National University
Of Computer and Emerging Sciences

Lab Task # 9

Name – Muhammad Taha

Roll NO – 23P-0559

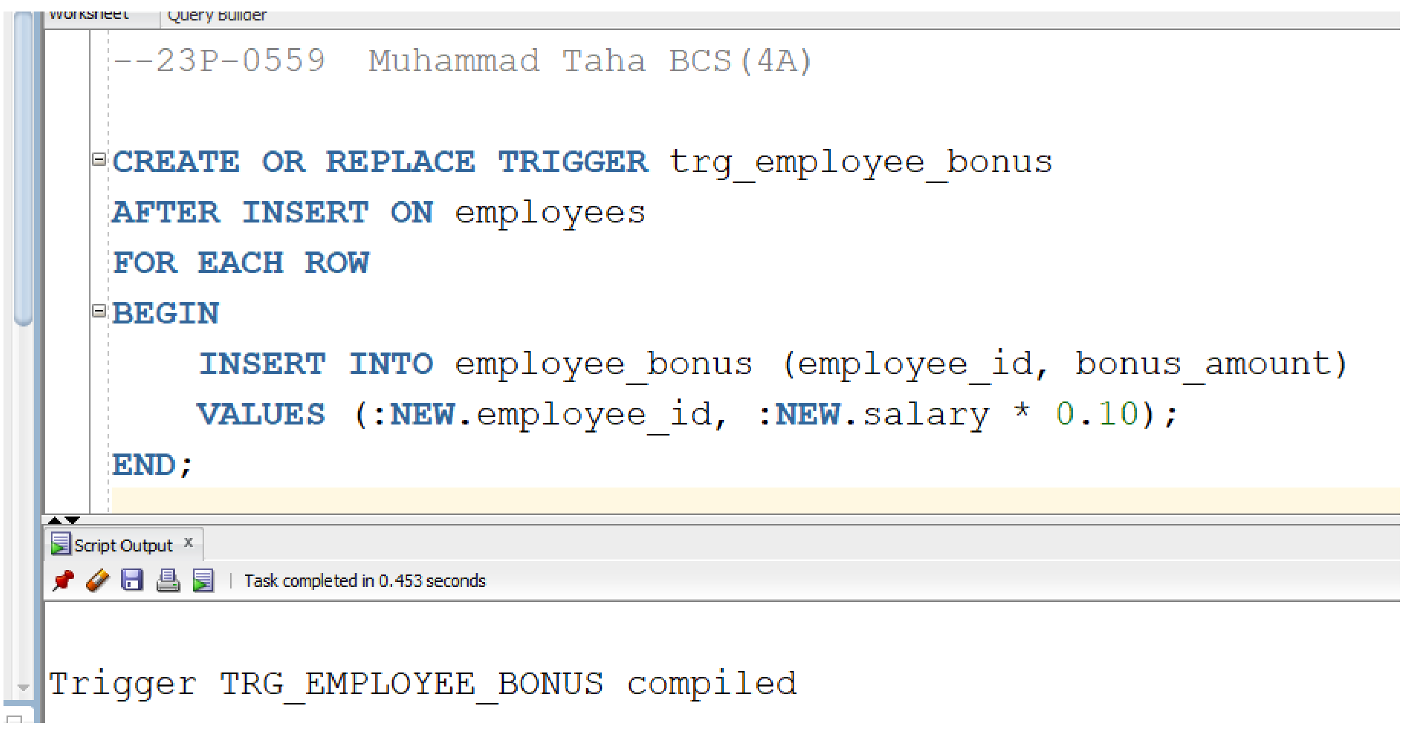
SECTION – BCS(4A)

Subject – Database Systems - LAB

DML Trigger Task

Create a trigger that automatically updates an employee's bonus table when a new record is added to the employees table. The bonus is set to 10% of the inserted salary. Create a table employee_bonus and populate it on each insert command.

```
CREATE OR REPLACE TRIGGER trg_employee_bonus
AFTER INSERT ON employees
FOR EACH ROW
BEGIN
  INSERT INTO employee_bonus (employee_id, bonus_amount)
  VALUES (:NEW.employee_id, :NEW.salary * 0.10);
END;
```

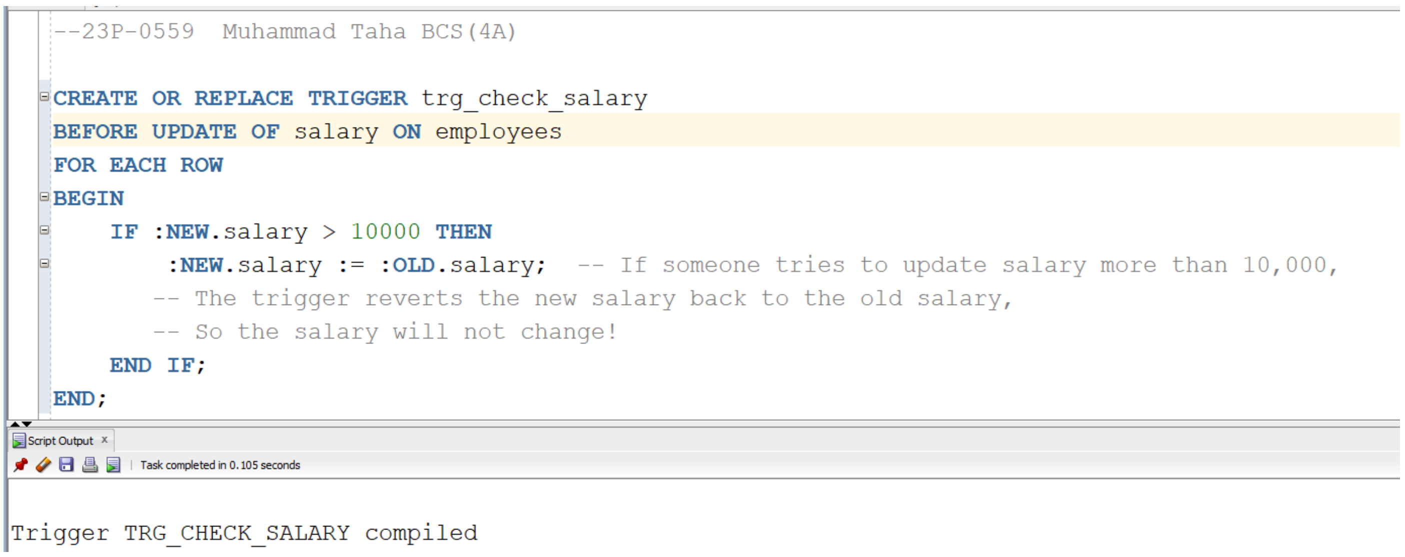


```
--23P-0559  Muhammad Taha BCS(4A)

CREATE OR REPLACE TRIGGER trg_employee_bonus
AFTER INSERT ON employees
FOR EACH ROW
BEGIN
    INSERT INTO employee_bonus (employee_id, bonus_amount)
    VALUES (:NEW.employee_id, :NEW.salary * 0.10);
END;
```

```
Trigger TRG_EMPLOYEE_BONUS compiled
```

