Practice 1 – 1:

A and b:

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
CREATE OR REPLACE PACKAGE JOB_PKG IS
    PROCEDURE add_job(p_id jobs.job_id%TYPE, p_title jobs.job_title%TYPE);
    PROCEDURE del_job(p_id jobs.job_id%TYPE);
    FUNCTION get_job(p_id IN jobs.job_id%TYPE) RETURN jobs.job_title%TYPE;
    PROCEDURE upd_job(p_id jobs.job_id%TYPE, p_title jobs.job_title%TYPE);
END JOB_PKG;
/
SHOW ERRORS
```

Script Output — Task completed in 0.06 seconds

```
Package JOB_PKG compiled
```

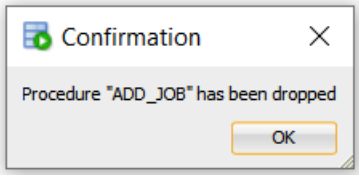
```
    FUNCTION get_job(p_id IN jobs.job_id%TYPE)
    RETURN jobs.job_title%TYPE IS v_title jobs.job_title%TYPE;
    BEGIN
        SELECT job_title
        INTO v_title
        FROM jobs
        WHERE job_id = p_id;
        RETURN v_title;
    END get_job;

    PROCEDURE UPD_JOB(p_id jobs.job_id%TYPE, p_title jobs.job_title%TYPE) IS
    BEGIN
        UPDATE JOBS
        SET job_title = p_title
        WHERE job_id = p_id;
        IF SQL%NOTFOUND THEN
            RAISE_APPLICATION_ERROR(-20202, 'NOTHING TO UPDATE');
        END IF;
    END UPD_JOB;
END JOB_PKG;
/
SHOW ERRORS
```

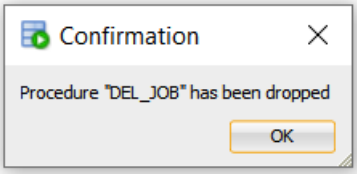
Script Output — Task completed in 0.051 seconds

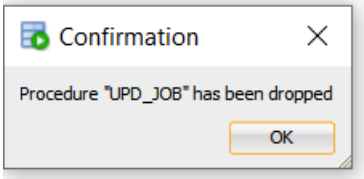
```
Package JOB_PKG compiled


