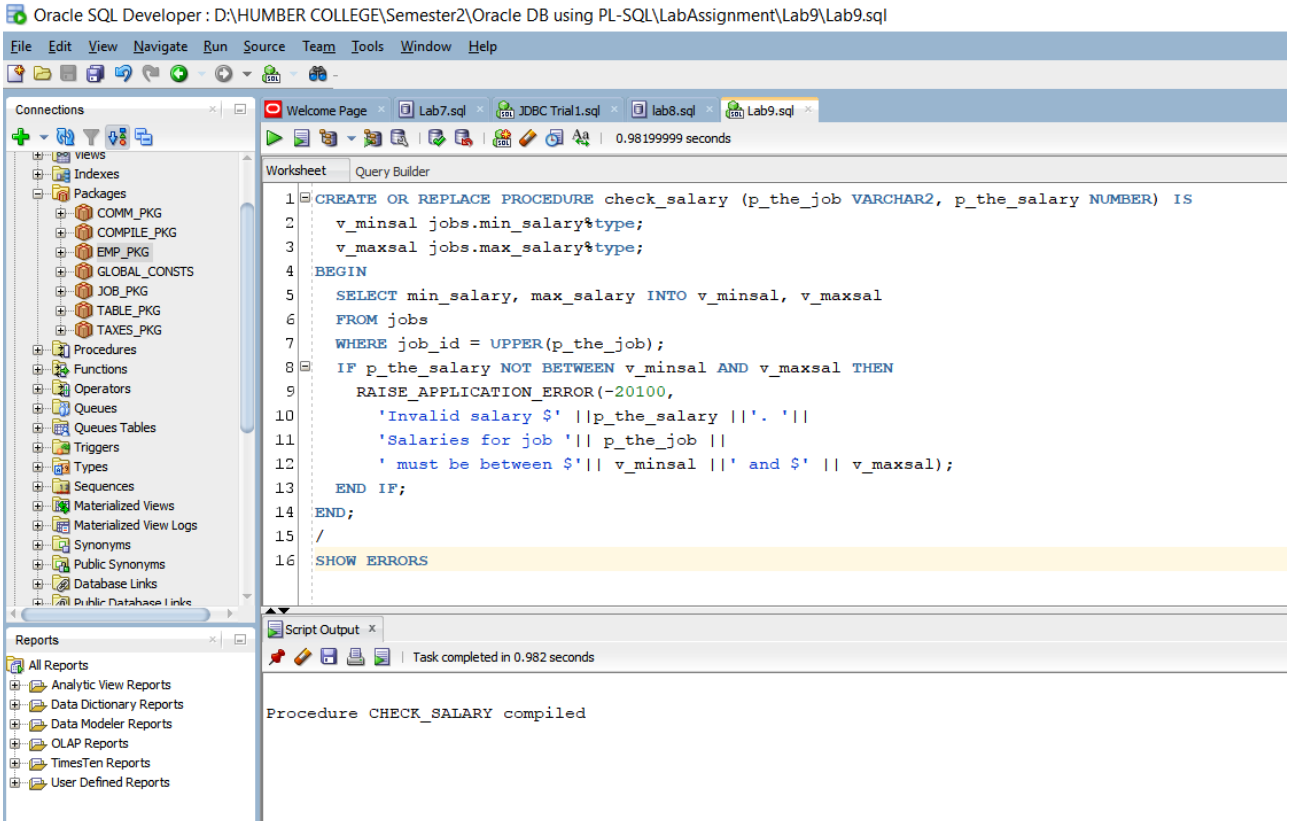
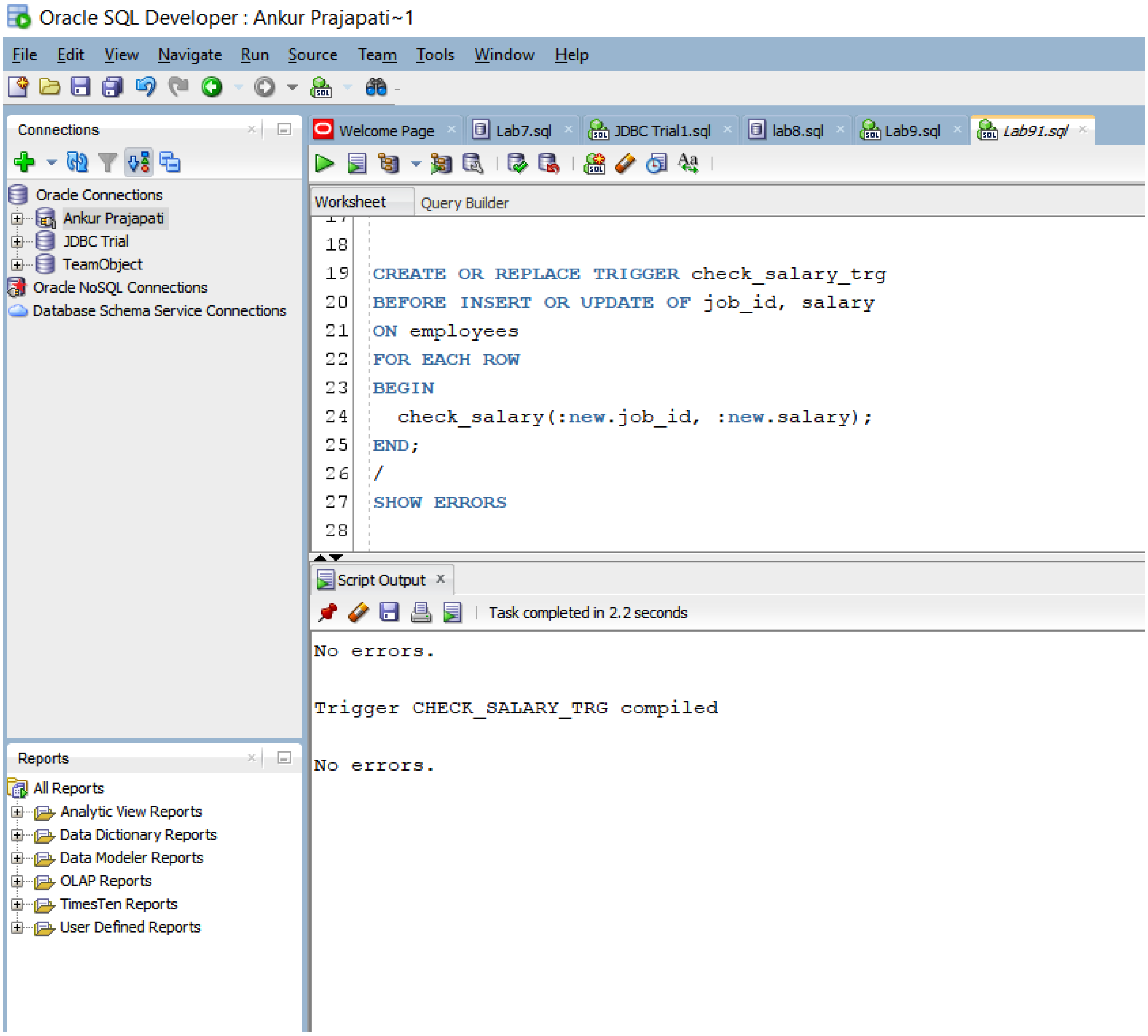
Practice – 1 - 1:

- Creation of procedure check_salary with parameters p_the_job with type varchar2 and p_the_salary with type number and we are declaring two variables v_minsal and v_maxsal with type of jobs.min_salary. After that we are selecting minimum and maximum salary from jobs where job_id matches with the parameters p_the_job. If the salary is not between min and max salary it will raise an error notifying that it is invalid salary.
- Creation of trigger check_salary_trg which will be triggered when an insert or update happens for job_id and salary for employees table.
- Note that we are calling this for each and every row for job_id. Trigger will pass the new job_id and salary to the procedure parameters.



```
CREATE OR REPLACE PROCEDURE check_salary (p_the_job VARCHAR2, p_the_salary NUMBER) IS
  v_minsal jobs.min_salary%type;
  v_maxsal jobs.max_salary%type;
BEGIN
  SELECT min_salary, max_salary INTO v_minsal, v_maxsal
  FROM jobs
  WHERE job_id = UPPER(p_the_job);
  IF p_the_salary NOT BETWEEN v_minsal AND v_maxsal THEN
    RAISE_APPLICATION_ERROR(-20100,
      'Invalid salary $' ||p_the_salary ||'. '||
      'Salaries for job '|| p_the_job ||
      ' must be between $'|| v_minsal ||' and $' || v_maxsal);
  END IF;
END;
/
SHOW ERRORS
```

```
Procedure CHECK_SALARY compiled
```

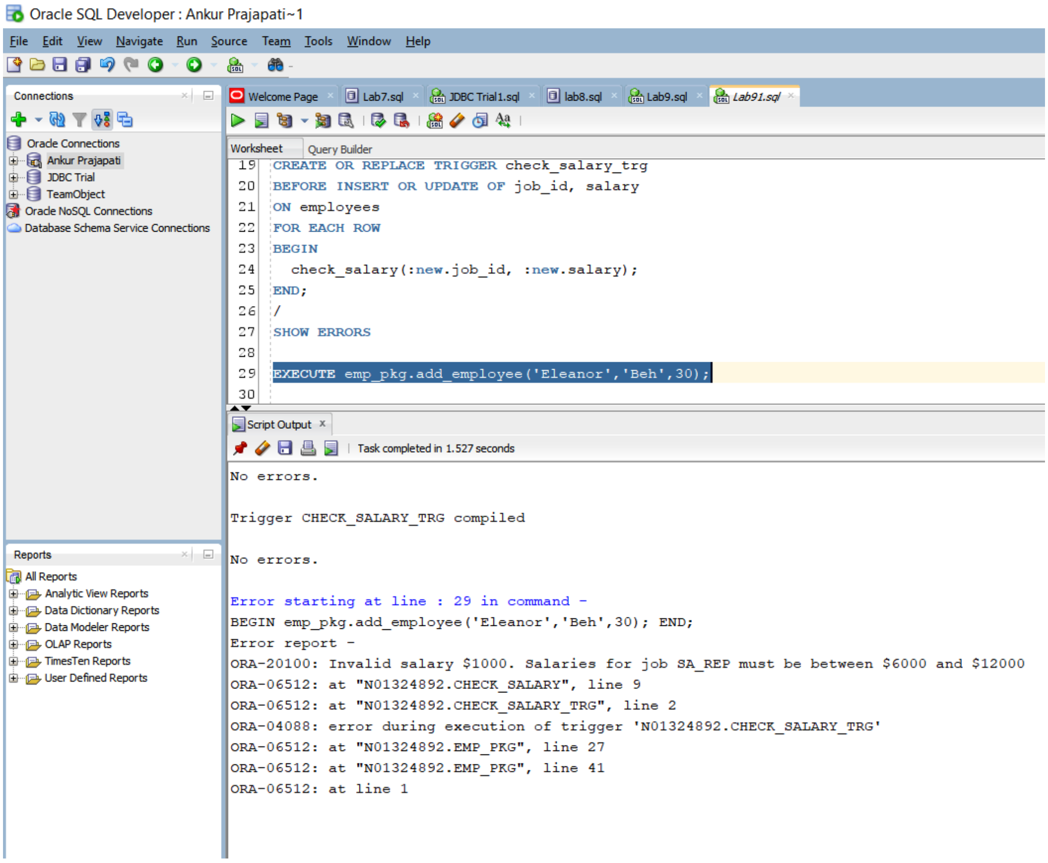


```
CREATE OR REPLACE TRIGGER check_salary_trg
BEFORE INSERT OR UPDATE OF job_id, salary
ON employees
FOR EACH ROW
BEGIN
  check_salary(:new.job_id, :new.salary);
END;
/
SHOW ERRORS
```

```
No errors.

Trigger CHECK_SALARY_TRG compiled

No errors.
```