Create a trigger that checks the new salary value being updated in the employees table. If the new salary is greater than a threshold (say 10,000), display an error message to the user.

```
CREATE OR REPLACE TRIGGER trg_check_salary
BEFORE UPDATE OF salary ON employees
FOR EACH ROW
BEGIN
   IF :NEW.salary > 10000 THEN
      :NEW.salary := :OLD.salary;  -- If someone tries to update salary more than 10,000,
     -- The trigger reverts the new salary back to the old salary,
     -- So the salary will not change!
   END IF;
END;
```



```
--23P-0559  Muhammad Taha BCS(4A)

CREATE OR REPLACE TRIGGER trg_check_salary
BEFORE UPDATE OF salary ON employees
FOR EACH ROW
BEGIN
    IF :NEW.salary > 10000 THEN
        :NEW.salary := :OLD.salary;   -- If someone tries to update salary more than 10,000,
       -- The trigger reverts the new salary back to the old salary,
       -- So the salary will not change!
    END IF;
END;
```

Script Output ×

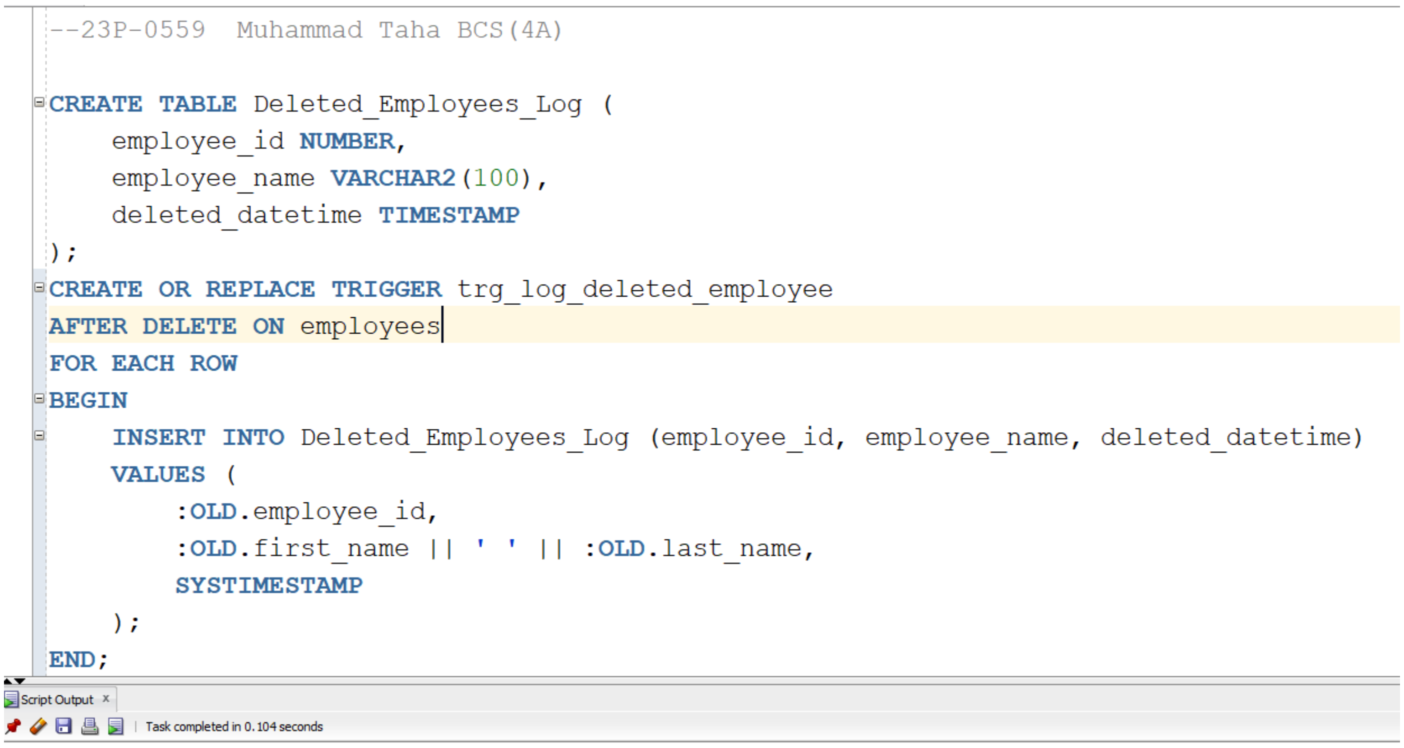
Task completed in 0.105 seconds

```
Trigger TRG_CHECK_SALARY compiled
```

Create a trigger that logs every deleted record from the Employees table into a Deleted_Employees_Log table.

```
CREATE TABLE Deleted_Employees_Log (
   employee_id NUMBER,
   employee_name VARCHAR2(100),
   deleted_datetime TIMESTAMP
);
CREATE OR REPLACE TRIGGER trg_log_deleted_employee
```

```
AFTER DELETE ON employees
FOR EACH ROW
BEGIN
    INSERT INTO Deleted_Employees_Log (employee_id, employee_name, deleted_datetime)
    VALUES (
        :OLD.employee_id,
        :OLD.first_name || ' ' || :OLD.last_name,
        SYSTIMESTAMP
    );
END;
```

