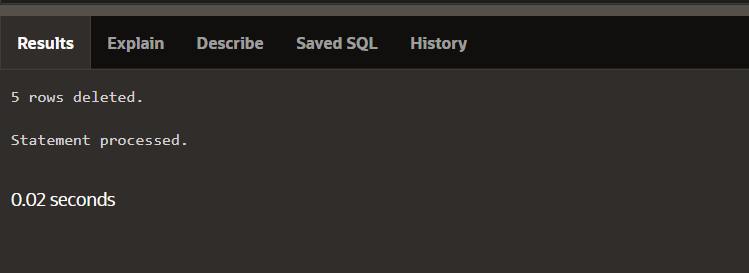
v\_count NUMBER;

BEGIN

v\_count := del\_rows('employee');

DBMS\_OUTPUT.PUT\_LINE(v\_count || ' rows deleted.');

END;



CREATE PROCEDURE add\_row(p\_table\_name VARCHAR2,

p\_id NUMBER, p\_name VARCHAR2) IS

BEGIN

EXECUTE IMMEDIATE 'INSERT INTO ' || p\_table\_name ||

'VALUES(' || p\_id || ', ''' || p\_name || ''')';

END;



CREATE PROCEDURE compile\_plsql

(p\_name VARCHAR2,p\_type VARCHAR2,p\_options VARCHAR2 := NULL)

IS

v\_stmt VARCHAR2(200);

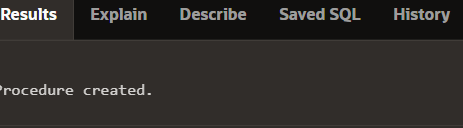
BEGIN

v\_stmt := 'ALTER ' || p\_type || ' ' || p\_name || ' COMPILE'

|| ' ' || p\_options;

EXECUTE IMMEDIATE v\_stmt;

END;



CREATE OR REPLACE FUNCTION del\_rows

(p\_table\_name VARCHAR2) RETURN NUMBER IS

v\_csr\_id INTEGER;

v\_rows\_del NUMBER;

BEGIN

v\_csr\_id := DBMS\_SQL.OPEN\_CURSOR;

DBMS\_SQL.PARSE(v\_csr\_id,

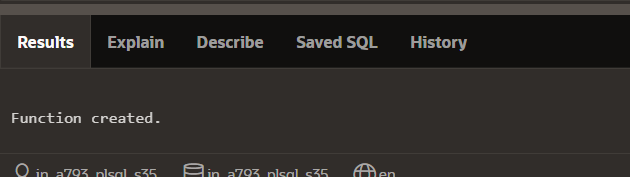
'DELETE FROM ' || p\_table\_name, DBMS\_SQL.NATIVE);

v\_rows\_del := DBMS\_SQL.EXECUTE(v\_csr\_id);

DBMS\_SQL.CLOSE\_CURSOR(v\_csr\_id);

RETURN v\_rows\_del;

END;



CREATE OR REPLACE PACKAGE emp\_pkg IS

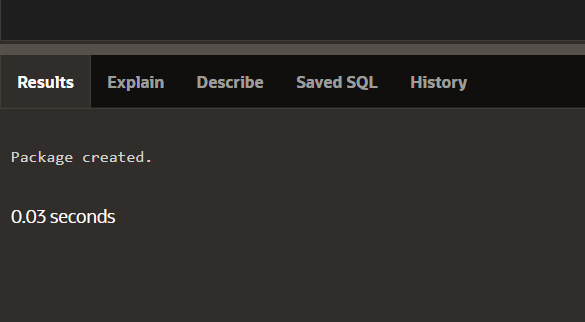
TYPE t\_emp IS TABLE OF employee%ROWTYPE

INDEX BY BINARY\_INTEGER;

PROCEDURE emp\_proc

(p\_small\_arg IN NUMBER, p\_big\_arg OUT t\_emp);

END emp\_pkg;



CREATE OR REPLACE PACKAGE emp\_pkg IS

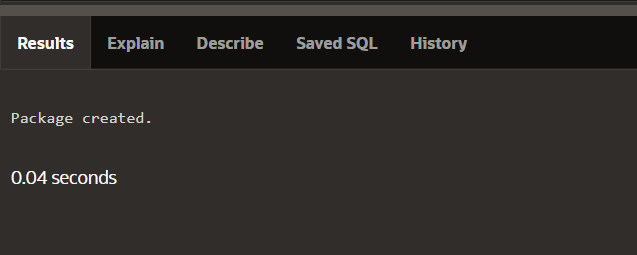
TYPE t\_emp IS TABLE OF employee%ROWTYPE

INDEX BY BINARY\_INTEGER;

PROCEDURE emp\_proc

(p\_small\_arg IN NUMBER, p\_big\_arg OUT t\_emp);

END emp\_pkg;



CREATE OR REPLACE PACKAGE emp\_pkg IS

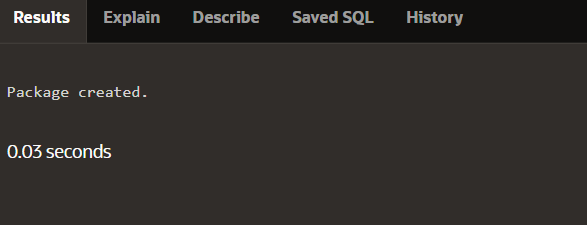
TYPE t\_emp IS TABLE OF employee%ROWTYPE

INDEX BY BINARY\_INTEGER;

PROCEDURE emp\_proc

(p\_small\_arg IN NUMBER, p\_big\_arg OUT NOCOPY t\_emp);

END emp\_pkg;



CREATE OR REPLACE PACKAGE emp\_pkg IS

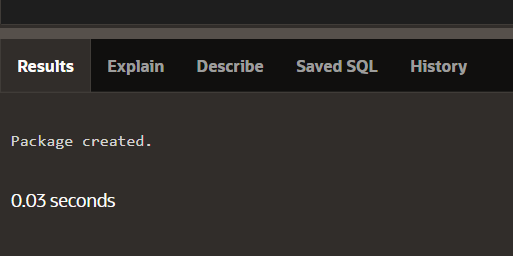
TYPE t\_emp IS TABLE OF employee%ROWTYPE

INDEX BY BINARY\_INTEGER;

PROCEDURE emp\_proc

(p\_small\_arg IN NUMBER, p\_big\_arg OUT NOCOPY t\_emp);

END emp\_pkg;



CREATE OR REPLACE FUNCTION twicenum

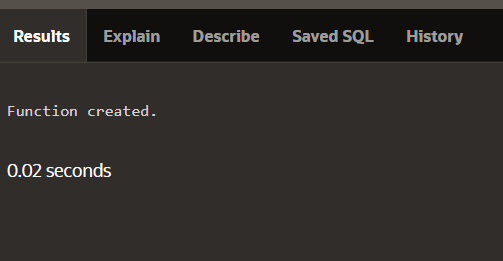
(p\_number IN NUMBER)

RETURN NUMBER IS

BEGIN

RETURN p\_number \* 2;

END twicenum;