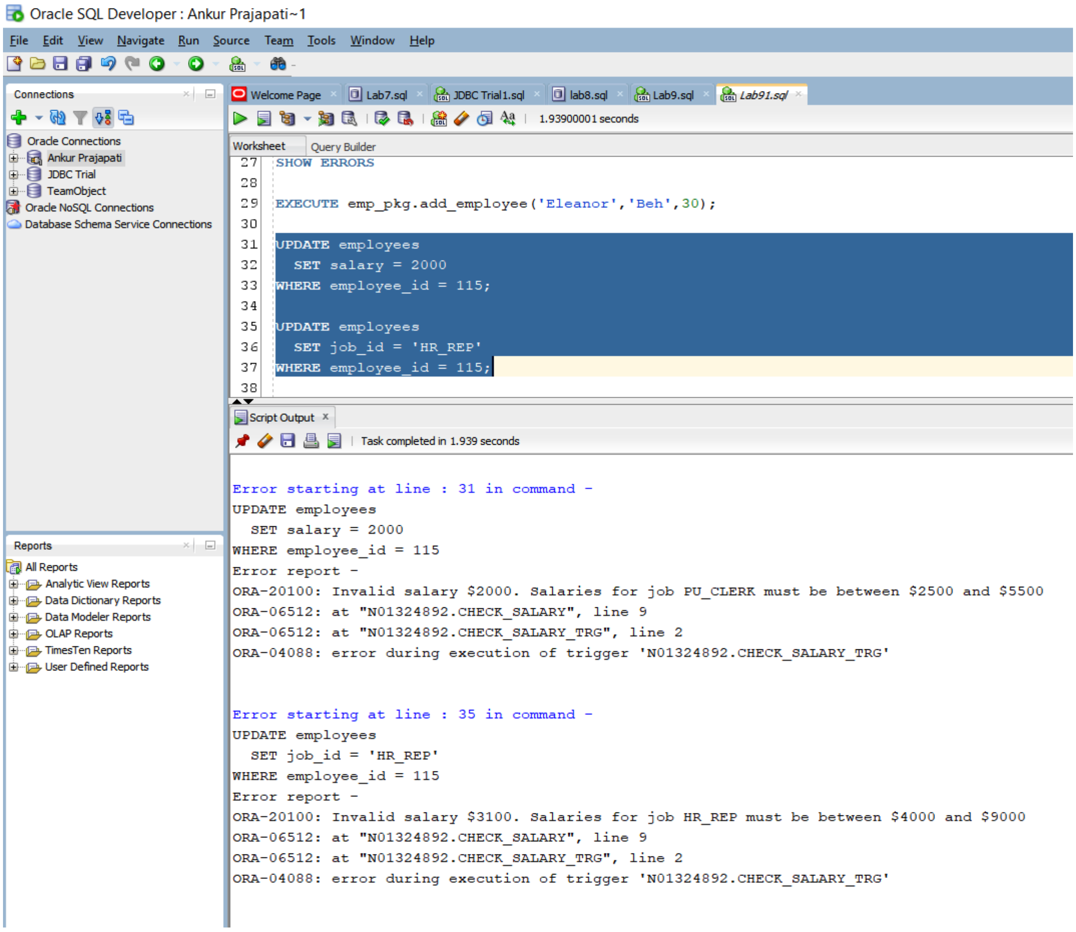
Practice – 1 - 2:

- We are executing emp_pkg.add_employee with parameters ‘Eleanor’, ‘Beh’, 30. Note that this salary ($1000) is invalid because it should be between min salary $6000 and max salary $12000. Due to check_salary_trg trigger we are actually getting this error. Here, procedure invokes an overloaded version of itself that uses the default salary of $1000 and default job_id of ‘SA_REP’
- Here, first update statement fails to set the salary to $2000 where employee_id = 115. Because min salary of job_id ‘PU_CLERK’ is $2500. That’s why an application error is invoke during updating data. In Second update statement same happens, here min salary of HR_REP is $4000 and current employee’s salary is less than that salary. That’s why it fails.
- Where in third update statement it gets updated because the new value of salary, we are assigning is greater than the min salary of employee whose id is 115.



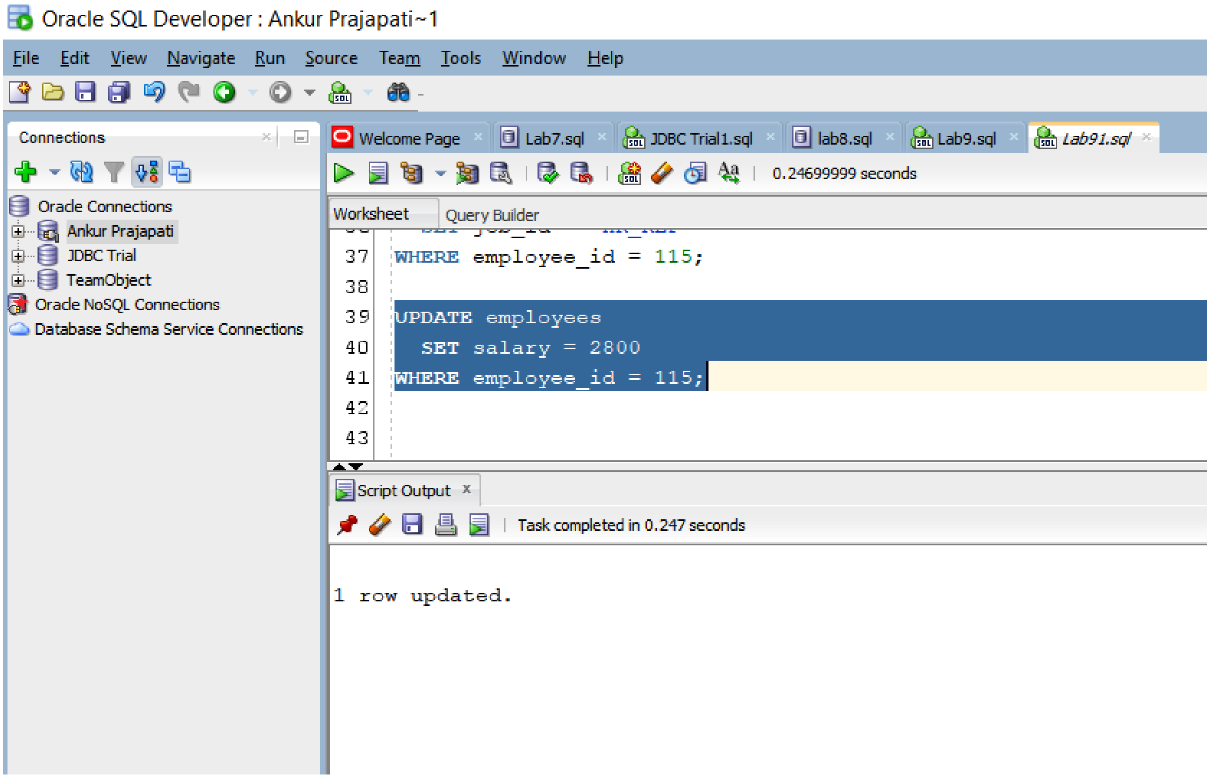
```
19  CREATE OR REPLACE TRIGGER check_salary_trg
20  BEFORE INSERT OR UPDATE OF job_id, salary
21  ON employees
22  FOR EACH ROW
23  BEGIN
24    check_salary(:new.job_id, :new.salary);
25  END;
26  /
27  SHOW ERRORS
28
29  EXECUTE emp_pkg.add_employee('Eleanor','Beh',30);
30
```

```
No errors.

Trigger CHECK_SALARY_TRG compiled

No errors.

Error starting at line : 29 in command -
BEGIN emp_pkg.add_employee('Eleanor','Beh',30); END;
Error report -
ORA-20100: Invalid salary $1000. Salaries for job SA_REP must be between $6000 and $12000
ORA-06512: at "N01324892.CHECK_SALARY", line 9
ORA-06512: at "N01324892.CHECK_SALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.CHECK_SALARY_TRG'
ORA-06512: at "N01324892.EMP_PKG", line 27
ORA-06512: at "N01324892.EMP_PKG", line 41
ORA-06512: at line 1
```



```
27  SHOW ERRORS
28
29  EXECUTE emp_pkg.add_employee('Eleanor','Beh',30);
30
31  UPDATE employees
32    SET salary = 2000
33  WHERE employee_id = 115;
34
35  UPDATE employees
36    SET job_id = 'HR_REP'
37  WHERE employee_id = 115;
38
```

```
Error starting at line : 31 in command -
UPDATE employees
  SET salary = 2000
WHERE employee_id = 115
Error report -
ORA-20100: Invalid salary $2000. Salaries for job PU_CLERK must be between $2500 and $5500
ORA-06512: at "N01324892.CHECK_SALARY", line 9
ORA-06512: at "N01324892.CHECK_SALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.CHECK_SALARY_TRG'

Error starting at line : 35 in command -
UPDATE employees
  SET job_id = 'HR_REP'
WHERE employee_id = 115
Error report -
ORA-20100: Invalid salary $3100. Salaries for job HR_REP must be between $4000 and $9000
ORA-06512: at "N01324892.CHECK_SALARY", line 9
ORA-06512: at "N01324892.CHECK_SALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.CHECK_SALARY_TRG'
```

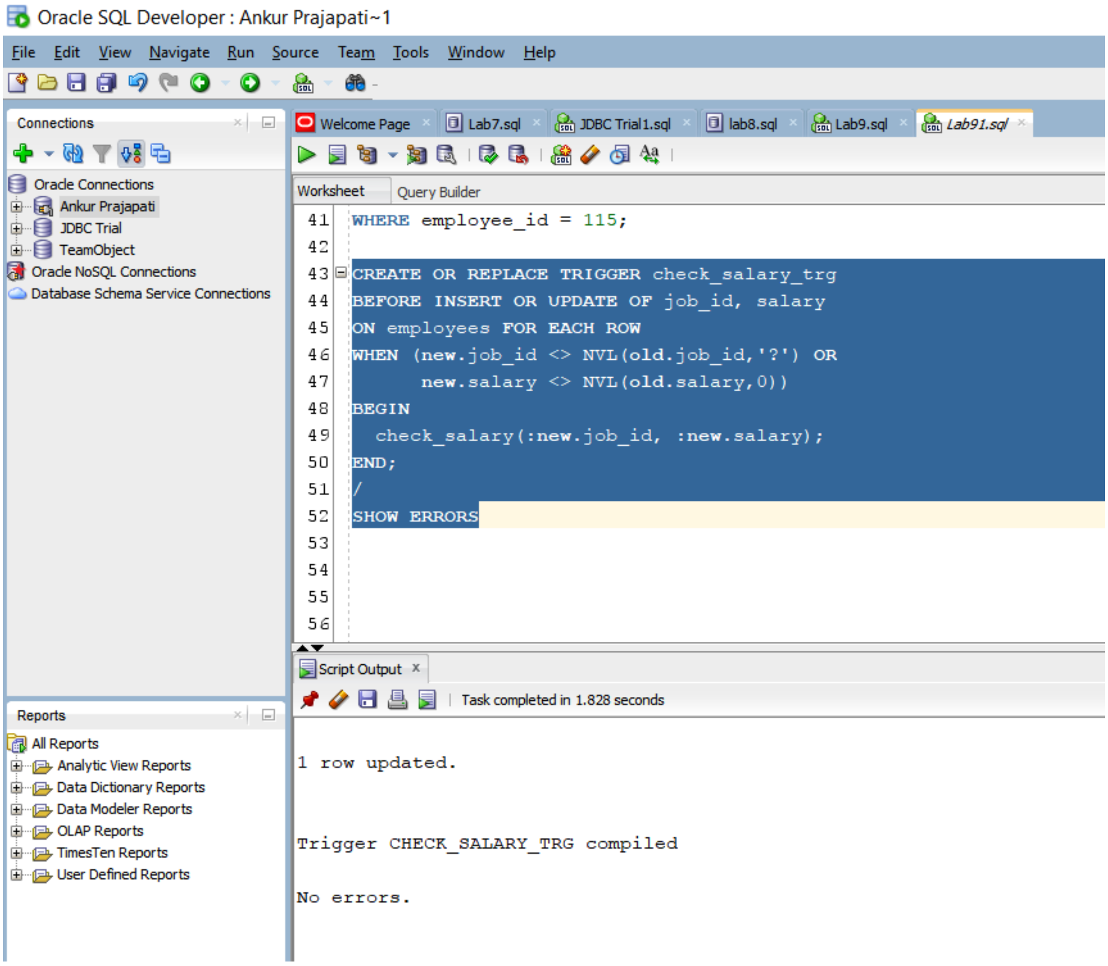


```
WHERE employee_id = 115;

UPDATE employees
  SET salary = 2800
WHERE employee_id = 115;
```

```
1 row updated.
```

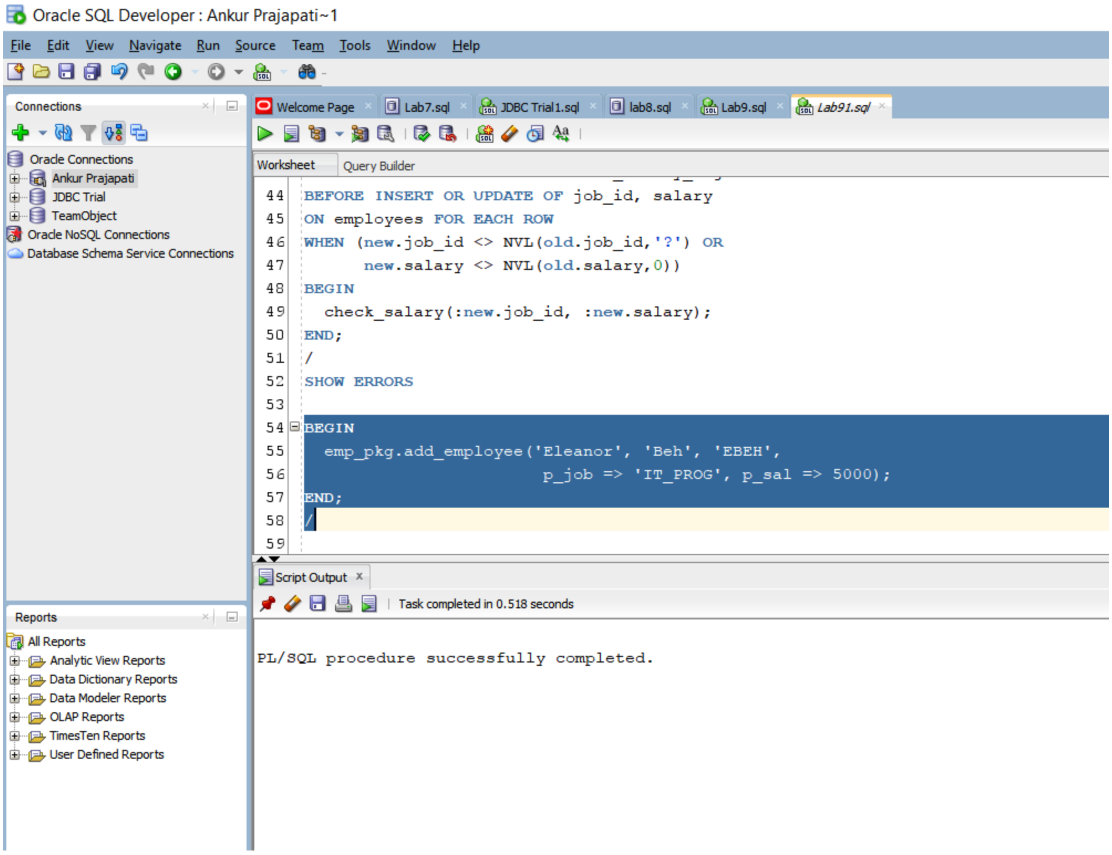
Practice – 1 – 3:

- We have created trigger check_salary_trg but here we are updating it by adding new business rule to that trigger. Basically, we are checking that whether job_id or salary values have changed or not using WHEN clause.
- After that we are executing emp_pkg.add_employee to test the trigger created. On executing it gets executed successfully.
- Here, we are updating salary of employees by $ 2000 where job_id is 'IT_PROG'. But we are not able to do that because of the trigger we created. Suppose we do that then it exceeds maximum salary of IT_PROG. It raised an application error notifying that salaries for IT_PROG must be between 4000 to 10000.
- Here we are updating salary for employee Beh by selecting employee_id of Beh. We are setting salary of employee Beh to $ 9000. And it gets executed successfully. Because 9000 is between min salary and max salary.
- Here we are changing job of Eleanor Beh to ST_MAN using update statement with another subquery but it dosen't get updated because that employee's salary is between 5500 and 8500. Where Beh's salary is more than maximum salary. That's why we are not able to change job.



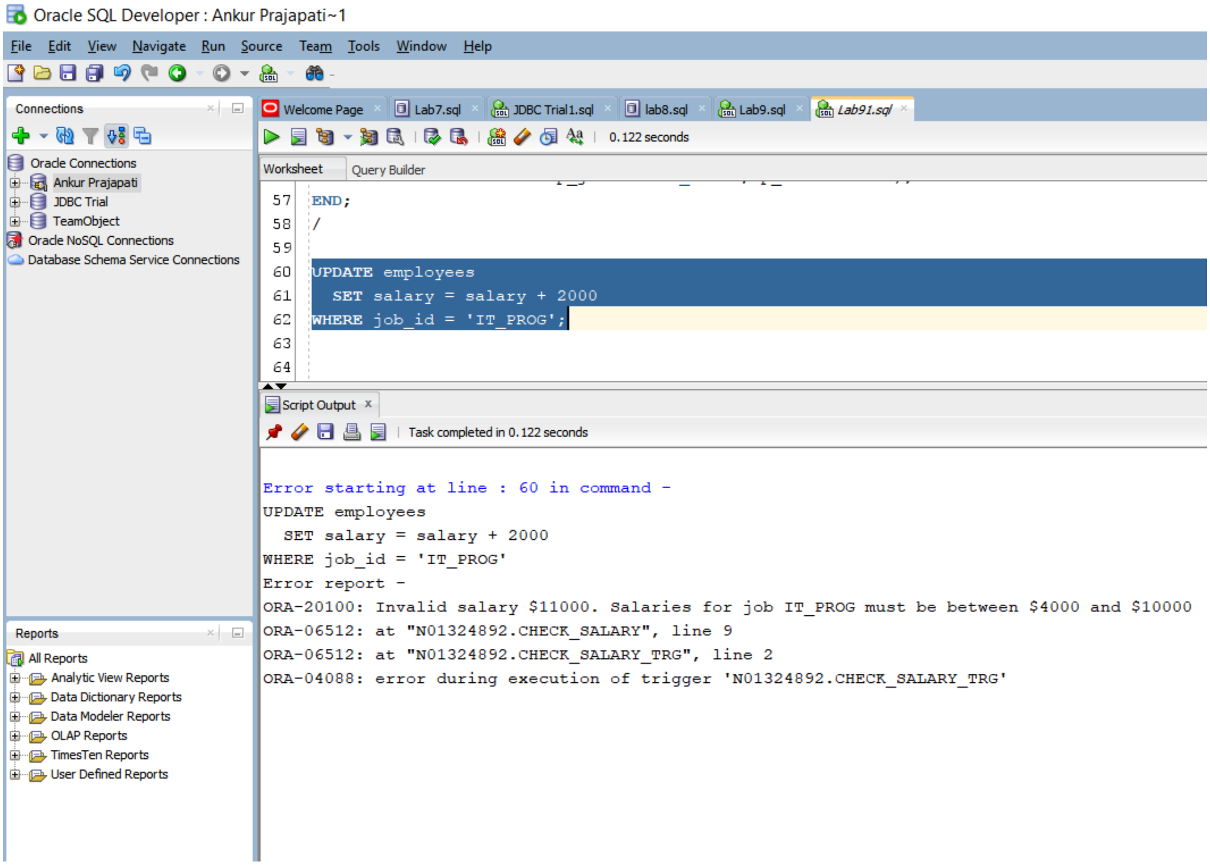
```
WHERE employee_id = 115;

CREATE OR REPLACE TRIGGER check_salary_trg
BEFORE INSERT OR UPDATE OF job_id, salary
ON employees FOR EACH ROW
WHEN (new.job_id <> NVL(old.job_id,'?') OR
      new.salary <> NVL(old.salary,0))
BEGIN
  check_salary(:new.job_id, :new.salary);
END;
/
SHOW ERRORS
```

```
1 row updated.


Trigger CHECK_SALARY_TRG compiled

No errors.
```



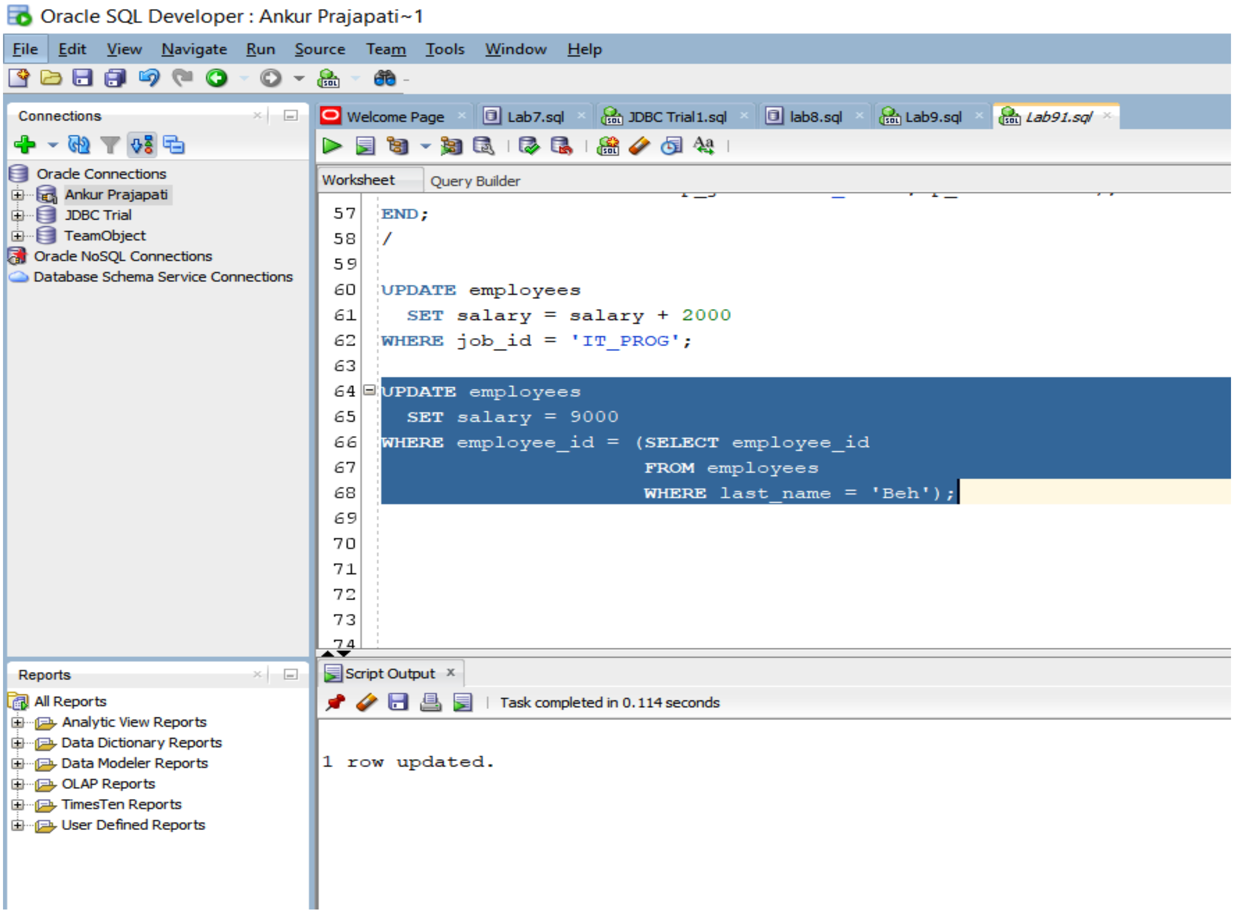
```
BEFORE INSERT OR UPDATE OF job_id, salary
ON employees FOR EACH ROW
WHEN (new.job_id <> NVL(old.job_id,'?') OR
      new.salary <> NVL(old.salary,0))
BEGIN
  check_salary(:new.job_id, :new.salary);
END;
/
SHOW ERRORS

BEGIN
  emp_pkg.add_employee('Eleanor', 'Beh', 'EBEH',
                       p_job => 'IT_PROG', p_sal => 5000);
END;
/
```

```
PL/SQL procedure successfully completed.
```



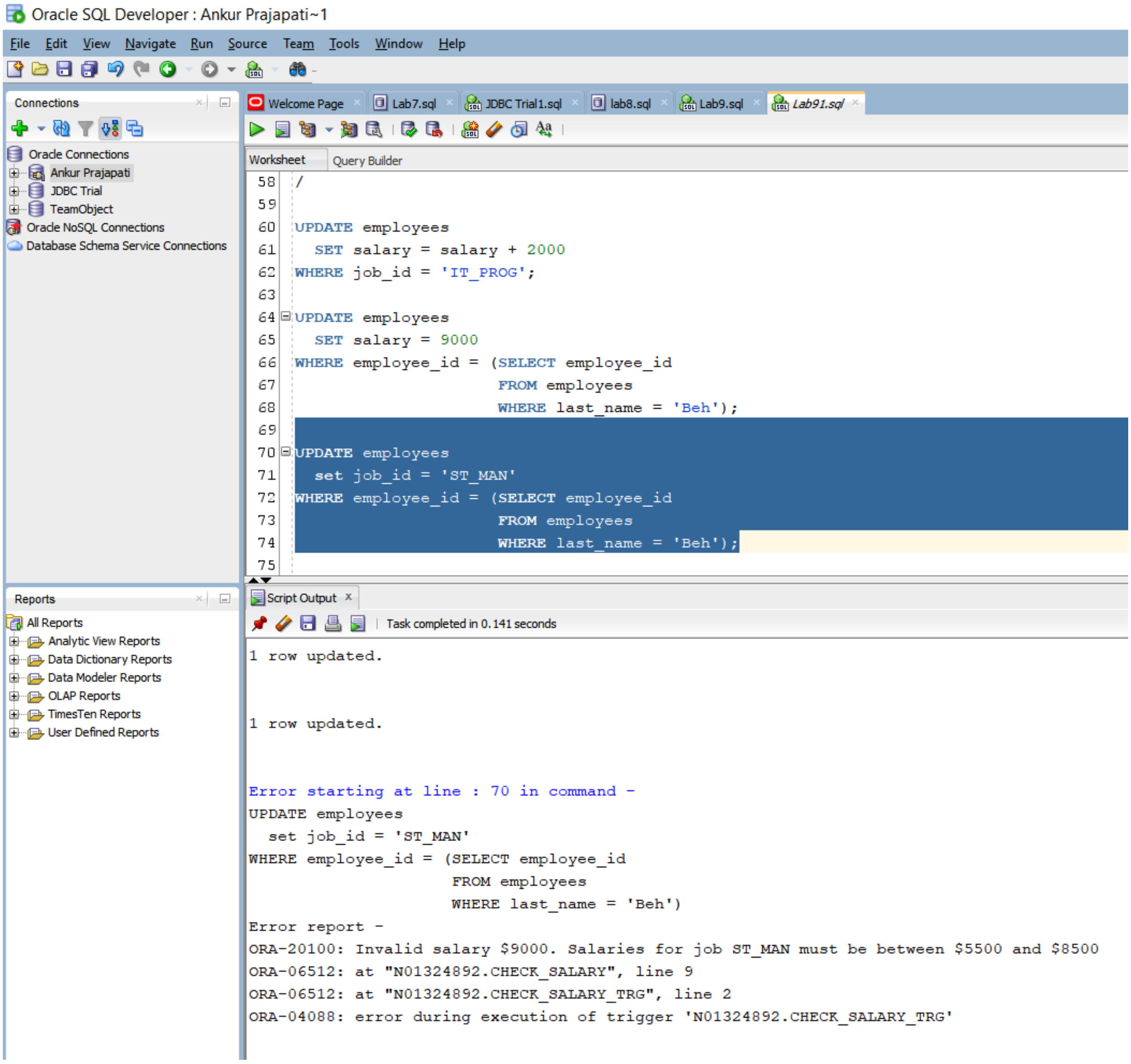
```
57  END;
58  /
59
60  UPDATE employees
61    SET salary = salary + 2000
62  WHERE job_id = 'IT_PROG';
63
64
```

```
Error starting at line : 60 in command -
UPDATE employees
  SET salary = salary + 2000
WHERE job_id = 'IT_PROG'
Error report -
ORA-20100: Invalid salary $11000. Salaries for job IT_PROG must be between $4000 and $10000
ORA-06512: at "N01324892.CHECK_SALARY", line 9
ORA-06512: at "N01324892.CHECK_SALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.CHECK_SALARY_TRG'
```



```
57  END;
58  /
59
60  UPDATE employees
61    SET salary = salary + 2000
62  WHERE job_id = 'IT_PROG';
63
64  UPDATE employees
65    SET salary = 9000
66  WHERE employee_id = (SELECT employee_id
67                       FROM employees
68                       WHERE last_name = 'Beh');
69
70
71
72
73
74
```

```
1 row updated.
```