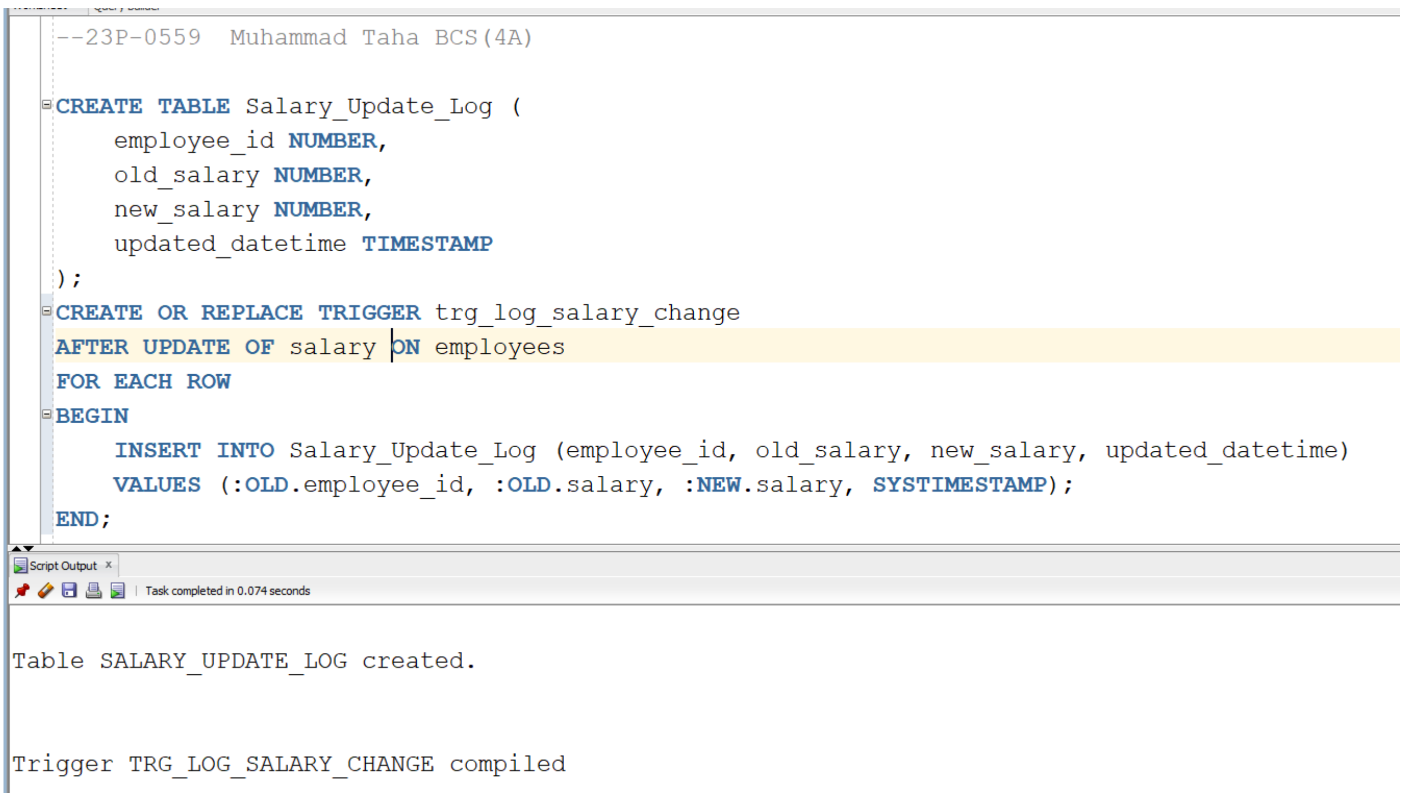


```
--23P-0559  Muhammad Taha BCS(4A)

CREATE TABLE Deleted_Employees_Log (
    employee_id NUMBER,
    employee_name VARCHAR2(100),
    deleted_datetime TIMESTAMP
);
CREATE OR REPLACE TRIGGER trg_log_deleted_employee
AFTER DELETE ON employees
FOR EACH ROW
BEGIN
    INSERT INTO Deleted_Employees_Log (employee_id, employee_name, deleted_datetime)
    VALUES (
        :OLD.employee_id,
        :OLD.first_name || ' ' || :OLD.last_name,
        SYSTIMESTAMP
    );
END;
```

```
Trigger TRG_LOG_DELETED_EMPLOYEE compiled
```

Create a trigger that logs the old and new values of a salary whenever an UPDATE occurs in the employees table.

```
CREATE TABLE Salary_Update_Log (
    employee_id NUMBER,
    old_salary NUMBER,
    new_salary NUMBER,
    updated_datetime TIMESTAMP
);
CREATE OR REPLACE TRIGGER trg_log_salary_change
AFTER UPDATE OF salary ON employees
FOR EACH ROW
BEGIN
    INSERT INTO Salary_Update_Log (employee_id, old_salary, new_salary, updated_datetime)
    VALUES (:OLD.employee_id, :OLD.salary, :NEW.salary, SYSTIMESTAMP);
END;
```



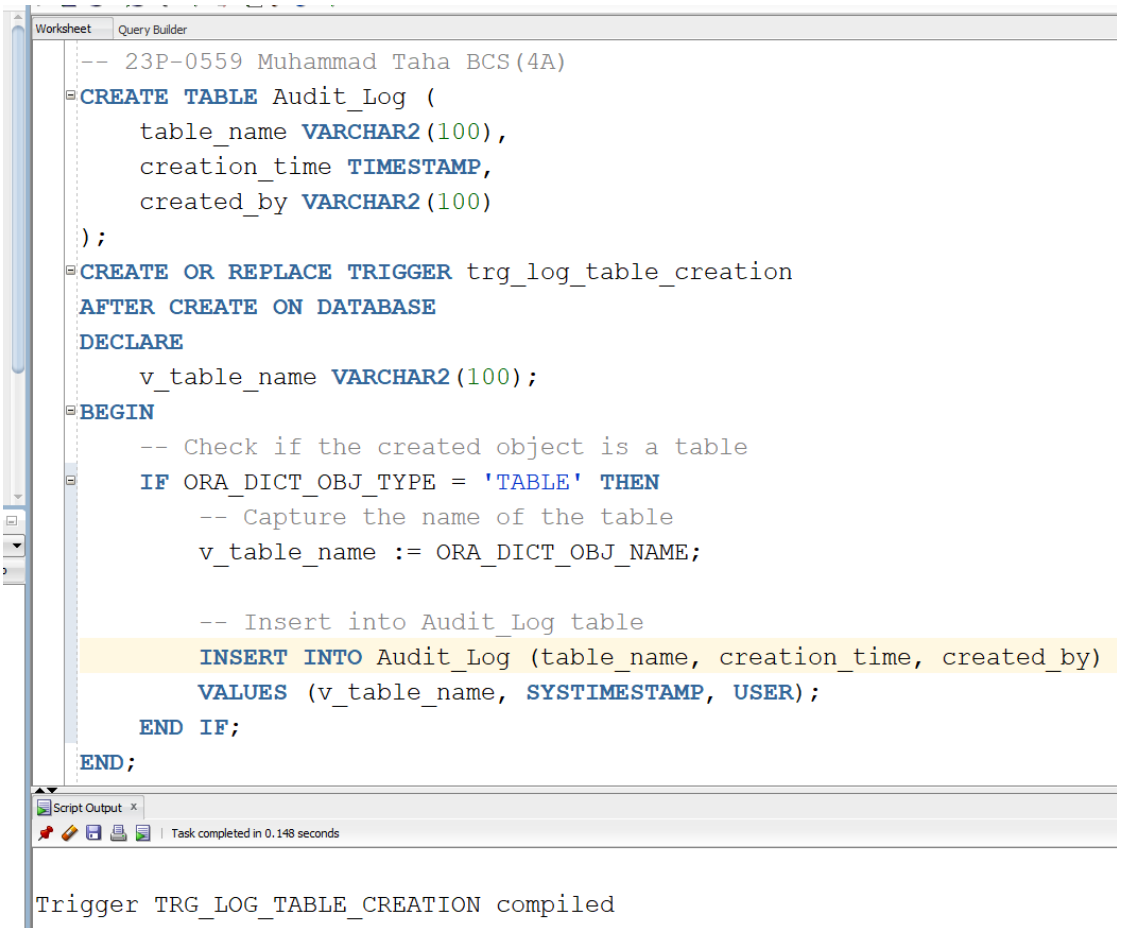
```
--23P-0559  Muhammad Taha BCS(4A)

CREATE TABLE Salary_Update_Log (
    employee_id NUMBER,
    old_salary NUMBER,
    new_salary NUMBER,
    updated_datetime TIMESTAMP
);
CREATE OR REPLACE TRIGGER trg_log_salary_change
AFTER UPDATE OF salary ON employees
FOR EACH ROW
BEGIN
    INSERT INTO Salary_Update_Log (employee_id, old_salary, new_salary, updated_datetime)
    VALUES (:OLD.employee_id, :OLD.salary, :NEW.salary, SYSTIMESTAMP);
END;
```

```
Table SALARY_UPDATE_LOG created.


Trigger TRG_LOG_SALARY_CHANGE compiled
```

