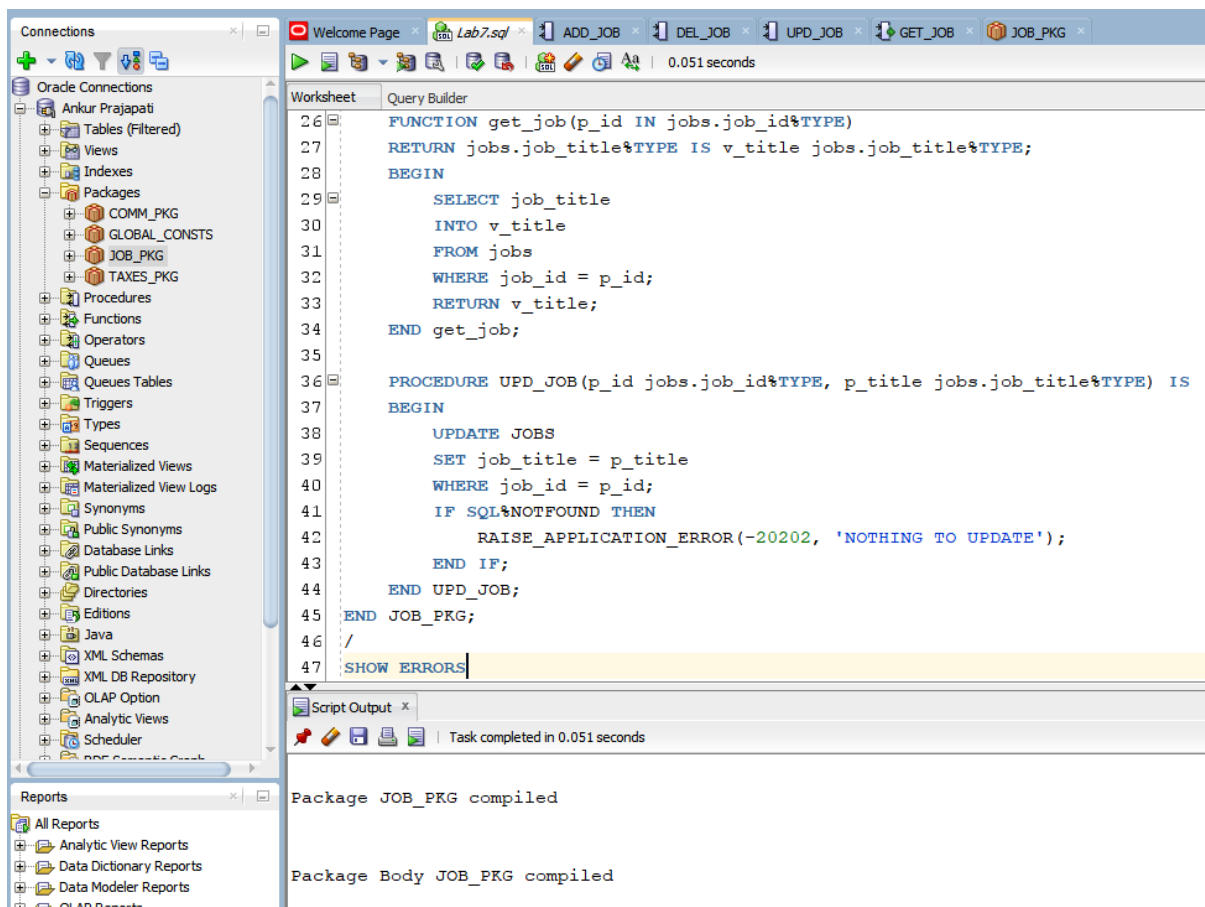
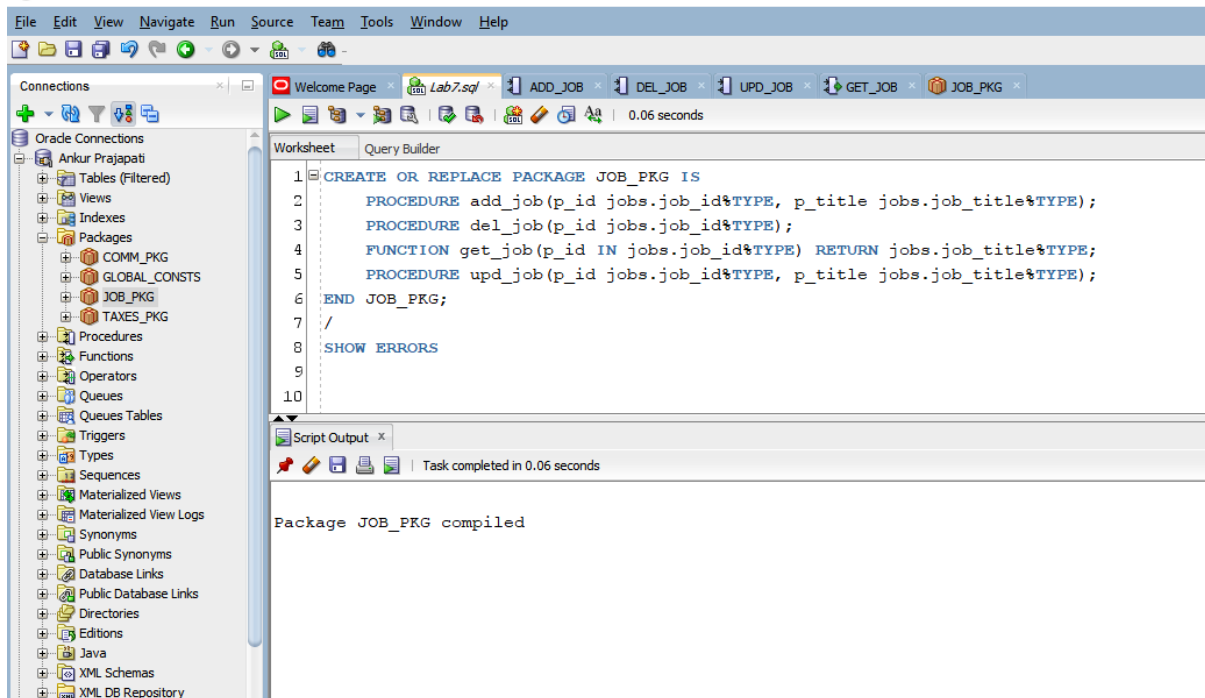


Practice 1 – 1:

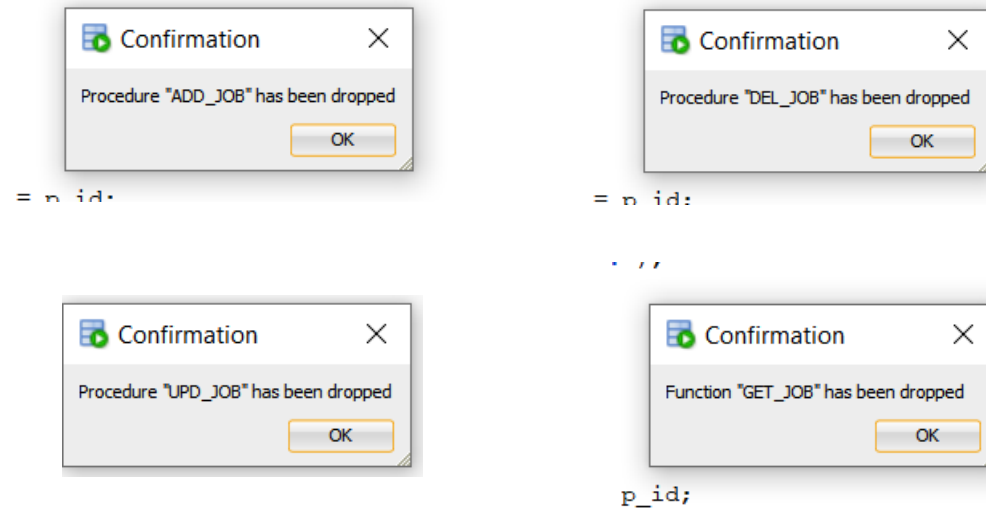
A and b:

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



- 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:



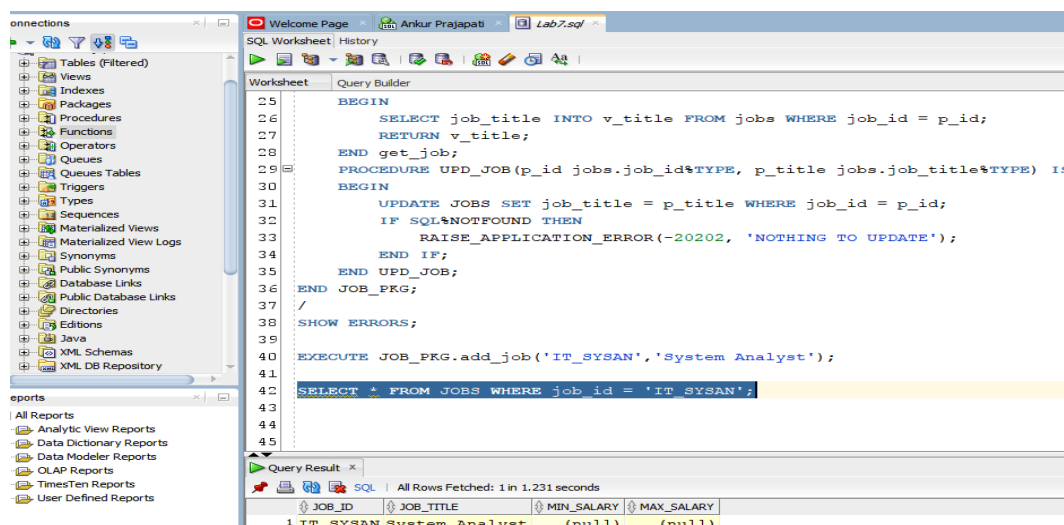
Dropping of all the procedures.