Package Body JOB_PKG compiled
```

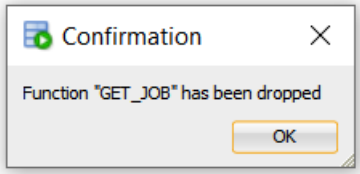
- Here in part a we are creating job_pkg specification with three procedures and one function headings. We are using procedures: add_job, del_job, get_job and upd_job function headings with formal parameters p_jobid with type of job_id of JOBS table.
- Here we have used SHOW ERROR, which will give us errors occurred in compilation time.
- As you can see here package job_pkg s compiled successfully for specification as well as body.

Practice Question 1 - C:









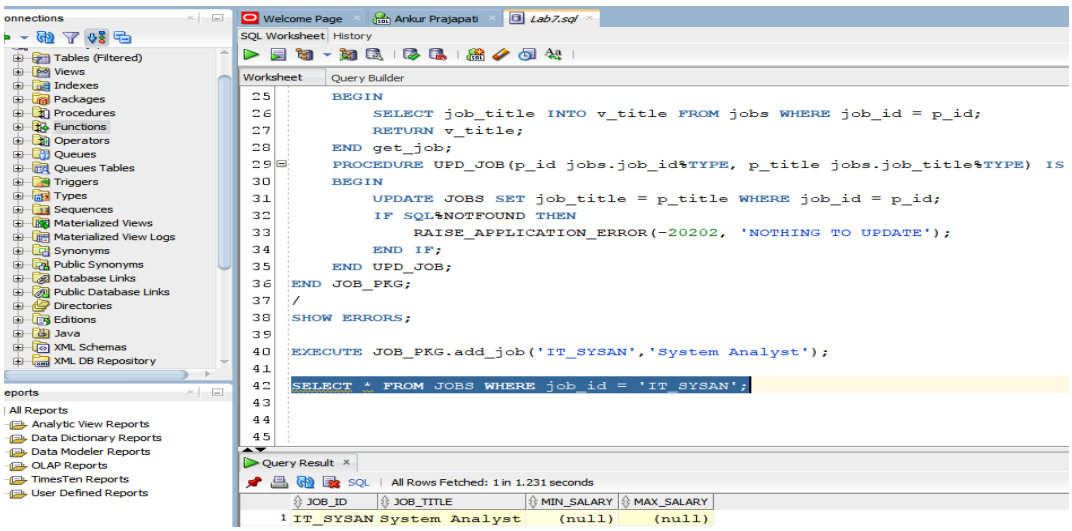
Dropping of all the procedures.

Practice Question 1 - d:

Executing of job_pkg.add_job('IT_SYSAN', 'System Analyst');

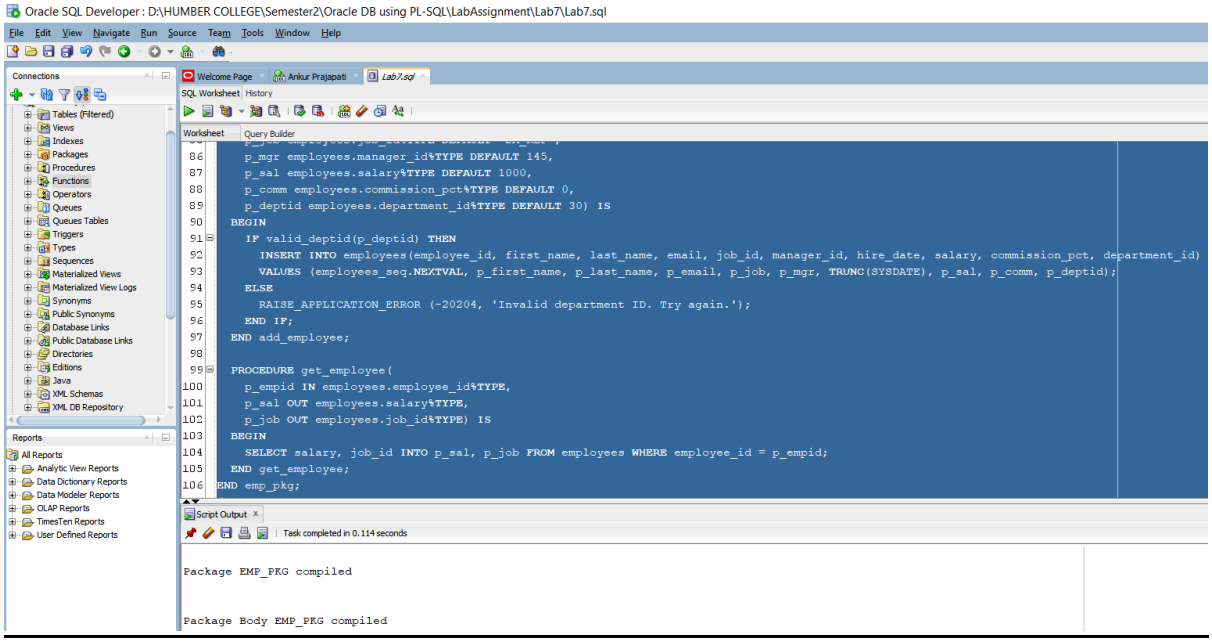
Basically, it adds job of System Analyst with job_id IT_SYSAN.

Practice Question 1 - e:



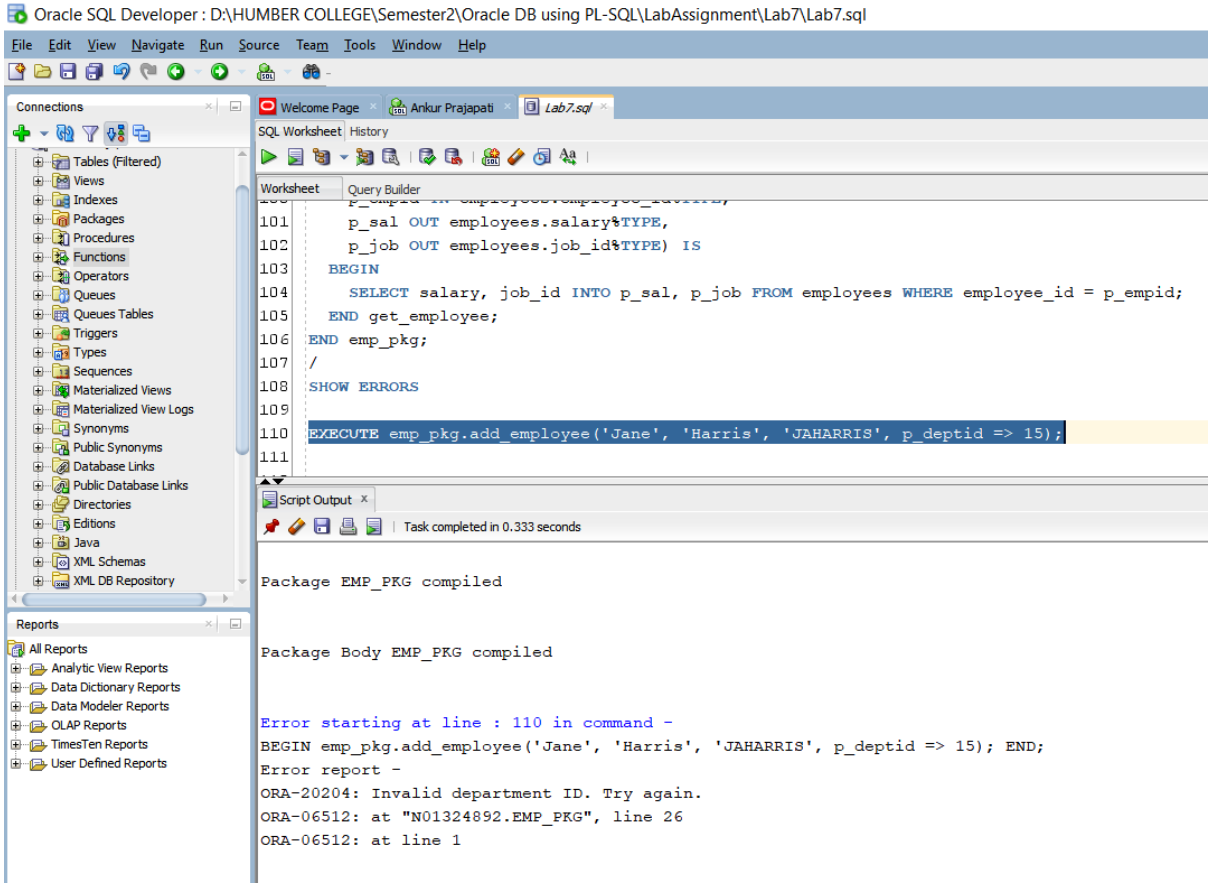
Here we are selecting job_id with IT_SYSAN. Basically, we are using select query.

Practice 1 – 2:

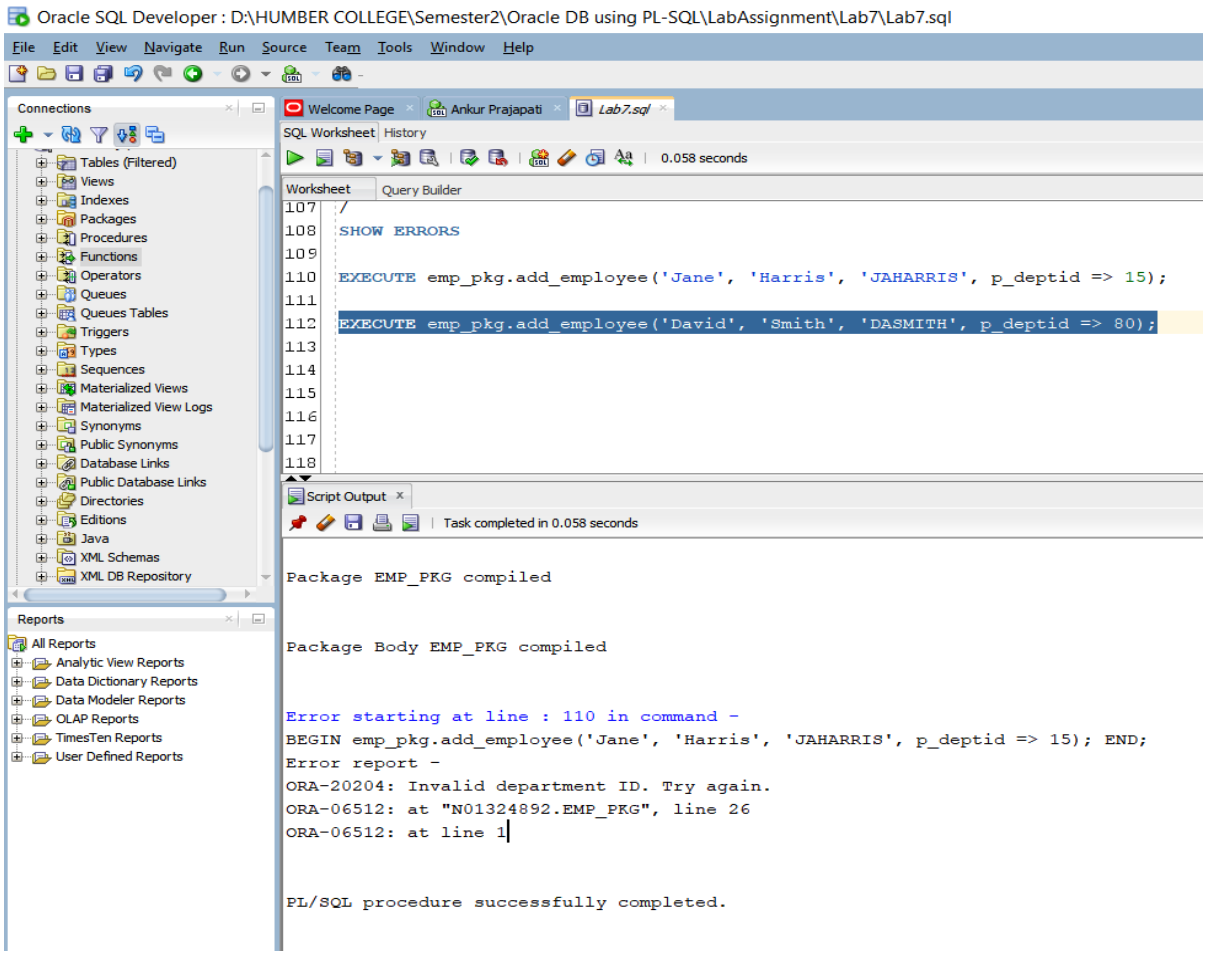


In this step we are basically adding add_employee and get_employee procedure with valid_deptid function in package specification and package body as public construct.

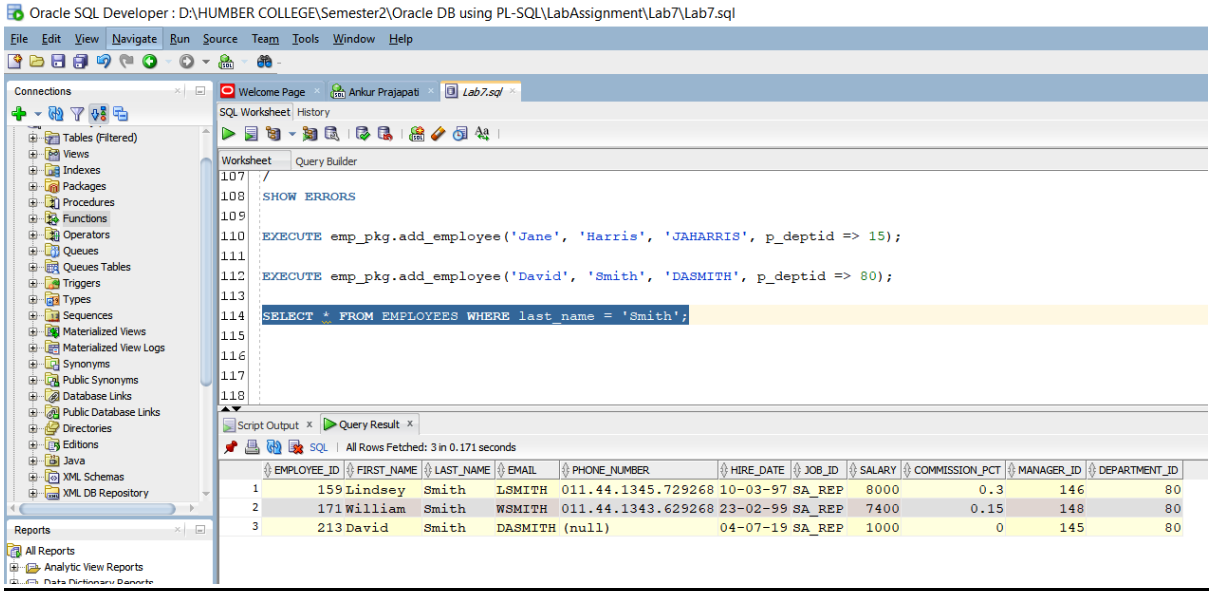
Here valid_deptid is private function that's why its not declared in specification.