CREATE OR REPLACE PROCEDURE fetch\_all\_emps IS

TYPE t\_emp IS TABLE OF employees%ROWTYPE INDEX BY BINARY\_INTEGER;

v\_emptab t\_emp;

BEGIN

SELECT \* BULK COLLECT INTO v\_emptab FROM employees;

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

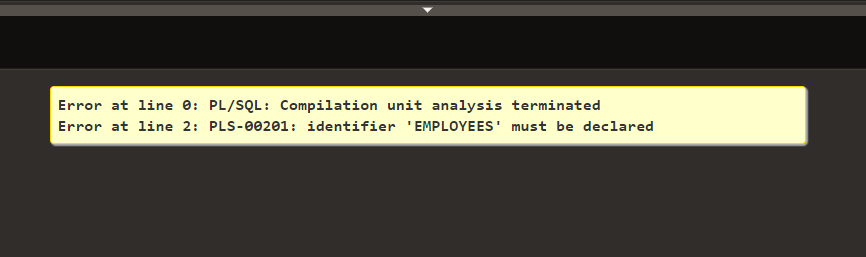
IF v\_emptab.EXISTS(i) THEN

DBMS\_OUTPUT.PUT\_LINE(v\_emptab(i).last\_name);

END IF;

END LOOP;

END fetch\_all\_emps;



CREATE OR REPLACE PROCEDURE update\_all\_emps

(p\_salary\_raise\_percent IN NUMBER) IS

TYPE t\_empid IS TABLE OF employees.employee\_id%TYPE

INDEX BY BINARY\_INTEGER;

TYPE t\_sal IS TABLE OF employees.salary%TYPE

INDEX BY BINARY\_INTEGER;

v\_empidtab t\_empid;

v\_saltab t\_sal;

BEGIN

SELECT employee\_id BULK COLLECT INTO v\_empidtab FROM employees;

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

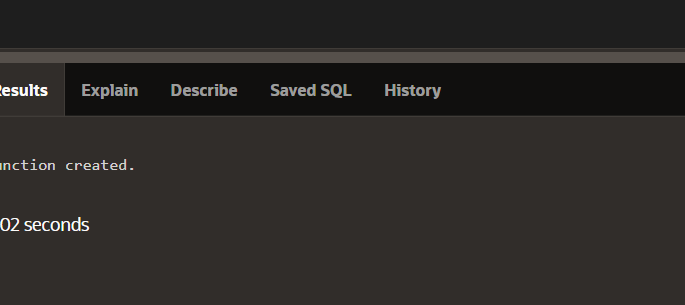
UPDATE employees

SET salary = salary \* (1 + p\_salary\_raise\_percent)

WHERE employee\_id = v\_empidtab(i)

RETURNING salary BULK COLLECT INTO v\_saltab;

END update\_all\_emps;



CREATE OR REPLACE PROCEDURE update\_one\_emp

(p\_emp\_id IN employees.employee\_id%TYPE,

p\_salary\_raise\_percent IN NUMBER) IS

v\_new\_salary employees.salary%TYPE;

BEGIN

UPDATE employees

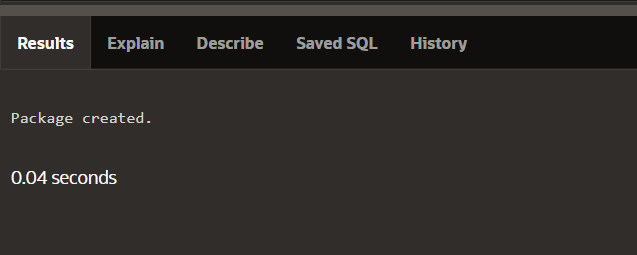
SET salary = salary \* (1 + p\_salary\_raise\_percent)

WHERE employee\_id = p\_emp\_id

RETURNING salary INTO v\_new\_salary;

DBMS\_OUTPUT.PUT\_LINE('New salary is: ' || v\_new\_salary);

END update\_one\_emp;



CREATE OR REPLACE PROCEDURE update\_one\_emp

(p\_emp\_id IN employees.employee\_id%TYPE,

p\_salary\_raise\_percent IN NUMBER) IS

v\_new\_salary employees.salary%TYPE;

BEGIN

UPDATE employees

SET salary = salary \* (1 + p\_salary\_raise\_percent)

WHERE employee\_id = p\_emp\_id;

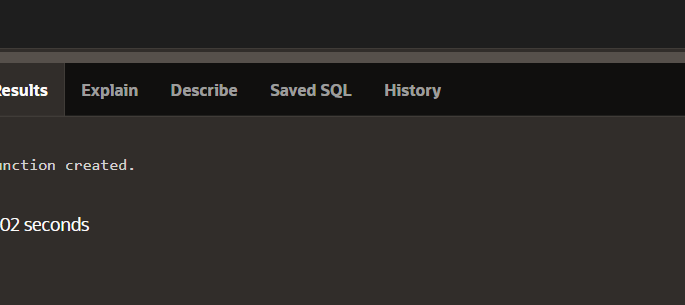
SELECT salary INTO v\_new\_salary

FROM employees

WHERE employee\_id = p\_emp\_id;

DBMS\_OUTPUT.PUT\_LINE('New salary is: ' || v\_new\_salary);

END update\_one\_emp;



CREATE OR REPLACE PROCEDURE insert\_emps IS

TYPE t\_emps IS TABLE OF employees%ROWTYPE INDEX BY BINARY\_INTEGER;

v\_emptab t\_emps;

BEGIN

SELECT \* BULK COLLECT INTO v\_emptab FROM employees;

FORALL i IN v\_emptab.FIRST..v\_emptab.LAST SAVE EXCEPTIONS

INSERT INTO employees VALUES v\_emptab(i);

EXCEPTION

WHEN OTHERS THEN

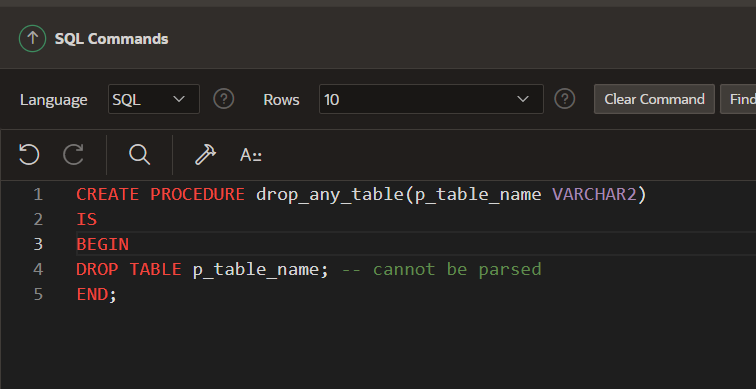
FOR j in 1..SQL%BULK\_EXCEPTIONS.COUNT LOOP