DDL Trigger Tasks:

Create a trigger that logs every new table created in the database into an Audit_Log table, including the table name, creation time and user name.

```
-- 23P-0559 Muhammad Taha BCS(4A)
CREATE TABLE Audit_Log (
    table_name VARCHAR2(100),
    creation_time TIMESTAMP,
    created_by VARCHAR2(100)
);
CREATE OR REPLACE TRIGGER trg_log_table_creation
AFTER CREATE ON DATABASE
DECLARE
    v_table_name VARCHAR2(100);
BEGIN
    -- Check if the created object is a table
    IF ORA_DICT_OBJ_TYPE = 'TABLE' THEN
        -- Capture the name of the table
        v_table_name := ORA_DICT_OBJ_NAME;

        -- Insert into Audit_Log table
        INSERT INTO Audit_Log (table_name, creation_time, created_by)
        VALUES (v_table_name, SYSTIMESTAMP, USER);
    END IF;
END;
```


Worksheet Query Builder

```
-- 23P-0559 Muhammad Taha BCS(4A)
CREATE TABLE Audit_Log (
    table_name VARCHAR2(100),
    creation_time TIMESTAMP,
    created_by VARCHAR2(100)
);
CREATE OR REPLACE TRIGGER trg_log_table_creation
AFTER CREATE ON DATABASE
DECLARE
    v_table_name VARCHAR2(100);
BEGIN
    -- Check if the created object is a table
    IF ORA_DICT_OBJ_TYPE = 'TABLE' THEN
        -- Capture the name of the table
        v_table_name := ORA_DICT_OBJ_NAME;

        -- Insert into Audit_Log table
        INSERT INTO Audit_Log (table_name, creation_time, created_by)
        VALUES (v_table_name, SYSTIMESTAMP, USER);
    END IF;
END;
```

Script Output ×

Task completed in 0.148 seconds

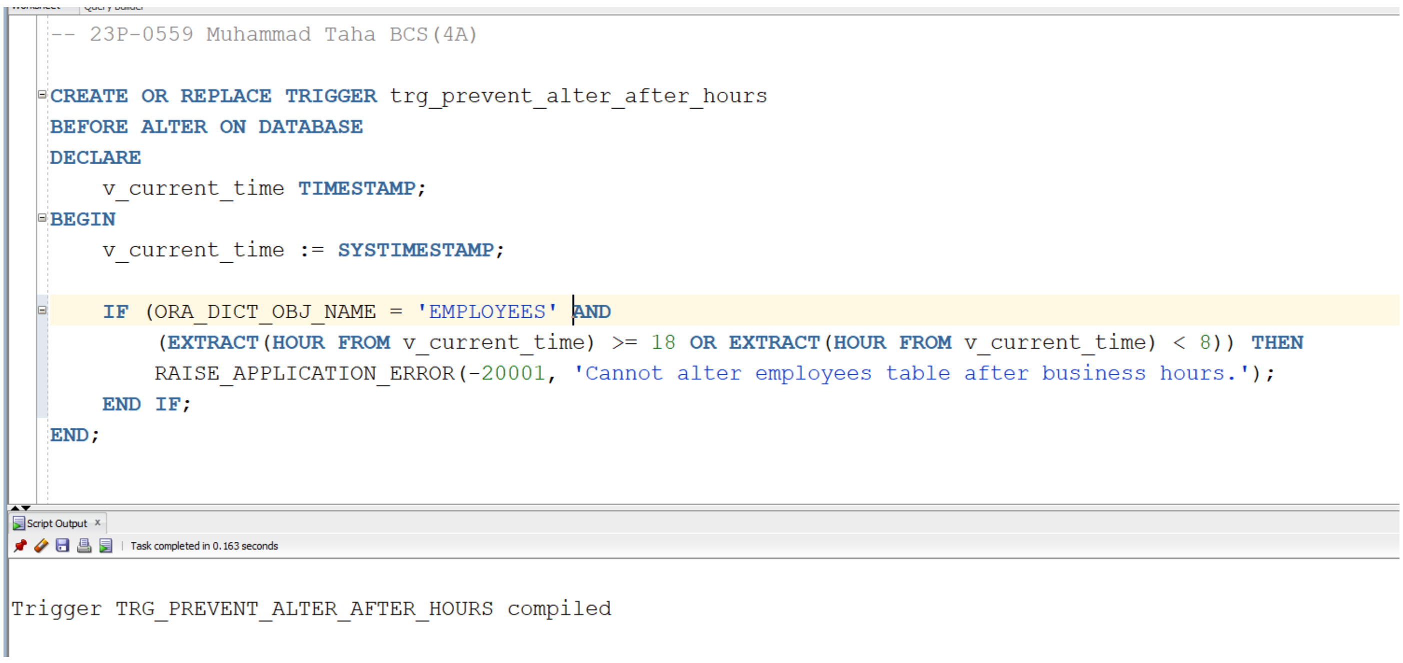
```
Trigger TRG_LOG_TABLE_CREATION compiled
```

Create a trigger that prevents changes (ALTER statements) to the employees table after business hours (e.g., 6 PM to 8 AM).