Here we are invoking emp_pkg.add_employee to add new employee in database for department id 15. But it gives an error that department_id is invalid.



Same as before we are trying to add employee for department id 80. But it's valid so it's get added.



Printing all the employees where last name is Smith.

Practice 2 – 1:

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
CREATE OR REPLACE PACKAGE emp_pkg IS
  PROCEDURE add_employee(
    p_first_name employees.first_name%TYPE,
    p_last_name employees.last_name%TYPE,
    p_email employees.email%TYPE,
    p_job employees.job_id%TYPE DEFAULT 'SA_REP',
    p_mgr employees.manager_id%TYPE DEFAULT 145,
    p_sal employees.salary%TYPE DEFAULT 1000,
    p_comm employees.commission_pct%TYPE DEFAULT 0,
    p_deptid employees.department_id%TYPE DEFAULT 30);

/*---------------------------- New overloaded add_employee----------- */
  PROCEDURE add_employee(
    p_first_name employees.first_name%TYPE,
    p_last_name employees.last_name%TYPE,
    p_deptid employees.department_id%TYPE);
/*-----------------------------------------------------------------*/
  PROCEDURE get_employee(
    p_empid IN employees.employee_id%TYPE,
    p_sal OUT employees.salary%TYPE,
    p_job OUT employees.job_id%TYPE);
END emp_pkg;
/
```

```
Package EMP_PKG compiled
```