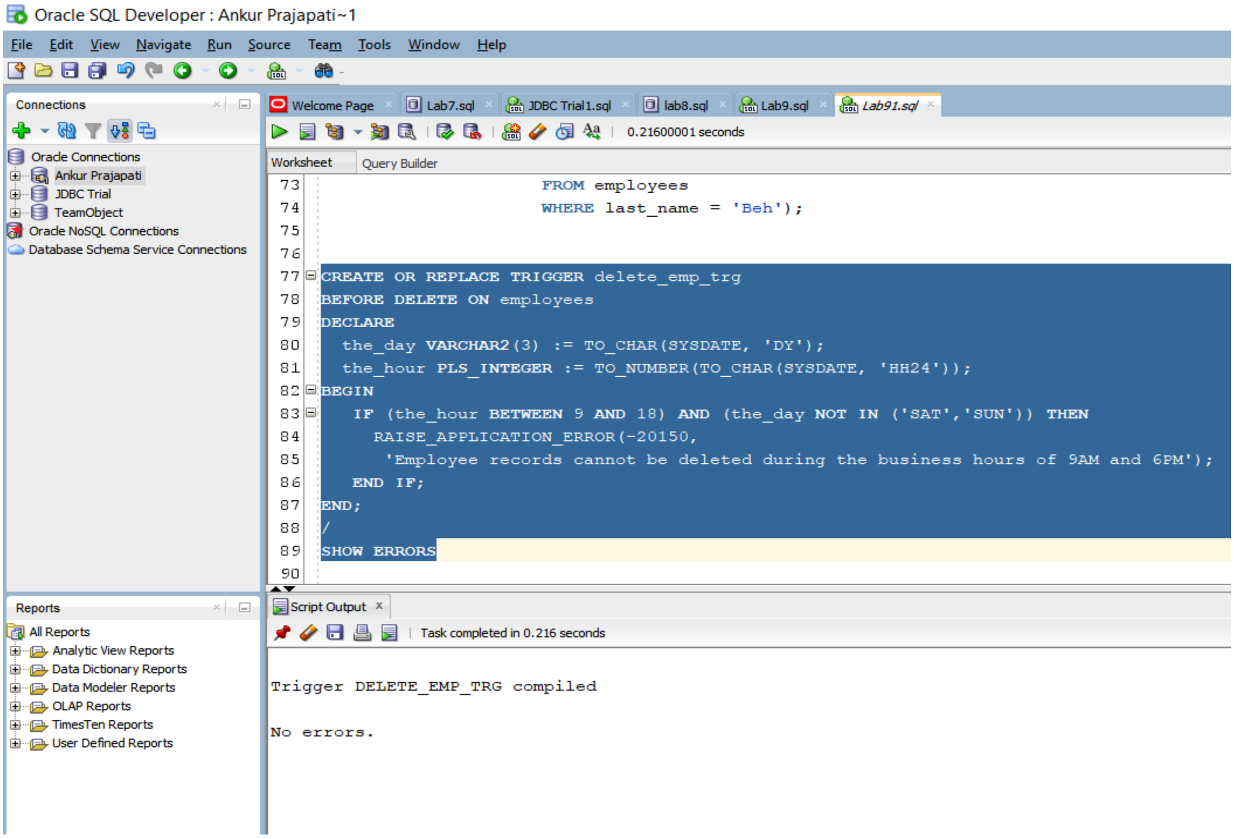


```
/

UPDATE employees
  SET salary = salary + 2000
WHERE job_id = 'IT_PROG';

UPDATE employees
  SET salary = 9000
WHERE employee_id = (SELECT employee_id
                     FROM employees
                     WHERE last_name = 'Beh');

UPDATE employees
  set job_id = 'ST_MAN'
WHERE employee_id = (SELECT employee_id
                     FROM employees
                     WHERE last_name = 'Beh');
```

```
1 row updated.


1 row updated.


Error starting at line : 70 in command -
UPDATE employees
  set job_id = 'ST_MAN'
WHERE employee_id = (SELECT employee_id
                     FROM employees
                     WHERE last_name = 'Beh')
Error report -
ORA-20100: Invalid salary $9000. Salaries for job ST_MAN must be between $5500 and $8500
ORA-06512: at "N01324892.CHECK_SALARY", line 9
ORA-06512: at "N01324892.CHECK_SALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.CHECK_SALARY_TRG'
```

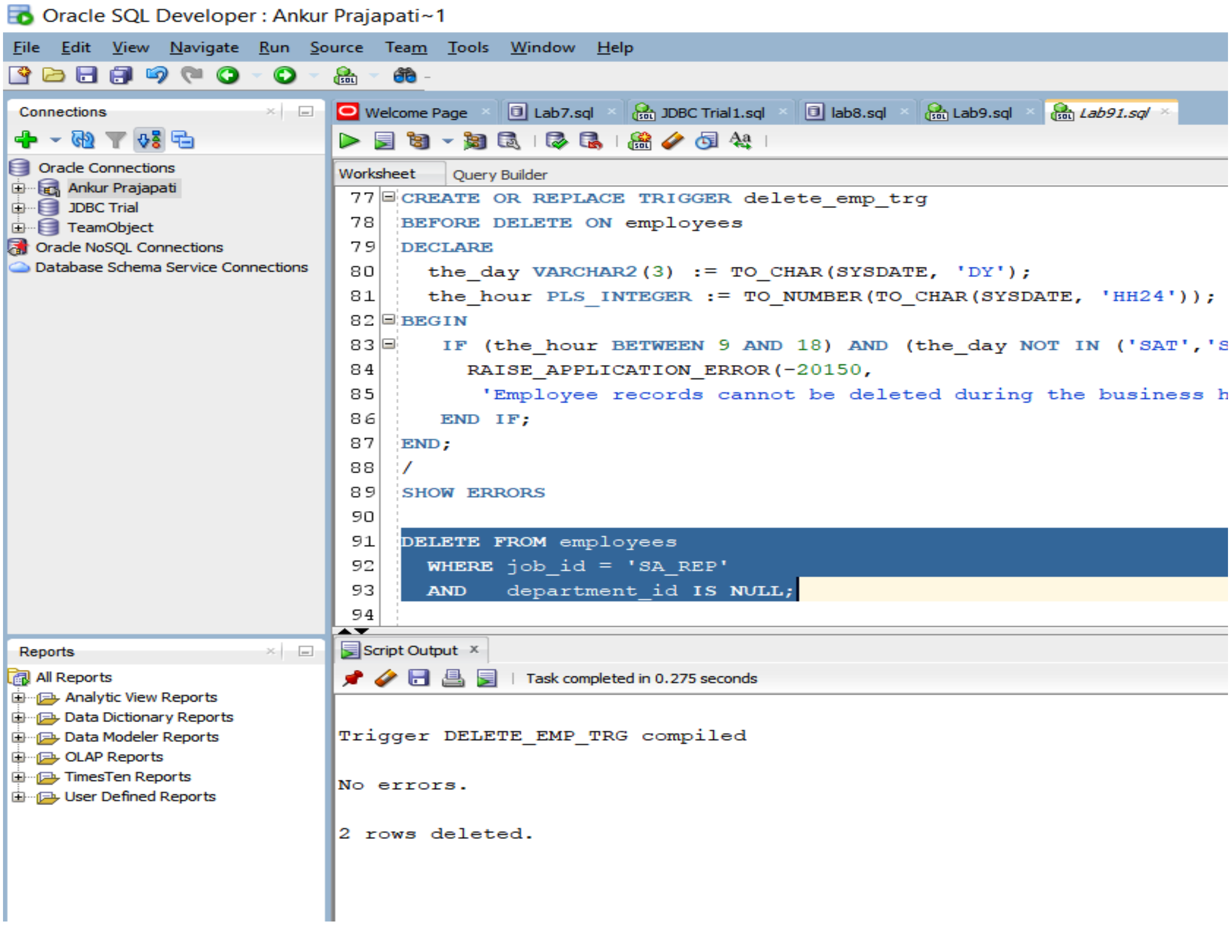
Practice - 1 – 4:

- We are writing a trigger called delete_emp_trg on the employees table to prevent an employee from being deleted during weekday business hours in between 9:00 AM to 6:00 PM. Trigger gets complied successfully.
- After that we are trying to delete from employees table. Note that it gets deleted because current user is trying to delete from employees table after 6:00 PM. That's why it displays 2 rows deleted.



```
                        FROM employees
                        WHERE last_name = 'Beh');


CREATE OR REPLACE TRIGGER delete_emp_trg
BEFORE DELETE ON employees
DECLARE
  the_day VARCHAR2(3)  := TO_CHAR(SYSDATE, 'DY');
  the_hour PLS_INTEGER := TO_NUMBER(TO_CHAR(SYSDATE, 'HH24'));
BEGIN
   IF (the_hour BETWEEN 9 AND 18) AND (the_day NOT IN ('SAT','SUN')) THEN
     RAISE_APPLICATION_ERROR(-20150,
      'Employee records cannot be deleted during the business hours of 9AM and 6PM');
   END IF;
END;
/
SHOW ERRORS