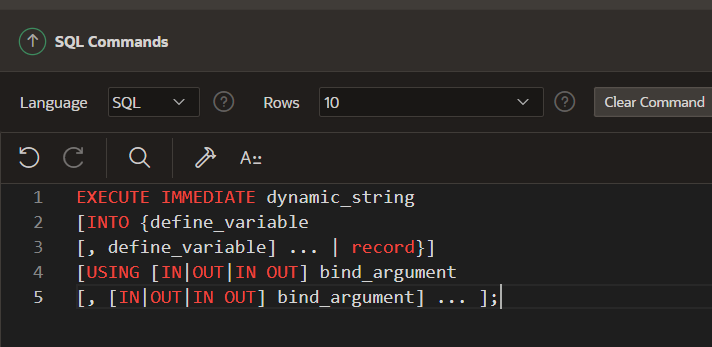
DBMS\_OUTPUT.PUT\_LINE(SQL%BULK\_EXCEPTIONS(j).ERROR\_INDEX);

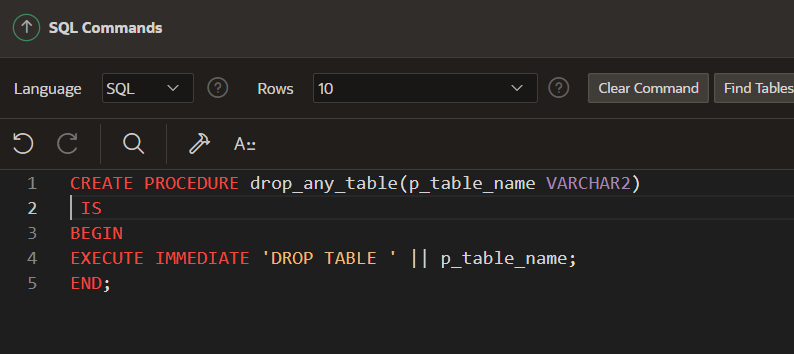
DBMS\_OUTPUT.PUT\_LINE(SQL%BULK\_EXCEPTIONS(j).ERROR\_CODE);

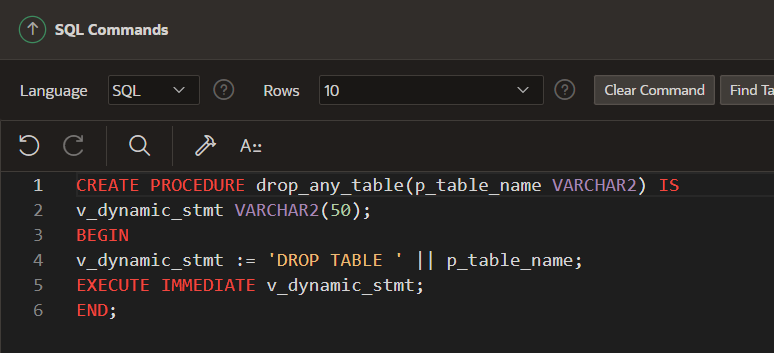
END LOOP;

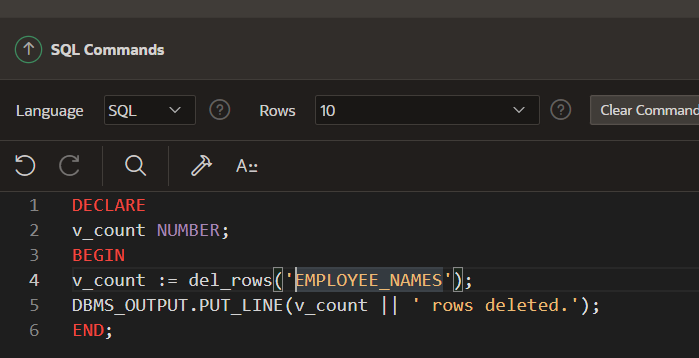
END insert\_emps;