We are adding new procedure with 3 different parameters to provide overloading of add_employee.

It gets compiled successfully with overloading.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

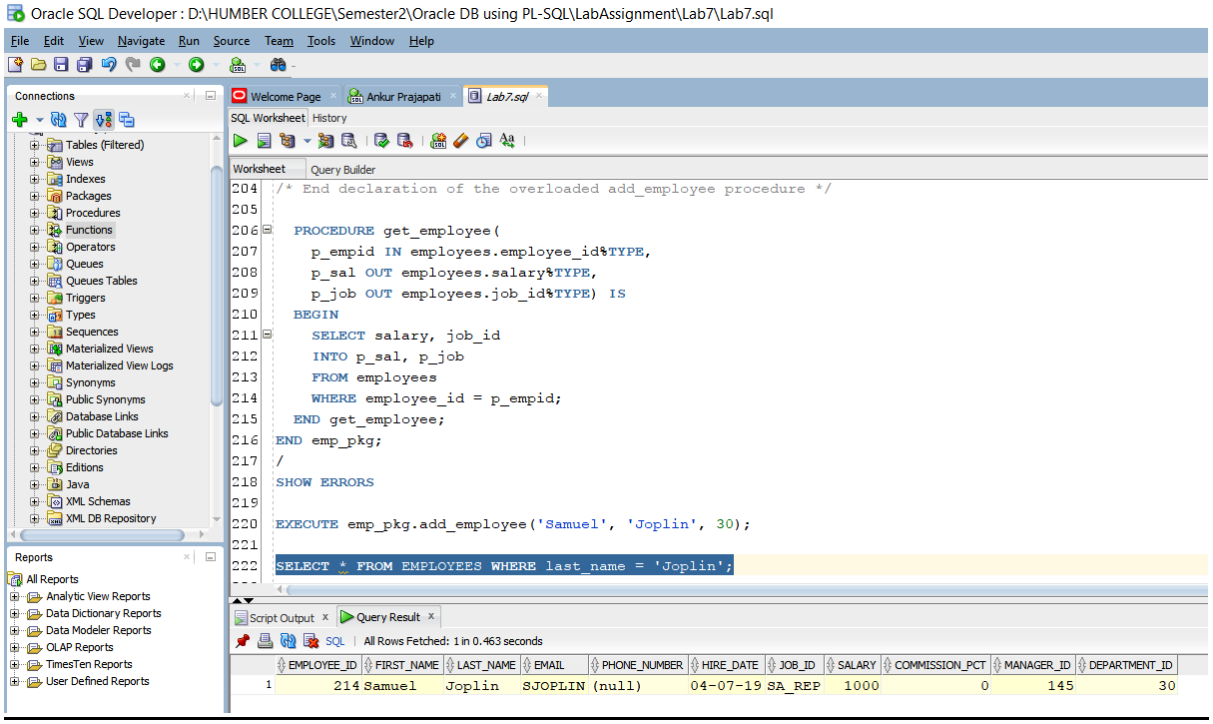
```
/*-----------------------------New overloaded add_employee procedure--*/

  PROCEDURE add_employee(
    p_first_name employees.first_name%TYPE,
    p_last_name employees.last_name%TYPE,
    p_deptid employees.department_id%TYPE) IS
    p_email employees.email%type;
  BEGIN
    p_email := UPPER(SUBSTR(p_first_name, 1, 1)||SUBSTR(p_last_name, 1, 7));
    add_employee(p_first_name, p_last_name, p_email, p_deptid => p_deptid);
  END;

/* End declaration of the overloaded add_employee procedure */

  PROCEDURE get_employee(
    p_empid IN employees.employee_id%TYPE,
    p_sal OUT employees.salary%TYPE,
    p_job OUT employees.job_id%TYPE) IS
  BEGIN
    SELECT salary, job_id
    INTO p_sal, p_job
    FROM employees
    WHERE employee_id = p_empid;
  END get_employee;
END emp_pkg;
/
SHOW ERRORS
```

```
Package EMP_PKG compiled


Package Body EMP_PKG compiled
```

Here new added overloaded procedure we are actually formatting email address in uppercase latter with first seven letters of last name.

We are calling procedure to insert operation.

On compiling it gets compiled successfully.



We have executed emp_pkg.add_employee with (‘Samuel’, ‘Joplin’, 30) to insert new record.

After that select query is used to print the inserted record where last_name is ‘Joplin’.