-- 23P-0559 Muhammad Taha BCS(4A)

```
CREATE OR REPLACE TRIGGER trg_prevent_alter_after_hours
BEFORE ALTER ON DATABASE
DECLARE
   v_current_time TIMESTAMP;
BEGIN
   v_current_time := SYSTIMESTAMP;

   IF (ORA_DICT_OBJ_NAME = 'EMPLOYEES' AND
       (EXTRACT(HOUR FROM v_current_time) >= 18 OR EXTRACT(HOUR FROM v_current_time) <
8)) THEN
       RAISE_APPLICATION_ERROR(-20001, 'Cannot alter employees table after business hours.');
   END IF;
```

END;

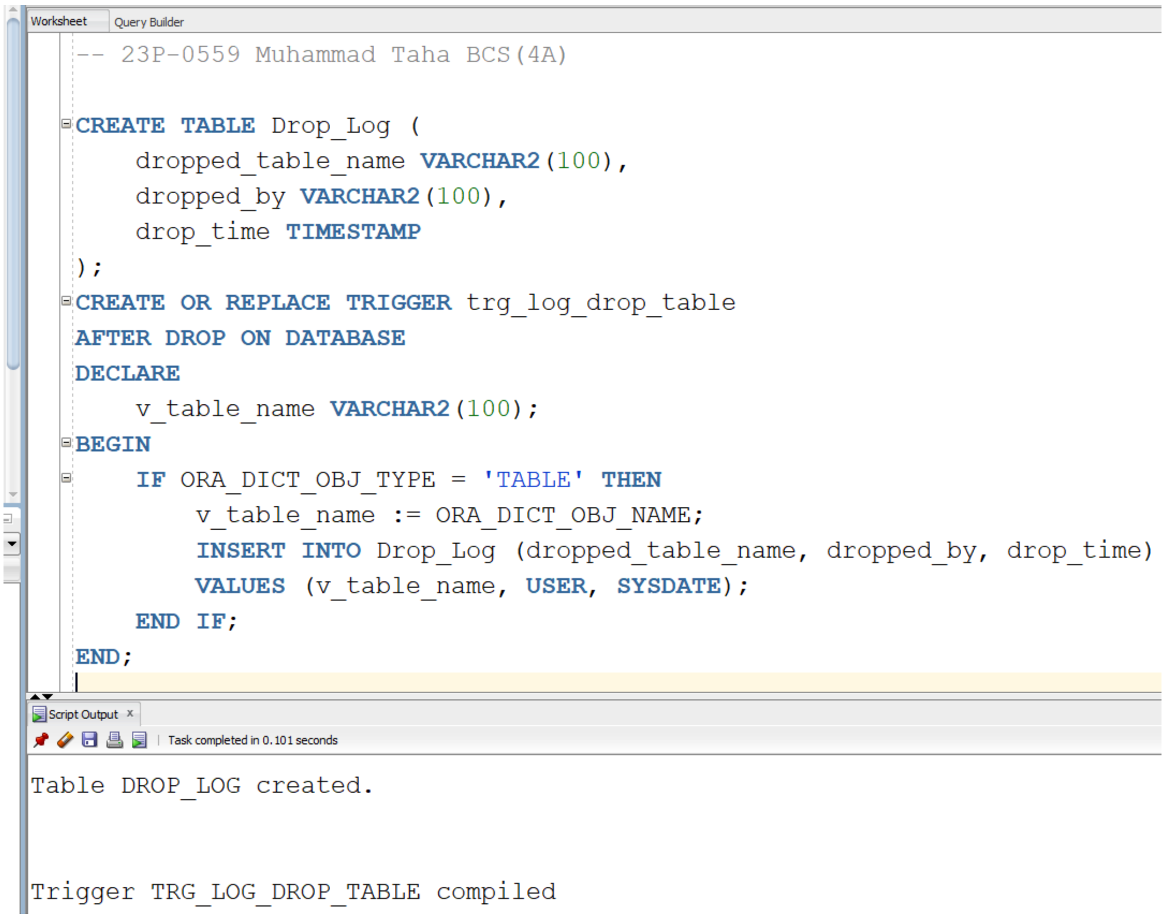


```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE TRIGGER trg_prevent_alter_after_hours
BEFORE ALTER ON DATABASE
DECLARE
    v_current_time TIMESTAMP;
BEGIN
    v_current_time := SYSTIMESTAMP;

    IF (ORA_DICT_OBJ_NAME = 'EMPLOYEES' AND
        (EXTRACT(HOUR FROM v_current_time) >= 18 OR EXTRACT(HOUR FROM v_current_time) < 8)) THEN
        RAISE_APPLICATION_ERROR(-20001, 'Cannot alter employees table after business hours.');
    END IF;
END;
```

```
Trigger TRG_PREVENT_ALTER_AFTER_HOURS compiled
```

Create a trigger that logs every DROP operation on any table in the database to a Drop_Log table, recording the user who performed the action and the time it occurred.

```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE TABLE Drop_Log (
    dropped_table_name VARCHAR2(100),
    dropped_by VARCHAR2(100),
    drop_time TIMESTAMP
);
CREATE OR REPLACE TRIGGER trg_log_drop_table
AFTER DROP ON DATABASE
DECLARE
    v_table_name VARCHAR2(100);
BEGIN
    IF ORA_DICT_OBJ_TYPE = 'TABLE' THEN
        v_table_name := ORA_DICT_OBJ_NAME;
        INSERT INTO Drop_Log (dropped_table_name, dropped_by, drop_time)
        VALUES (v_table_name, USER, SYSDATE);
    END IF;
END;
```