Practice Question 1 - d:

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

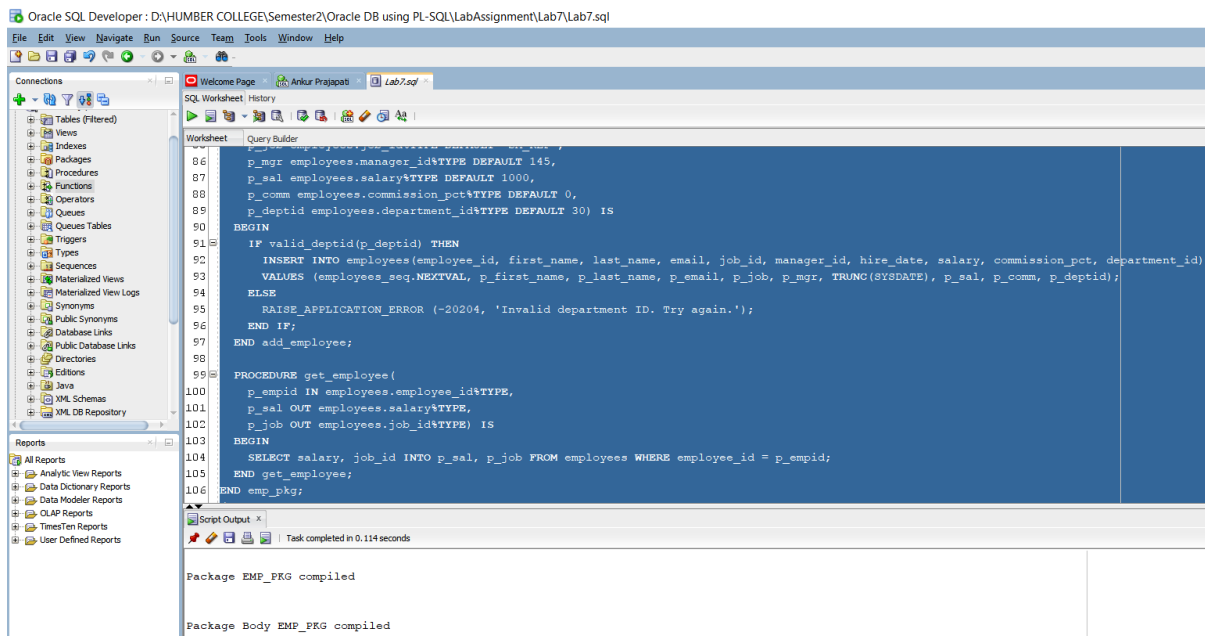
Basically, it adds job of System Analyst with job_id IT_SYSAN.

Practice Question 1 - e:



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

Practice 1 – 2:



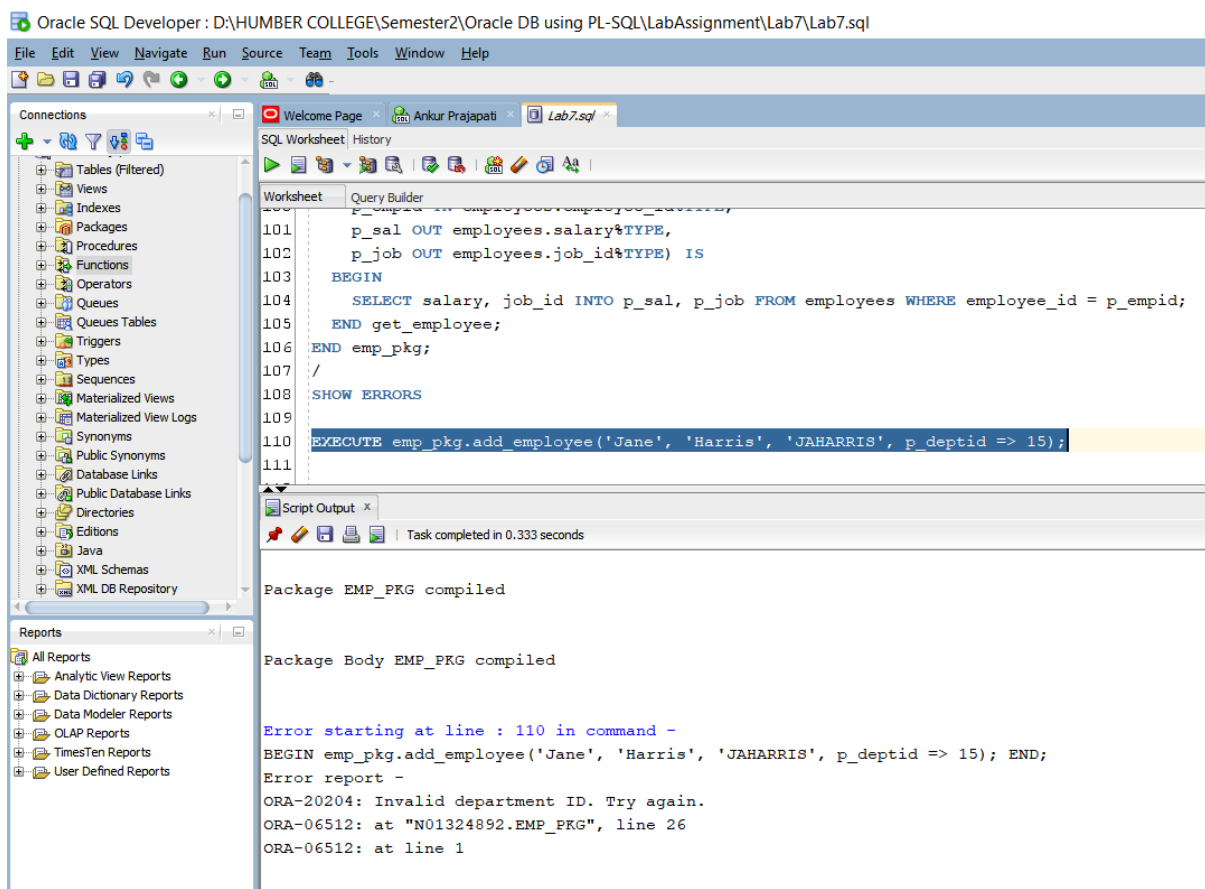
The screenshot shows the Oracle SQL Developer interface. The main window displays the SQL Worksheet with the following code:

```
86 p_mgr employees.manager_id%TYPE DEFAULT 145,  
87 p_sal employees.salary%TYPE DEFAULT 1000,  
88 p_comm employees.commission_pct%TYPE DEFAULT 0,  
89 p_deptid employees.department_id%TYPE DEFAULT 30) IS  
90 BEGIN  
91 IF valid_deptid(p_deptid) THEN  
92 INSERT INTO employees(employee_id, first_name, last_name, email, job_id, manager_id, hire_date, salary, commission_pct, department_id)  
93 VALUES (employees_seq.NEXTVAL, p_first_name, p_last_name, p_email, p_job, p_mgr, TRUNC(SYSDATE), p_sal, p_comm, p_deptid);  
94 ELSE  
95 RAISE_APPLICATION_ERROR (-20204, 'Invalid department ID. Try again.');
```

The bottom status bar indicates: "Package EMP_PKG compiled" and "Package Body EMP_PKG compiled".

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.



The screenshot shows the Oracle SQL Developer interface. The main window displays the SQL Worksheet with the following code:

```
101 p_sal OUT employees.salary%TYPE,  
102 p_job OUT employees.job_id%TYPE) IS  
103 BEGIN  
104 SELECT salary, job_id INTO p_sal, p_job FROM employees WHERE employee_id = p_empid;  
105 END get_employee;  
106 END emp_pkg;  
107 /  
108 SHOW ERRORS  
109  
110 EXECUTE emp_pkg.add_employee('Jane', 'Harris', 'JAHARRIS', p_deptid => 15);  
111
```

The bottom status bar indicates: "Package EMP_PKG compiled" and "Package Body EMP_PKG compiled".

The Script Output window shows the following error message:

```
Error starting at line : 110 in command -  
BEGIN emp_pkg.add_employee('Jane', 'Harris', 'JAHARRIS', p_deptid => 15); END;  
Error report -  
ORA-20204: Invalid department ID. Try again.  
ORA-06512: at "N01324892.EMP_PKG", line 26  
ORA-06512: at line 1
```

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.

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

File Edit View Navigate Run Source Team Tools Window Help

Connections

Tables (Filtered)
Views
Indexes
Packages
Procedures
Functions
Operators
Queues
Queues Tables
Triggers
Types
Sequences
Materialized Views
Materialized View Logs
Synonyms
Public Synonyms
Database Links
Public Database Links
Directories
Editions
Java
XML Schemas
XML DB Repository

Reports

All Reports
Analytic View Reports
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
TimesTen Reports
User Defined Reports

Worksheet

107 /
108 SHOW ERRORS
109
110 EXECUTE emp_pkg.add_employee('Jane', 'Harris', 'JAHARRIS', p_deptid => 15);
111
112 EXECUTE emp_pkg.add_employee('David', 'Smith', 'DASMITH', p_deptid => 80);
113
114
115
116
117
118

Script Output

Task completed in 0.058 seconds

Package EMP_PKG compiled

Package Body EMP_PKG compiled

Error starting at line : 110 in command -
BEGIN emp_pkg.add_employee('Jane', 'Harris', 'JAHARRIS', p_deptid => 15); END;
Error report -
ORA-20204: Invalid department ID. Try again.
ORA-06512: at "N01324892.EMP_PKG", line 26
ORA-06512: at line 1

PL/SQL procedure successfully completed.

