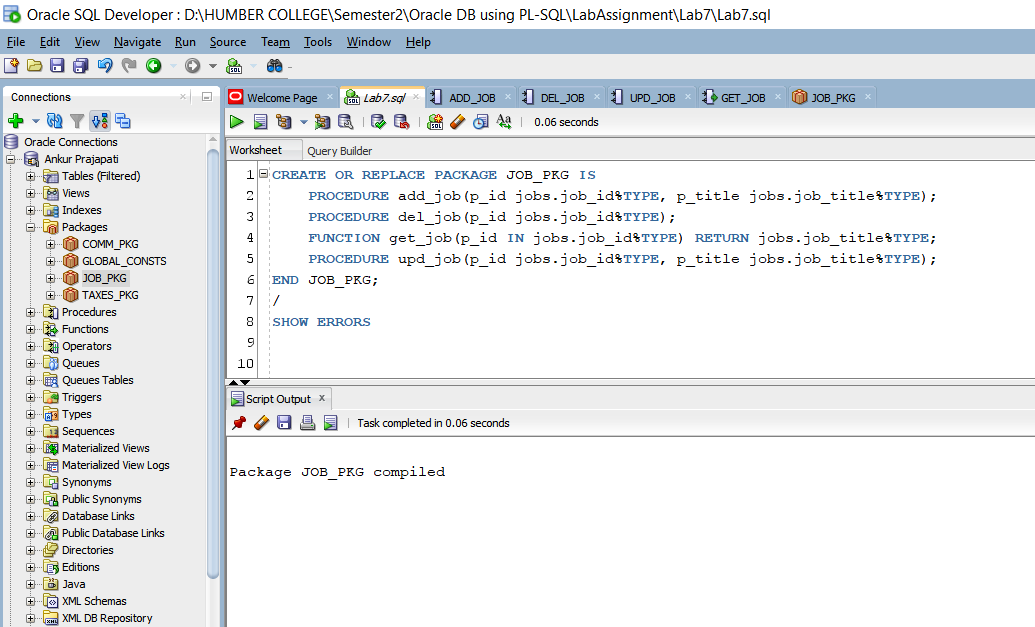
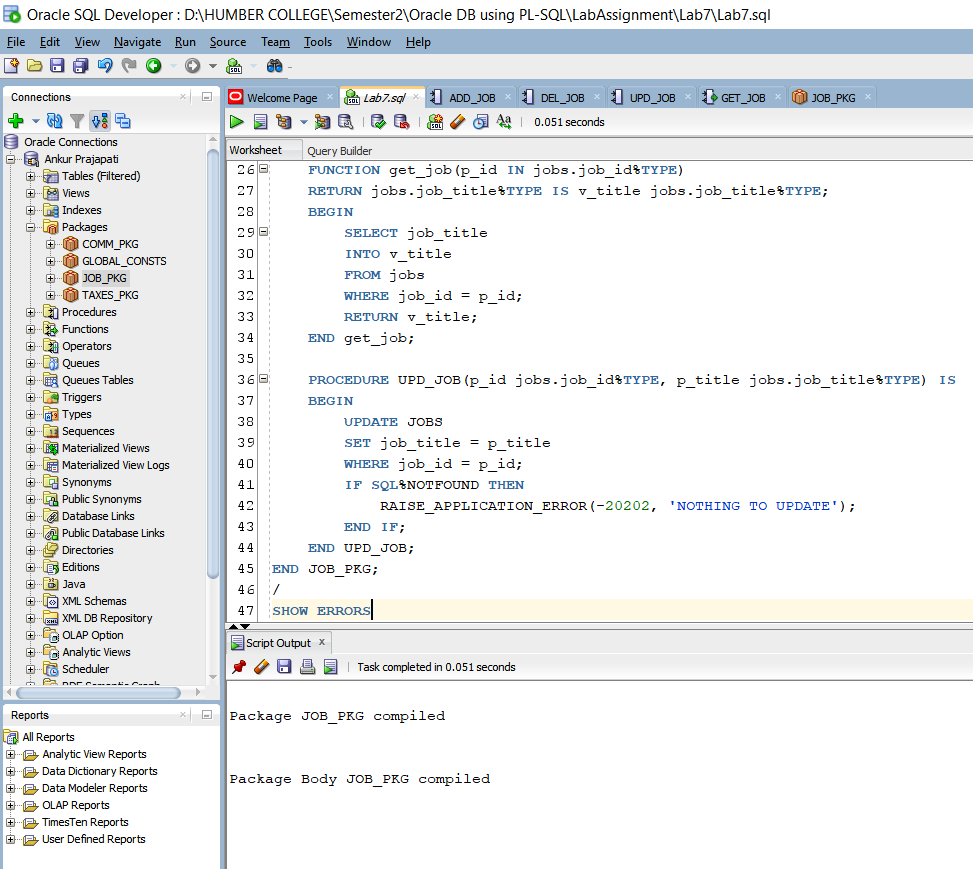
**Practice 1 – 1:**

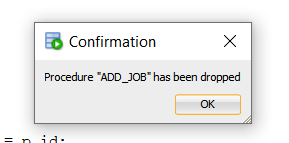
**A and b:**

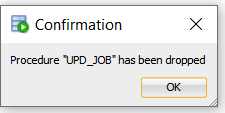
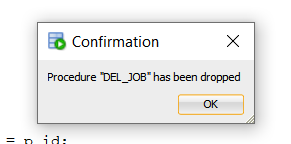
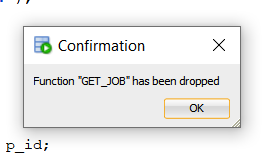




* Here in part a we are creating job\_pkg specification with three procedures and one function headings. We are using procedures: add\_job, del\_job, get\_job and upd\_job function headings with formal parameters p\_jobid with type of job\_id of JOBS table.
* Here we have used SHOW ERROR, which will 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:**

****



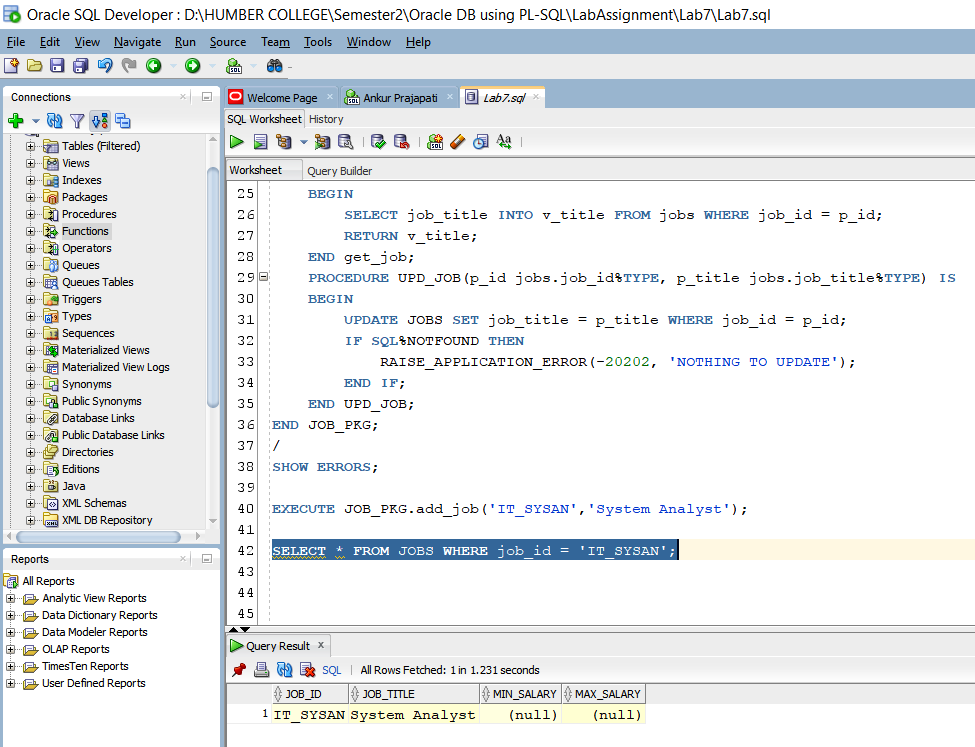
Dropping of all the procedures.