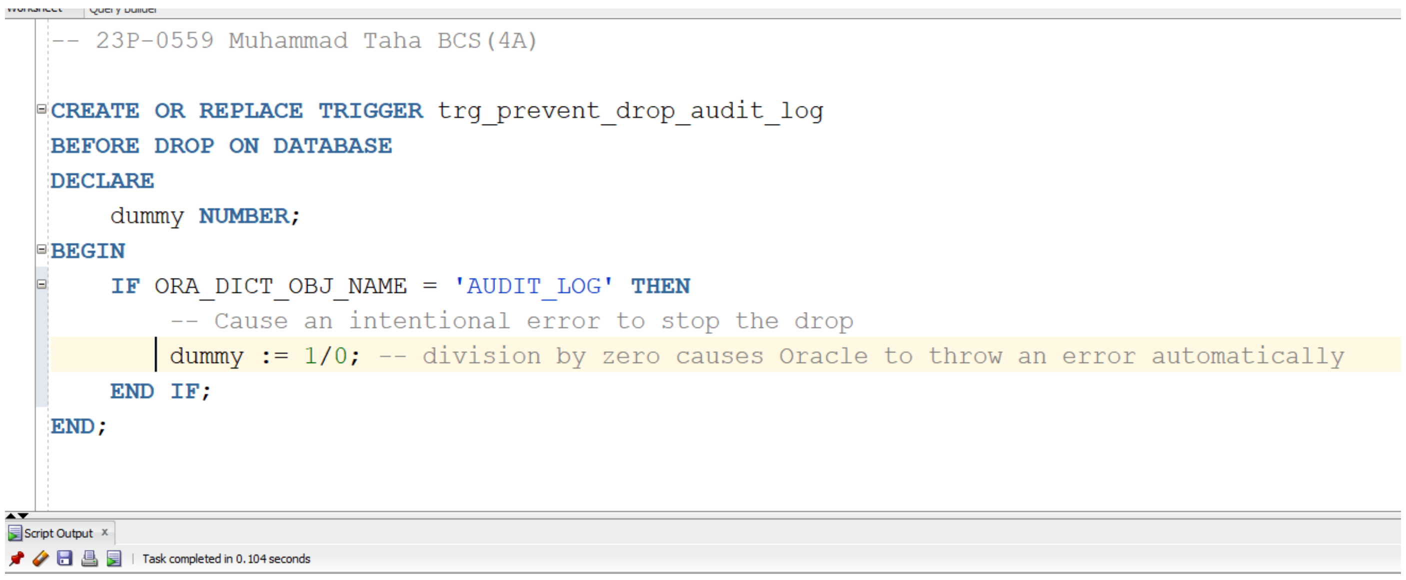


```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE TABLE Drop_Log (
    dropped_table_name VARCHAR2(100),
    dropped_by VARCHAR2(100),
    drop_time TIMESTAMP
);
CREATE OR REPLACE TRIGGER trg_log_drop_table
AFTER DROP ON DATABASE
DECLARE
    v_table_name VARCHAR2(100);
BEGIN
    IF ORA_DICT_OBJ_TYPE = 'TABLE' THEN
        v_table_name := ORA_DICT_OBJ_NAME;
        INSERT INTO Drop_Log (dropped_table_name, dropped_by, drop_time)
        VALUES (v_table_name, USER, SYSDATE);
    END IF;
END;
```

```
Table DROP_LOG created.


Trigger TRG_LOG_DROP_TABLE compiled
```

Create a trigger that prevents dropping the Audit_Log table under any circumstance and display a warning message instead.

```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE TRIGGER trg_prevent_drop_audit_log
BEFORE DROP ON DATABASE
DECLARE
    dummy NUMBER;
BEGIN
    IF ORA_DICT_OBJ_NAME = 'AUDIT_LOG' THEN
        -- Cause an intentional error to stop the drop
        dummy := 1/0; -- division by zero causes Oracle to throw an error automatically
    END IF;
END;
```



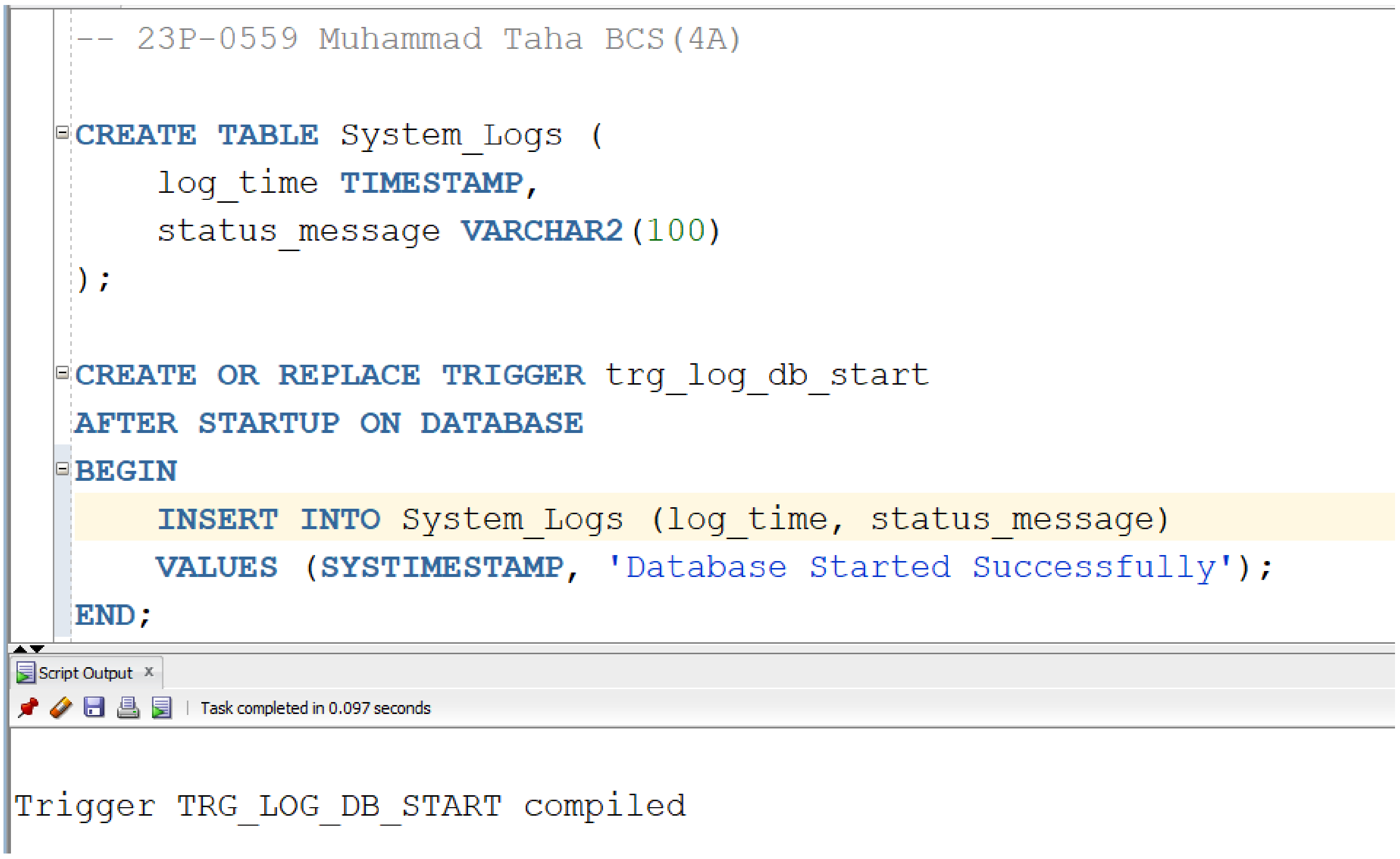
```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE TRIGGER trg_prevent_drop_audit_log
BEFORE DROP ON DATABASE
DECLARE
    dummy NUMBER;
BEGIN
    IF ORA_DICT_OBJ_NAME = 'AUDIT_LOG' THEN
        -- Cause an intentional error to stop the drop
        dummy := 1/0; -- division by zero causes Oracle to throw an error automatically
    END IF;
END;
```

```
Trigger TRG_PREVENT_DROP_AUDIT_LOG compiled
```