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

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

File Edit View Navigate Run Source Team Tools Window Help

Connections

Tables (Filtered)
Views
Indexes
Packages
Procedures
Functions
Operators
Queues
Queues Tables
Triggers
Types
Sequences
Materialized Views
Materialized View Logs
Synonyms
Public Synonyms
Database Links
Public Database Links
Directories
Editions
Java
XML Schemas
XML DB Repository

Reports

All Reports
Analytic View Reports
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
TimesTen Reports
User Defined Reports

Worksheet

107 /
108 SHOW ERRORS
109
110 EXECUTE emp_pkg.add_employee('Jane', 'Harris', 'JAHARRIS', p_deptid => 15);
111
112 EXECUTE emp_pkg.add_employee('David', 'Smith', 'DASMITH', p_deptid => 80);
113
114 SELECT * FROM EMPLOYEES WHERE last_name = 'Smith';
115
116
117
118

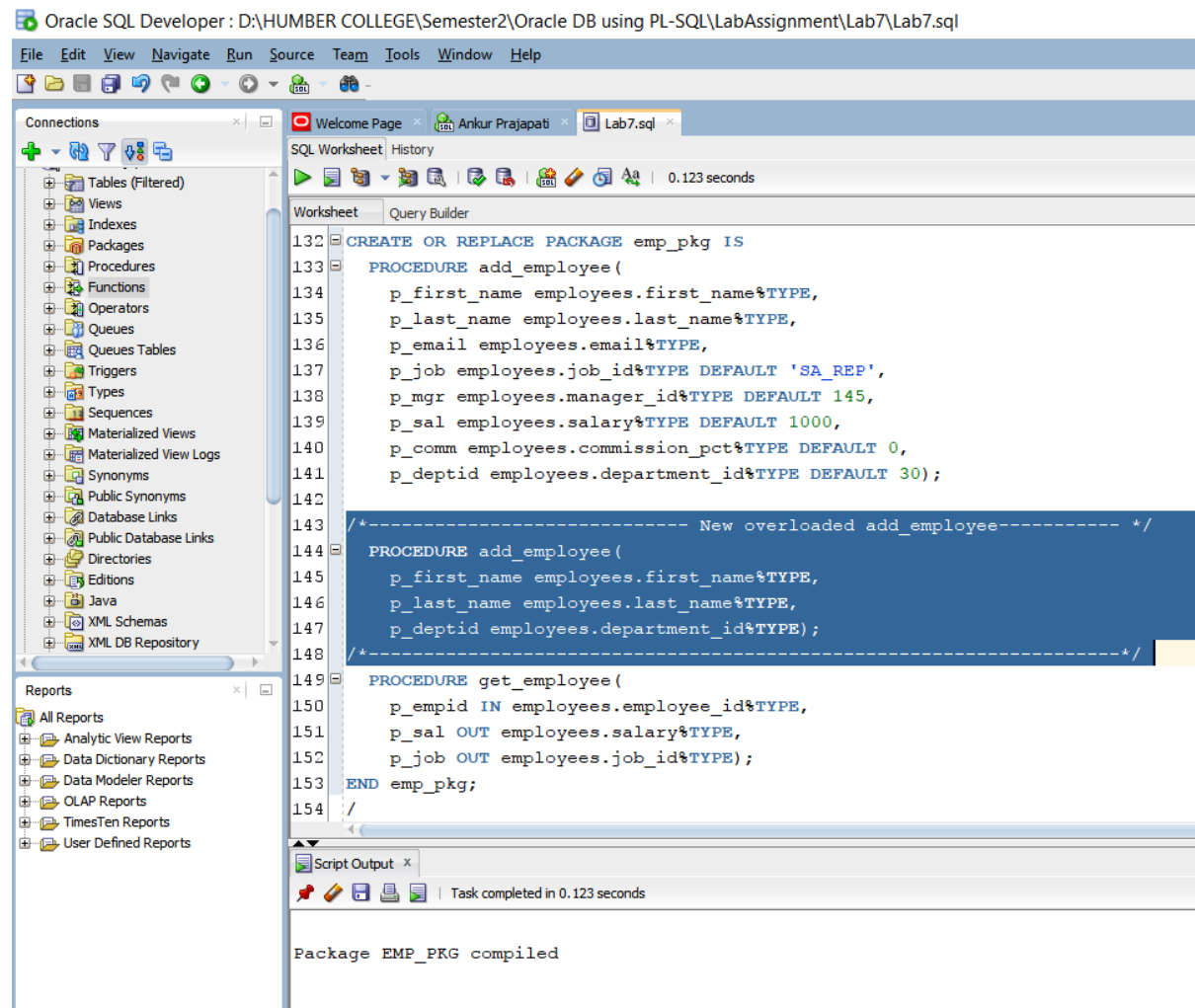
Script Output

SQL All Rows Fetched: 3 in 0.171 seconds

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	159	Lindsey	Smith	LSMITH	011.44.1345.729268	10-03-97	SA_REP	8000	0.3	146	80
2	171	William	Smith	WSMITH	011.44.1343.629268	23-02-99	SA_REP	7400	0.15	148	80
3	213	David	Smith	DASMITH (null)		04-07-19	SA_REP	1000	0	145	80

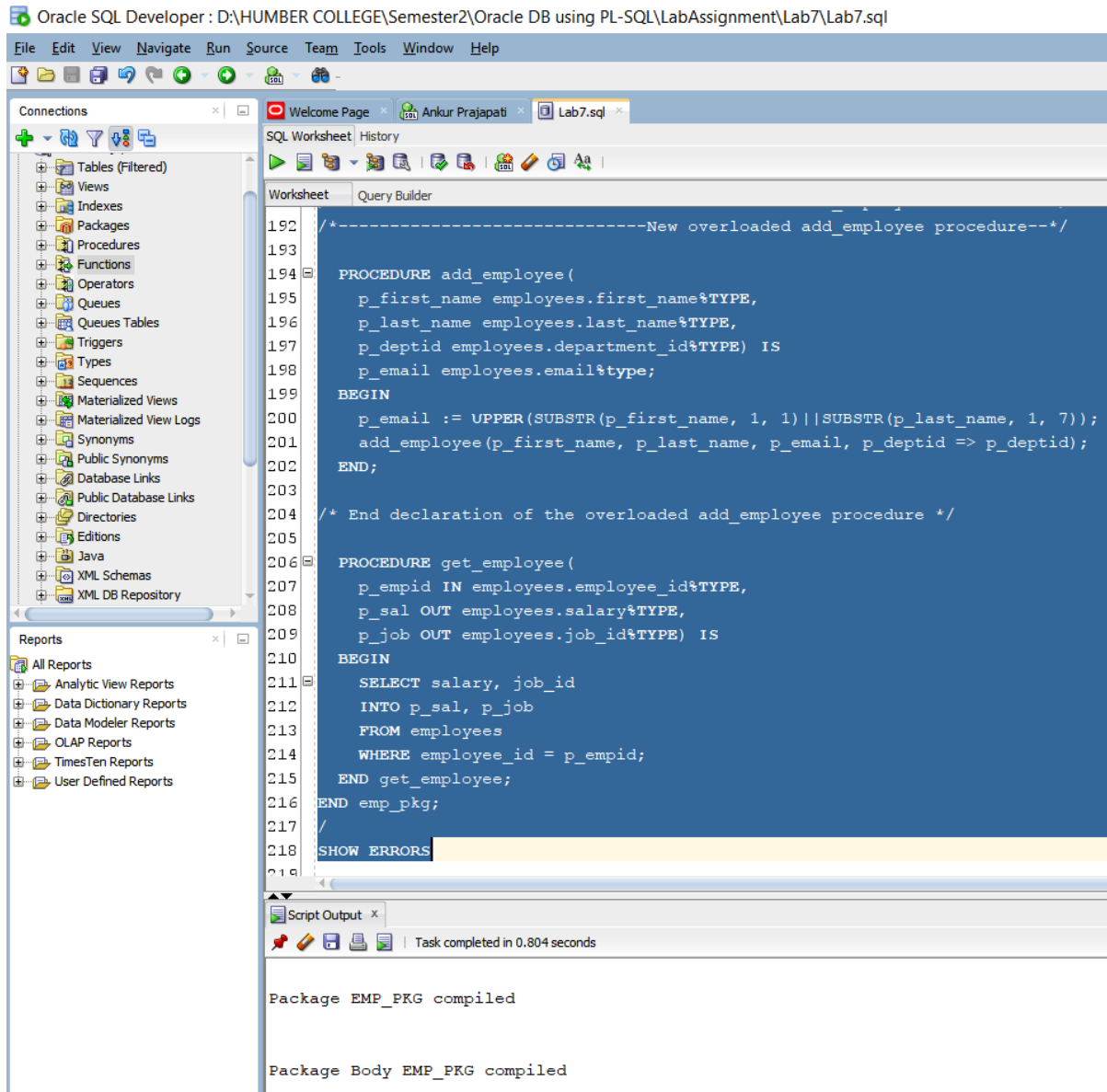
Printing all the employees where last name is Smith.

Practice 2 – 1:



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

It gets compiled successfully with overloading.