**Practice Question 1 - d:**

Executing of job\_pkg.add\_job(‘IT\_SYSAN’, ‘System Analyst’);

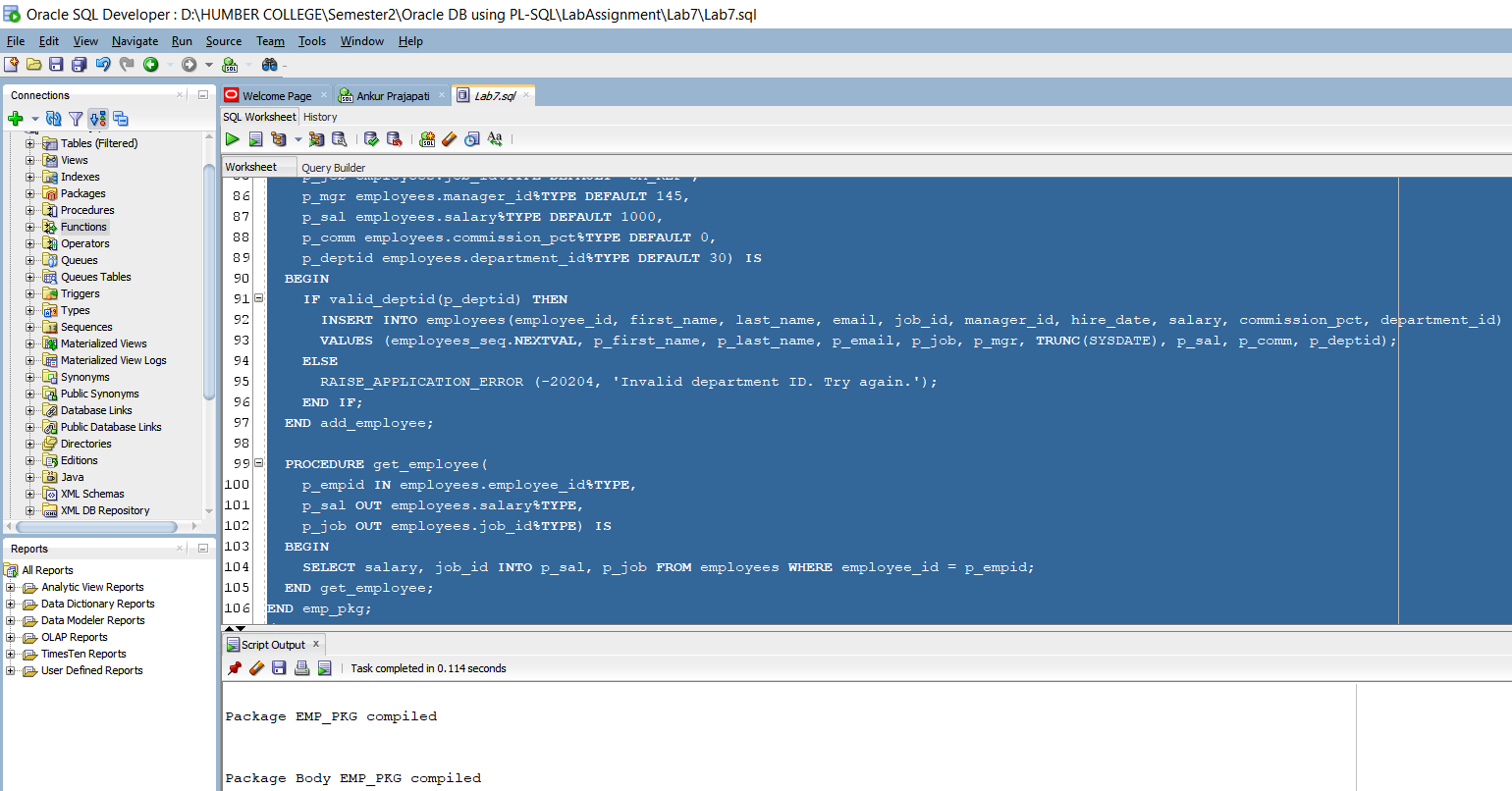
Basically, it adds job of System Analyst with job\_id IT\_SYSAN.

**Practice Question 1 - e:**

****

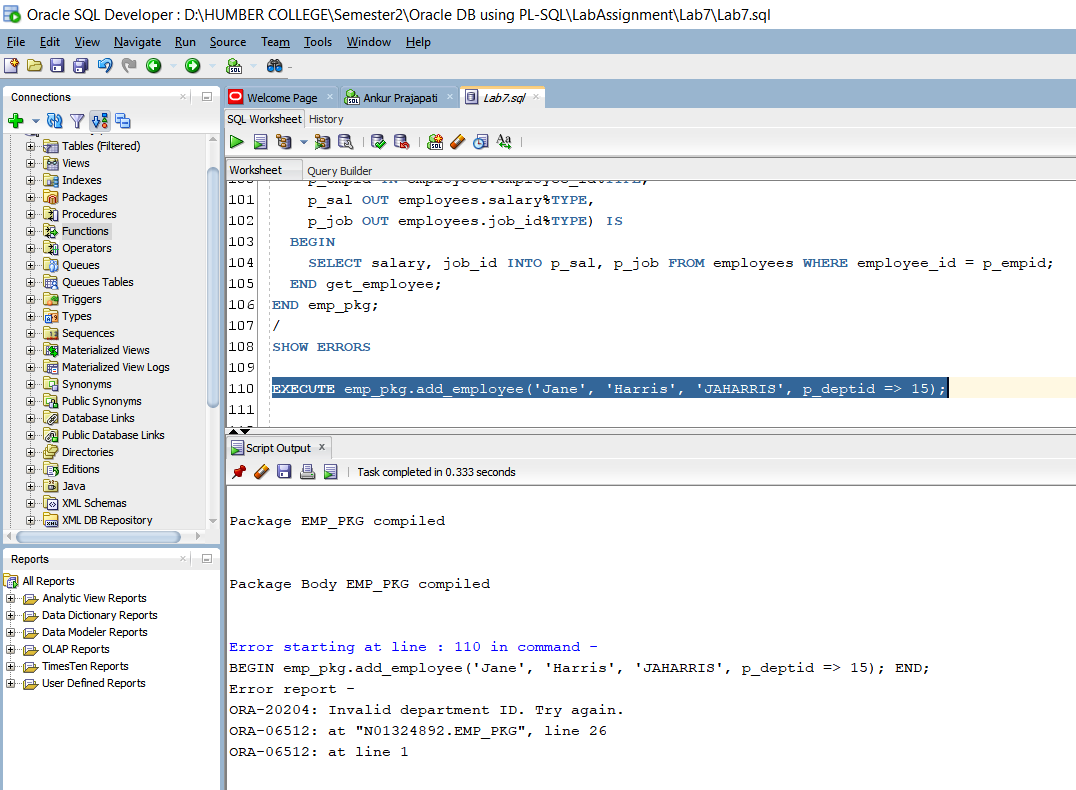
Here we are selecting job\_id with IT\_SYSAN. Basically, we are using select query.

**Practice 1 – 2:**

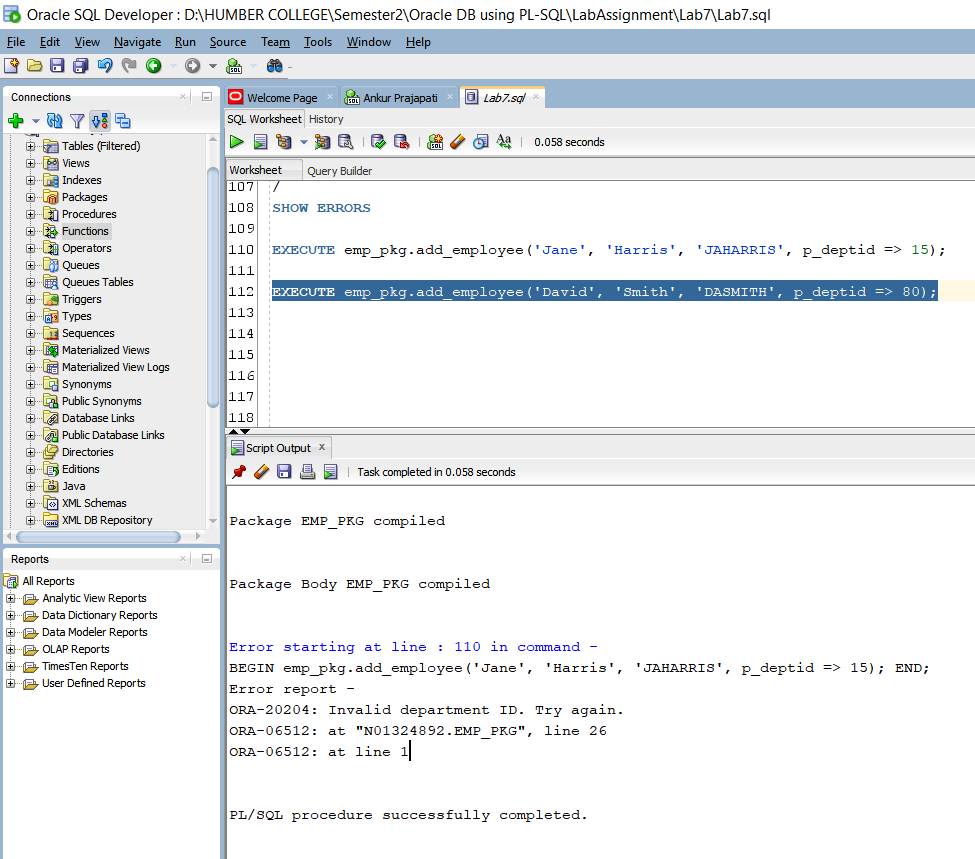
****

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.