```

```
Trigger DELETE_EMP_TRG compiled

No errors.
```

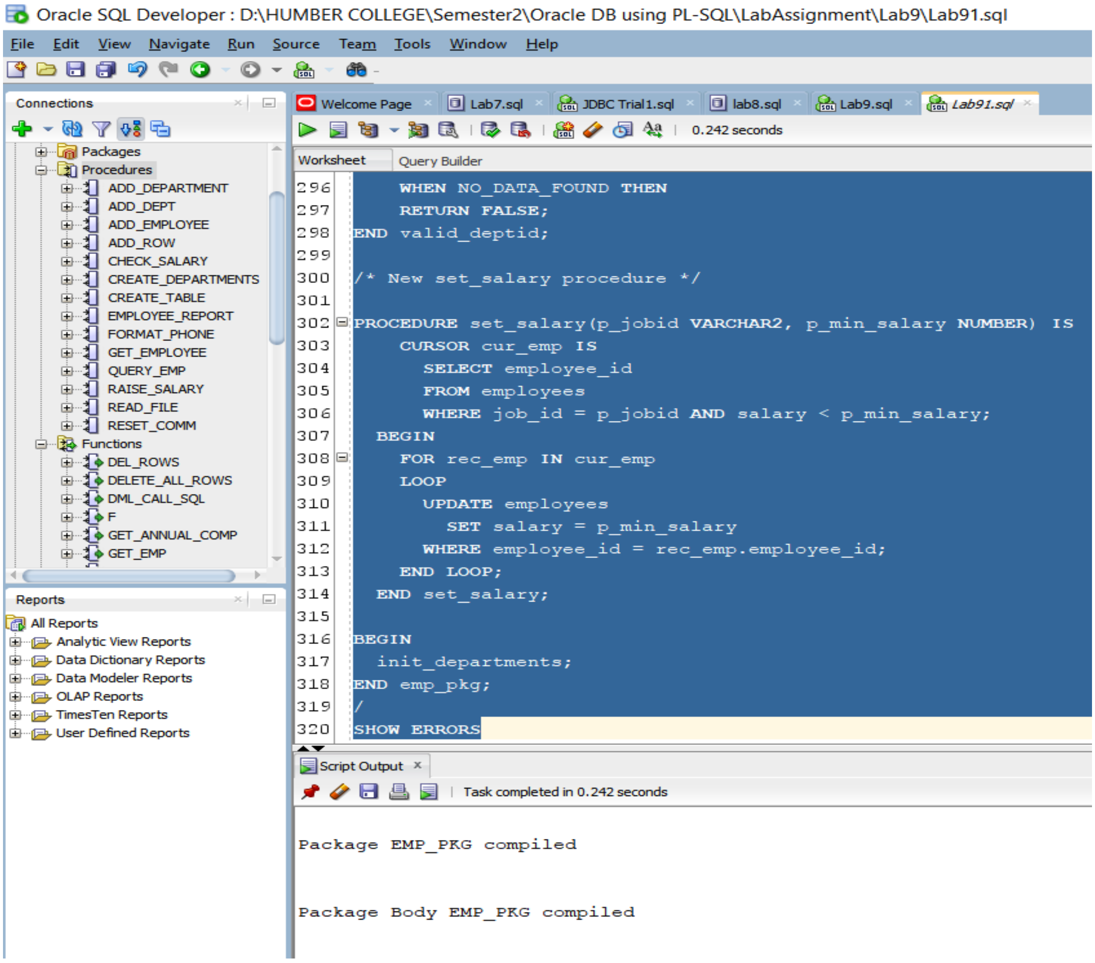


```
CREATE OR REPLACE TRIGGER delete_emp_trg
BEFORE DELETE ON employees
DECLARE
  the_day VARCHAR2(3)  := TO_CHAR(SYSDATE, 'DY');
  the_hour PLS_INTEGER := TO_NUMBER(TO_CHAR(SYSDATE, 'HH24'));
BEGIN
   IF (the_hour BETWEEN 9 AND 18) AND (the_day NOT IN ('SAT','S
     RAISE_APPLICATION_ERROR(-20150,
      'Employee records cannot be deleted during the business h
   END IF;
END;
/
SHOW ERRORS

DELETE FROM employees
  WHERE job_id = 'SA_REP'
  AND   department_id IS NULL;

```

```
Trigger DELETE_EMP_TRG compiled

No errors.

2 rows deleted.
```

Practice – 2 – 1:

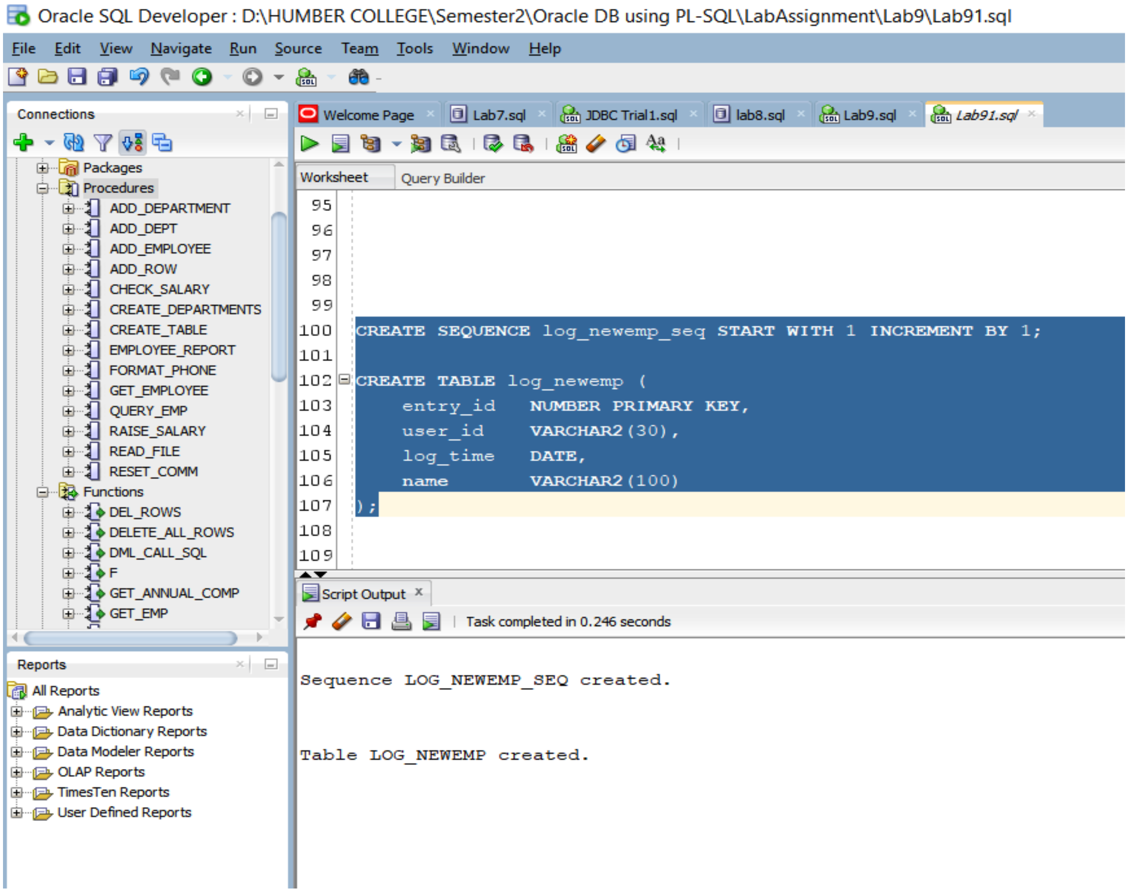
- We will implement a business rule to ensure data integrity of employee's salaries with respect to the valid salary range for their jobs. For that employees receive an automatic increase in salary if the minimum salary for a job is increased to a value larger than their salaries. To implement this, we have to update emp_pkg. On updating data, it will call the check_salary trigger to read the salaries from jobs table. We have added procedure called SET_SALARY that updates the employee's salaries.
- Note that we weren't able to run that package emp_pkg because we need to create table called log_newemp_seq and to create that table we have created sequence that starts with 1 and increments by 1. After creating this table when we run this emp_pkg gets complied successfully.
- After that we are creating a trigger named upd_minsalary_trg on the JOBS table that will invoke emp_pkg.set_salary procedure when minimum salary in the JOBS table is updated for a specified job ID.
- In next part we are writing a query to display employeeID, last name, job ID, current salary and minimum salary for employees whose job_id is 'IT-PROG'. After that we are updating minimum salary by $ 1000. Note that we were successful to fetch the 6 rows using select query. But we cannot update salary because upd_minsalary_trg calls setSalary, which invokes check_salary_trg which calls check_salary which selects the minimum salary from jobs which puts implicit lock and so mutating error is thrown.



Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab9\Lab91.sql

```
    WHEN NO_DATA_FOUND THEN
    RETURN FALSE;
END valid_deptid;

/* New set_salary procedure */

PROCEDURE set_salary(p_jobid VARCHAR2, p_min_salary NUMBER) IS
    CURSOR cur_emp IS
      SELECT employee_id
      FROM employees
      WHERE job_id = p_jobid AND salary < p_min_salary;
  BEGIN
    FOR rec_emp IN cur_emp
    LOOP
      UPDATE employees
        SET salary = p_min_salary
      WHERE employee_id = rec_emp.employee_id;
    END LOOP;
  END set_salary;

BEGIN
  init_departments;
END emp_pkg;
/
SHOW ERRORS
```

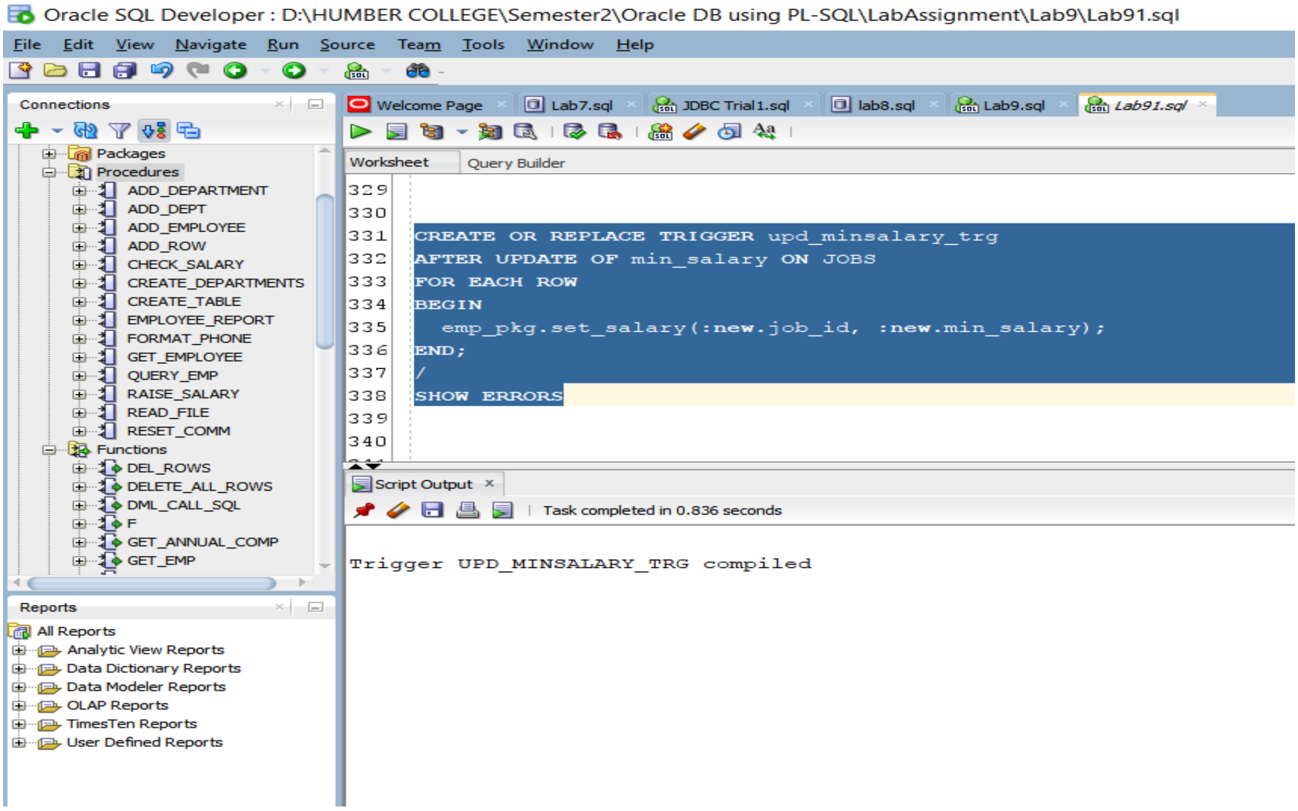
```
Package EMP_PKG compiled


Package Body EMP_PKG compiled
```



Oracle SQL Developer : D:\HUMBER COLLEGE\Semester2\Oracle DB using PL-SQL\LabAssignment\Lab9\Lab91.sql