The screenshot displays the Oracle SQL Developer interface. The title bar indicates the connection to 'D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql'. The left-hand 'Connections' pane shows a tree view of database objects, including Tables, Views, Indexes, Packages, Procedures, Functions, Operators, Queues, Queues Tables, Triggers, Types, Sequences, Materialized Views, Materialized View Logs, Synonyms, Public Synonyms, Database Links, Public Database Links, Directories, Editions, Java, XML Schemas, and XML DB Repository. The main 'SQL Worksheet' pane contains the following PL/SQL code:

```
192 /*-----New overloaded add_employee procedure---*/
193
194 PROCEDURE add_employee(
195     p_first_name employees.first_name%TYPE,
196     p_last_name employees.last_name%TYPE,
197     p_deptid employees.department_id%TYPE) IS
198     p_email employees.email%type;
199 BEGIN
200     p_email := UPPER(SUBSTR(p_first_name, 1, 1)||SUBSTR(p_last_name, 1, 7));
201     add_employee(p_first_name, p_last_name, p_email, p_deptid => p_deptid);
202 END;
203
204 /* End declaration of the overloaded add_employee procedure */
205
206 PROCEDURE get_employee(
207     p_empid IN employees.employee_id%TYPE,
208     p_sal OUT employees.salary%TYPE,
209     p_job OUT employees.job_id%TYPE) IS
210 BEGIN
211     SELECT salary, job_id
212     INTO p_sal, p_job
213     FROM employees
214     WHERE employee_id = p_empid;
215 END get_employee;
216 END emp_pkg;
217 /
218 SHOW ERRORS
```

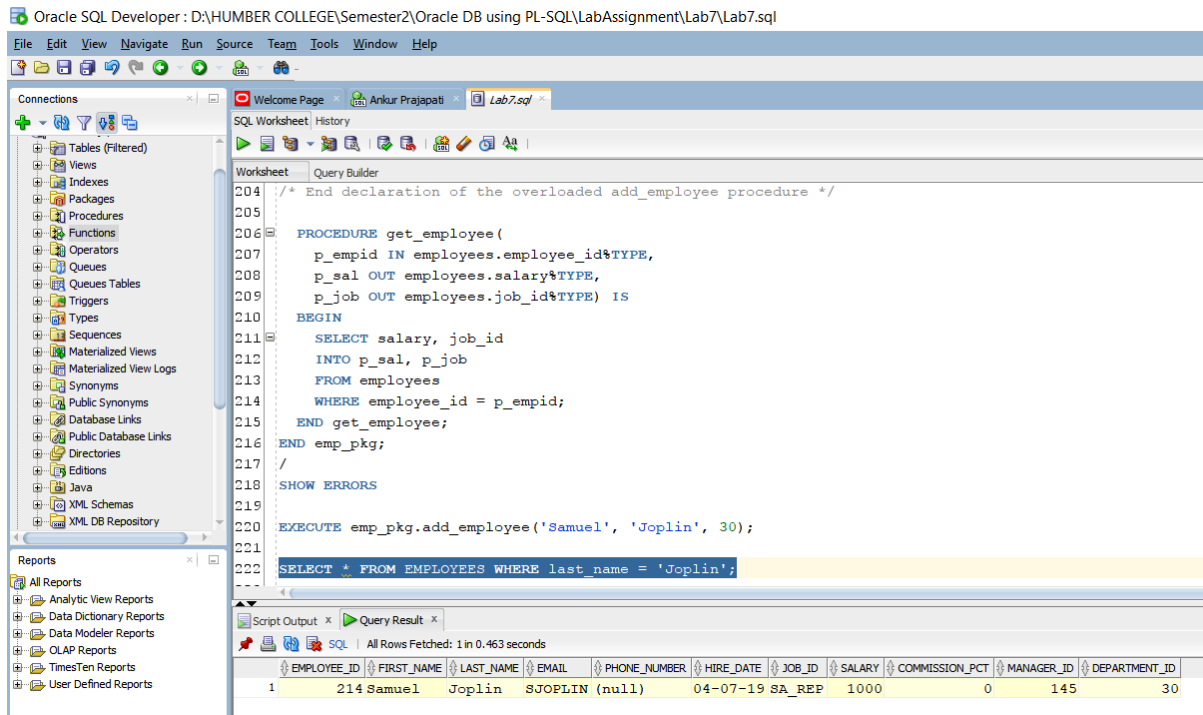
The 'Script Output' pane at the bottom shows the successful compilation of the package:

```
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.



The screenshot shows the Oracle SQL Developer interface. The main window displays a SQL worksheet with the following code:

```
204 /* End declaration of the overloaded add_employee procedure */
205
206 PROCEDURE get_employee(
207     p_empid IN employees.employee_id%TYPE,
208     p_sal OUT employees.salary%TYPE,
209     p_job OUT employees.job_id%TYPE) IS
210 BEGIN
211     SELECT salary, job_id
212     INTO p_sal, p_job
213     FROM employees
214     WHERE employee_id = p_empid;
215 END get_employee;
216 END emp_pkg;
217 /
218 SHOW ERRORS
219
220 EXECUTE emp_pkg.add_employee('Samuel', 'Joplin', 30);
221
222 SELECT * FROM EMPLOYEES WHERE last_name = 'Joplin';
```

The bottom pane shows the query result for the last query. It displays a single row with the following data:

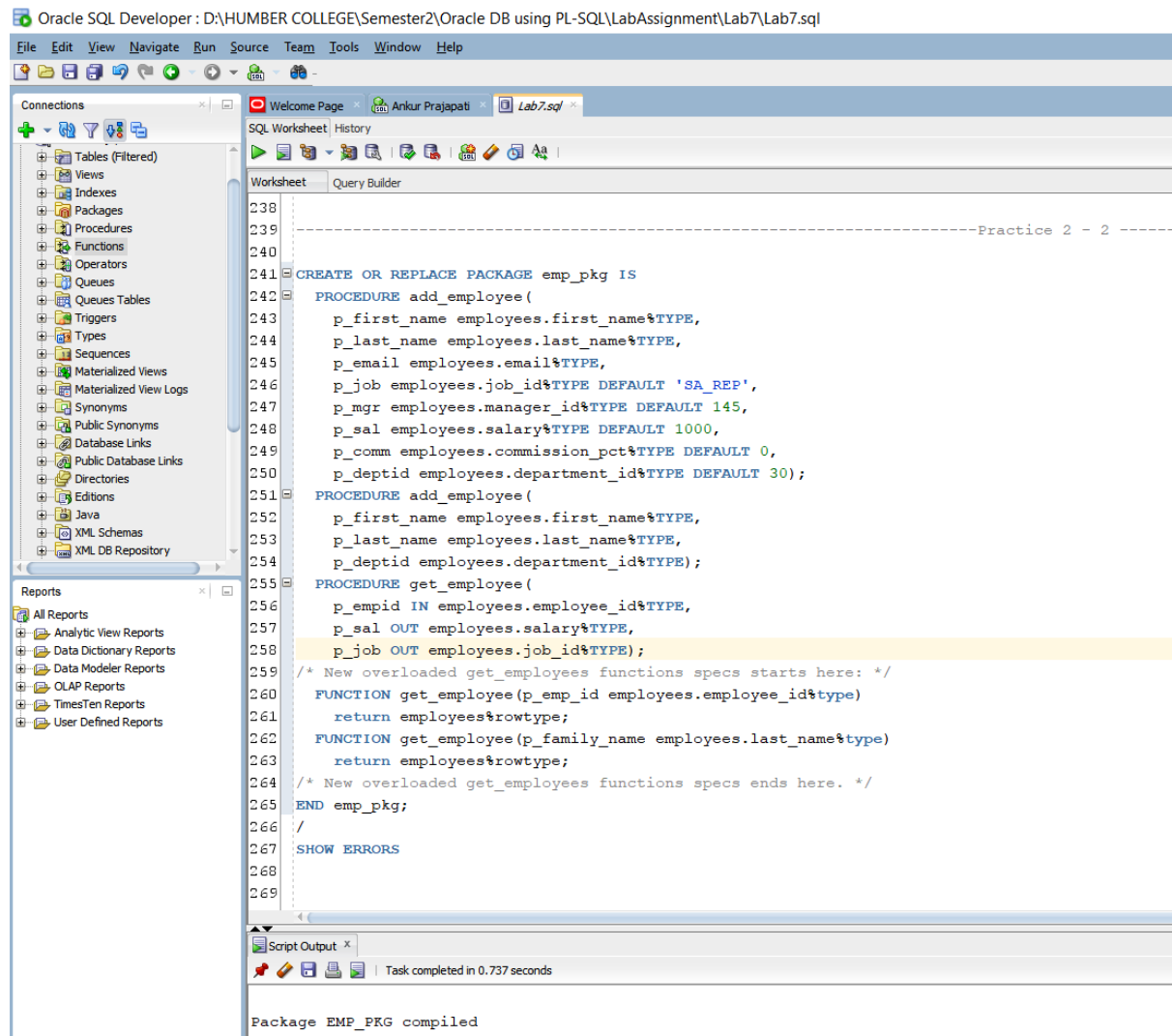
EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
214	Samuel	Joplin	SJOPLIN (null)		04-07-19	SA_REP	1000	0	145	30

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:



The screenshot displays the Oracle SQL Developer interface. The title bar indicates the connection to 'D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql'. The main window shows a SQL worksheet with the following code:

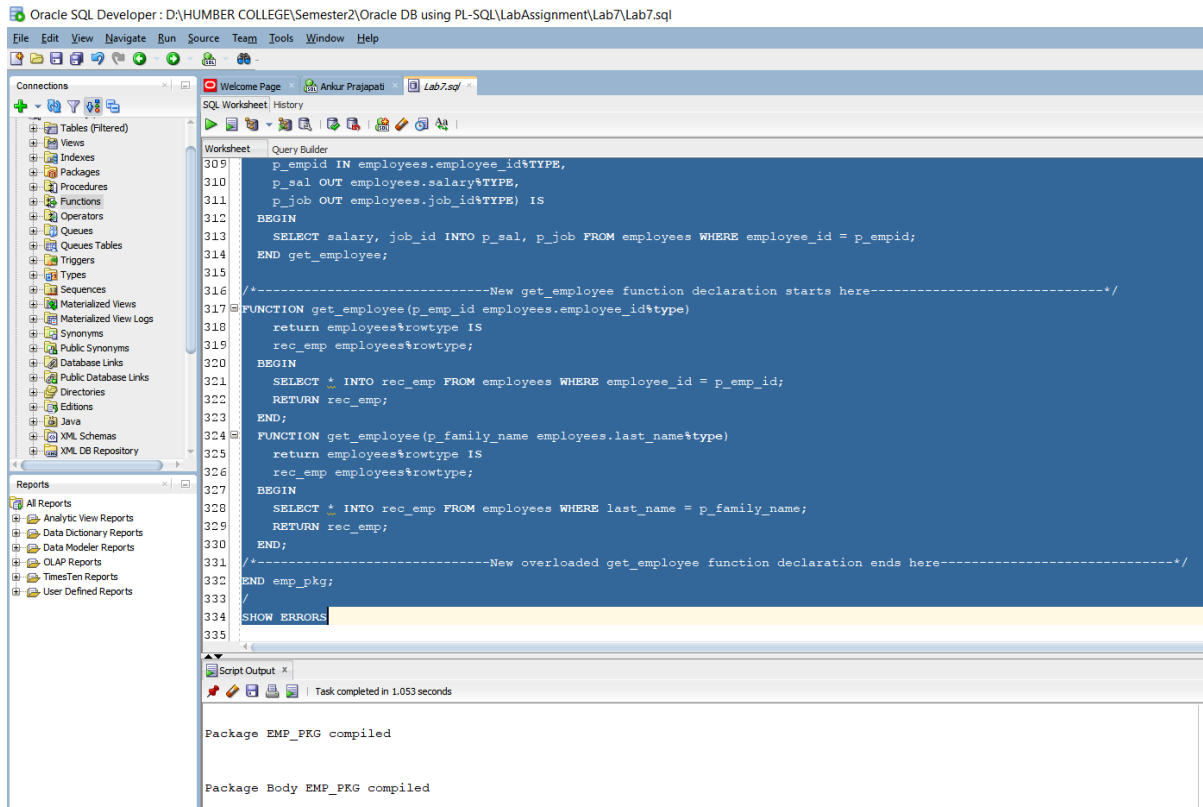
```
238
239 -----Practice 2 - 2 -----
240
241 CREATE OR REPLACE PACKAGE emp_pkg IS
242   PROCEDURE add_employee(
243     p_first_name employees.first_name%TYPE,
244     p_last_name employees.last_name%TYPE,
245     p_email employees.email%TYPE,
246     p_job employees.job_id%TYPE DEFAULT 'SA_REP',
247     p_mgr employees.manager_id%TYPE DEFAULT 145,
248     p_sal employees.salary%TYPE DEFAULT 1000,
249     p_comm employees.commission_pct%TYPE DEFAULT 0,
250     p_deptid employees.department_id%TYPE DEFAULT 30);
251   PROCEDURE add_employee(
252     p_first_name employees.first_name%TYPE,
253     p_last_name employees.last_name%TYPE,
254     p_deptid employees.department_id%TYPE);
255   PROCEDURE get_employee(
256     p_empid IN employees.employee_id%TYPE,
257     p_sal OUT employees.salary%TYPE,
258     p_job OUT employees.job_id%TYPE);
259   /* New overloaded get_employees functions specs starts here: */
260   FUNCTION get_employee(p_emp_id employees.employee_id%type)
261     return employees%rowtype;
262   FUNCTION get_employee(p_family_name employees.last_name%type)
263     return employees%rowtype;
264   /* New overloaded get_employees functions specs ends here. */
265 END emp_pkg;
266 /
267 SHOW ERRORS
268
269
```

The 'Script Output' window at the bottom shows the message: '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 compiled successfully.