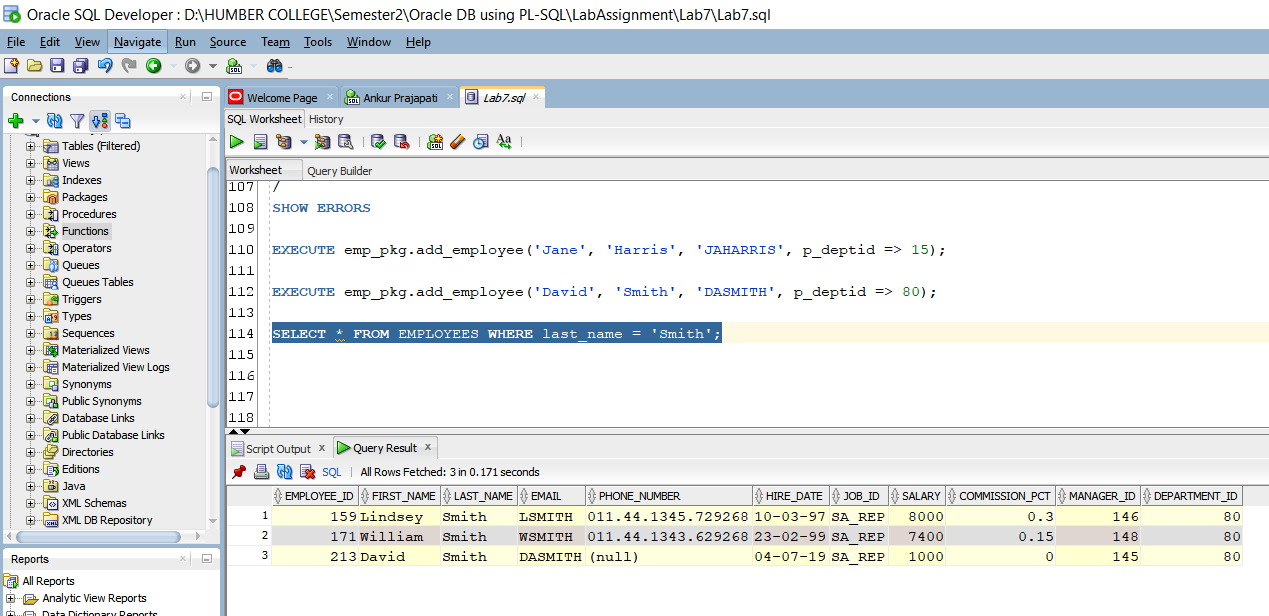
Here valid\_deptid is private function that’s why its not declared in specification.

****

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.

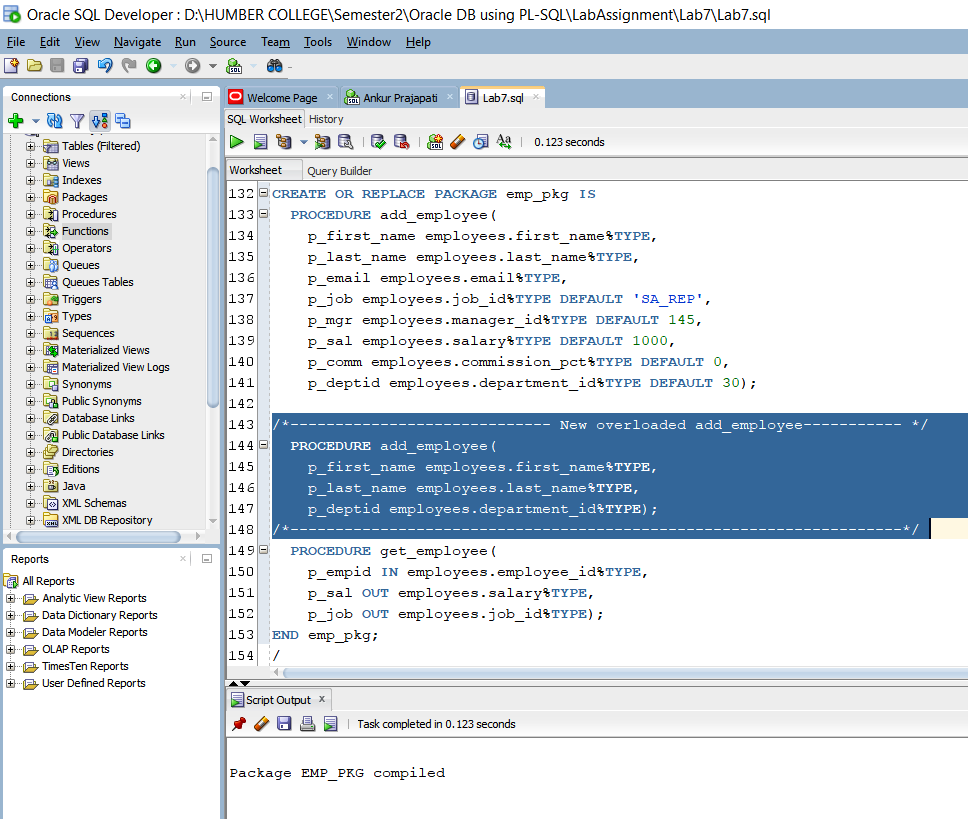
****

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

****

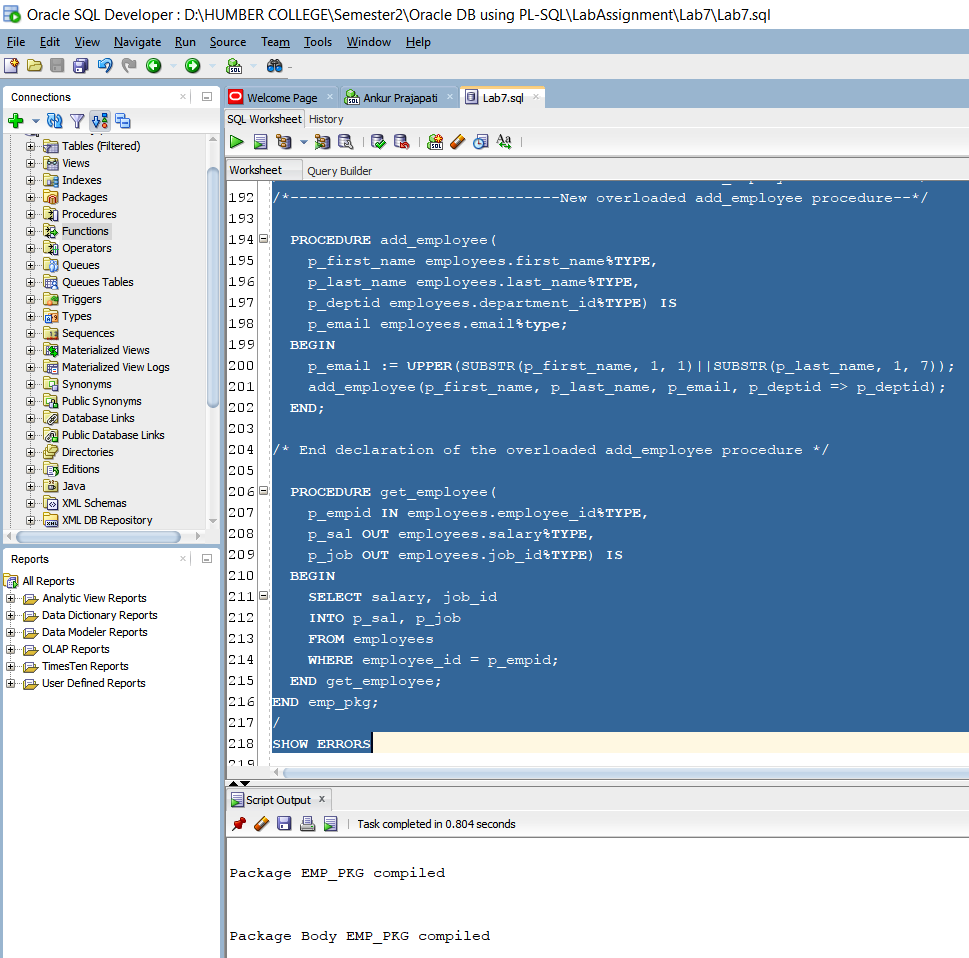
Printing all the employees where last name is Smith.

**Practice 2 – 1:**

****

We are adding new procedure with 3 different parameters to provide overloading of add\_employee.

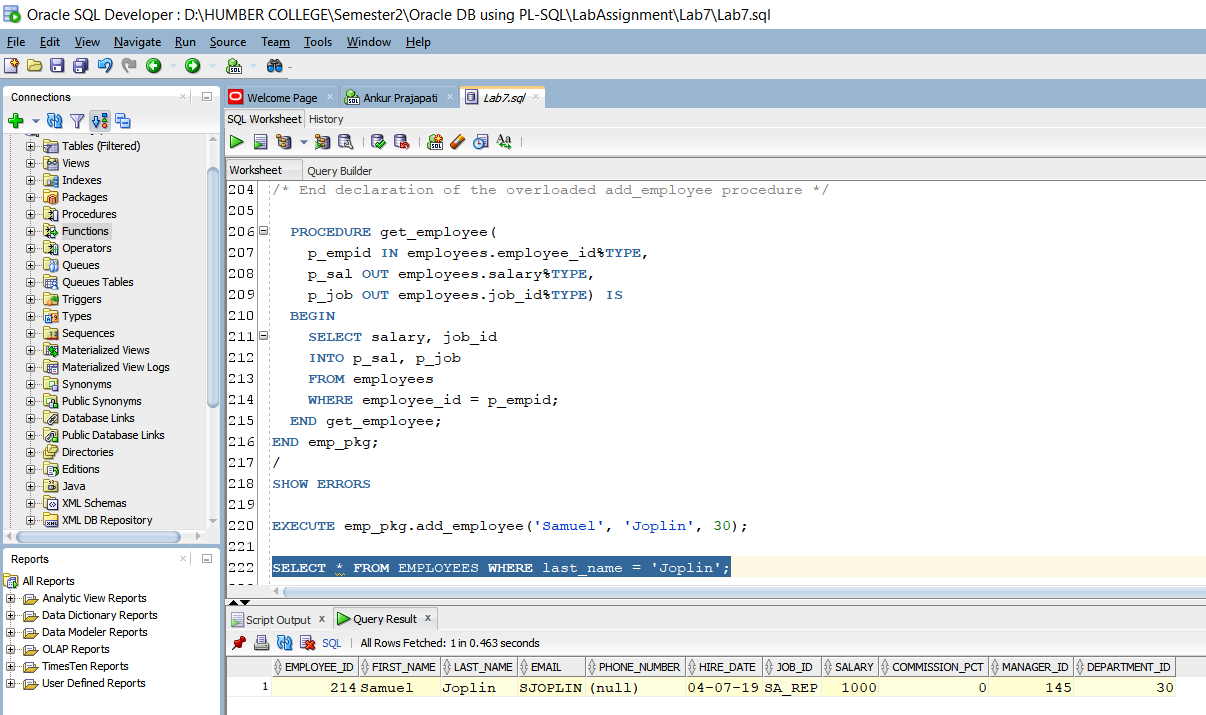
It gets compiled successfully with overloading.