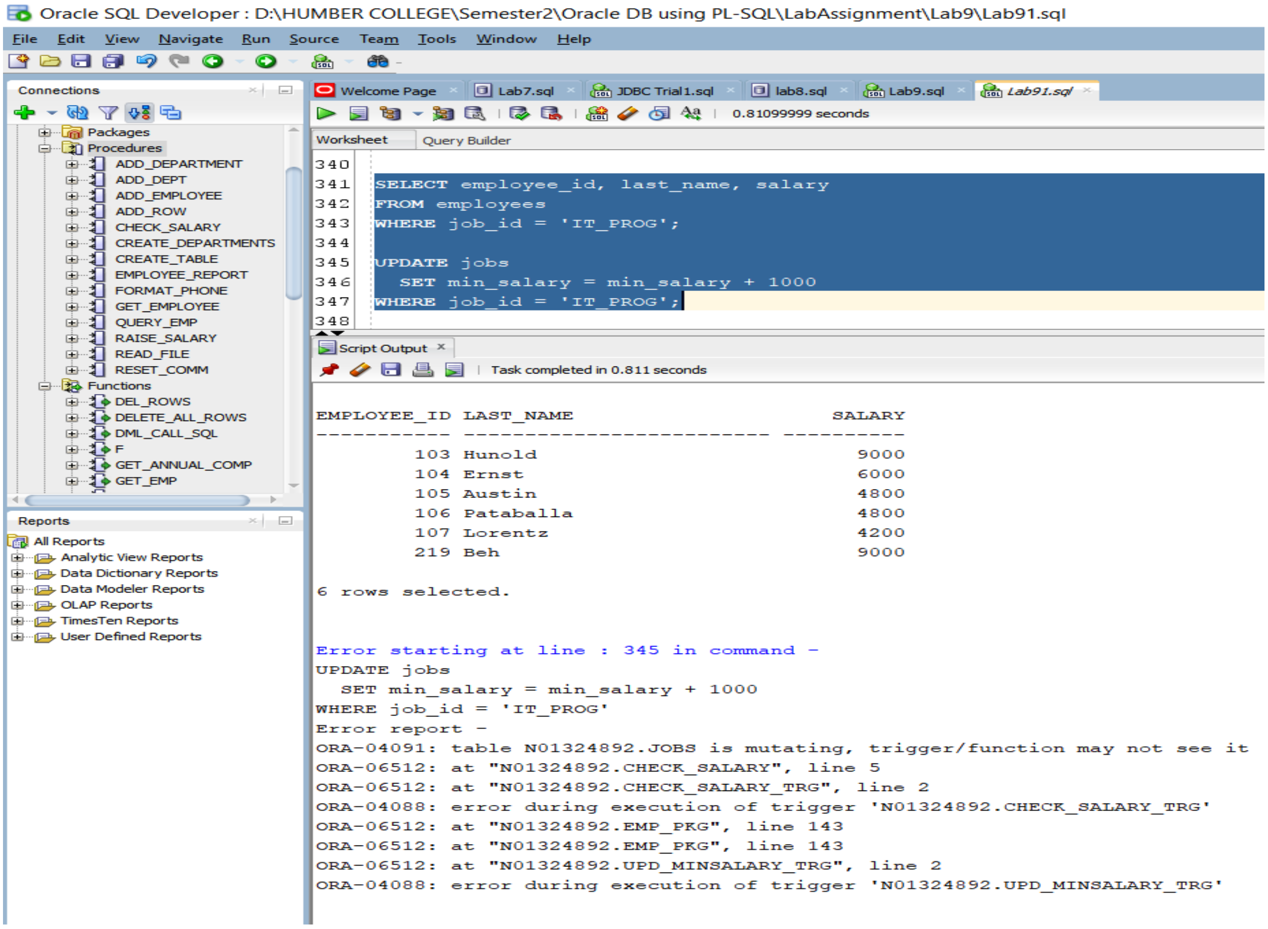
```
CREATE SEQUENCE log_newemp_seq START WITH 1 INCREMENT BY 1;

CREATE TABLE log_newemp (
    entry_id    NUMBER PRIMARY KEY,
    user_id     VARCHAR2(30),
    log_time    DATE,
    name        VARCHAR2(100)
);
```

```
Sequence LOG_NEWEMP_SEQ created.


Table LOG_NEWEMP created.
```



```
CREATE OR REPLACE TRIGGER upd_minsalary_trg
AFTER UPDATE OF min_salary ON JOBS
FOR EACH ROW
BEGIN
  emp_pkg.set_salary(:new.job_id, :new.min_salary);
END;
/
SHOW ERRORS
```

```
Trigger UPD_MINSALARY_TRG compiled
```

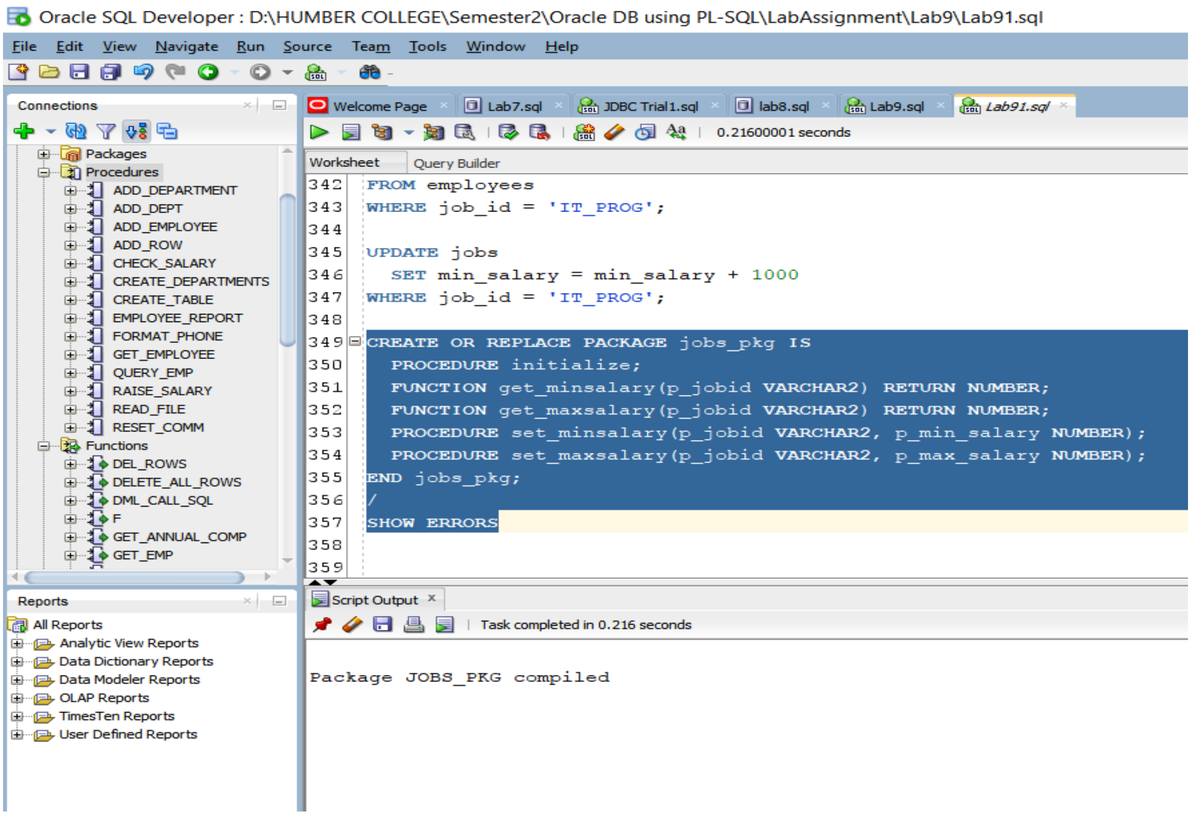


```
SELECT employee_id, last_name, salary
FROM employees
WHERE job_id = 'IT_PROG';

UPDATE jobs
  SET min_salary = min_salary + 1000
WHERE job_id = 'IT_PROG';
```

```
EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
        103 Hunold                          9000
        104 Ernst                           6000
        105 Austin                          4800
        106 Pataballa                       4800
        107 Lorentz                         4200
        219 Beh                             9000

6 rows selected.


Error starting at line : 345 in command -
UPDATE jobs
  SET min_salary = min_salary + 1000
WHERE job_id = 'IT_PROG'
Error report -
ORA-04091: table N01324892.JOBS is mutating, trigger/function may not see it
ORA-06512: at "N01324892.CHECK_SALARY", line 5
ORA-06512: at "N01324892.CHECK_SALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.CHECK_SALARY_TRG'
ORA-06512: at "N01324892.EMP_PKG", line 143
ORA-06512: at "N01324892.EMP_PKG", line 143
ORA-06512: at "N01324892.UPD_MINSALARY_TRG", line 2
ORA-04088: error during execution of trigger 'N01324892.UPD_MINSALARY_TRG'
```

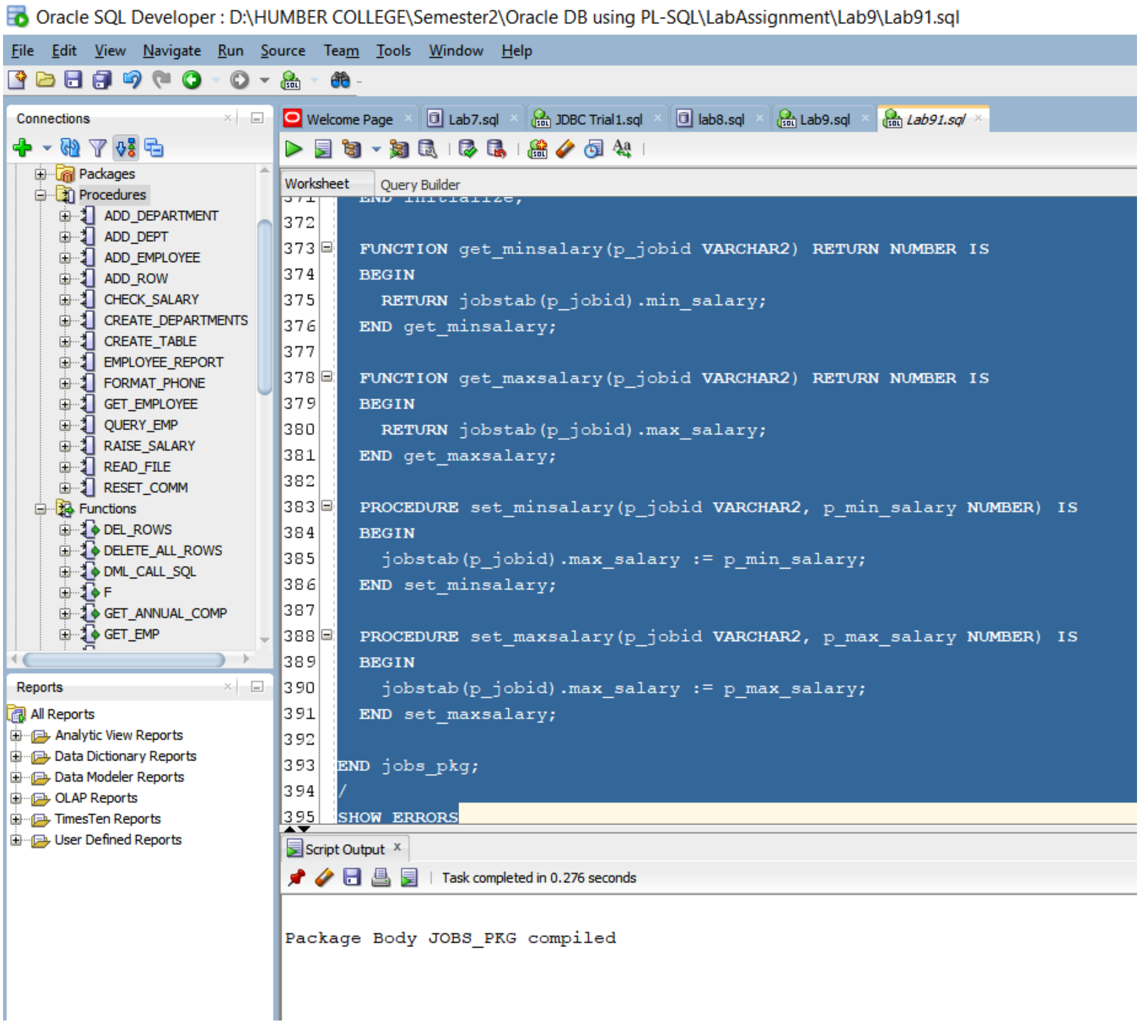
Practice – 2 – 2:

- We are creating jobs_pkg with procedure initialize and functions named get_minsalary and get_maxsalary with two other procedures set_minsalary and set_maxsalary. It gets complied successfully. Note that we are using getter and setter method to get the min and max salary and after that to set the min and max salary.
- Declaring a private PL/SQL index by table called jobs_tab_type that is indexed by a string type based on the jobs.job_id%type.the get_minsalary function uses a p_jobid and returns min_salary, max_salary.
- After that we are modifying the code by replacing the query on the jobs table with statements to set the local minsal and maxsal varribles with values from the jobs_pkg data by calling appropriate get_*salary functions.
- Now we will implement BEFORE INSERT OR UPDATE statement in trigger INIT_JOBPKG_TRG that uses the CALL syntax to invoke the JOBS_PKG.INITIALIZE procedure to ensure that the package state is current before the DML operations are performed.
- In next query employees with name Austin, Pataballa and Lorentz have had all their salaries updated. No exceptions occurred during this process and we have implemented a solution for the mutating table trigger exceptions.