```
309 p_empid IN employees.employee_id%TYPE,  
310 p_sal OUT employees.salary%TYPE,  
311 p_job OUT employees.job_id%TYPE) IS  
312 BEGIN  
313 SELECT salary, job_id INTO p_sal, p_job FROM employees WHERE employee_id = p_empid;  
314 END get_employee;  
315  
316 /*-----New get_employee function declaration starts here-----*/  
317 FUNCTION get_employee(p_emp_id employees.employee_id%TYPE)  
318 return employees%rowtype IS  
319 rec_emp employees%rowtype;  
320 BEGIN  
321 SELECT * INTO rec_emp FROM employees WHERE employee_id = p_emp_id;  
322 RETURN rec_emp;  
323 END;  
324 FUNCTION get_employee(p_family_name employees.last_name%TYPE)  
325 return employees%rowtype IS  
326 rec_emp employees%rowtype;  
327 BEGIN  
328 SELECT * INTO rec_emp FROM employees WHERE last_name = p_family_name;  
329 RETURN rec_emp;  
330 END;  
331 /*-----New overloaded get_employee function declaration ends here-----*/  
332 END emp_pkg;  
333 /  
334 SHOW ERRORS  
335
```

Script Output x
Task completed in 1.053 seconds

Package EMP_PKG compiled

Package Body EMP_PKG compiled

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 compiled successfully.

Adding print_employee specification and body to package.

```

257     p_sal OUT employees.salary%TYPE,
258     p_job OUT employees.job_id%TYPE);
259 /*-----New overloaded get_employees functions specs starts here:-----*/
260 FUNCTION get_employee(p_emp_id employees.employee_id%TYPE)
261     return employees%rowtype;
262 FUNCTION get_employee(p_family_name employees.last_name%TYPE)
263     return employees%rowtype;
264 /*-----New overloaded get_employees functions specs ends here.-----*/
265 /*-----New print_employee print_employee procedure spec-----*/
266 PROCEDURE print_employee(p_rec_emp employees%rowtype);
267 END emp_pkg;
268 /
269 SHOW ERRORS
270
271
272 -- Package BODY
273 CREATE OR REPLACE PACKAGE BODY emp_pkg IS