****

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.

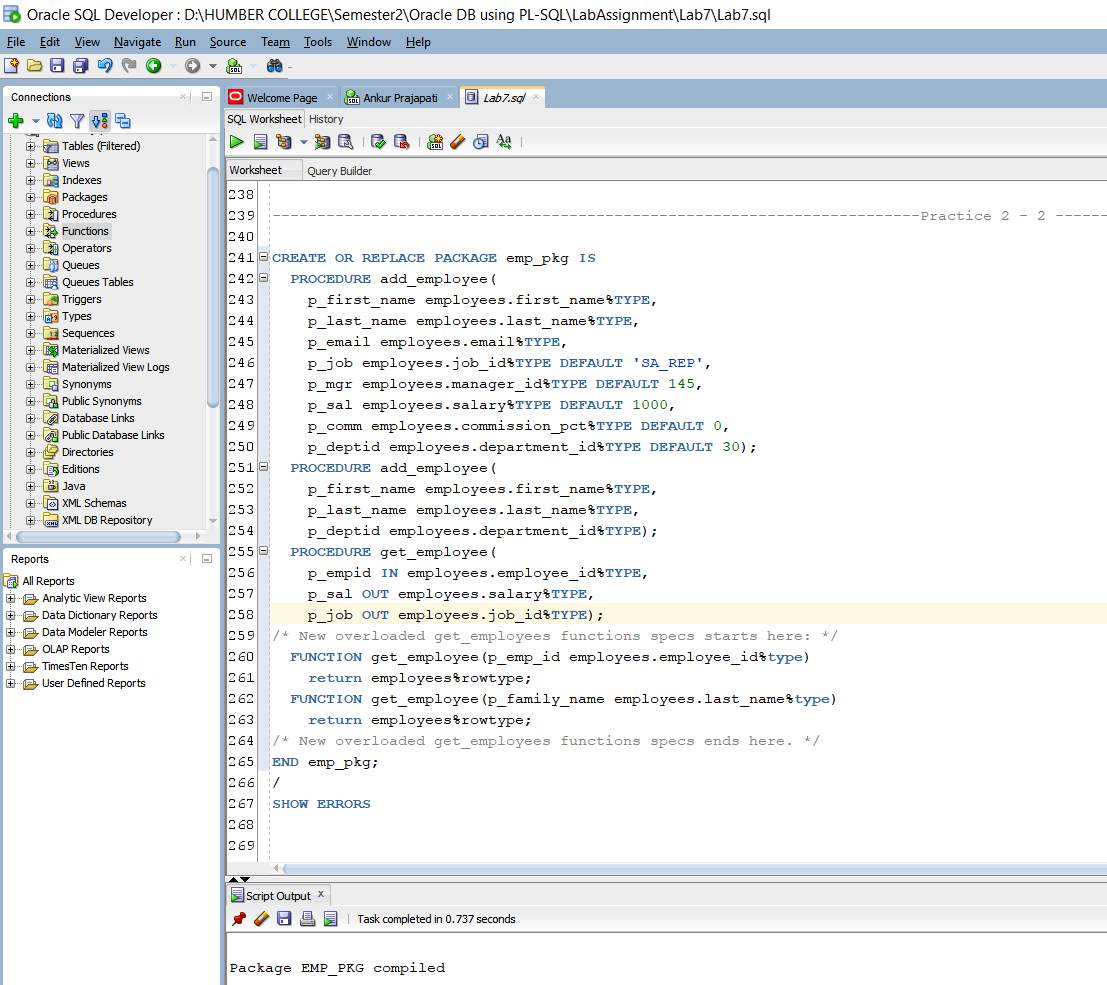
****

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.

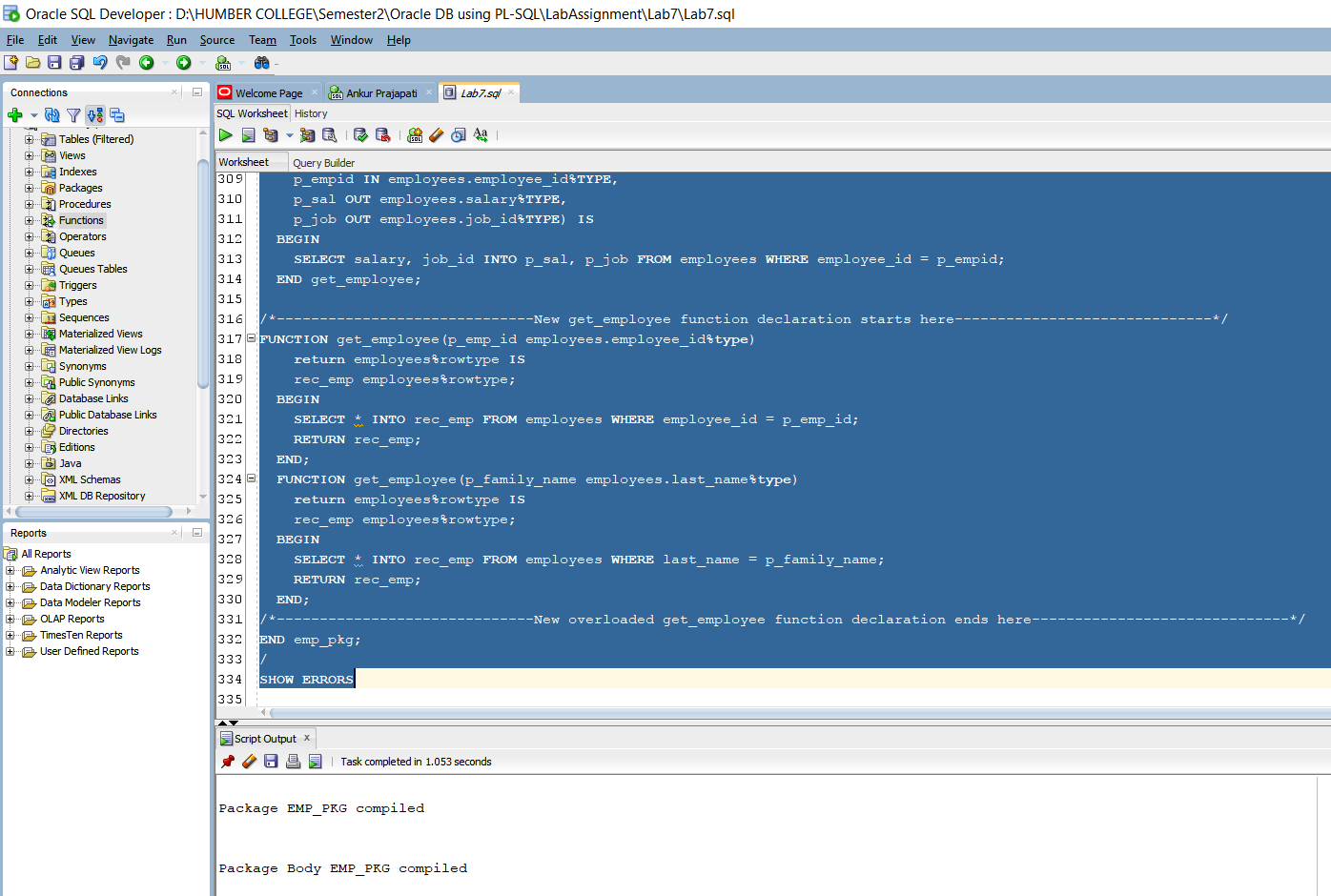
Practice 2 – 2:



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.

