**Practice 7**

1. Create and invoke the GET\_JOB function to return a job title.

* + - 1. Create and compile a function called GET\_JOB to return a job title.

CREATE OR REPLACE FUNCTION GET\_JOB(p\_job\_id JOBS.JOB\_ID%TYPE) RETURN JOBS.JOB\_TITLE%TYPE AS

v\_job\_title JOBS.JOB\_TITLE%TYPE;

BEGIN

SELECT JOB\_TITLE INTO v\_job\_title

FROM JOBS

WHERE JOB\_ID = p\_job\_id;

RETURN v\_job\_title;

END;

/

* + - 1. Call the created function from an anonymous block for the SA\_REP job ID and print the result.

DECLARE

v\_job\_title JOBS.JOB\_TITLE%TYPE;

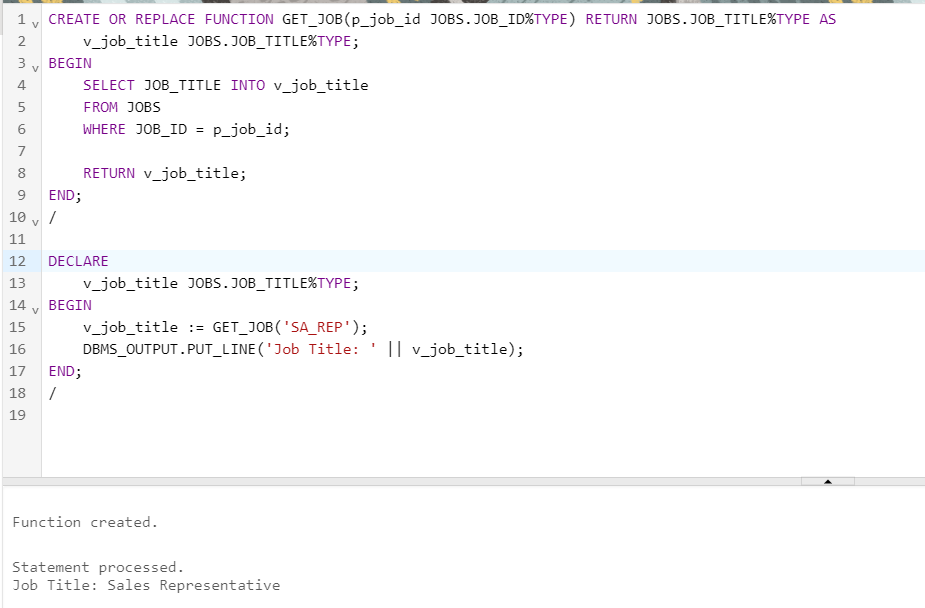
BEGIN

v\_job\_title := GET\_JOB('SA\_REP');

DBMS\_OUTPUT.PUT\_LINE('Job Title: ' || v\_job\_title);

END;

/



2. Create a function called GET\_ANNUAL\_COMP to return the annual salary computed from an employee’s monthly salary and commission passed as parameters.

a. Create the function, which accepts parameter values for the monthly salary and commission. Either or both values passed can be NULL, but the function should still return a non-NULL annual salary. Use the following basic formula to calculate the annual salary:  
 (salary\*12) + (commission\_pct\*salary\*12)

CREATE OR REPLACE FUNCTION GET\_ANNUAL\_COMP(

p\_monthly\_salary NUMBER,

p\_commission NUMBER

) RETURN NUMBER AS

v\_annual\_salary NUMBER;

BEGIN

v\_annual\_salary := (NVL(p\_monthly\_salary, 0) \* 12) + (NVL(p\_commission, 0) \* NVL(p\_monthly\_salary, 0) \* 12);

RETURN v\_annual\_salary;

END;