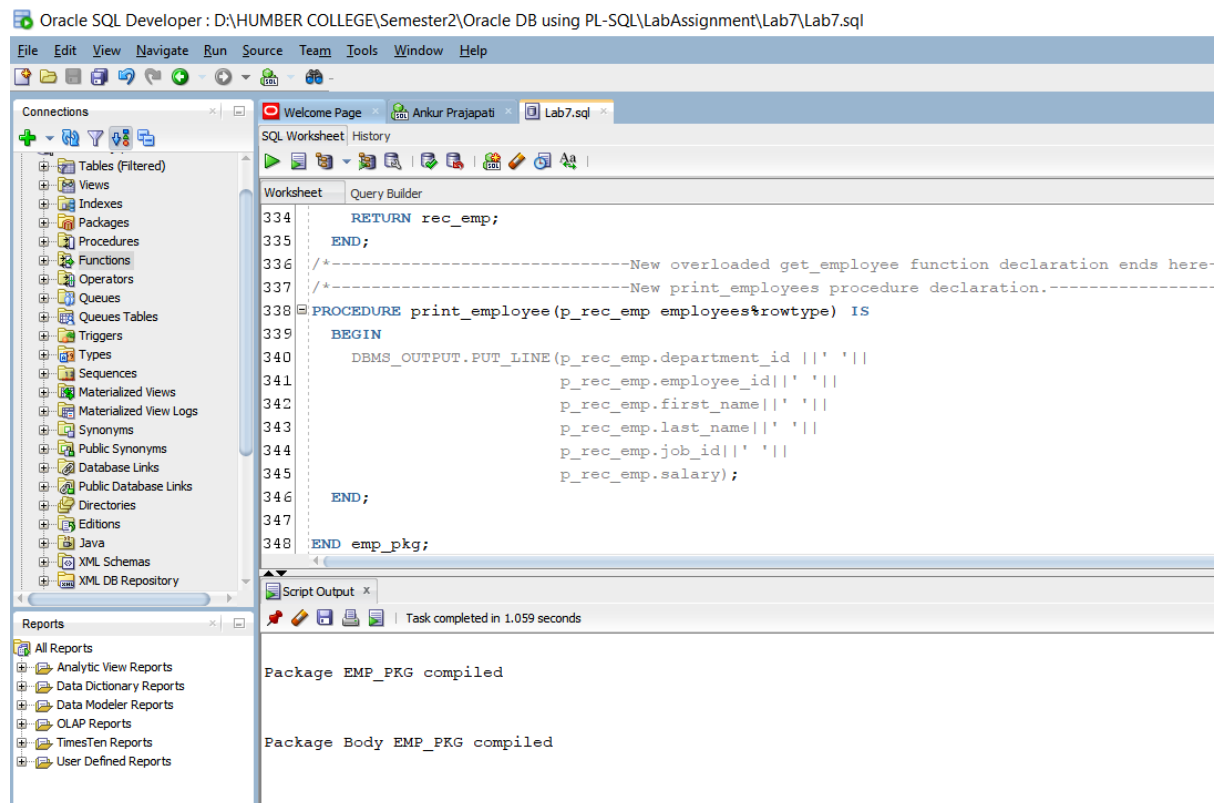
```

```

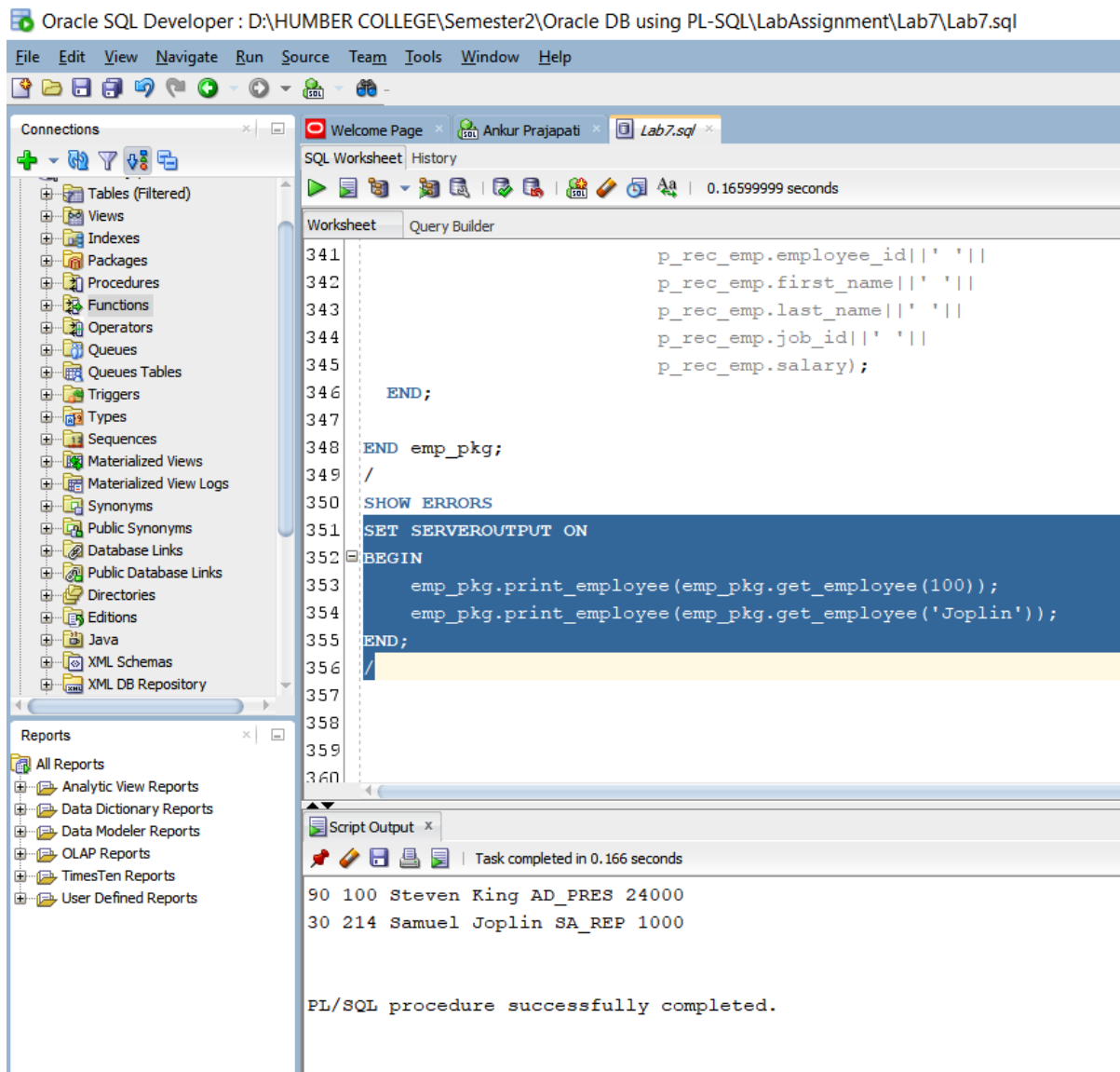
334     END;
335 /*-----New overloaded get_employee function declaration ends here-----*/
336 /*-----New print_employees procedure declaration.-----*/
337 PROCEDURE print_employee(p_rec_emp employees%rowtype) IS
338 BEGIN
339     DBMS_OUTPUT.PUT_LINE(p_rec_emp.department_id || ' ' ||
340         p_rec_emp.employee_id || ' ' ||
341         p_rec_emp.first_name || ' ' ||
342         p_rec_emp.last_name || ' ' ||
343         p_rec_emp.job_id || ' ' ||
344         p_rec_emp.salary);
345 END;
346
347 END emp_pkg;

```

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



It gets compiled successfully.



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
File Edit View Navigate Run Source Team Tools Window Help

Connections
+ Oracle Connections
+ Ankur Prajapati
+ JDBC Trial
+ TeamObject
+ Oracle NoSQL Connections
+ Database Schema Service Connections

Reports
+ All Reports
+ Analytic View Reports
+ Data Dictionary Reports
+ Data Modeler Reports
+ OLAP Reports
+ TimesTen Reports
+ User Defined Reports

SQL Worksheet: History
Worksheet Query Builder

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
File Edit View Navigate Run Source Team Tools Window Help

Connections
+ Oracle Connections
+ Ankur Prajapati
+ JDBC Trial
+ TeamObject
+ Oracle NoSQL Connections
+ Database Schema Service Connections

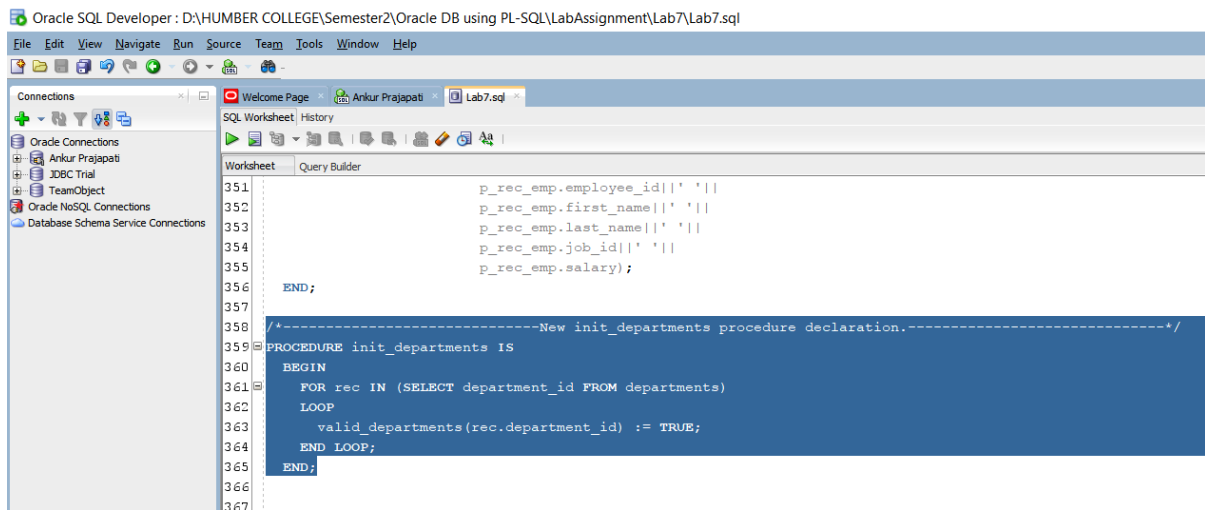
Reports
+ All Reports
+ Analytic View Reports
+ Data Dictionary Reports
+ Data Modeler Reports

SQL Worksheet: History
Worksheet Query Builder

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

```

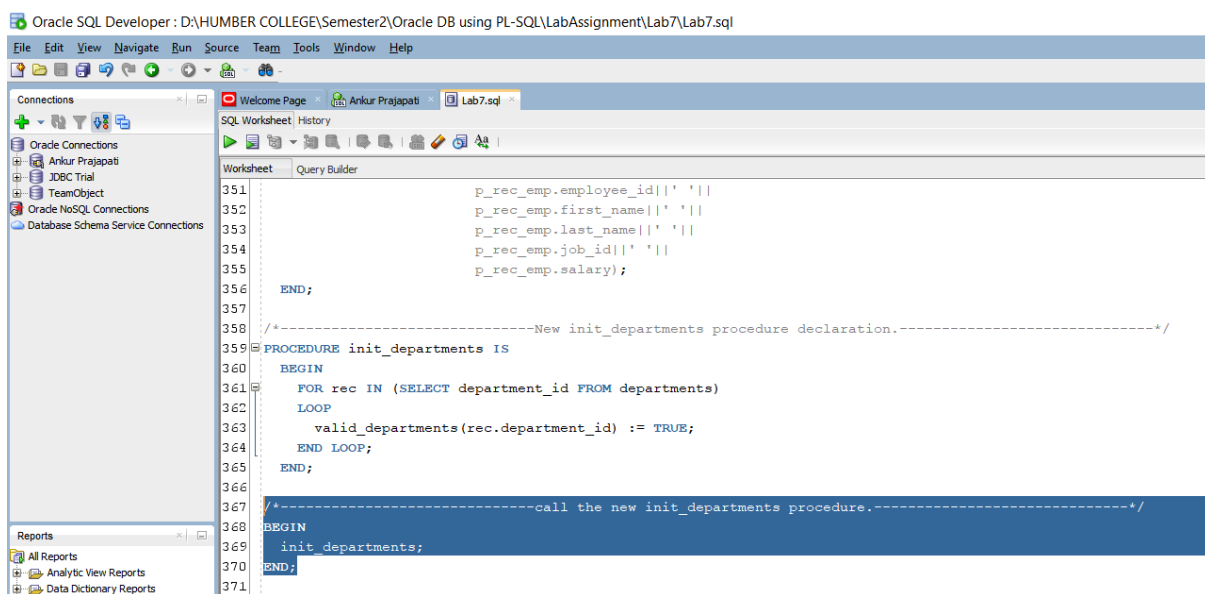
Implementation of init_departments procedures to store all department ids in private PL/SQL block. Boolean_tab_type is a table which has binary_integer for it's index where valid_departments is a variable of that table.



The screenshot shows the Oracle SQL Developer interface. The title bar indicates the file path: D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab7\Lab7.sql. The main window displays a SQL Worksheet with the following code:

```
351 p_rec_emp.employee_id||' '||
352 p_rec_emp.first_name||' '||
353 p_rec_emp.last_name||' '||
354 p_rec_emp.job_id||' '||
355 p_rec_emp.salary);
356
357 END;
358
359 /*-----New init_departments procedure declaration.-----*/
360 PROCEDURE init_departments IS
361 BEGIN
362   FOR rec IN (SELECT department_id FROM departments)
363   LOOP
364     valid_departments(rec.department_id) := TRUE;
365   END LOOP;
366 END;
```

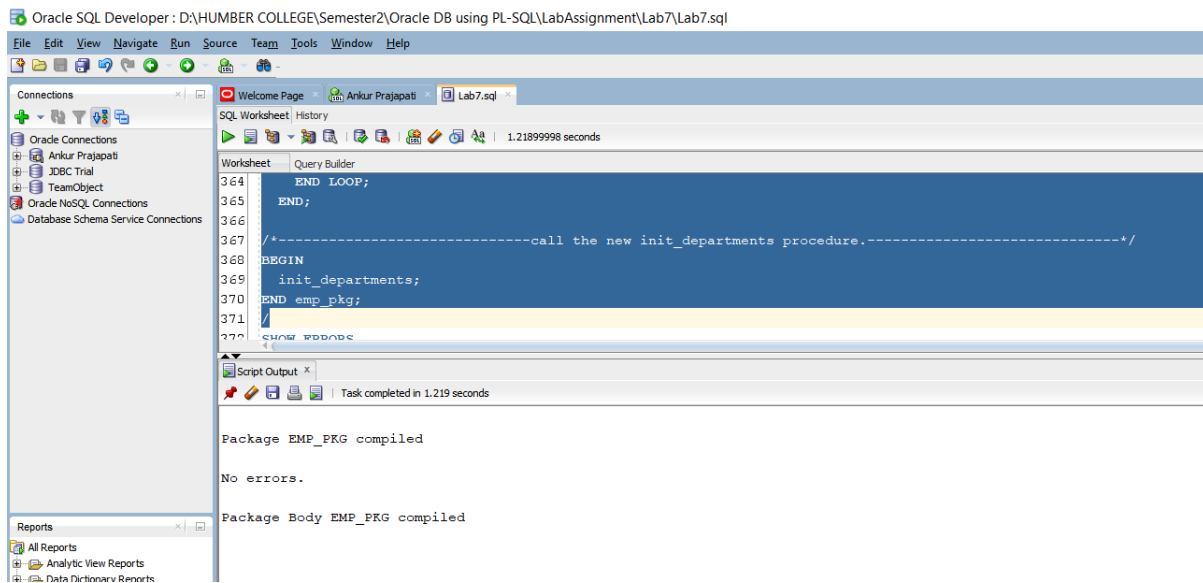
In this procedure init_departments it takes department_id from departments and it prints/sets true for each department id using FOR LOOP.



