**Zhumagali Kanat IT2-2003**

**e**

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

create or replace function get\_job(j\_id jobs.job\_id%type) return jobs.job\_title%type is j\_title jobs.job\_title%type;

begin

select job\_title into j\_title from jobs where job\_id = j\_id;

return j\_title;

end get\_job;

declare

job\_t jobs.job\_title%type;

cursor j\_cursor is select e.last\_name, e.salary, j.job\_title from employees e inner join jobs j on e.job\_id = j.job\_id where j.job\_title = job\_t;

begin

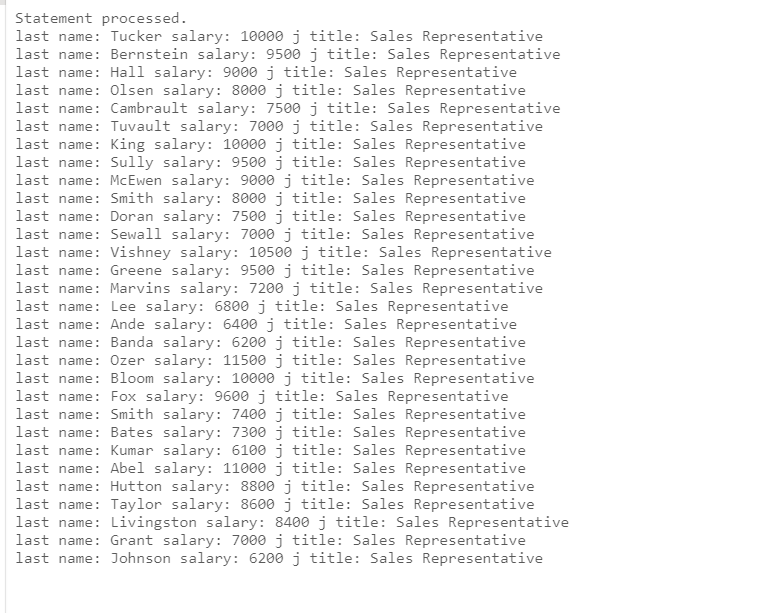
job\_t := get\_job('SA\_REP');

for j\_rec in j\_cursor loop

dbms\_output.put\_line('last name: ' || j\_rec.last\_name || ' salary: ' || j\_rec.salary || ' j title: ' || j\_rec.job\_title);

end loop;

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)

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

create or replace function get\_annual\_comp (salary employees.salary%type, commission\_pct employees.commission\_pct%type) return employees.salary%type is com\_pct employees.commission\_pct%type;

begin

if commission\_pct is null then

com\_pct := 0.5;

return ((salary\*12) + (com\_pct\*salary\*12));

else return ((salary\*12) + (commission\_pct\*salary\*12));

end if;

end get\_annual\_comp;

select last\_name, salary, get\_annual\_comp(salary, commission\_pct ) 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 (department\_id departments.department\_id%type ) return boolean is

begin

if department\_id is null then return false;

else return true;

end if;

end valid\_deptid;

create or replace procedure add\_employee(

first\_name employees.first\_name%TYPE,

last\_name employees.last\_name%TYPE,

email employees.email%TYPE := 'mail',

job employees.job\_id%TYPE := 'SA\_REP',

mgr employees.manager\_id%TYPE := 145,

sal employees.salary%TYPE := 100,

comm employees.commission\_pct%TYPE := 0,

deptid employees.department\_id%TYPE := 30

)

is

begin

IF valid\_deptid(deptid) 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, first\_name, last\_name, email, TRUNC(SYSDATE), job, sal, comm, mgr, deptid);

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

else

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

end if;

end;

execute add\_employee('Jane', 'Haris', deptid => 15);

EXECUTE add\_employee(first\_name => 'Joe', last\_name => 'Haris', deptid => 80);

select \* from employees where first\_name = 'Jane';

select \* from employees where first\_name = 'Joe';

