Post Graduate Diploma in Advance Computing

Database Technologies

Assignment – 7

Prepared By –

Md Farazul Haque

PRN - 210980420078

1. Create a table jobs that have two column jobid and jobtitle.

a) **Create a procedure called ADD_JOB to insert a new job into the JOBS table. Provide the ID and title of the job, using two parameters.**
b) **Compile the code, and invoke the procedure with IT_DBA as job ID and Database Administrator as job title. Query the JOBS table to view the results.**

```
SQL> CREATE TABLE jobs1(jobid VARCHAR2(15),jobtitle VARCHAR2(30));

SQL> ed
CREATE PROCEDURE ADD_JOB(jid VARCHAR2, jtitle VARCHAR2)
IS
BEGIN
        INSERT INTO jobs1 VALUES(jid, jtitle);
END;
/

SQL> exec ADD_JOB('IT_DBA', 'DATABASE ADMINISTRATOR');

SQL> select * from jobs1;

JOBID         JOBTITLE
--------------- -------------------------------------
IT_DBA        DATABASE ADMINISTRATOR
```

2. Create a procedure called UPD_JOB to modify a job in the JOBS table.

a) **Create a procedure called UPD_JOB to update the job title. Provide the job ID and a new title, using two parameters. Include the necessary exception handling if no update occurs.**
b) **Compile the code; invoke the procedure to change the job title of the job ID IT_DBA to Data Administrator. Query the JOBS table to view the results. Also check the exception handling by trying to update a job that does not exist (you can use job ID IT_WEB and job title Web Master).**

```
SQL> ed
CREATE OR REPLACE PROCEDURE upd_job(jid VARCHAR2, jtitle VARCHAR2)
IS
BEGIN
        UPDATE jobs1 SET jobtitle=jtitle WHERE jobid=jid;
        IF SQL%NOTFOUND THEN
                RAISE_APPLICATION_ERROR(-20001,'Job ID does not exist');
        END IF;
END;
/

SQL> EXEC upd_job('IT_DBA', 'DATA ADMINISTRATOR');
PL/SQL procedure successfully completed.
```

```
SQL> EXEC upd_job('IT_WEB', 'WEB MASTER');
BEGIN upd_job('IT_WEB', 'WEB MASTER'); END;

*
ERROR at line 1:
ORA-20001: Job ID does not exist
ORA-06512: at "HR.UPD_JOB", line 6
ORA-06512: at line 1
```

3. Create a procedure called QUERY_EMP to query the EMPLOYEES table, retrieving the salary and job ID for an employee when provided with the employee ID.

```
CREATE OR REPLACE PROCEDURE query_emp(eid in number, sal out number, jid out varchar2)
IS
BEGIN
        select to_number(salary), job_id into sal,jid
        from employees
        where employee_id=eid;
END;
/
```

4. Create and invoke the Q_JOB function to return a job title.

a) **Create a function called Q_JOB to return a job title to a host variable.**
b) **Compile the code; create a host variable G_TITLE and invoke the function with job ID SA_REP. Query the host variable to view the result.**

```
CREATE OR REPLACE FUNCTION Q_JOB ( jid in jobs.job_id%type)
RETURN jobs.job_title%type IS
jtitle jobs.job_title%type;
BEGIN
        SELECT job_title INTO jtitle FROM jobs WHERE job_id=jid;
        RETURN jtitle;
end;
/

SQL> /
Function created.

SQL> variable G_TITLE varchar2(30)
SQL> exec :G_TITLE := Q_JOB('SA_REP');

PL/SQL procedure successfully completed.
```

```
SQL> print G_TITLE

G_TITLE
-----------------------------------
Sales Representative
```

5. Create a procedure, NEW_EMP, to insert a new employee into the EMPLOYEES table. The procedure should contain a call to the VALID_DEPTID function to check whether the department ID specified for the new employee exists in the DEPARTMENTS table.

a) Create a function VALID_DEPTID to validate a specified department ID. The function should return a BOOLEAN value.

```
CREATE OR REPLACE FUNCTION VALID_DEPTID( did IN
departments.department_id%TYPE)
RETURN BOOLEAN IS
valid INTEGER;
BEGIN
    SELECT 1 INTO valid FROM departments WHERE department_id=did;
    RETURN TRUE;
    EXCEPTION
        WHEN NO_DATA_FOUND THEN
        RETURN FALSE;
end;
/
```

b) Create the procedure NEW_EMP to add an employee to the EMPLOYEES table. A new row should be added to EMPLOYEES table if the function returns TRUE. If the function returns FALSE, the procedure should alert the user with an appropriate message. Define DEFAULT values for most parameters. The default commission is 0, the default salary is 1000, the default department ID is 30, the default job is SA_REP and the default manager ID is 145. For the employee's ID, use the sequence EMPLOYEES _SEQ. Provide the last name, first name and e-mail address of the employee.

```
CREATE OR REPLACE PROCEDURE new_emp (
    fname employees.first_name%TYPE,
    lname employees.last_name%TYPE,
    mail  employees.email%TYPE,
    jid   employees.job_id%TYPE DEFAULT 'SA_REP',
    mid   employees.manager_id%TYPE DEFAULT 145,
    sal   employees.salary%type DEFAULT 1000,
    comm  employees.commission_pct%TYPE DEFAULT 0,
    did   employees.department_id%TYPE DEFAULT 30
)
IS
```

```
BEGIN
    IF valid_deptid(did) THEN
            insert into employees(employee_id, first_name, last_name, email, job_id,
            manager_id, hire_date, salary, commission_pct, department_id)
            VALUES (employees_seq.NEXTVAL, fname, lname, mail, jid, mid,
            TRUNC(SYSDATE), sal, comm, did );
    ELSE
            RAISE_APPLICATION_ERROR(-20205, 'Invalid department ID.');
    END IF;
END;
/

SQL> exec new_emp('Faraz', 'Haque', 'FAHAQUE', did=>10);

PL/SQL procedure successfully completed.

SQL> exec new_emp('Faraz', 'Haq', 'FHAQ', did=>101);
BEGIN new_emp('Faraz', 'Haq', 'FHAQ', did=>101); END;

*
ERROR at line 1:
ORA-20205: Invalid department ID.
ORA-06512: at "HR.NEW_EMP", line 16
ORA-06512: at line 1
```

6. Update the description of a department. The user supplies the department number and the new name. If the user enters a department number that does not exist, no rows will be updated in the DEPARTMENTS table. Raise an exception and print a message for the user that an invalid department number was entered.

```
CREATE OR REPLACE PROCEDURE upd_dept_desc(
        dept_id   departments.department_id%TYPE,
        dept_name departments.department_name%TYPE
)
IS
BEGIN
        UPDATE departments SET department_name=dept_name  WHERE
        department_id=dept_id;
        IF SQL%NOTFOUND THEN
                RAISE_APPLICATION_ERROR(-20001,'Invalid department number entered');
         END IF;
END;
/

Procedure created.
```

```
SQL> exec upd_dept_desc(101, 'Software Engineer');
BEGIN upd_dept_desc(101, 'Software Engineer'); END;

*
ERROR at line 1:
ORA-20001: Invalid department number entered
ORA-06512: at "HR.UPD_DEPT_DESC", line 9
ORA-06512: at line 1
```