The screenshot shows the Oracle SQL Developer interface with the same file path. The main window displays the SQL Worksheet with the following code:

```
351 p_rec_emp.employee_id||' '||
352 p_rec_emp.first_name||' '||
353 p_rec_emp.last_name||' '||
354 p_rec_emp.job_id||' '||
355 p_rec_emp.salary);
356
357 END;
358
359 /*-----New init_departments procedure declaration.-----*/
360 PROCEDURE init_departments IS
361 BEGIN
362   FOR rec IN (SELECT department_id FROM departments)
363   LOOP
364     valid_departments(rec.department_id) := TRUE;
365   END LOOP;
366 END;
367
368 /*-----call the new init_departments procedure.-----*/
369 BEGIN
370   init_departments;
371 END;
```

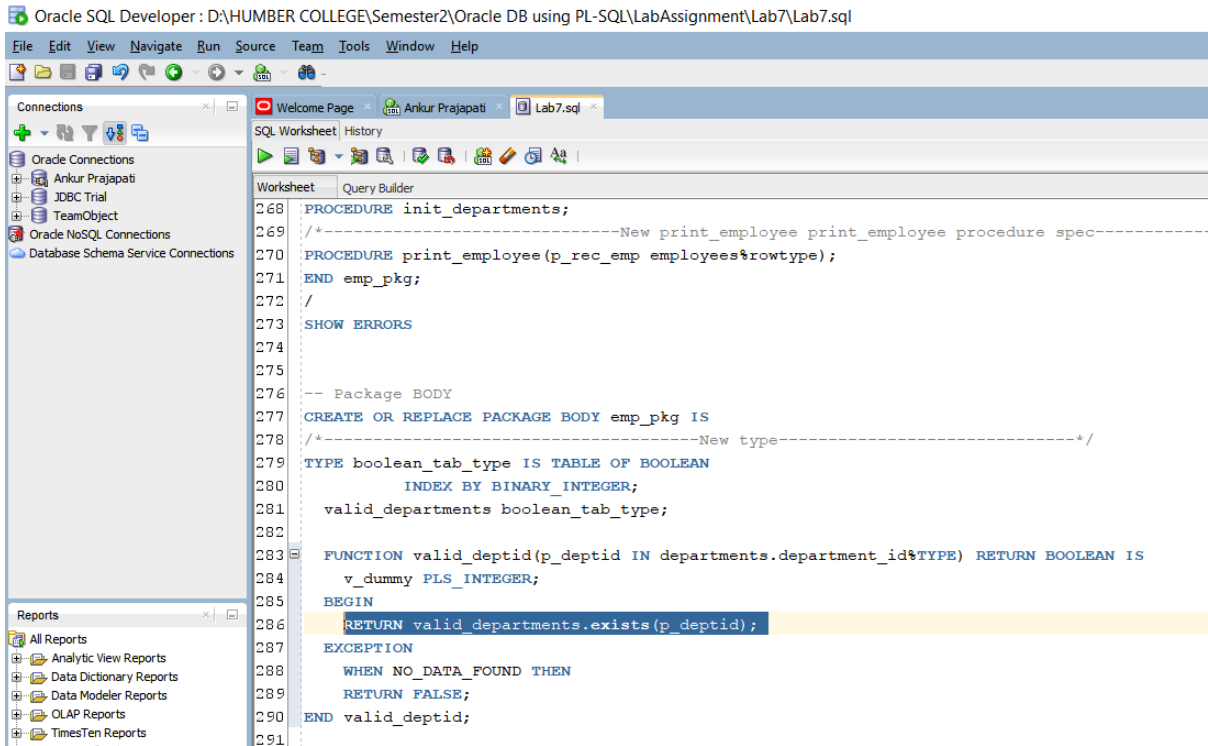
Creating initial block to call procedure init_departments.



Compiling emp_pkg and It gets compiled successfully.

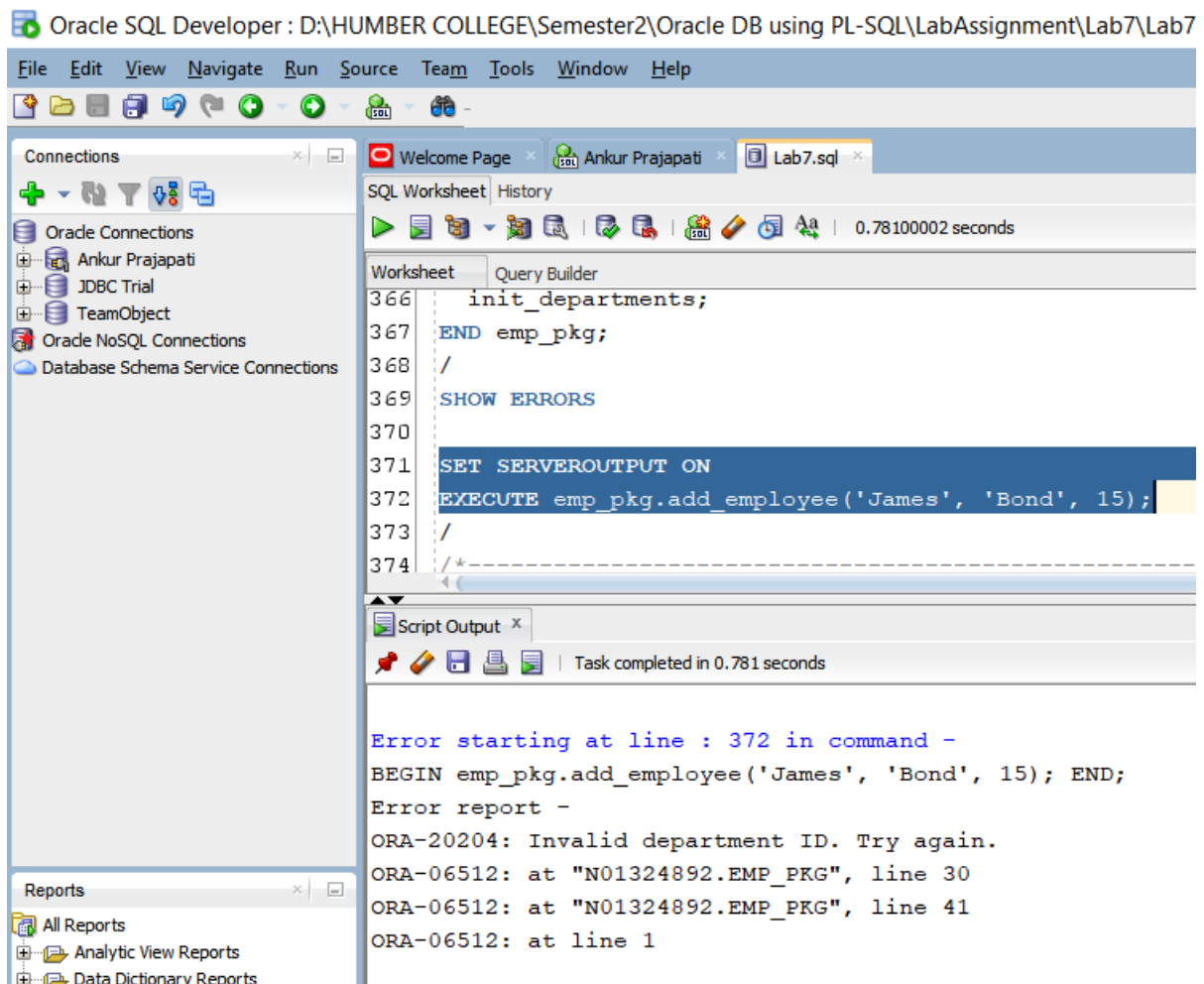
Practice 2 – 4:

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



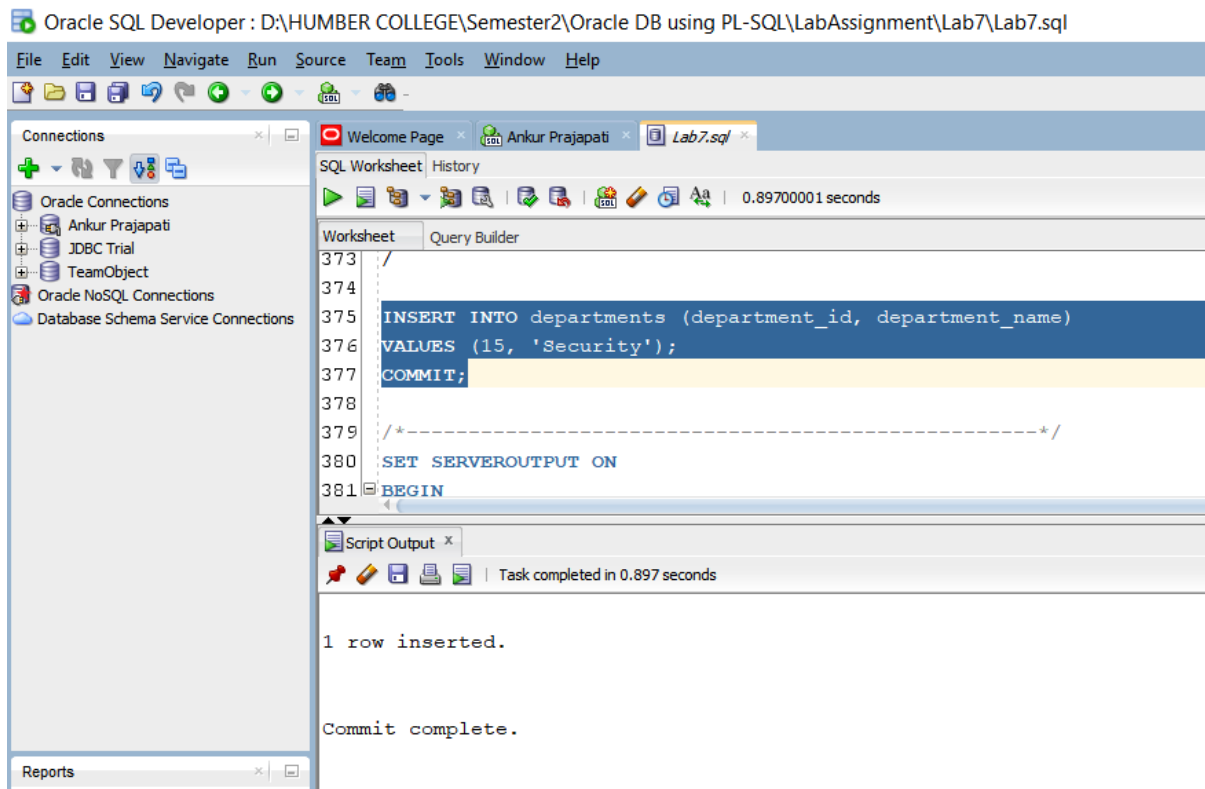
```
268 PROCEDURE init_departments;
269 /-----New print_employee print_employee procedure spec-----
270 PROCEDURE print_employee(p_rec_emp employees%rowtype);
271 END emp_pkg;
272 /
273 SHOW ERRORS
274
275
276 -- Package BODY
277 CREATE OR REPLACE PACKAGE BODY emp_pkg IS
278 /-----New type-----*/
279 TYPE boolean_tab_type IS TABLE OF BOOLEAN
280     INDEX BY BINARY_INTEGER;
281     valid_departments boolean_tab_type;
282
283 FUNCTION valid_deptid(p_deptid IN departments.department_id%TYPE) RETURN BOOLEAN IS
284     v_dummy PLS_INTEGER;
285 BEGIN
286     RETURN valid_departments.exists(p_deptid);
287 EXCEPTION
288     WHEN NO_DATA_FOUND THEN
289         RETURN FALSE;
290 END valid_deptid;
291
```

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



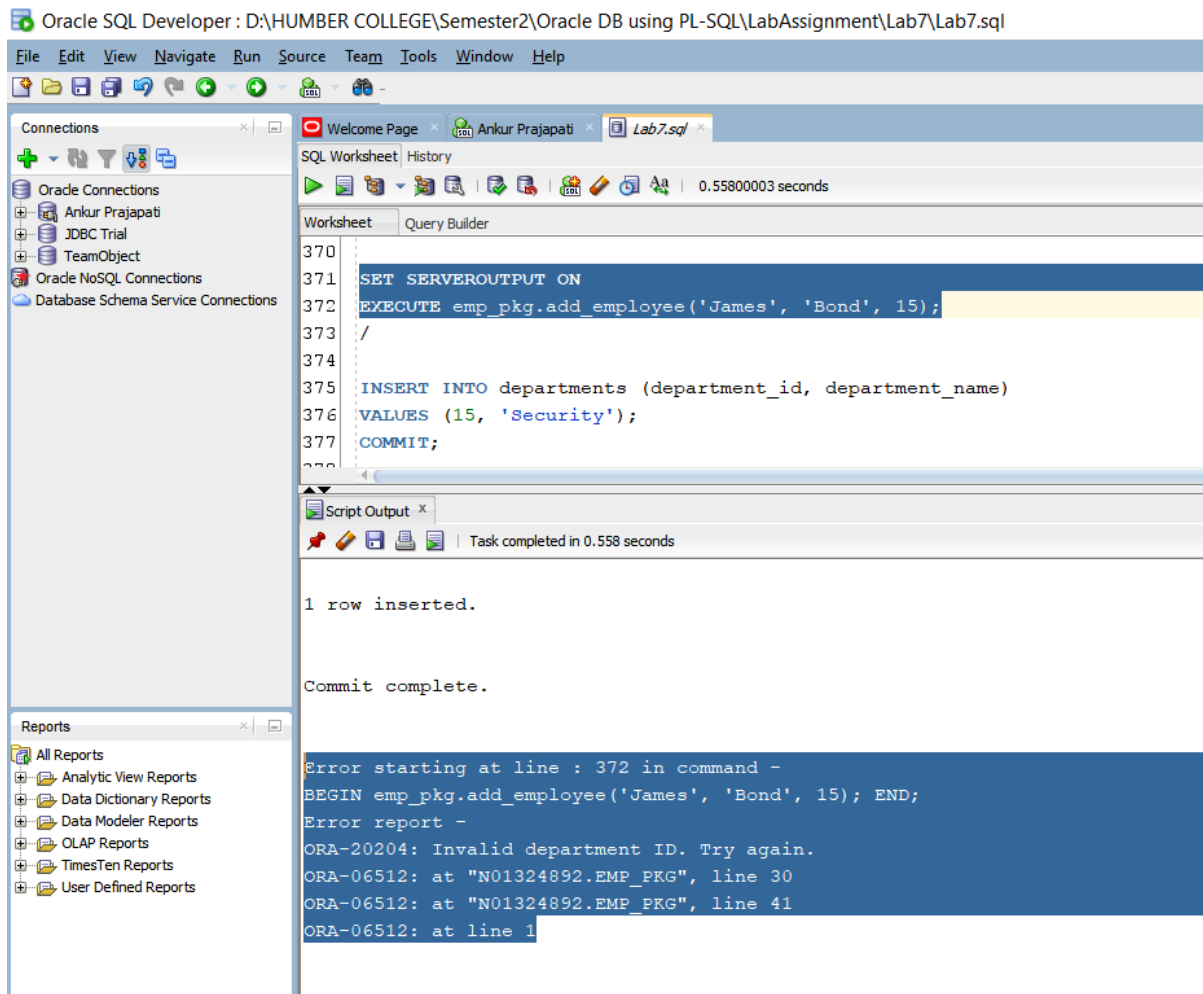
We are trying to add new employee record with first name James and last name of Bond and department id 15.

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.

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

File Edit View Navigate Run Source Team Tools Window Help

Connections

Oracle Connections

Ankur Prajapati

Tables (Filtered)

- COUNTRIES
- CUSTOMERS
- DEPARTMENTS
- EMP
- EMPL
- EMPLOYEES
- EMPLOYEES1
- JOB_HISTORY
- JOBS
- LOCATIONS
- MESSAGES
- REGIONS
- RETIRED_EMPS
- TOP_SALARIES

Views

Indexes

Packages

Procedures

Functions

Operators

Reports

All Reports

- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimesTen Reports
- User Defined Reports

SQL Worksheet

History

0.085 seconds

Worksheet

Query Builder

```

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;

```

Script Output

Task completed in 0.085 seconds

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

File Edit View Navigate Run Source Team Tools Window Help

Connections

Oracle Connections

Ankur Prajapati

Tables (Filtered)

- COUNTRIES
- CUSTOMERS
- DEPARTMENTS
- EMP
- EMPL
- EMPLOYEES
- EMPLOYEES1
- JOB_HISTORY
- JOBS
- LOCATIONS
- MESSAGES
- REGIONS
- RETIRED_EMPS
- TOP_SALARIES

Views

Indexes

Packages

Procedures

Functions

Operators

Reports

All Reports

- Analytic View Reports
- Data Dictionary Reports
- Data Modeler Reports
- OLAP Reports
- TimesTen Reports
- User Defined Reports

SQL Worksheet

History

0.096 seconds

Worksheet

Query Builder

```

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
380 INSERT INTO departments (department_id, department_name)
381 VALUES (15, 'Security');
382 COMMIT;
383
384 /*-----*/
385 SET SERVEROUTPUT ON
386 BEGIN
387     emp_pkg.print_employee(emp_pkg.get_employee(100));
388     emp_pkg.print_employee(emp_pkg.get_employee('Joplin'));
389 END;

```

Script Output

Task completed in 0.096 seconds

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	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
File Edit View Navigate Run Source Team Tools Window Help

Connections
Oracle Connections
Ankur Prajapati
Tables (Filtered)
COUNTRIES
CUSTOMERS
DEPARTMENTS
EMP
EMPL
EMPLOYEES
EMPLOYEES1
JOB_HISTORY
JOBS
LOCATIONS
MESSAGES
REGIONS
RETIRED_EMPS
TOP_SALARIES
Views
Indexes
Packages
Procedures
Functions
Operators

SQL Worksheet History
Welcome Page x Ankur Prajapati x Lab7.sql x EMPLOYEES x
1.74800003 seconds

Worksheet Query Builder
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) VALUES (15, 'Securit
380 COMMIT;
381 /*-----Practice 2 - 4 - g*/
382 DELETE FROM employees
383 WHERE first_name = 'James' AND last_name = 'Bond';
384 DELETE FROM departments WHERE department_id = 15;
385 COMMIT;
386 EXECUTE EMP_PKG.INIT_DEPARTMENTS
387
388 /*-----*/

Reports
All Reports
Analytic View Reports
Data Dictionary Reports
Data Modeler Reports
OLAP Reports
TimesTen Reports
User Defined Reports

Script Output x
Task completed in 1.748 seconds

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.