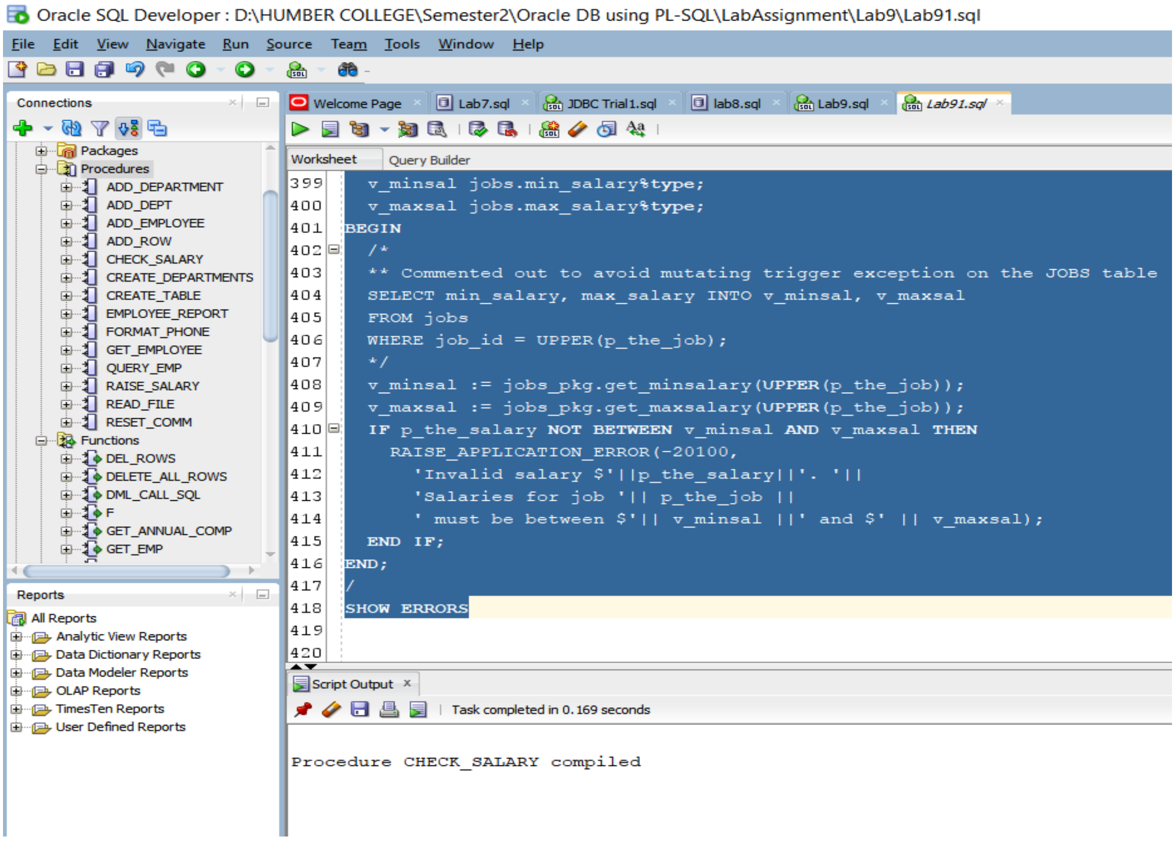
```
FROM employees
WHERE job_id = 'IT_PROG';

UPDATE jobs
  SET min_salary = min_salary + 1000
WHERE job_id = 'IT_PROG';

CREATE OR REPLACE PACKAGE jobs_pkg IS
  PROCEDURE initialize;
  FUNCTION get_minsalary(p_jobid VARCHAR2) RETURN NUMBER;
  FUNCTION get_maxsalary(p_jobid VARCHAR2) RETURN NUMBER;
  PROCEDURE set_minsalary(p_jobid VARCHAR2, p_min_salary NUMBER);
  PROCEDURE set_maxsalary(p_jobid VARCHAR2, p_max_salary NUMBER);
END jobs_pkg;
/
SHOW ERRORS
```

```
Package JOBS_PKG compiled
```



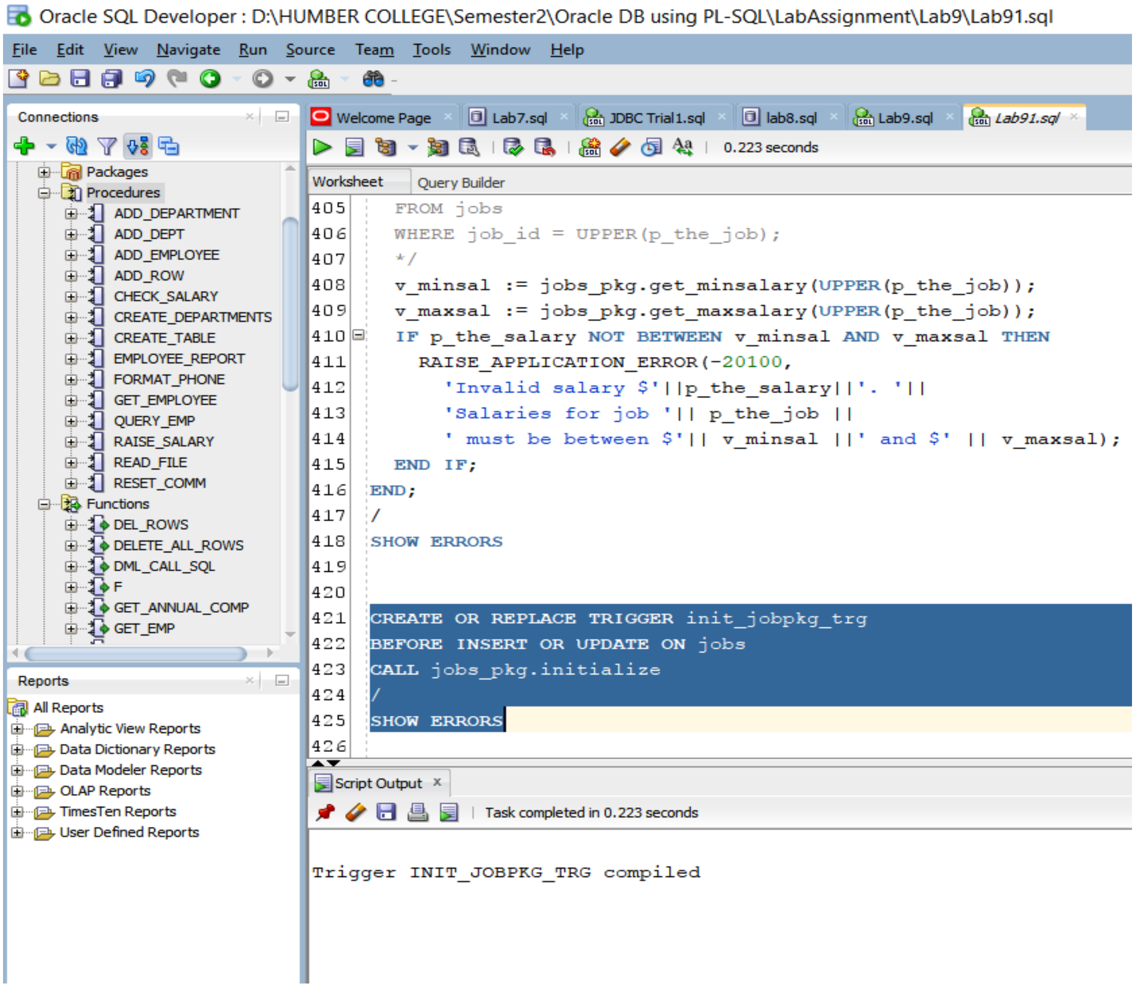
```
  END initialize;

  FUNCTION get_minsalary(p_jobid VARCHAR2) RETURN NUMBER IS
  BEGIN
    RETURN jobstab(p_jobid).min_salary;
  END get_minsalary;

  FUNCTION get_maxsalary(p_jobid VARCHAR2) RETURN NUMBER IS
  BEGIN
    RETURN jobstab(p_jobid).max_salary;
  END get_maxsalary;

  PROCEDURE set_minsalary(p_jobid VARCHAR2, p_min_salary NUMBER) IS
  BEGIN
    jobstab(p_jobid).max_salary := p_min_salary;
  END set_minsalary;

  PROCEDURE set_maxsalary(p_jobid VARCHAR2, p_max_salary NUMBER) IS
  BEGIN
    jobstab(p_jobid).max_salary := p_max_salary;
  END set_maxsalary;

END jobs_pkg;
/
SHOW ERRORS
```

```
Package Body JOBS_PKG compiled
```



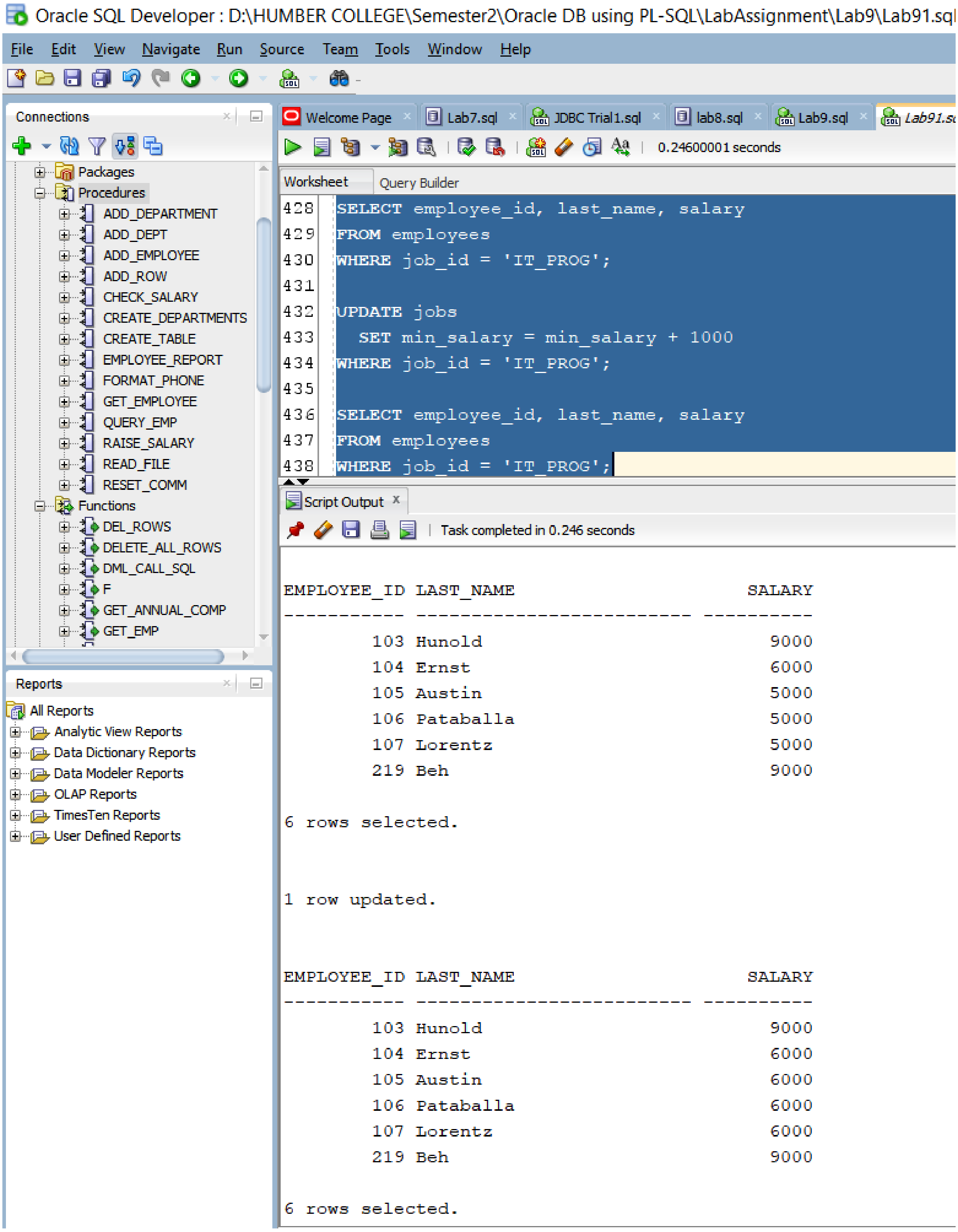
```
  v_minsal jobs.min_salary%type;
  v_maxsal jobs.max_salary%type;
BEGIN
  /*
  ** Commented out to avoid mutating trigger exception on the JOBS table
  SELECT min_salary, max_salary INTO v_minsal, v_maxsal
  FROM jobs
  WHERE job_id = UPPER(p_the_job);
  */
  v_minsal := jobs_pkg.get_minsalary(UPPER(p_the_job));
  v_maxsal := jobs_pkg.get_maxsalary(UPPER(p_the_job));
  IF p_the_salary NOT BETWEEN v_minsal AND v_maxsal THEN
    RAISE_APPLICATION_ERROR(-20100,
      'Invalid salary $'||p_the_salary||'. '||
      'Salaries for job '|| p_the_job ||
      ' must be between $'|| v_minsal ||' and $' || v_maxsal);
  END IF;
END;
/
SHOW ERRORS
```

```
Procedure CHECK_SALARY compiled
```



```
  FROM jobs
  WHERE job_id = UPPER(p_the_job);
  */
  v_minsal := jobs_pkg.get_minsalary(UPPER(p_the_job));
  v_maxsal := jobs_pkg.get_maxsalary(UPPER(p_the_job));
  IF p_the_salary NOT BETWEEN v_minsal AND v_maxsal THEN
    RAISE_APPLICATION_ERROR(-20100,
      'Invalid salary $'||p_the_salary||'. '||
      'Salaries for job '|| p_the_job ||
      ' must be between $'|| v_minsal ||' and $' || v_maxsal);
  END IF;
END;
/
SHOW ERRORS

CREATE OR REPLACE TRIGGER init_jobpkg_trg
BEFORE INSERT OR UPDATE ON jobs
CALL jobs_pkg.initialize
/
SHOW ERRORS
```

```
Trigger INIT_JOBPKG_TRG compiled
```