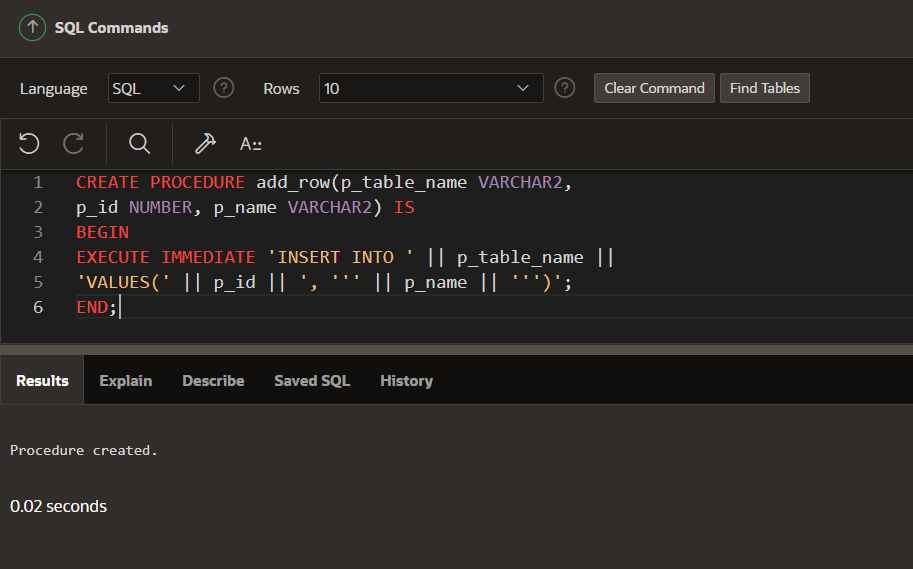
****

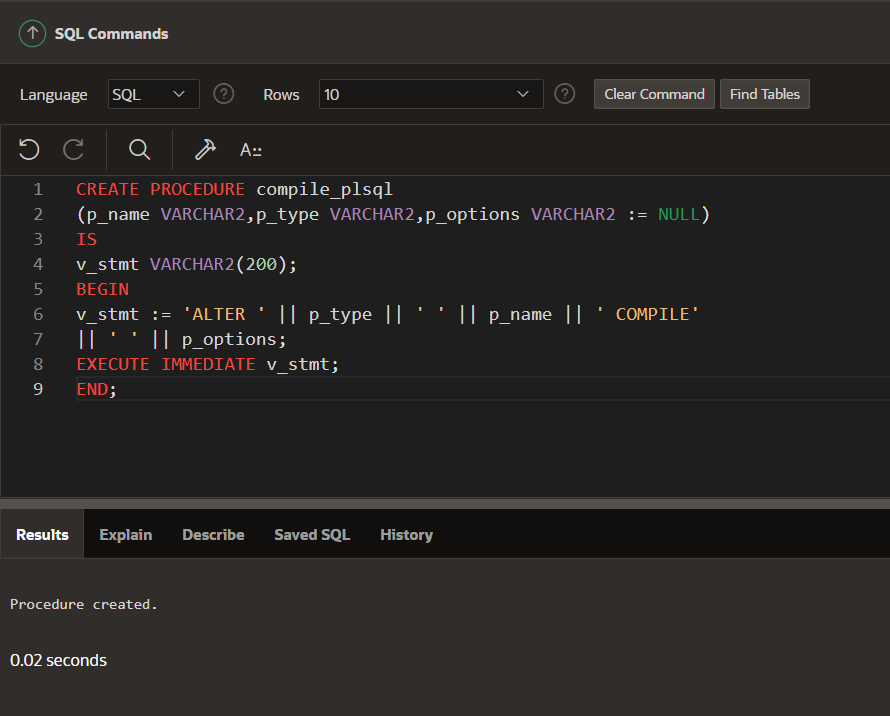
****

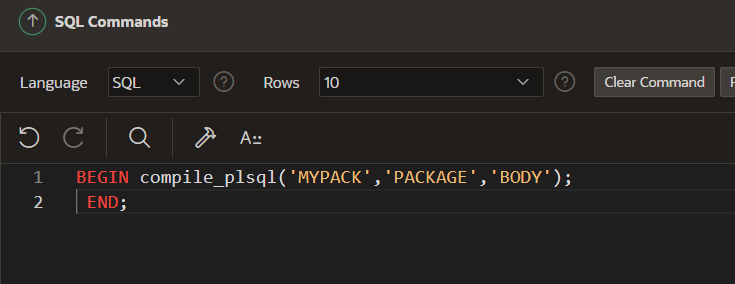
****

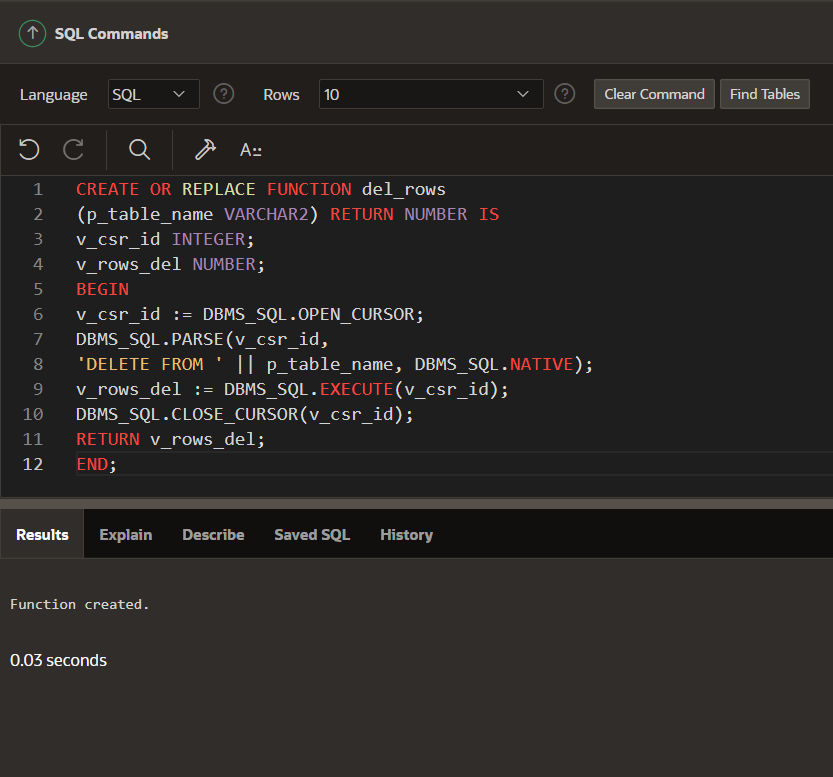
****

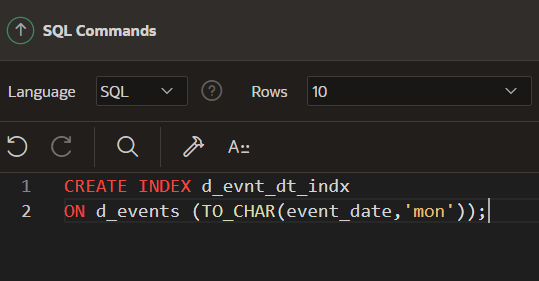