Note that the email is SJOPLIN here from first name and last name.

Practice 2 – 2:

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
------------------------------------------------------------------------Practice 2 - 2 ------

CREATE OR REPLACE PACKAGE emp_pkg IS
  PROCEDURE add_employee(
    p_first_name employees.first_name%TYPE,
    p_last_name employees.last_name%TYPE,
    p_email employees.email%TYPE,
    p_job employees.job_id%TYPE DEFAULT 'SA_REP',
    p_mgr employees.manager_id%TYPE DEFAULT 145,
    p_sal employees.salary%TYPE DEFAULT 1000,
    p_comm employees.commission_pct%TYPE DEFAULT 0,
    p_deptid employees.department_id%TYPE DEFAULT 30);
  PROCEDURE add_employee(
    p_first_name employees.first_name%TYPE,
    p_last_name employees.last_name%TYPE,
    p_deptid employees.department_id%TYPE);
  PROCEDURE get_employee(
    p_empid IN employees.employee_id%TYPE,
    p_sal OUT employees.salary%TYPE,
    p_job OUT employees.job_id%TYPE);
/* New overloaded get_employees functions specs starts here: */
  FUNCTION get_employee(p_emp_id employees.employee_id%type)
    return employees%rowtype;
  FUNCTION get_employee(p_family_name employees.last_name%type)
    return employees%rowtype;
/* New overloaded get_employees functions specs ends here. */
END emp_pkg;
/
SHOW ERRORS
```

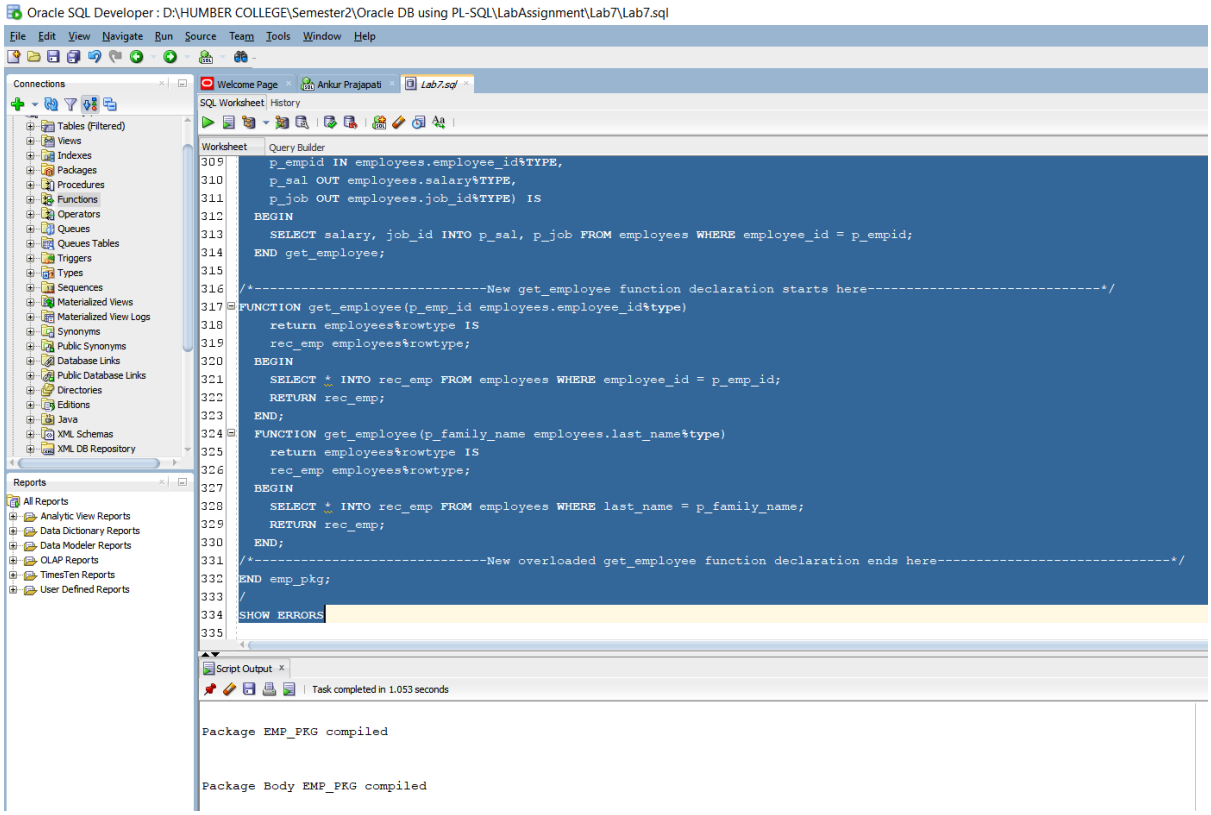
Task completed in 0.737 seconds

```
Package EMP_PKG compiled
```

Here get_employee function which accept parameters p_empid of type employee_id and it returns value of type employees%rowtype.

It also accept p_family_name in type of employees.last_name%type and returns employees%rowtype.

It gets complied successfully.



Two functions included in declaration with their parameters which returns record of type employees.

Where we are using employee's id to query in first function and family name in second function. Note that here = (equality operator is used) to find last name from family name.

It gets complied successfully.

Adding print_employee specification and body to package.