System/Database Trigger Task:

Create a trigger that logs the time and status when the database starts into a System_Logs table.

```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE TABLE System_Logs (
    log_time TIMESTAMP,
    status_message VARCHAR2(100)
);

CREATE OR REPLACE TRIGGER trg_log_db_start
AFTER STARTUP ON DATABASE
BEGIN
    INSERT INTO System_Logs (log_time, status_message)
    VALUES (SYSTIMESTAMP, 'Database Started Successfully');
END;
/
```

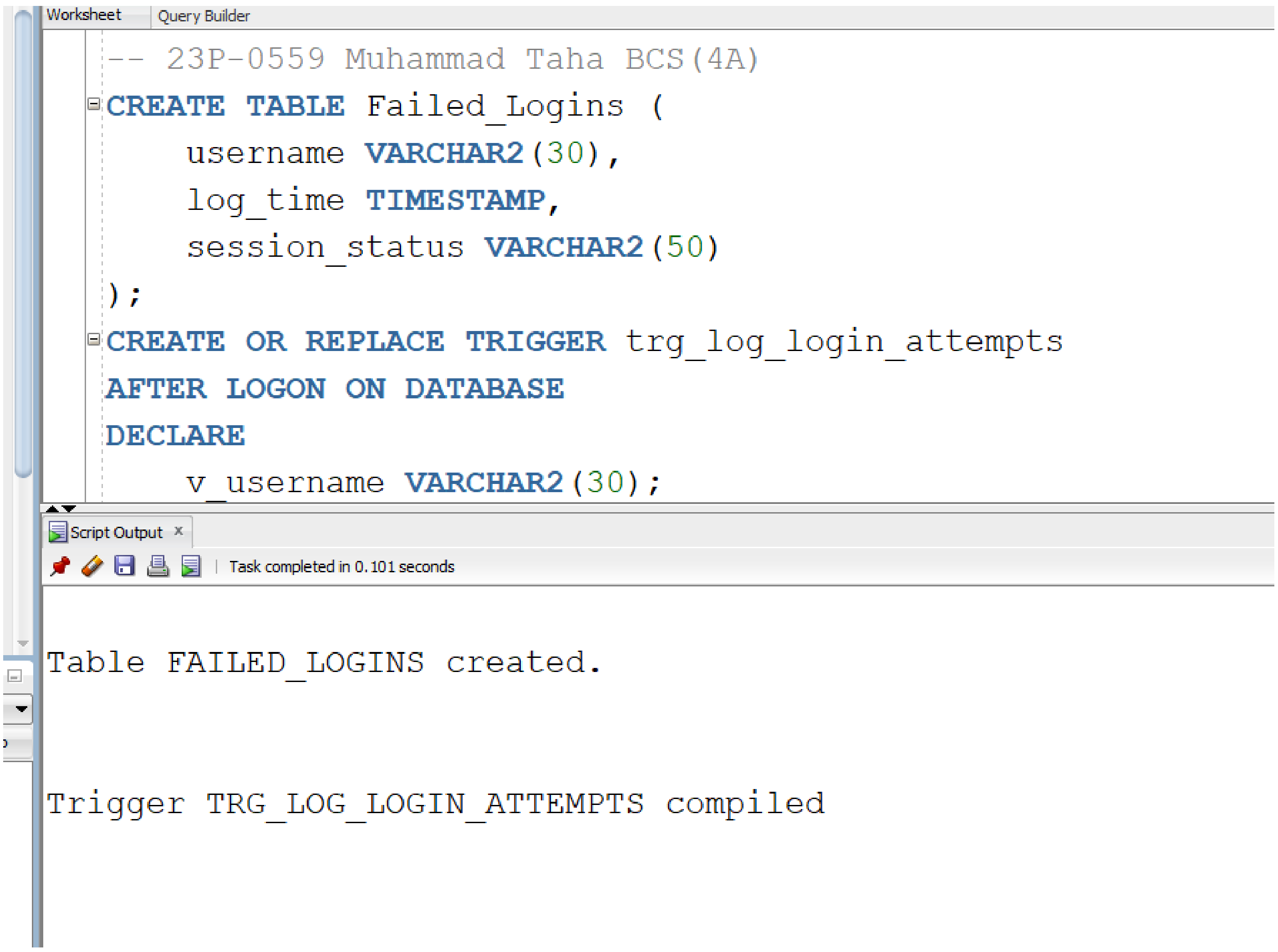


```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE TABLE System_Logs (
    log_time TIMESTAMP,
    status_message VARCHAR2(100)
);

CREATE OR REPLACE TRIGGER trg_log_db_start
AFTER STARTUP ON DATABASE
BEGIN
    INSERT INTO System_Logs (log_time, status_message)
    VALUES (SYSTIMESTAMP, 'Database Started Successfully');
END;
```

```
Trigger TRG_LOG_DB_START compiled
```

Create a trigger that tracks the login attempts of users and logs unsuccessful attempts into a Failed_Logins table.

```
-- 23P-0559 Muhammad Taha BCS(4A)
CREATE TABLE Failed_Logins (
    username VARCHAR2(30),
    log_time TIMESTAMP,
    session_status VARCHAR2(50)
);
CREATE OR REPLACE TRIGGER trg_log_login_attempts
AFTER LOGON ON DATABASE
DECLARE
    v_username VARCHAR2(30);
BEGIN
    v_username := SYS_CONTEXT('USERENV', 'SESSION_USER');

    IF v_username = 'UNKNOWN' THEN
        -- It's a failed login (username could not be resolved)
        INSERT INTO Failed_Logins (username, log_time, session_status)
        VALUES ('UNKNOWN', SYSTIMESTAMP, 'Failed Login');
    ELSE
        -- It's a successful login
        NULL; -- or you can log successful logins elsewhere if you want
    END IF;
END;
/
```