****

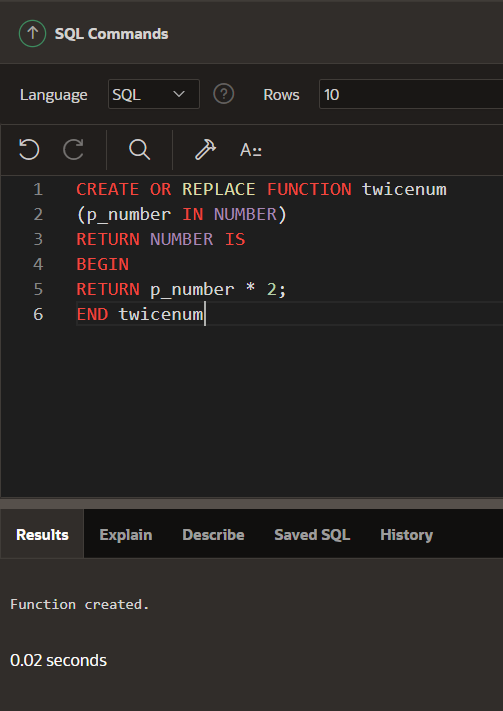
****

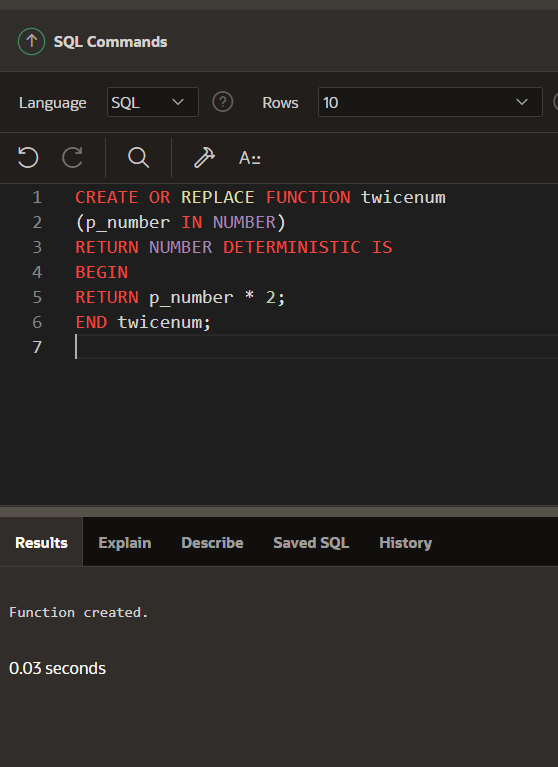
****

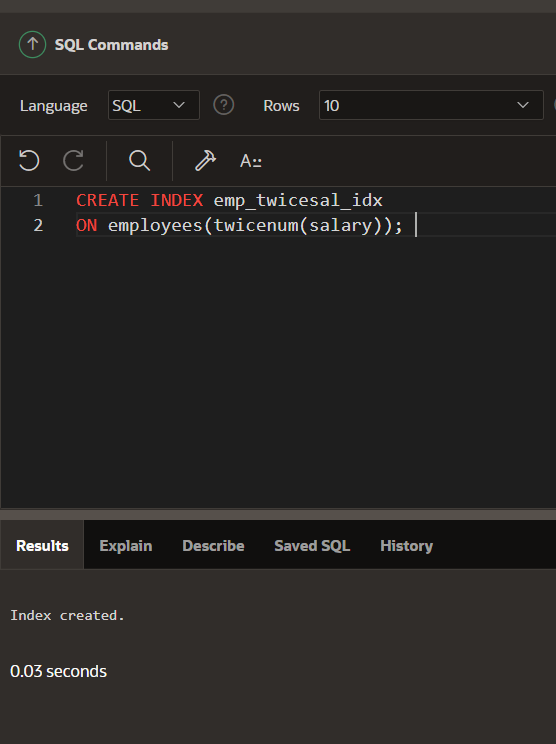
****

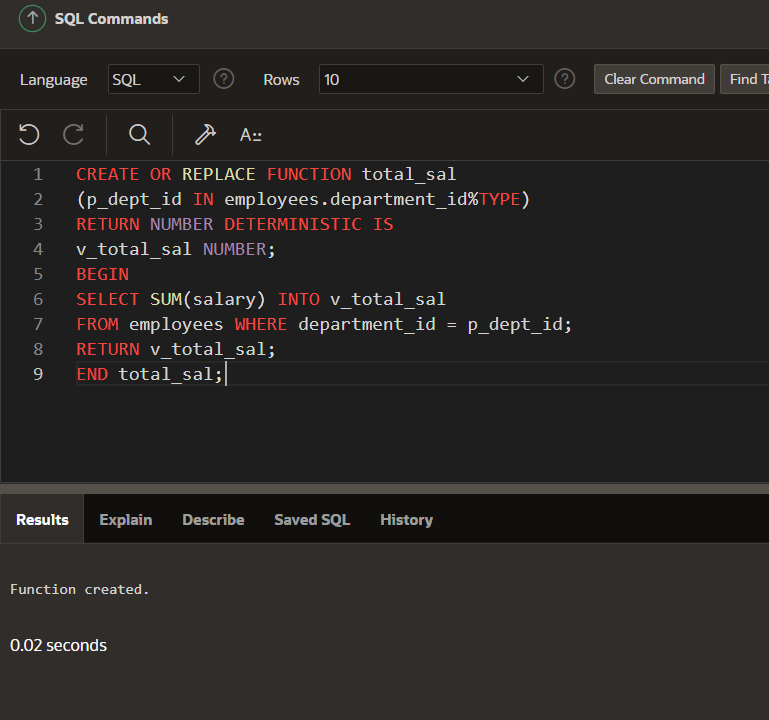
****

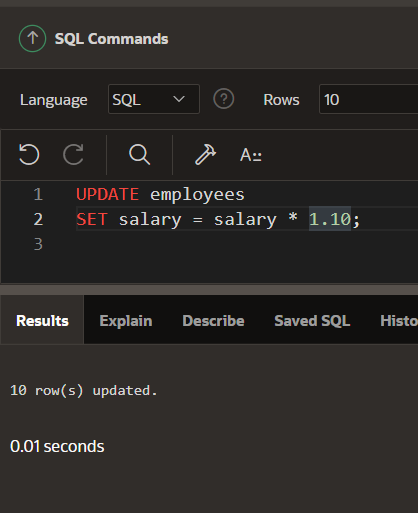