```
    p_sal OUT employees.salary%TYPE,
    p_job OUT employees.job_id%TYPE);
/*-----------------------------New overloaded get_employees functions specs starts here:-----------------------------*/
  FUNCTION get_employee(p_emp_id employees.employee_id%type)
    return employees%rowtype;
  FUNCTION get_employee(p_family_name employees.last_name%type)
    return employees%rowtype;
/*-----------------------------New overloaded get_employees functions specs ends here.-----------------------------*/
/*-----------------------------New print_employee print_employee procedure spec-----------------------------*/
PROCEDURE print_employee(p_rec_emp employees%rowtype);
END emp_pkg;
/
SHOW ERRORS


-- Package BODY
CREATE OR REPLACE PACKAGE BODY emp_pkg IS
```

```
  END;
/*-----------------------------New overloaded get_employee function declaration ends here-----------------------------*/
/*-----------------------------New print_employees procedure declaration.-----------------------------*/
PROCEDURE print_employee(p_rec_emp employees%rowtype) IS
  BEGIN
    DBMS_OUTPUT.PUT_LINE(p_rec_emp.department_id ||' '||
                         p_rec_emp.employee_id||' '||
                         p_rec_emp.first_name||' '||
                         p_rec_emp.last_name||' '||
                         p_rec_emp.job_id||' '||
                         p_rec_emp.salary);
  END;

END emp_pkg;
```

It displays department id, employee id, first name, last name, job id and salary.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
    RETURN rec_emp;
  END;
/*-----------------------------New overloaded get_employee function declaration ends here-
/*-----------------------------New print_employees procedure declaration.-----------------
PROCEDURE print_employee(p_rec_emp employees%rowtype) IS
  BEGIN
    DBMS_OUTPUT.PUT_LINE(p_rec_emp.department_id ||' '||
                         p_rec_emp.employee_id||' '||
                         p_rec_emp.first_name||' '||
                         p_rec_emp.last_name||' '||
                         p_rec_emp.job_id||' '||
                         p_rec_emp.salary);
  END;

END emp_pkg;
```

```
Package EMP_PKG compiled


Package Body EMP_PKG compiled
```

It gets complied successfully.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
                        p_rec_emp.employee_id||' '||
                        p_rec_emp.first_name||' '||
                        p_rec_emp.last_name||' '||
                        p_rec_emp.job_id||' '||
                        p_rec_emp.salary);
  END;

END emp_pkg;
/
SHOW ERRORS
SET SERVEROUTPUT ON
BEGIN
    emp_pkg.print_employee(emp_pkg.get_employee(100));
    emp_pkg.print_employee(emp_pkg.get_employee('Joplin'));
END;
/
```

```
90 100 Steven King AD_PRES 24000
30 214 Samuel Joplin SA_REP 1000


PL/SQL procedure successfully completed.
```

Use of anonymous block to invoke emp_pkg.print_employee with employee id and last name.

Practice 2 – 3:

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
243   PROCEDURE add_employee(
244     p_first_name employees.first_name%TYPE,
245     p_last_name employees.last_name%TYPE,
246     p_email employees.email%TYPE,
247     p_job employees.job_id%TYPE DEFAULT 'SA_REP',
248     p_mgr employees.manager_id%TYPE DEFAULT 145,
249     p_sal employees.salary%TYPE DEFAULT 1000,
250     p_comm employees.commission_pct%TYPE DEFAULT 0,
251     p_deptid employees.department_id%TYPE DEFAULT 30);
252   PROCEDURE add_employee(
253     p_first_name employees.first_name%TYPE,
254     p_last_name employees.last_name%TYPE,
255     p_deptid employees.department_id%TYPE);
256   PROCEDURE get_employee(
257     p_empid IN employees.employee_id%TYPE,
258     p_sal OUT employees.salary%TYPE,
259     p_job OUT employees.job_id%TYPE);
260 /*----------------------------New overloaded get_employees functions specs starts here:-----------------------------*/
261   FUNCTION get_employee(p_emp_id employees.employee_id%type)
262     return employees%rowtype;
263   FUNCTION get_employee(p_family_name employees.last_name%type)
264     return employees%rowtype;
265 /*----------------------------New overloaded get_employees functions specs ends here.-----------------------------*/
266
267 /*----------------------------New procedure init_departments spec------------------------------------------ */
268 PROCEDURE init_departments;
269 /*----------------------------New print_employee print_employee procedure spec----------------------------*/
270 PROCEDURE print_employee(p_rec_emp employees%rowtype);
271 END emp_pkg;
272 /
273 SHOW ERRORS
274
```

We are adding public procedure to emp_pkg to improve its' performance. Here init_departments procedure is created in package specification.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
273 SHOW ERRORS
274
275
276 -- Package BODY
277 CREATE OR REPLACE PACKAGE BODY emp_pkg IS
278
279 /*-------------------------------------New type-----------------------------*/
280
281 TYPE boolean_tab_type IS TABLE OF BOOLEAN
282           INDEX BY BINARY_INTEGER;
283   valid_departments boolean_tab_type;
284
285   FUNCTION valid_deptid(p_deptid IN departments.department_id%TYPE) RETURN BOOLEAN IS
286     v_dummy PLS_INTEGER;
287   BEGIN
288     SELECT 1 INTO v_dummy FROM departments WHERE department_id = p_deptid;
289     RETURN TRUE;
290   EXCEPTION
291     WHEN NO_DATA_FOUND THEN
292     RETURN FALSE;
293 END valid_deptid;
294
```

Implementation of init_departments procedures to store all department ids in private PL/SQL block. Boolean_tab_type is a table which has binary_integer for it's index where valid_departments is a variable of that table.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

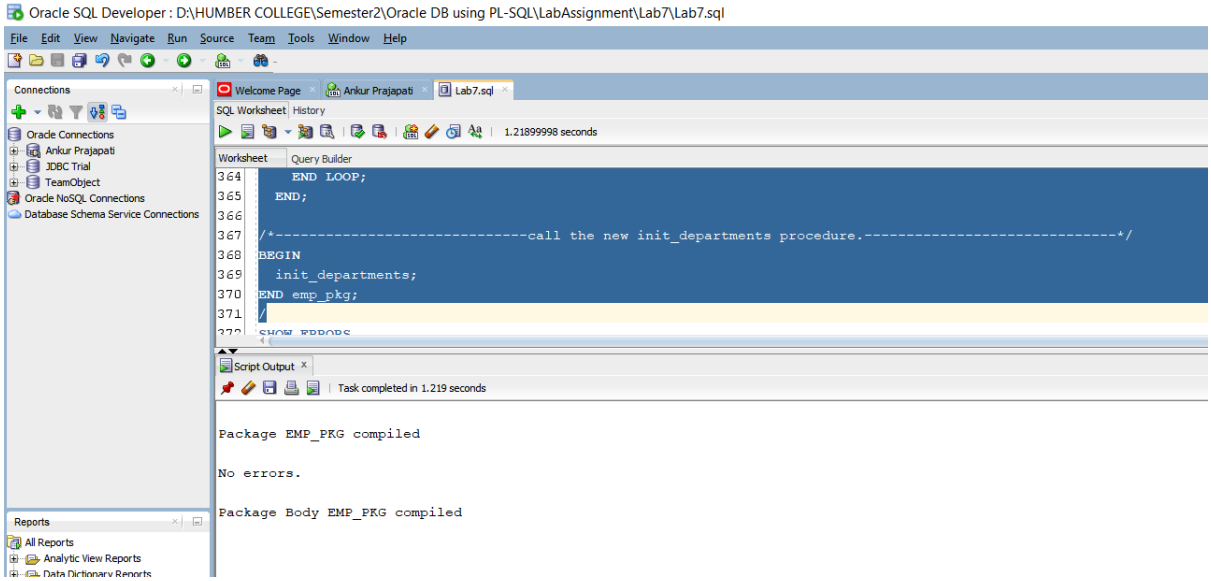
```
                        p_rec_emp.employee_id||' '||
                        p_rec_emp.first_name||' '||
                        p_rec_emp.last_name||' '||
                        p_rec_emp.job_id||' '||
                        p_rec_emp.salary);
  END;