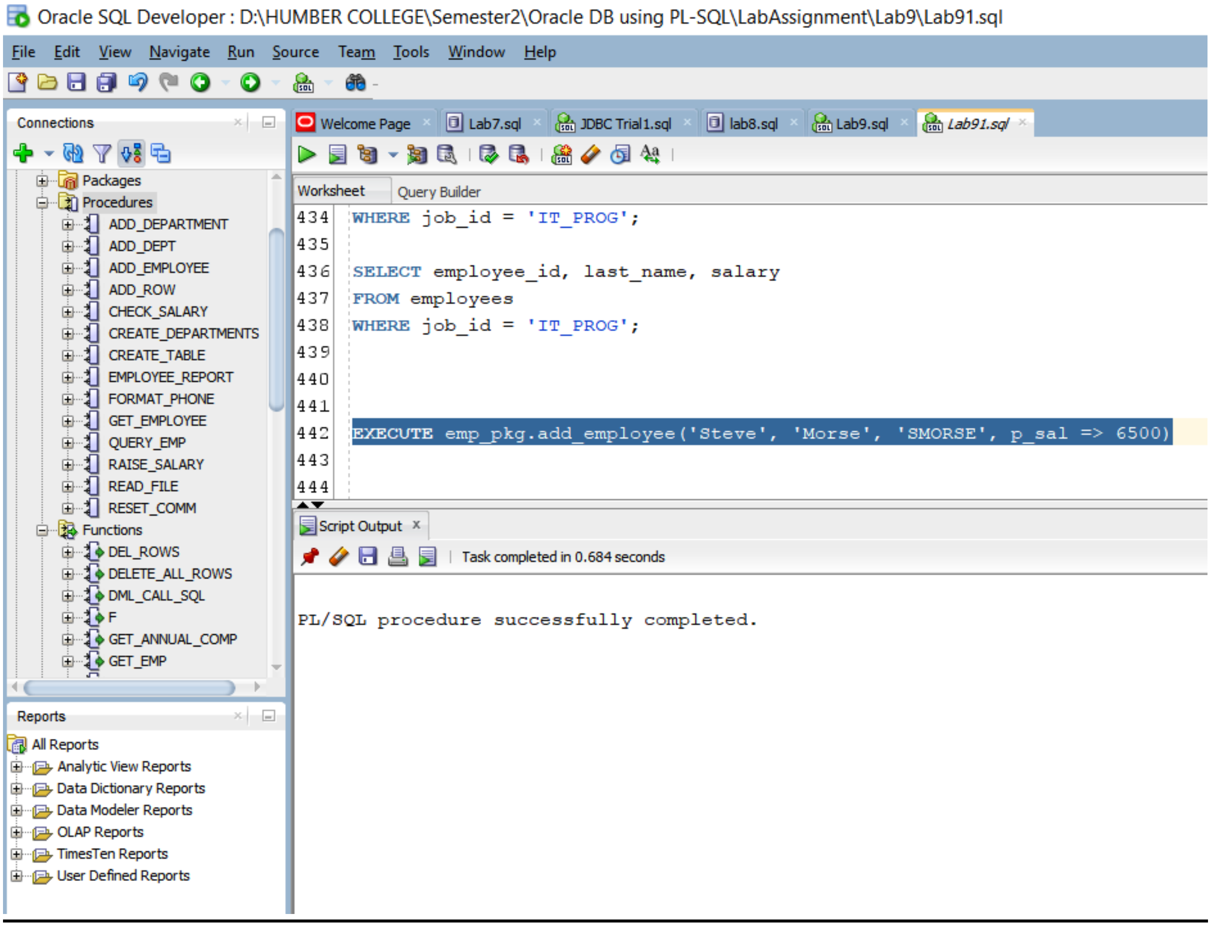


```
SELECT employee_id, last_name, salary
FROM employees
WHERE job_id = 'IT_PROG';

UPDATE jobs
  SET min_salary = min_salary + 1000
WHERE job_id = 'IT_PROG';

SELECT employee_id, last_name, salary
FROM employees
WHERE job_id = 'IT_PROG';
```

```
EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
        103 Hunold                          9000
        104 Ernst                           6000
        105 Austin                          5000
        106 Pataballa                       5000
        107 Lorentz                         5000
        219 Beh                             9000

6 rows selected.


1 row updated.


EMPLOYEE_ID LAST_NAME                     SALARY
----------- ------------------------- ----------
        103 Hunold                          9000
        104 Ernst                           6000
        105 Austin                          6000
        106 Pataballa                       6000
        107 Lorentz                         6000
        219 Beh                             9000

6 rows selected.
```

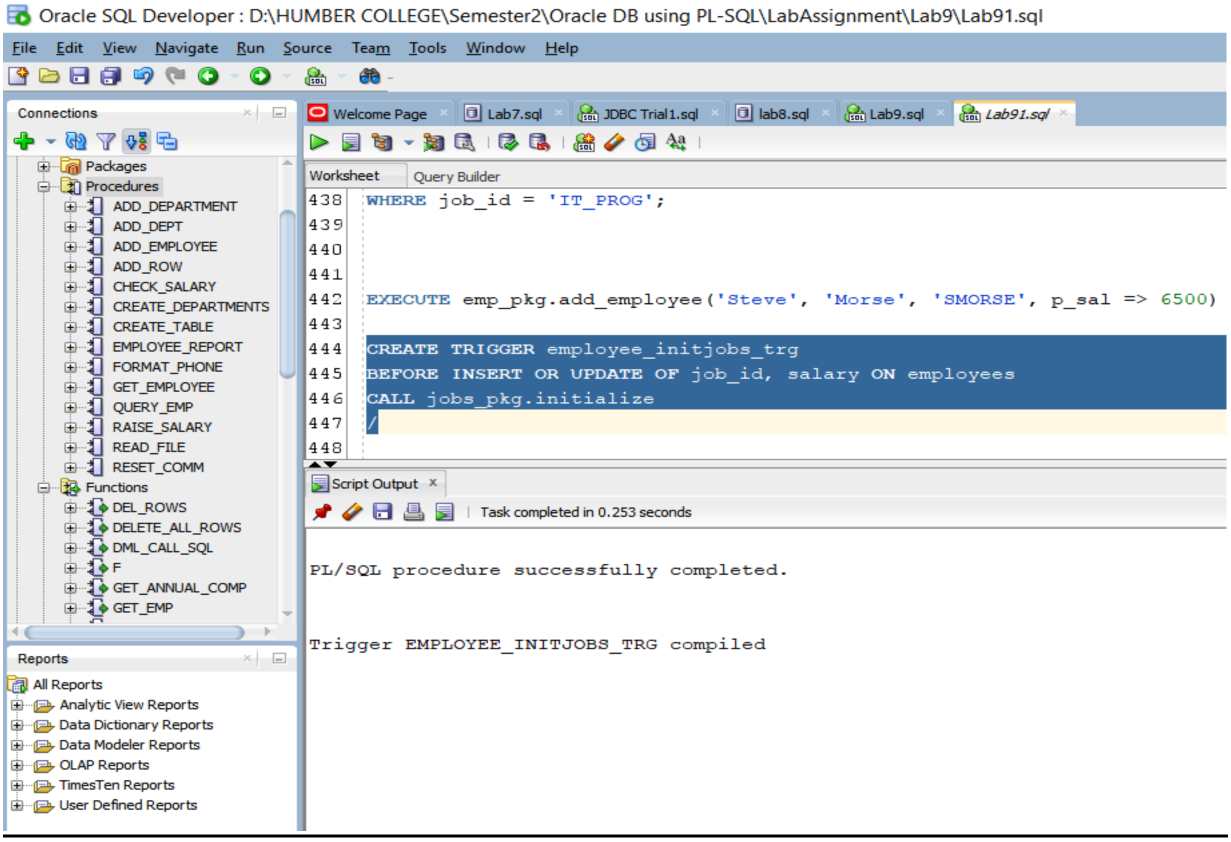
Practice - 2- 3:

- Now we will check whether this trigger works. To test this we will call emp_pkg.add_employee with parameters (Steave, Morse, Smorse, p_sal => 6500). It gets executed successfully.
- Now we are creating BEFORE INSERT OR UPDATE statement trigger called employee_initjobs_trg on the employees table that calls that procedure. We have also implemented CALL syntax in the trigger body.
- Now we are again adding this Steave Morse name and after calling select query we can fetch this but we can not add that name because it is already there.



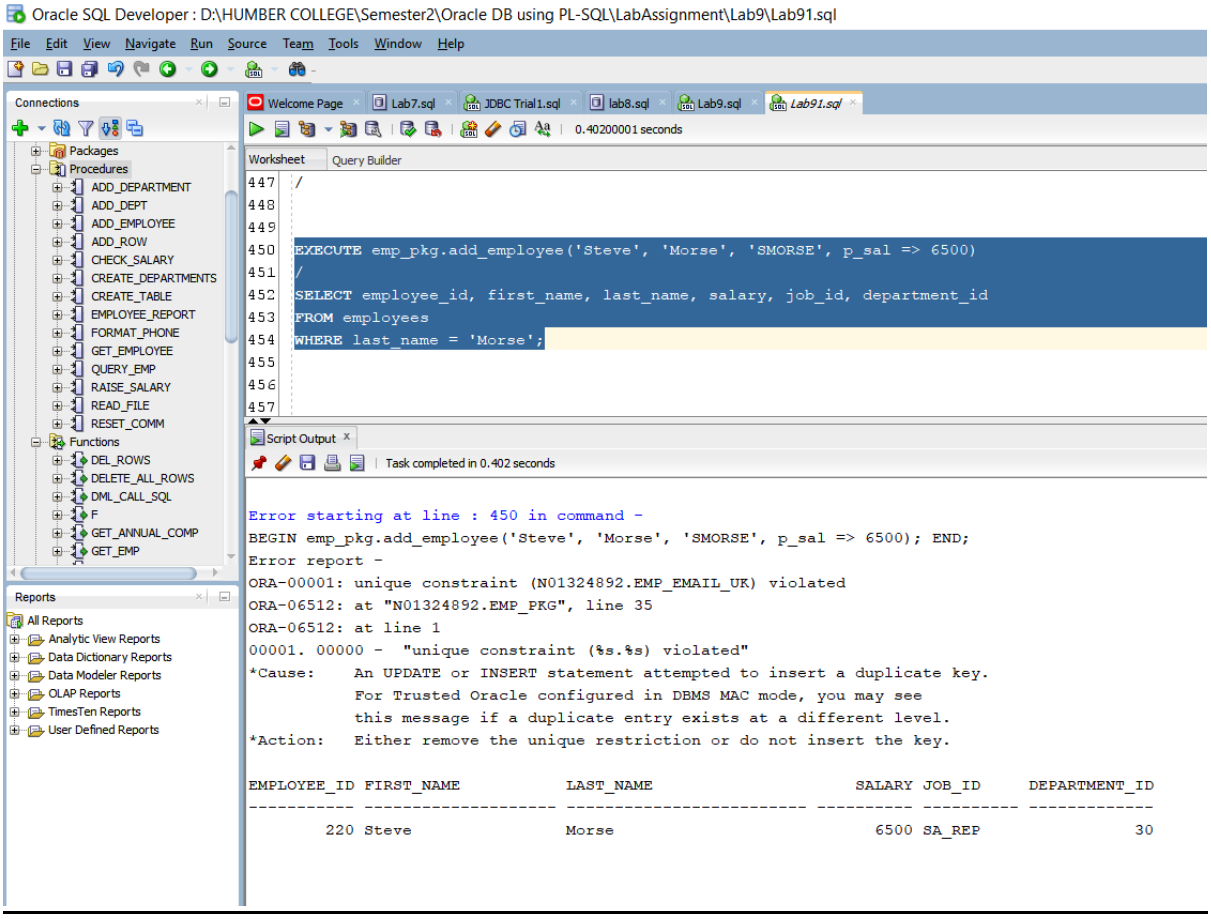
```
WHERE job_id = 'IT_PROG';

SELECT employee_id, last_name, salary
FROM employees
WHERE job_id = 'IT_PROG';



EXECUTE emp_pkg.add_employee('Steve', 'Morse', 'SMORSE', p_sal => 6500)
```

```
PL/SQL procedure successfully completed.
```



```
WHERE job_id = 'IT_PROG';


EXECUTE emp_pkg.add_employee('Steve', 'Morse', 'SMORSE', p_sal => 6500)

CREATE TRIGGER employee_initjobs_trg
BEFORE INSERT OR UPDATE OF job_id, salary ON employees
CALL jobs_pkg.initialize
/
```

```
PL/SQL procedure successfully completed.


Trigger EMPLOYEE_INITJOBS_TRG compiled
```



```
/


EXECUTE emp_pkg.add_employee('Steve', 'Morse', 'SMORSE', p_sal => 6500)
/
SELECT employee_id, first_name, last_name, salary, job_id, department_id
FROM employees
WHERE last_name = 'Morse';
```

```
Error starting at line : 450 in command -
BEGIN emp_pkg.add_employee('Steve', 'Morse', 'SMORSE', p_sal => 6500); END;
Error report -
ORA-00001: unique constraint (N01324892.EMP_EMAIL_UK) violated
ORA-06512: at "N01324892.EMP_PKG", line 35
ORA-06512: at line 1
00001. 00000 -  "unique constraint (%s.%s) violated"
*Cause:    An UPDATE or INSERT statement attempted to insert a duplicate key.
           For Trusted Oracle configured in DBMS MAC mode, you may see
           this message if a duplicate entry exists at a different level.
*Action:   Either remove the unique restriction or do not insert the key.

EMPLOYEE_ID FIRST_NAME           LAST_NAME                     SALARY JOB_ID     DEPARTMENT_ID
----------- -------------------- ------------------------- ---------- ---------- -------------
        220 Steve                Morse                           6500 SA_REP                30
```