****

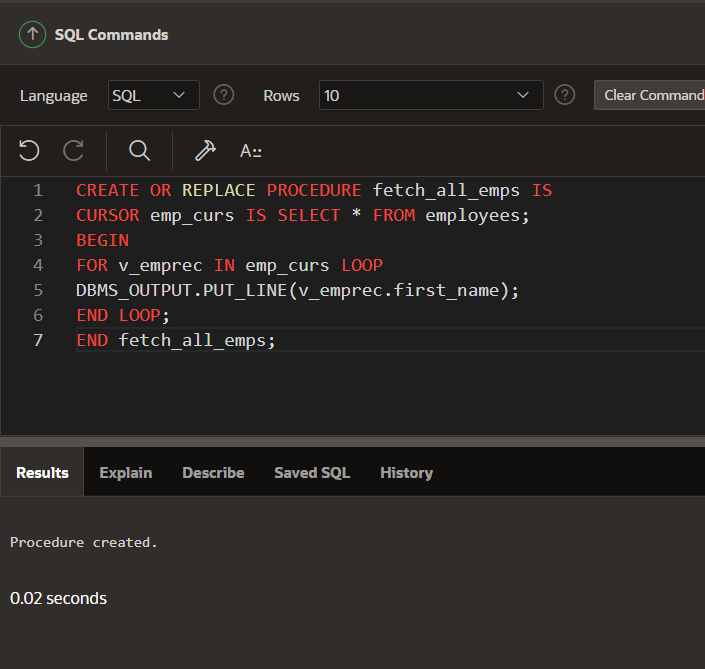
****

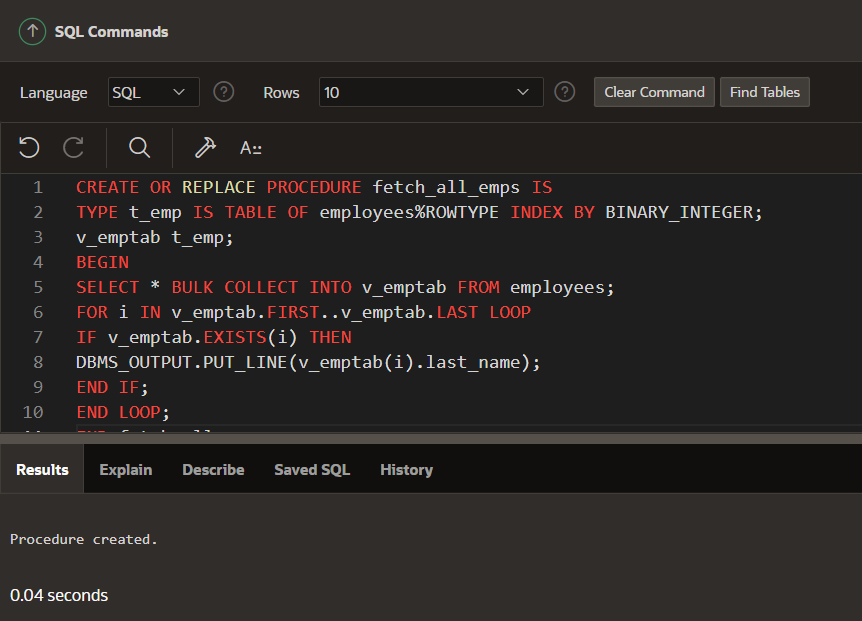
****

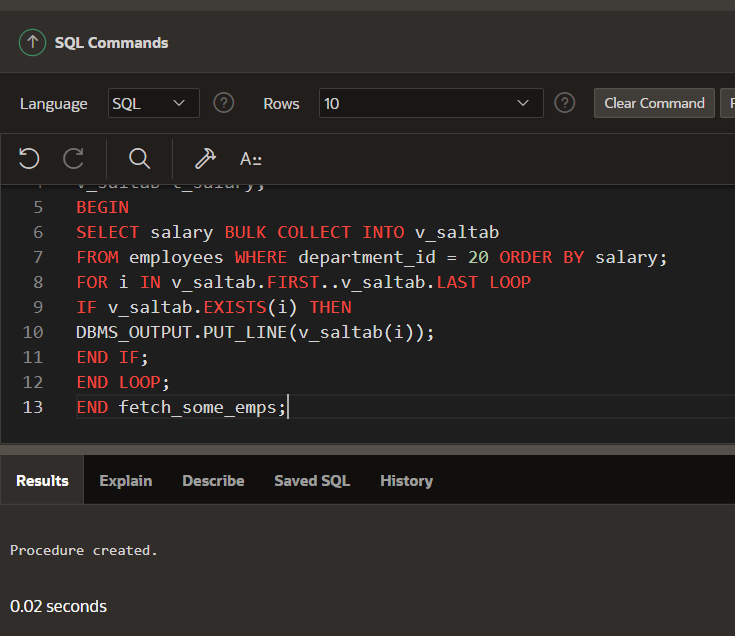
****

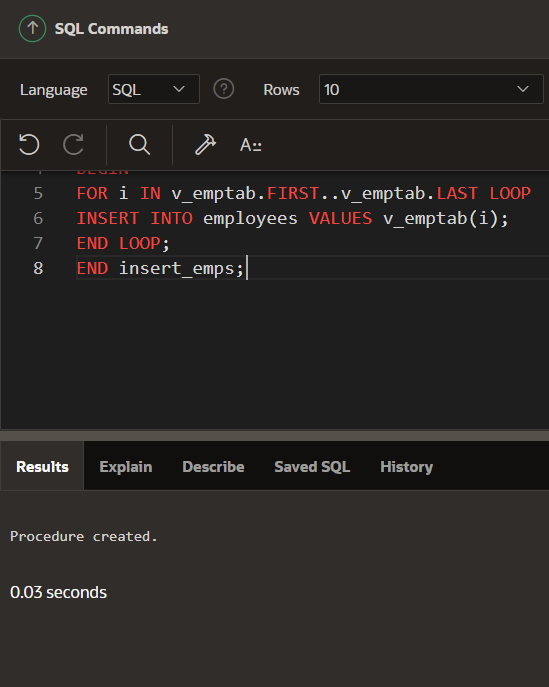
****

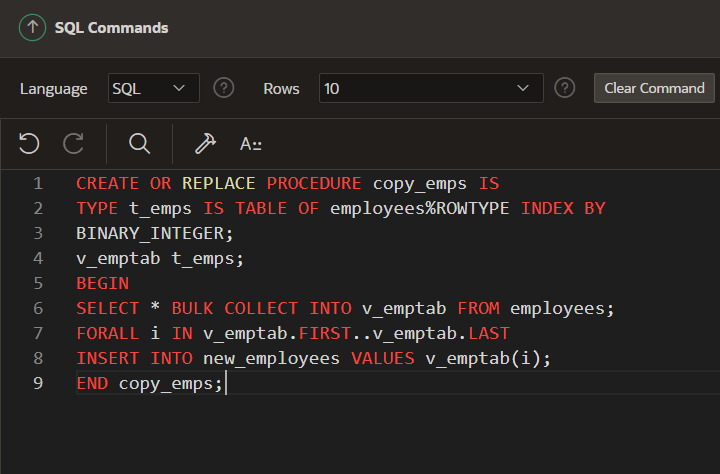