/*-----------------------------New init_departments procedure declaration.-----------------------------*/
PROCEDURE init_departments IS
  BEGIN
    FOR rec IN (SELECT department_id FROM departments)
    LOOP
      valid_departments(rec.department_id) := TRUE;
    END LOOP;
  END;
```

In this procedure init_departments it takes department_id from departments and it prints/sets true for each department id using FOR LOOP.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
                        p_rec_emp.employee_id||' '||
                        p_rec_emp.first_name||' '||
                        p_rec_emp.last_name||' '||
                        p_rec_emp.job_id||' '||
                        p_rec_emp.salary);
  END;

/*-----------------------------New init_departments procedure declaration.-----------------------------*/
PROCEDURE init_departments IS
  BEGIN
    FOR rec IN (SELECT department_id FROM departments)
    LOOP
      valid_departments(rec.department_id) := TRUE;
    END LOOP;
  END;

/*-----------------------------call the new init_departments procedure.-----------------------------*/
BEGIN
  init_departments;
END;
```

Creating initial block to call procedure init_departments.



Compiling emp_pkg and It gets complied successfully.

Practice 2 – 4:

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
PROCEDURE init_departments;
/*-----------------------------New print_employee print_employee procedure spec------------
PROCEDURE print_employee(p_rec_emp employees%rowtype);
END emp_pkg;
/
SHOW ERRORS


-- Package BODY
CREATE OR REPLACE PACKAGE BODY emp_pkg IS
/*-------------------------------------New type------------------------------*/
TYPE boolean_tab_type IS TABLE OF BOOLEAN
          INDEX BY BINARY_INTEGER;
  valid_departments boolean_tab_type;

  FUNCTION valid_deptid(p_deptid IN departments.department_id%TYPE) RETURN BOOLEAN IS
    v_dummy PLS_INTEGER;
  BEGIN
    RETURN valid_departments.exists(p_deptid);
  EXCEPTION
    WHEN NO_DATA_FOUND THEN
    RETURN FALSE;
END valid_deptid;
```

Modification of valid_departmentid instead of looping through to find valid department_id here we are calling our index type variable.

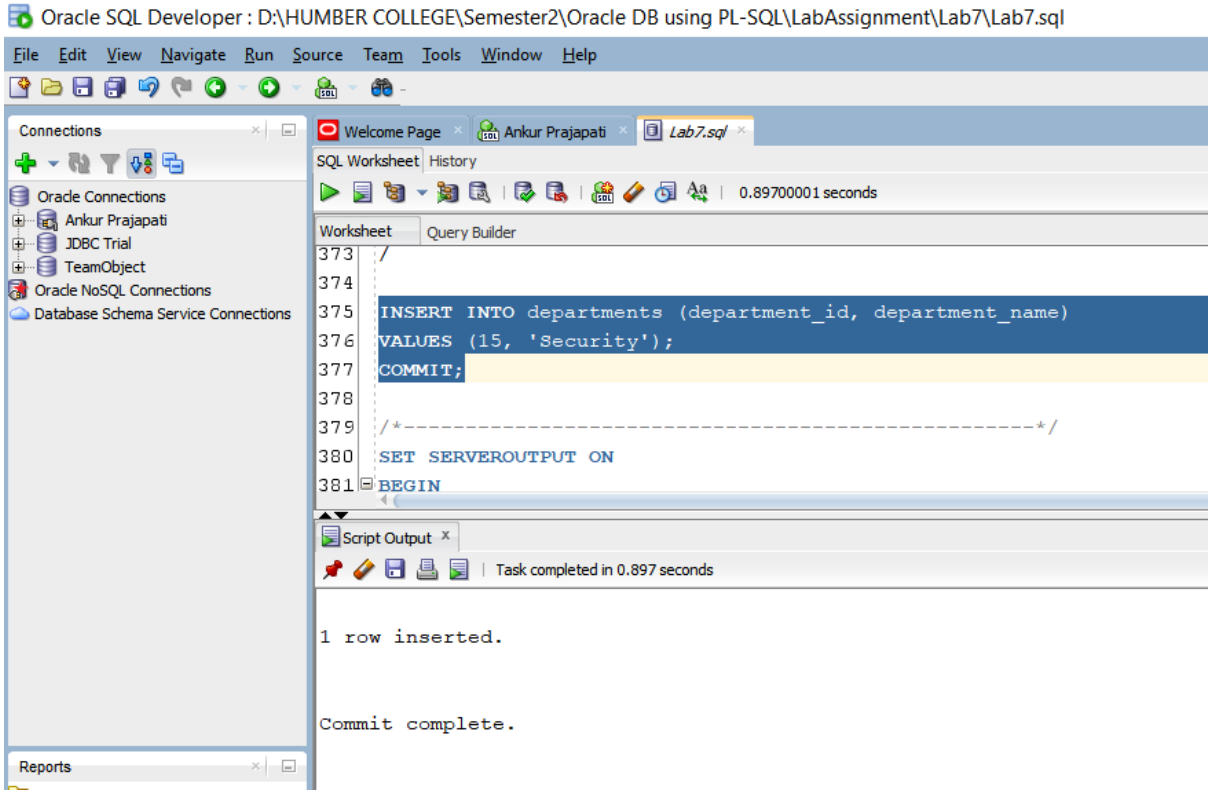
Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7