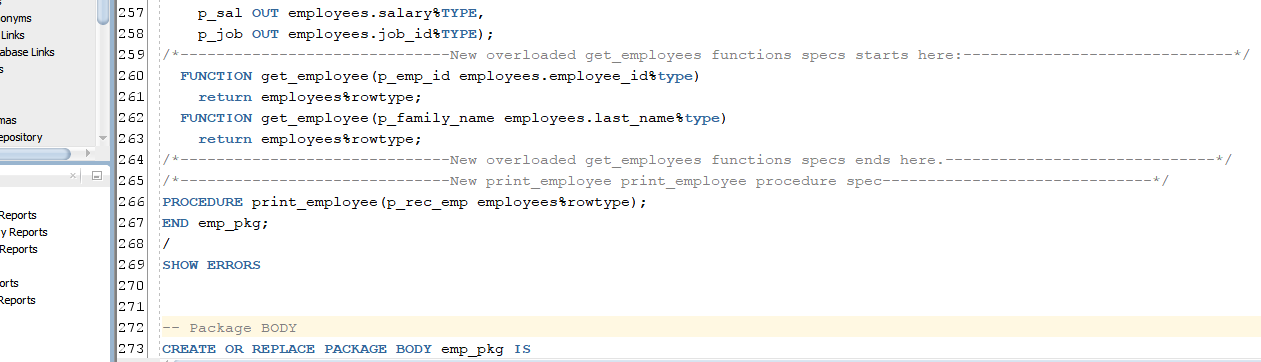


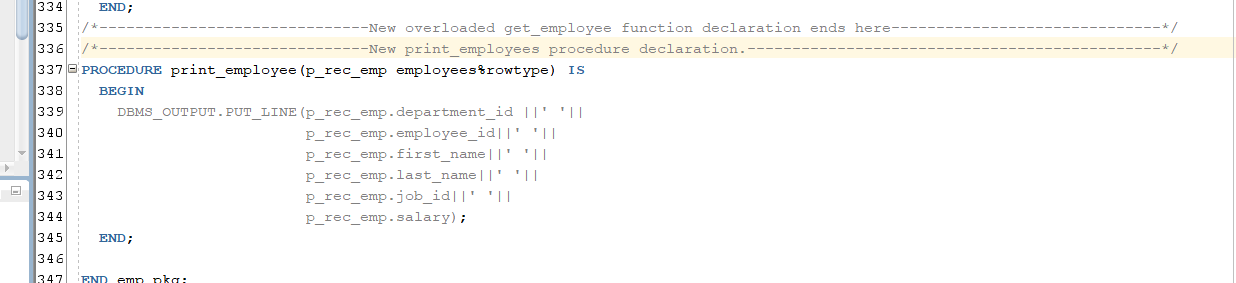
Two functions included in declaration with their parameters which returns record of type employees.

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.

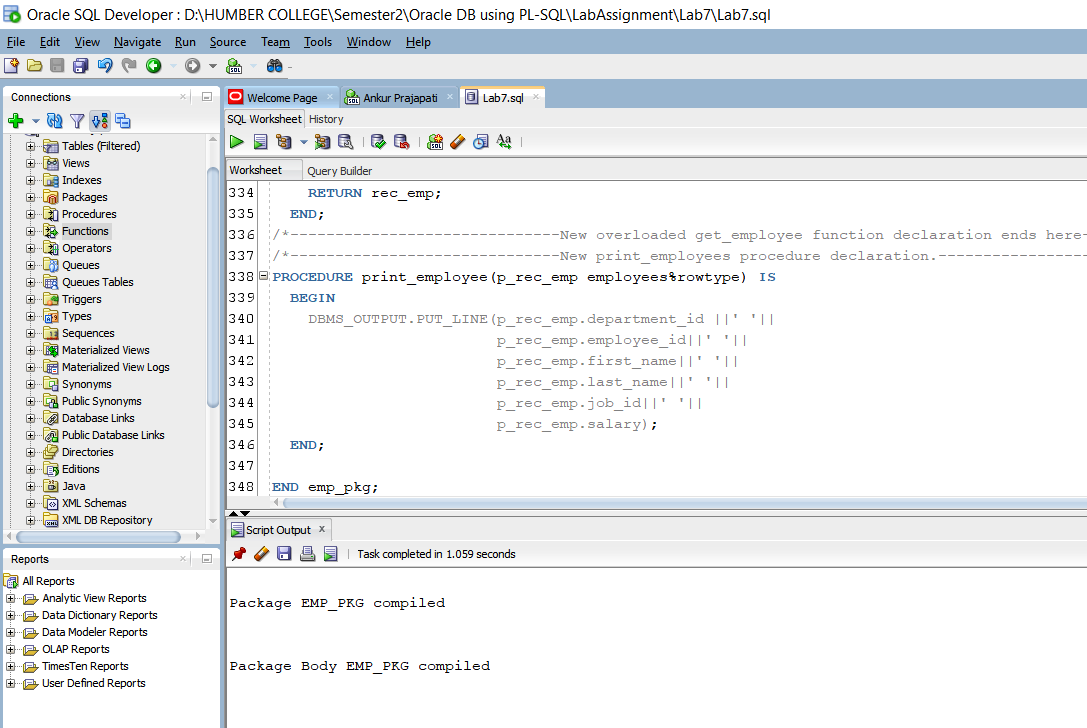
It gets complied successfully.

Adding print\_employee specification and body to package.

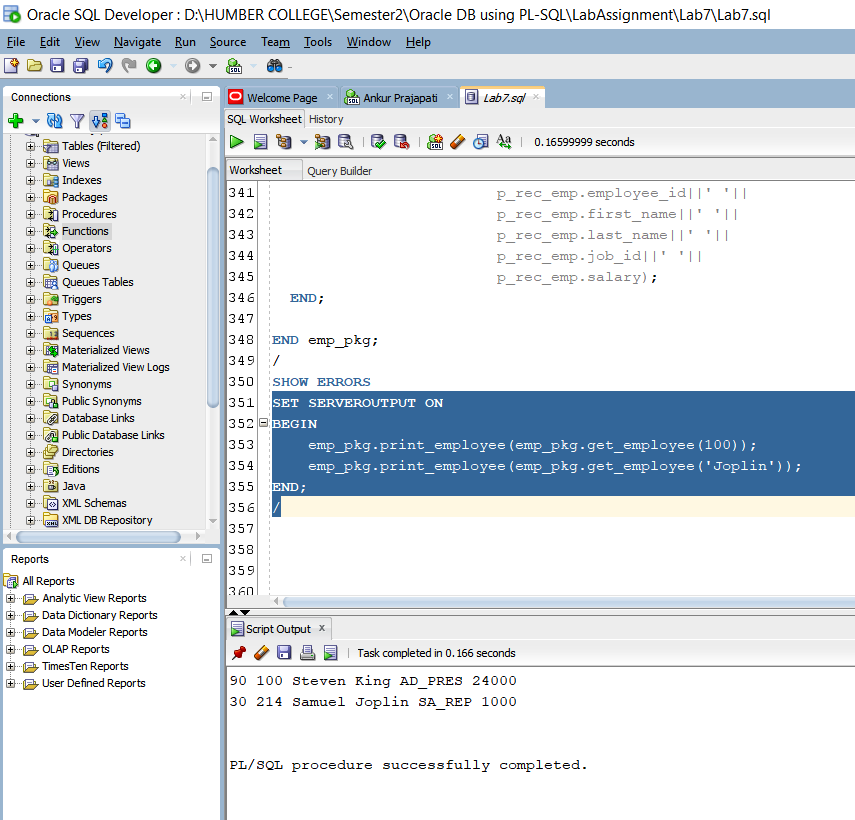




It displays department id, employee id, first name, last name, job id and salary.

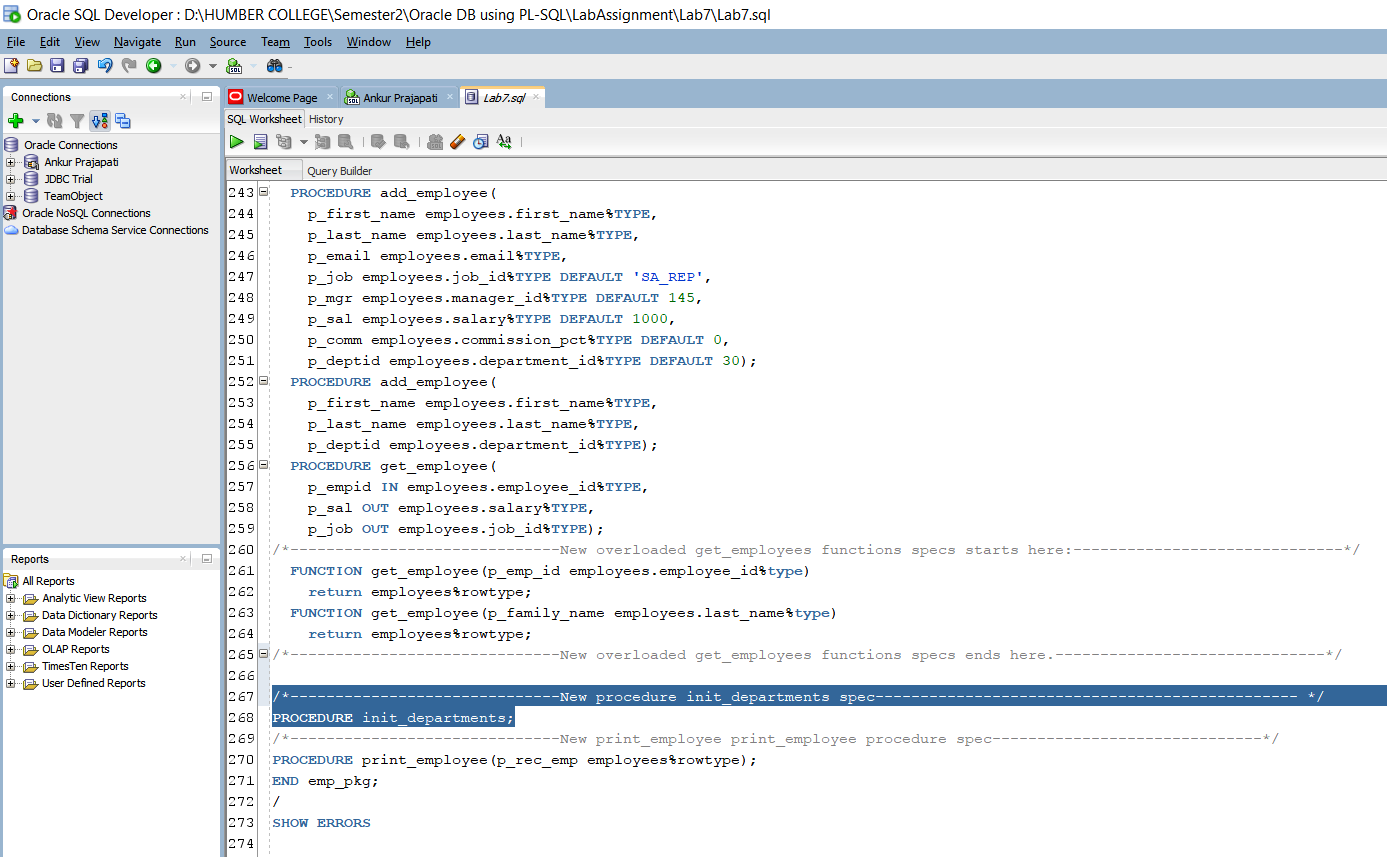


It gets complied successfully.