****

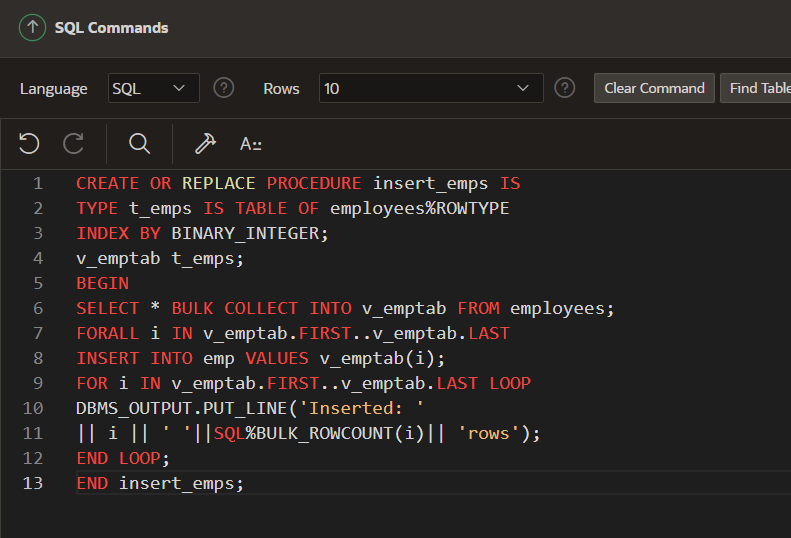
****

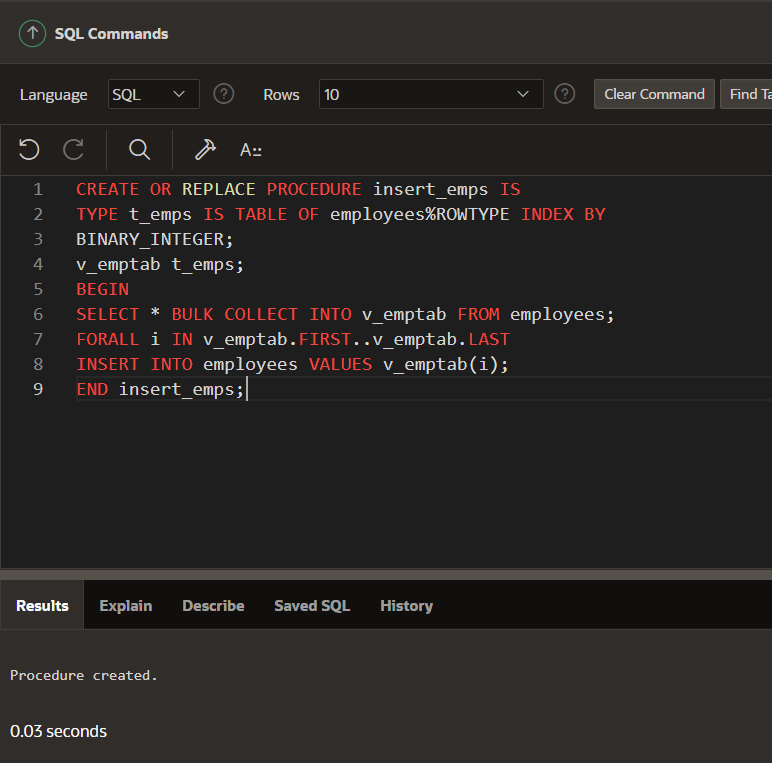
****

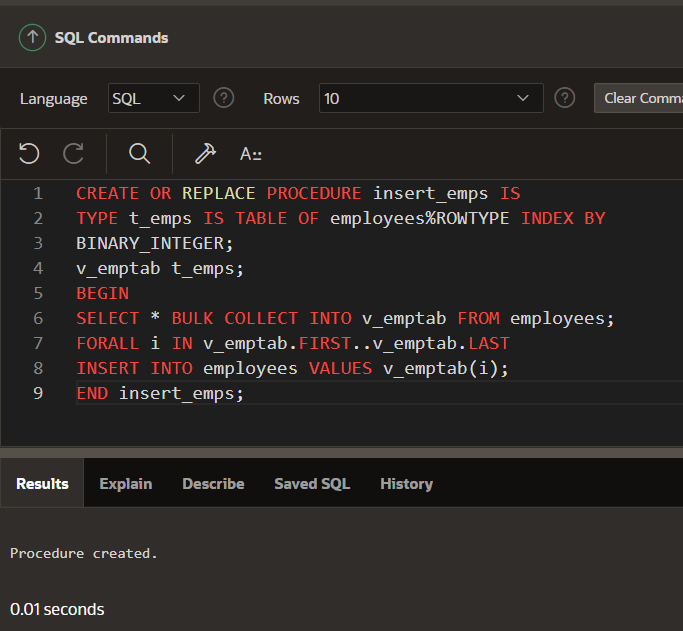
****

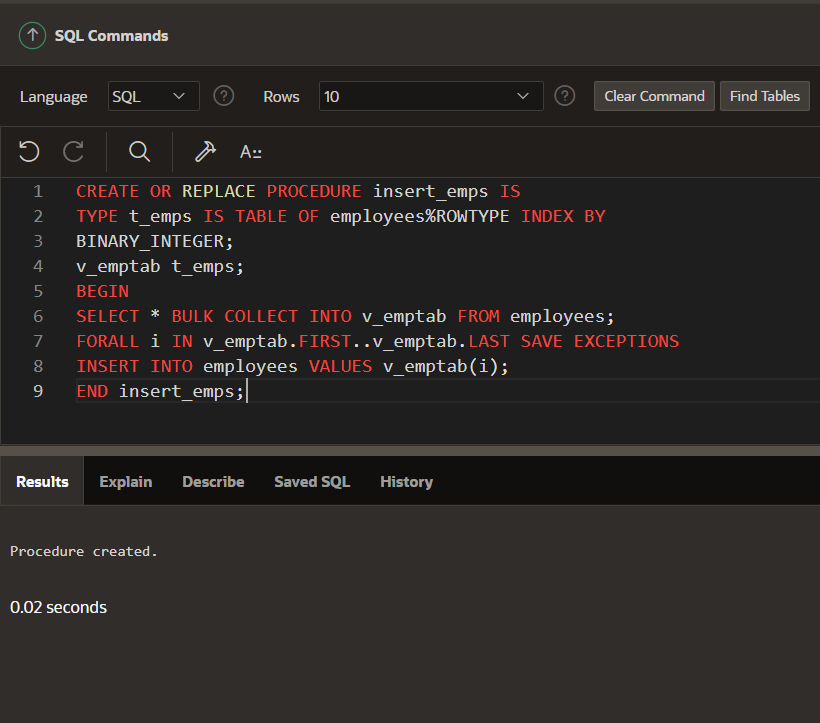
****

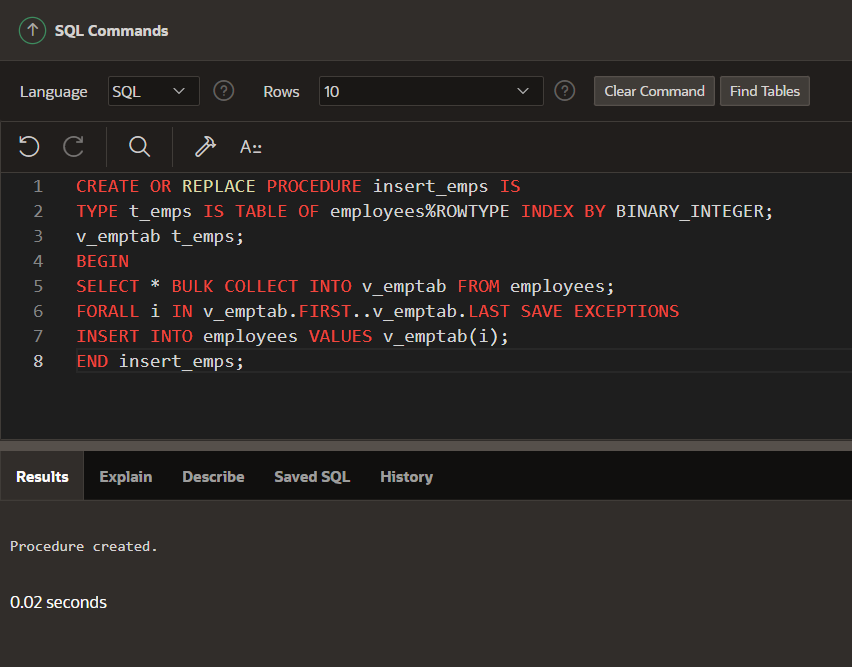
****

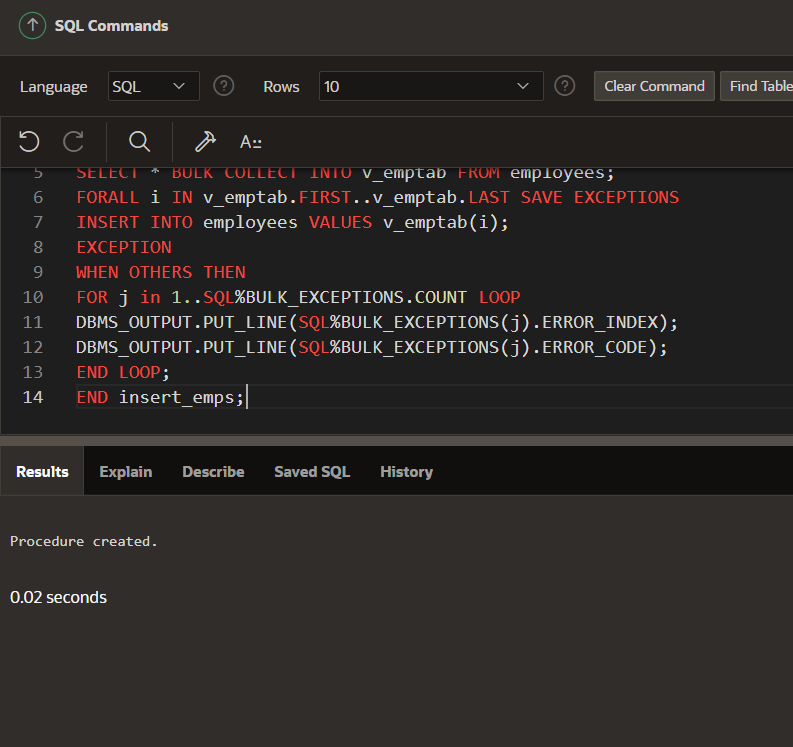