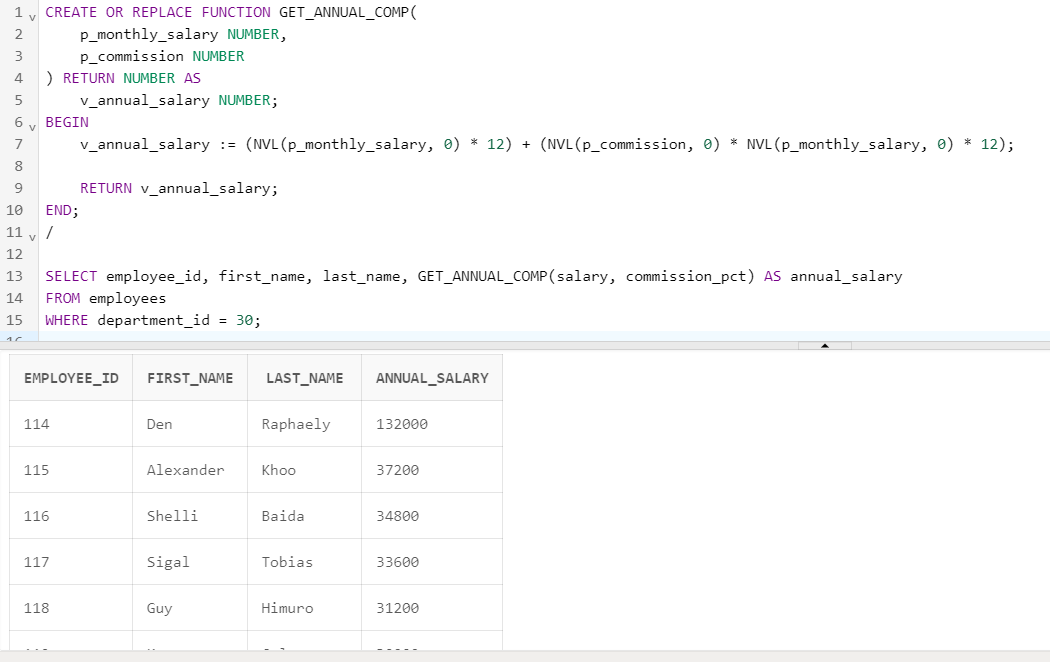
/

1. Use the function in a SELECT statement against the EMPLOYEES table for employees in department 30.

SELECT employee\_id, first\_name, last\_name, GET\_ANNUAL\_COMP(salary, commission\_pct) AS annual\_salary

FROM employees

WHERE department\_id = 30;



3. Create a procedure, ADD\_EMPLOYEE, to insert a new employee into the EMPLOYEES table. The procedure should call a VALID\_DEPTID function to check whether the department ID specified for the new employee exists in the DEPARTMENTS table.

a. Create a function called VALID\_DEPTID to validate a specified department ID and return a BOOLEAN value of TRUE if the department exists.

b. Create the ADD\_EMPLOYEE procedure to add an employee to the EMPLOYEES table. The row should be added to the EMPLOYEES table if the VALID\_DEPTID function returns TRUE; otherwise, alert the user with an appropriate message. Provide the following parameters:

* + - * + first\_name
        + last\_name
        + email
        + job: Use ‘SA\_REP’ as the default
        + mgr: Use 145 as the default
        + sal: Use 1000 as the default
        + comm: Use 0 as the default
        + deptid: Use 30 as the default
        + Use the EMPLOYEES\_SEQ sequence to set the employee\_id column
        + Set the hire\_date column to TRUNC(SYSDATE)
      1. Call ADD\_EMPLOYEE for the name Jane Harris in department 15, leaving other parameters with their default values. What is the result?
      2. Add another employee named Joe Harris in department 80, leaving remaining parameters with their default values. What is the result?



CREATE OR REPLACE FUNCTION VALID\_DEPTID(

p\_dept\_id NUMBER

) RETURN BOOLEAN AS

v\_dept\_count NUMBER;

BEGIN

SELECT COUNT(\*)

INTO v\_dept\_count

FROM departments

WHERE department\_id = p\_dept\_id;

RETURN (v\_dept\_count > 0);

END;

/

CREATE OR REPLACE PROCEDURE ADD\_EMPLOYEE(

p\_first\_name VARCHAR2,

p\_last\_name VARCHAR2,

p\_email VARCHAR2,

p\_job\_id VARCHAR2 DEFAULT 'SA\_REP',

p\_manager\_id NUMBER DEFAULT 145,

p\_salary NUMBER DEFAULT 1000,

p\_commission\_pct NUMBER DEFAULT 0,

p\_department\_id NUMBER DEFAULT 30

) AS

BEGIN

IF VALID\_DEPTID(p\_department\_id) THEN

INSERT INTO employees (employee\_id, first\_name, last\_name, email, hire\_date, job\_id, salary, commission\_pct, manager\_id, department\_id)

VALUES (employees\_seq.NEXTVAL, p\_first\_name, p\_last\_name, p\_email, TRUNC(SYSDATE), p\_job\_id, p\_salary, p\_commission\_pct, p\_manager\_id, p\_department\_id);

COMMIT;

DBMS\_OUTPUT.PUT\_LINE('Employee added successfully.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('Invalid department ID. Employee not added.');

END IF;

END;

/

BEGIN

ADD\_EMPLOYEE('Jane', 'Harris', 'jane@gmail.com', 'SA\_REP', NULL, NULL, NULL, 15);

END;

/

BEGIN

ADD\_EMPLOYEE('Joe', 'Harris', 'joe@gmail.com', 'SA\_REP', NULL, NULL, NULL, 80);

END;

/

SELECT \* FROM employees;