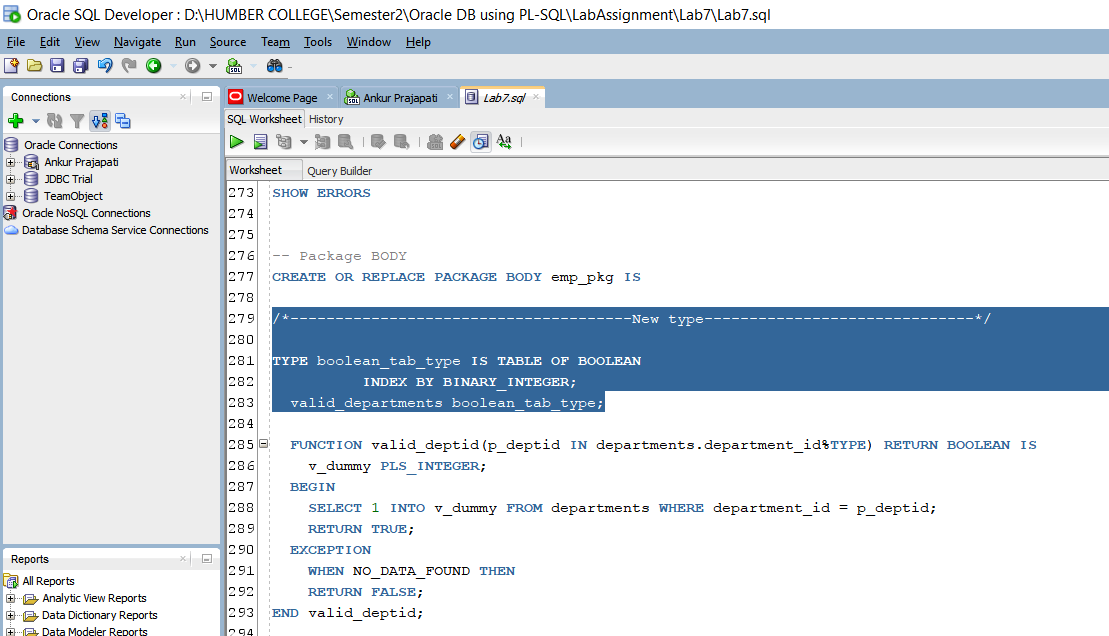


Use of anonymous block to invoke emp\_pkg.print\_employee with employee id and last name.

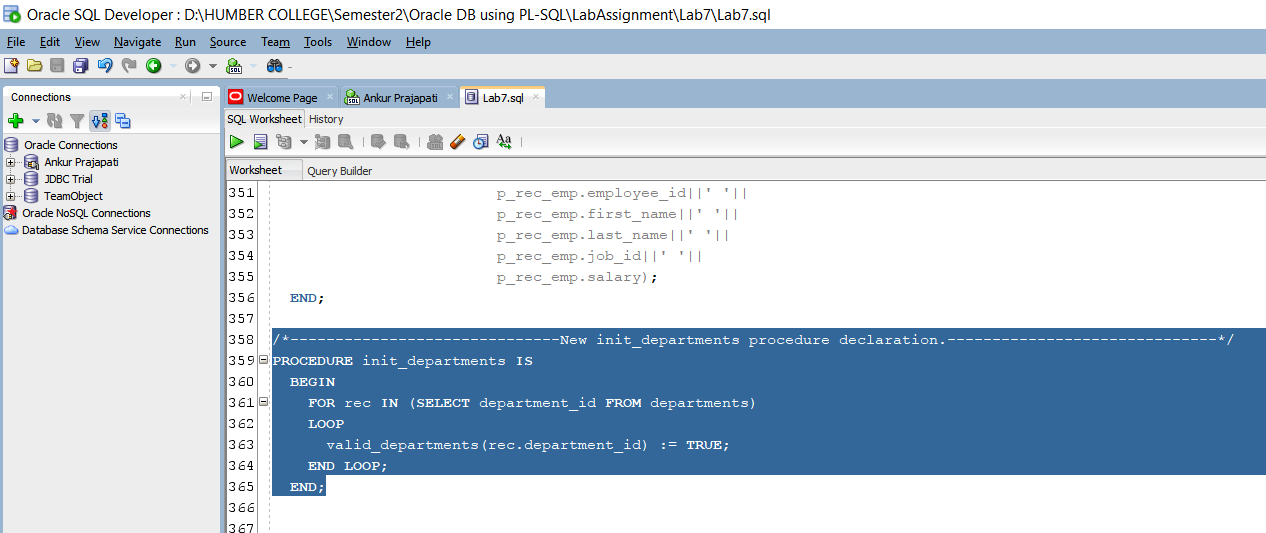
Practice 2 – 3:



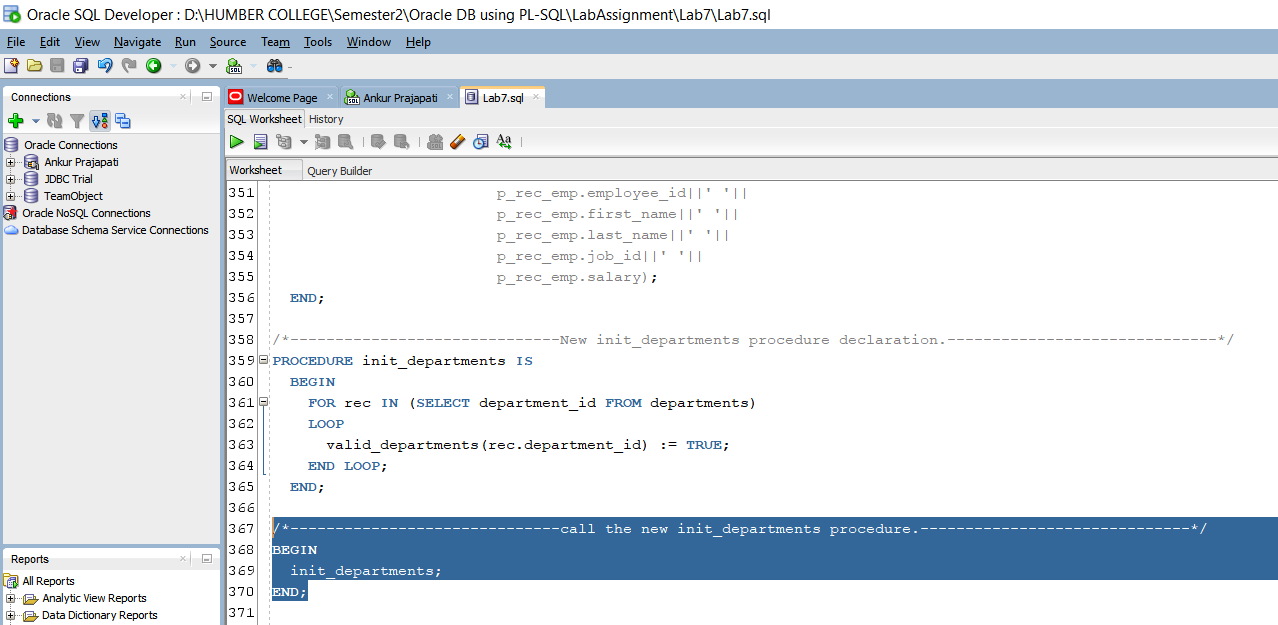
We are adding public procedure to emp\_pkg to improve its’ performance. Here init\_departments procedure is created in package specification.