```
366    init_departments;
367  END emp_pkg;
368  /
369  SHOW ERRORS
370
371  SET SERVEROUTPUT ON
372  EXECUTE emp_pkg.add_employee('James', 'Bond', 15);
373  /
374  /*----------------------------------------------
```

```
Error starting at line : 372 in command -
BEGIN emp_pkg.add_employee('James', 'Bond', 15); END;
Error report -
ORA-20204: Invalid department ID. Try again.
ORA-06512: at "N01324892.EMP_PKG", line 30
ORA-06512: at "N01324892.EMP_PKG", line 41
ORA-06512: at line 1
```

We are trying to add new employee record with first name James and alst name of Bond and department id 15.

It gives an error because department_id is invalid.



For that we have to add new department with department id 15 and department name 'Security'.

Note that we have to commit to apply changes to database.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
SET SERVEROUTPUT ON
EXECUTE emp_pkg.add_employee('James', 'Bond', 15);
/

INSERT INTO departments (department_id, department_name)
VALUES (15, 'Security');
COMMIT;
```

```
1 row inserted.


Commit complete.


Error starting at line : 372 in command -
BEGIN emp_pkg.add_employee('James', 'Bond', 15); END;
Error report -
ORA-20204: Invalid department ID. Try again.
ORA-06512: at "N01324892.EMP_PKG", line 30
ORA-06512: at "N01324892.EMP_PKG", line 41
ORA-06512: at line 1
```

Again, we are trying to add this record here but it's not added successfully.

Note that this department id of 15 with department name security is not there in init_departments.

To add this record successfully we have to call init_departments to initialize and add this department id 15 to table.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
372
373 EXECUTE EMP_PKG.INIT_DEPARTMENTS;
374
375 EXECUTE emp_pkg.add_employee('James', 'Bond', 15);
376 /
377
378
379 INSERT INTO departments (department_id, department_name)
380 VALUES (15, 'Security');
381 COMMIT;
382
383 /*-----------------------------------------------------*/
384 SET SERVEROUTPUT ON
385 BEGIN
386     emp_pkg.print_employee(emp_pkg.get_employee(100));
387     emp_pkg.print_employee(emp_pkg.get_employee('Joplin'));
388 END;
```

```
PL/SQL procedure successfully completed.

PL/SQL procedure successfully completed.
```

Calling of init_departments.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
372
373 EXECUTE EMP_PKG.INIT_DEPARTMENTS;
374
375 EXECUTE emp_pkg.add_employee('James', 'Bond', 15);
376 /
377
378 SELECT * FROM EMPLOYEES WHERE department_id = 15;
379 INSERT INTO departments (department_id, department_name)
380 VALUES (15, 'Security');
381 COMMIT;
382
383 /*-----------------------------------------------------*/
384 SET SERVEROUTPUT ON
385 BEGIN
386     emp_pkg.print_employee(emp_pkg.get_employee(100));
387     emp_pkg.print_employee(emp_pkg.get_employee('Joplin'));
388 END;
```

```
EMPLOYEE_ID FIRST_NAME           LAST_NAME                EMAIL                    PHONE_NUMBER         HIRE_DAT JOB_ID     SALARY COMMISSION_PCT MANAGER_ID DEPARTMENT_ID
----------- -------------------- ------------------------ ------------------------ -------------------- -------- --------- ------ -------------- ---------- -------------
        217 James                Bond                     JBOND                                         08-07-19 SA_REP      1000              0        145            15
```

Select Query to get the data from to employees where department id is 15.

Note that we can now see the data here because we have called init_departments here first.

Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql

```
EXECUTE EMP_PKG.INIT_DEPARTMENTS;

EXECUTE emp_pkg.add_employee('James', 'Bond', 15);
/
/*-----------------------------------------------------------*/
SELECT * FROM EMPLOYEES WHERE department_id = 15;
INSERT INTO departments (department_id, department_name) VALUES (15, 'Securit
COMMIT;
/*---------------------------Practice 2 - 4 - g*/
DELETE FROM employees
WHERE first_name = 'James' AND last_name = 'Bond';
DELETE FROM departments WHERE department_id = 15;
COMMIT;
EXECUTE EMP_PKG.INIT_DEPARTMENTS

/*-----------------------------------------------*/
```

```
0 rows deleted.

1 row deleted.

Commit complete.

PL/SQL procedure successfully completed.
```

Now here we are deleting from employees where first name is ‘James’ and last name is ‘Bond’.

We are also deleting record from departments also where department id is 15.

After that commit is used to save changes.

After that we are calling init_departments again to store the data changes in database.