****

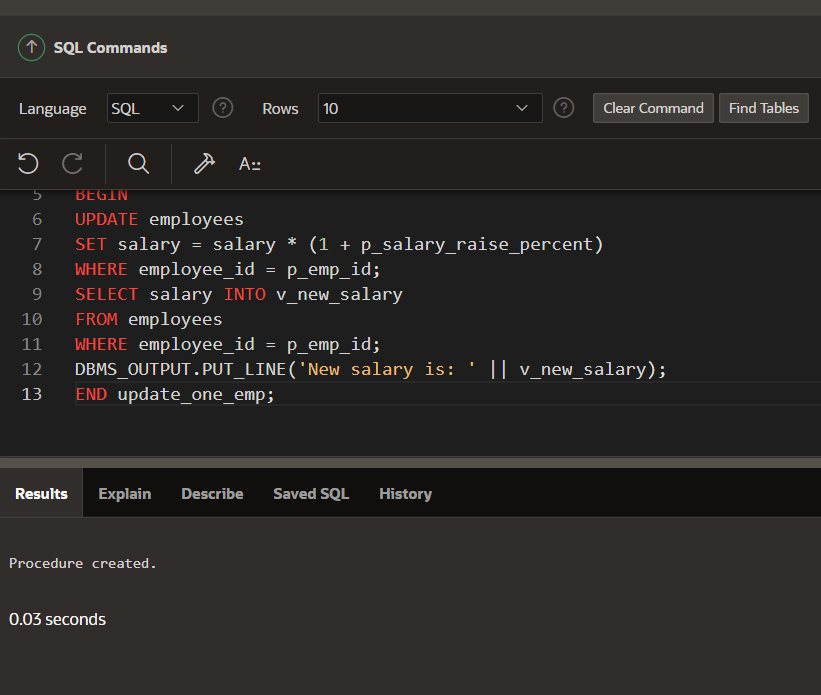
****

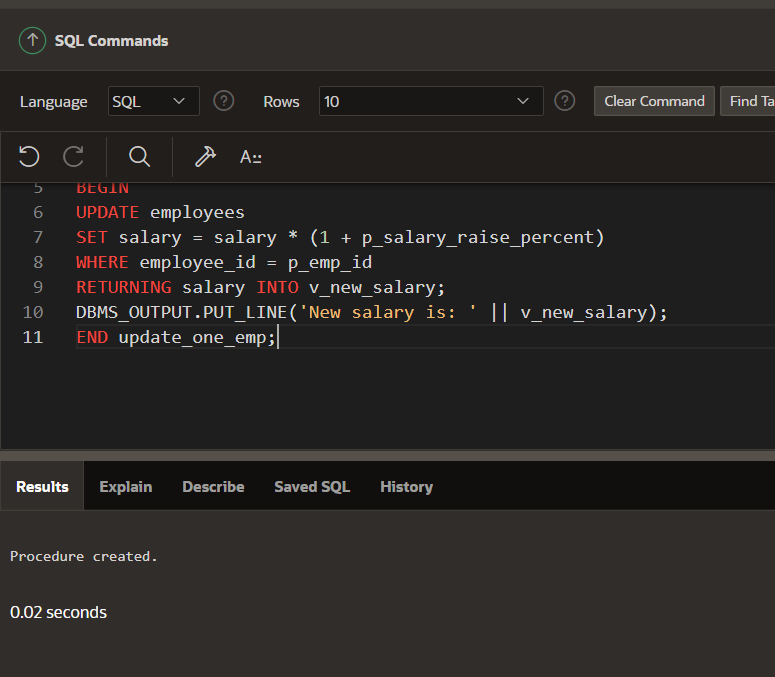
****

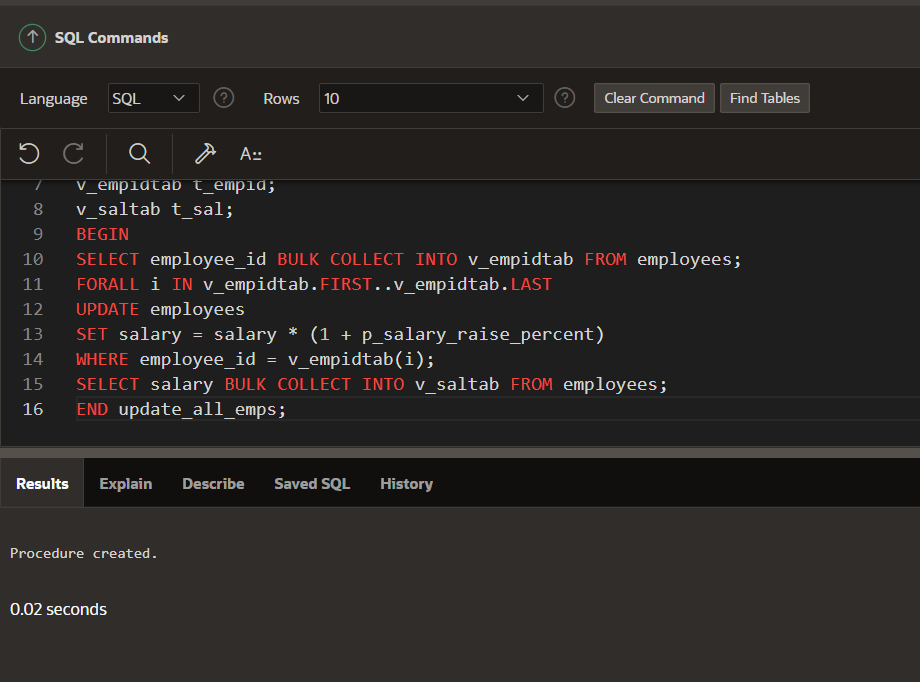
****

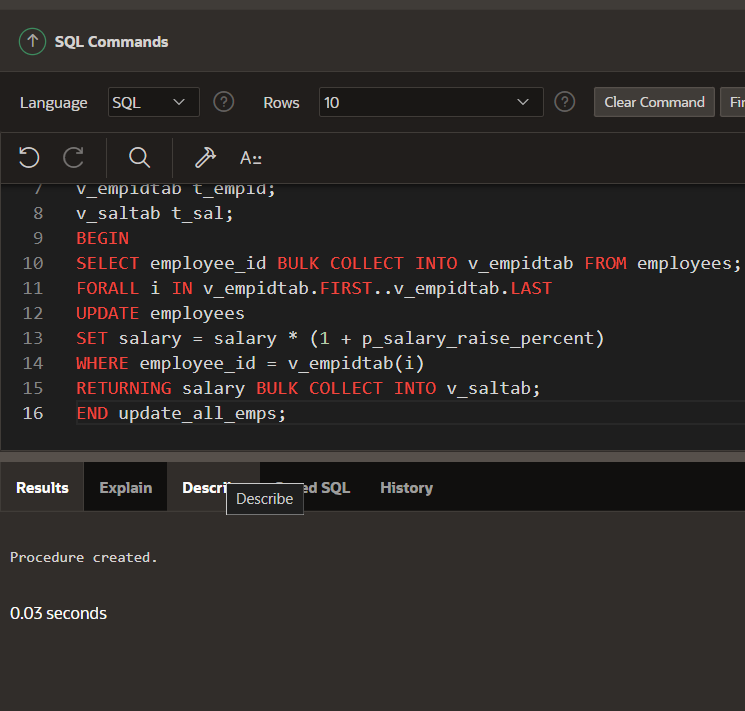
****

****

****

****

****

****