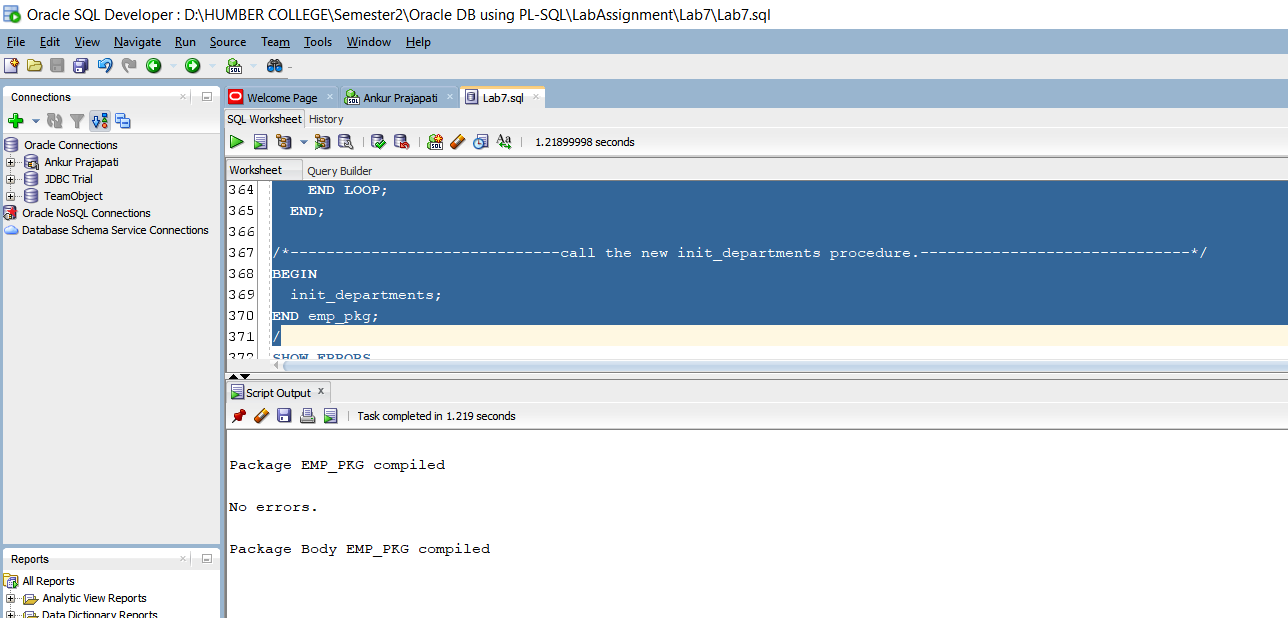
Implementation of init\_departments procedures to store all department ids in private PL/SQL block. Boolean\_tab\_type is a table which has binary\_integer for it’s index where valid\_departments is a variable of that table.



In this procedure init\_departments it takes department\_id from departments and it prints/sets true for each department id using FOR LOOP.

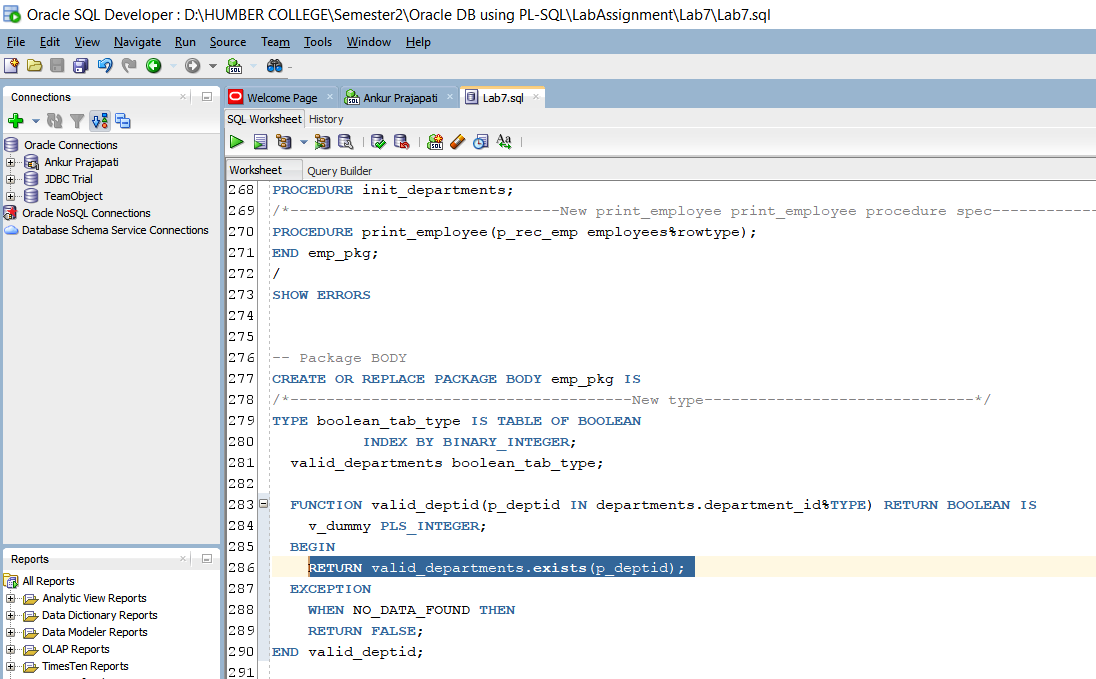


Creating initial block to call procedure init\_departments.

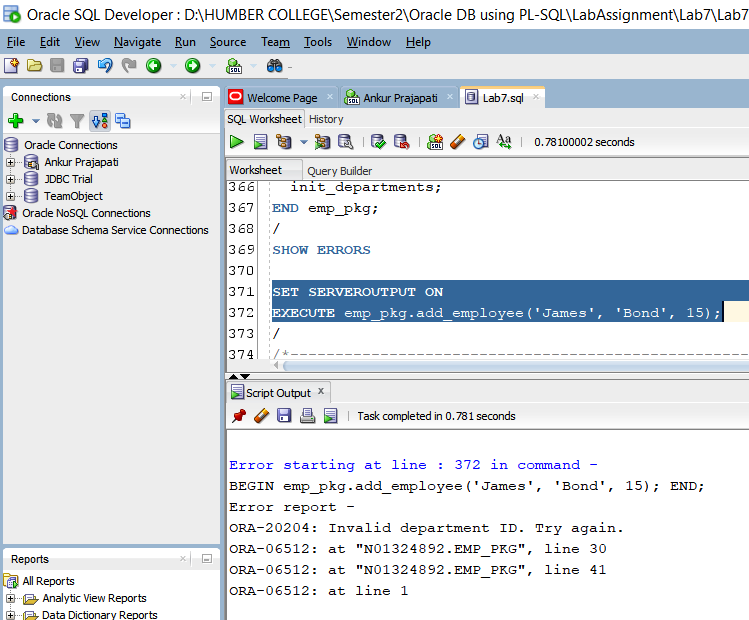


Compiling emp\_pkg and It gets complied successfully.

Practice 2 – 4:

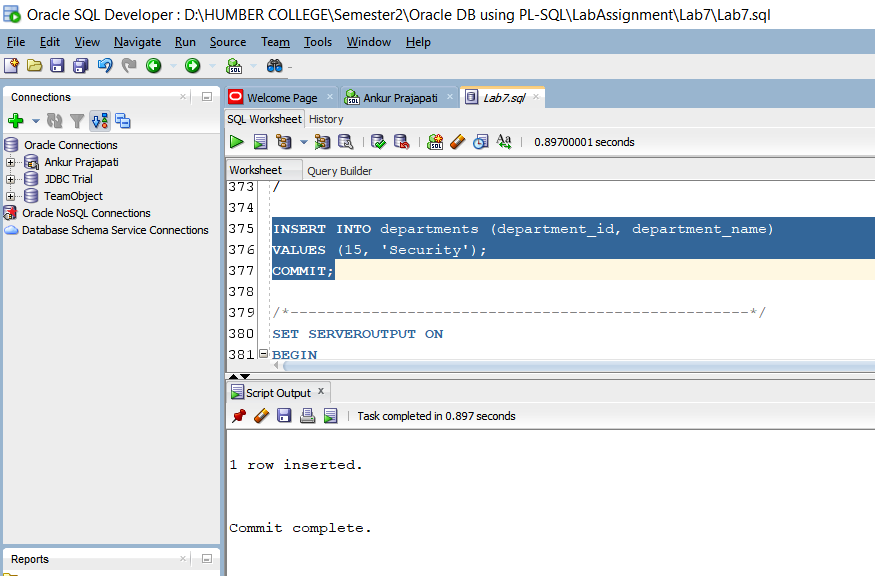


Modification of valid\_departmentid instead of looping through to find valid department\_id here we are calling our index type variable.



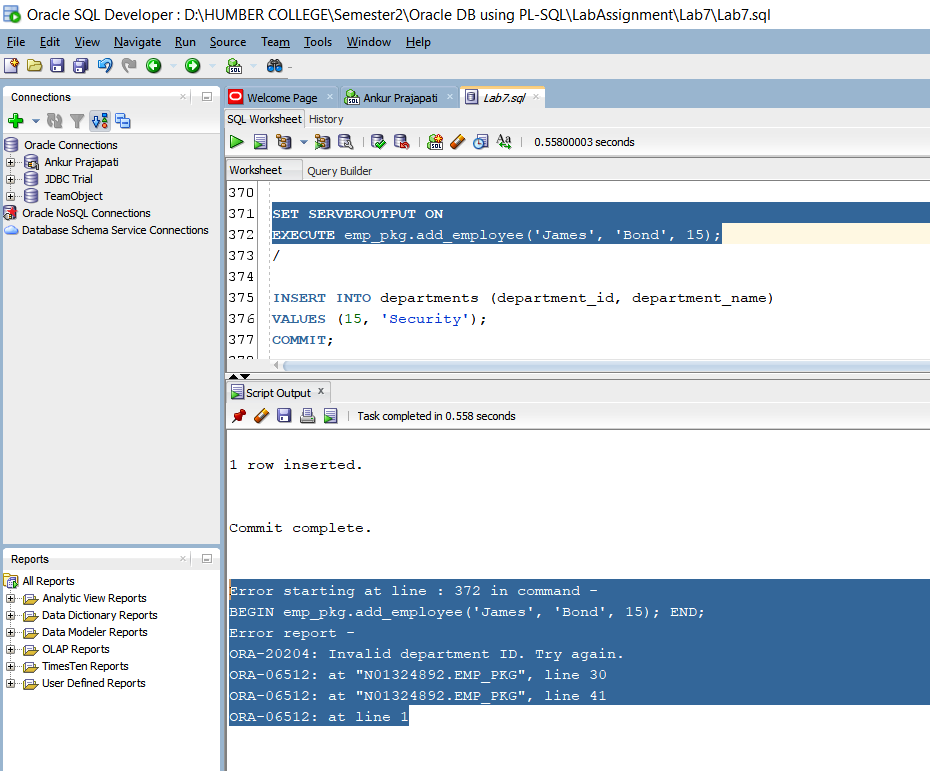
We are trying to add new employee record with first name James and alst name of Bond and department id 15.

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’.

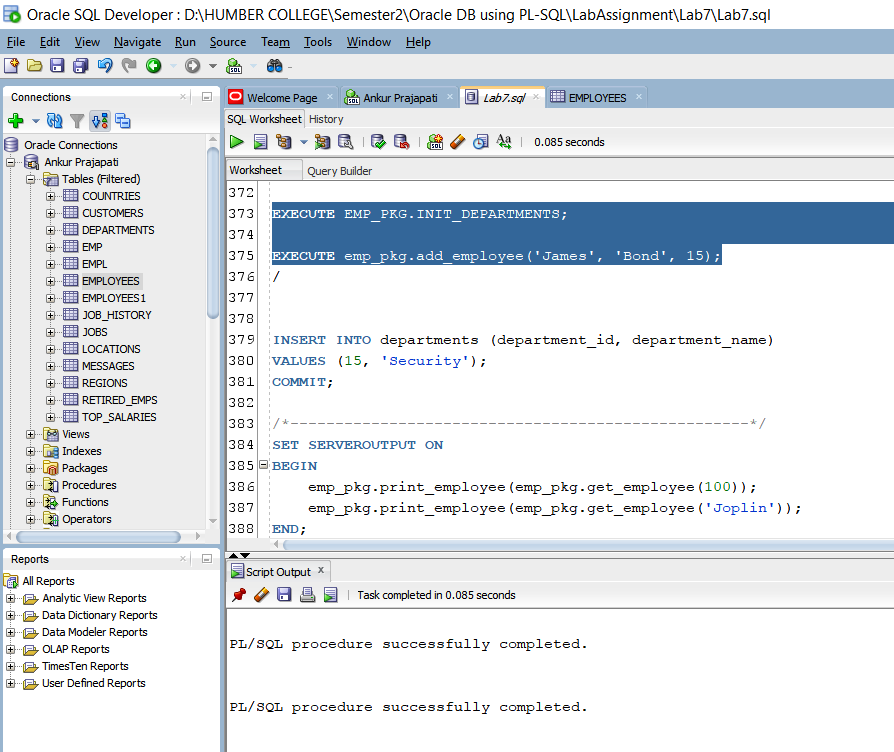
Note that we have to commit to apply changes to database.



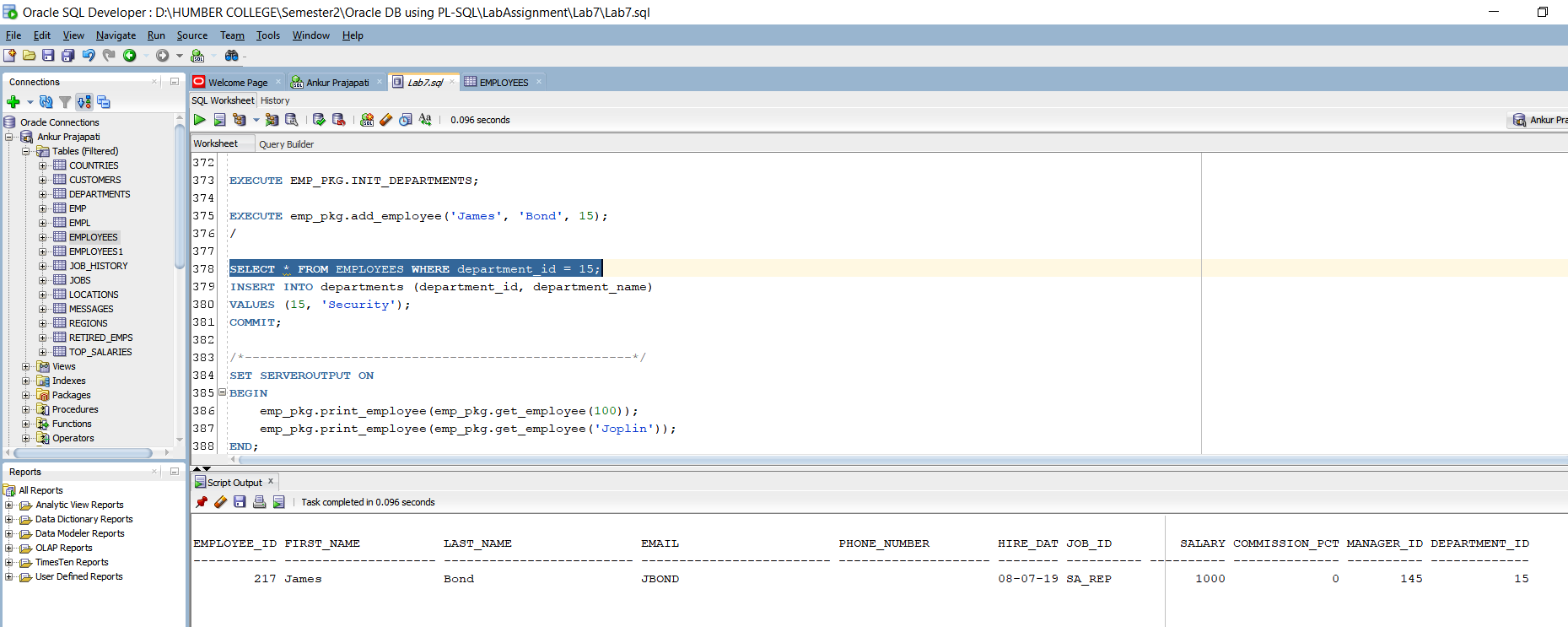
Again, we are trying to add this record here but it’s not added successfully.

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.

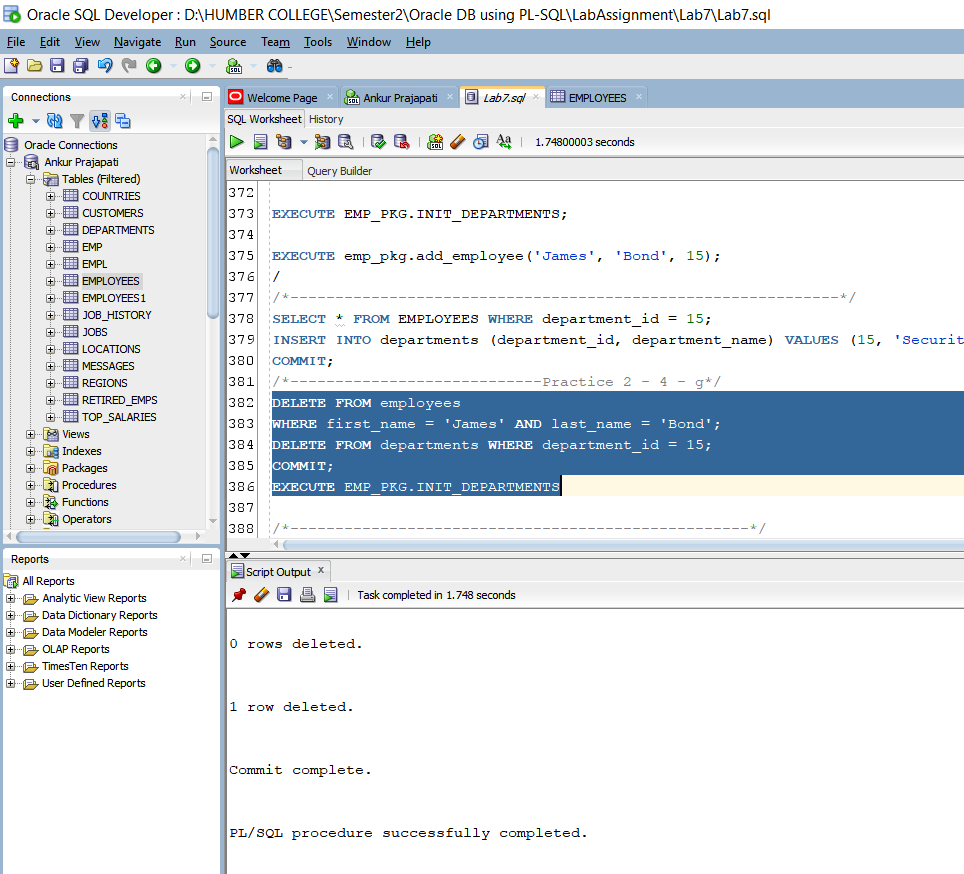


Calling of init\_departments.



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.



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.