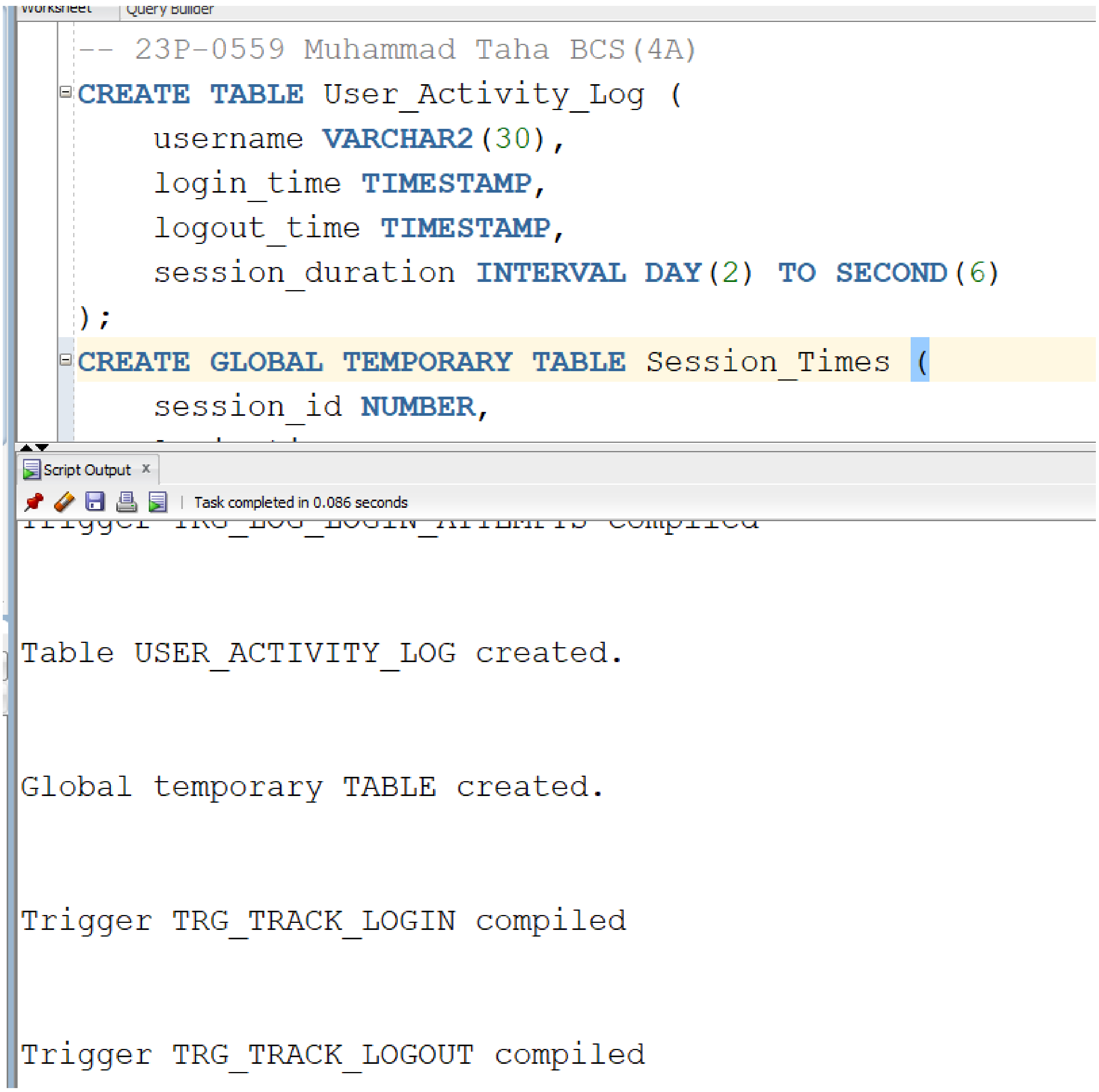
```
-- 23P-0559 Muhammad Taha BCS(4A)
CREATE TABLE Failed_Logins (
    username VARCHAR2(30),
    log_time TIMESTAMP,
    session_status VARCHAR2(50)
);
CREATE OR REPLACE TRIGGER trg_log_login_attempts
AFTER LOGON ON DATABASE
DECLARE
    v_username VARCHAR2(30);
```

```
Table FAILED_LOGINS created.


Trigger TRG_LOG_LOGIN_ATTEMPTS compiled
```

Create a trigger that logs every successful logout along with the session duration into a User_Activity_Log table.

```
-- 23P-0559 Muhammad Taha BCS(4A)
CREATE TABLE User_Activity_Log (
   username VARCHAR2(30),
   login_time TIMESTAMP,
   logout_time TIMESTAMP,
   session_duration INTERVAL DAY(2) TO SECOND(6)
);
CREATE GLOBAL TEMPORARY TABLE Session_Times (
   session_id NUMBER,
   login_time TIMESTAMP
) ON COMMIT PRESERVE ROWS;

CREATE OR REPLACE TRIGGER trg_track_login
AFTER LOGON ON DATABASE
BEGIN
   INSERT INTO Session_Times (session_id, login_time)
   VALUES (SYS_CONTEXT('USERENV', 'SESSIONID'), SYSTIMESTAMP);
END;
/
```

```
CREATE OR REPLACE TRIGGER trg_track_logout
BEFORE LOGOFF ON DATABASE
DECLARE
   v_login_time TIMESTAMP;
   v_session_duration INTERVAL DAY(2) TO SECOND(6);
BEGIN
   SELECT login_time INTO v_login_time
   FROM Session_Times
   WHERE session_id = SYS_CONTEXT('USERENV', 'SESSIONID');

   v_session_duration := SYSTIMESTAMP - v_login_time;

   INSERT INTO User_Activity_Log (username, login_time, logout_time, session_duration)
   VALUES (
      SYS_CONTEXT('USERENV', 'SESSION_USER'),
      v_login_time,
      SYSTIMESTAMP,
      v_session_duration
   );
END;
/
```



```
-- 23P-0559 Muhammad Taha BCS(4A)
CREATE TABLE User_Activity_Log (
    username VARCHAR2(30),
    login_time TIMESTAMP,
    logout_time TIMESTAMP,
    session_duration INTERVAL DAY(2) TO SECOND(6)
);
CREATE GLOBAL TEMPORARY TABLE Session_Times (
    session_id NUMBER,
```

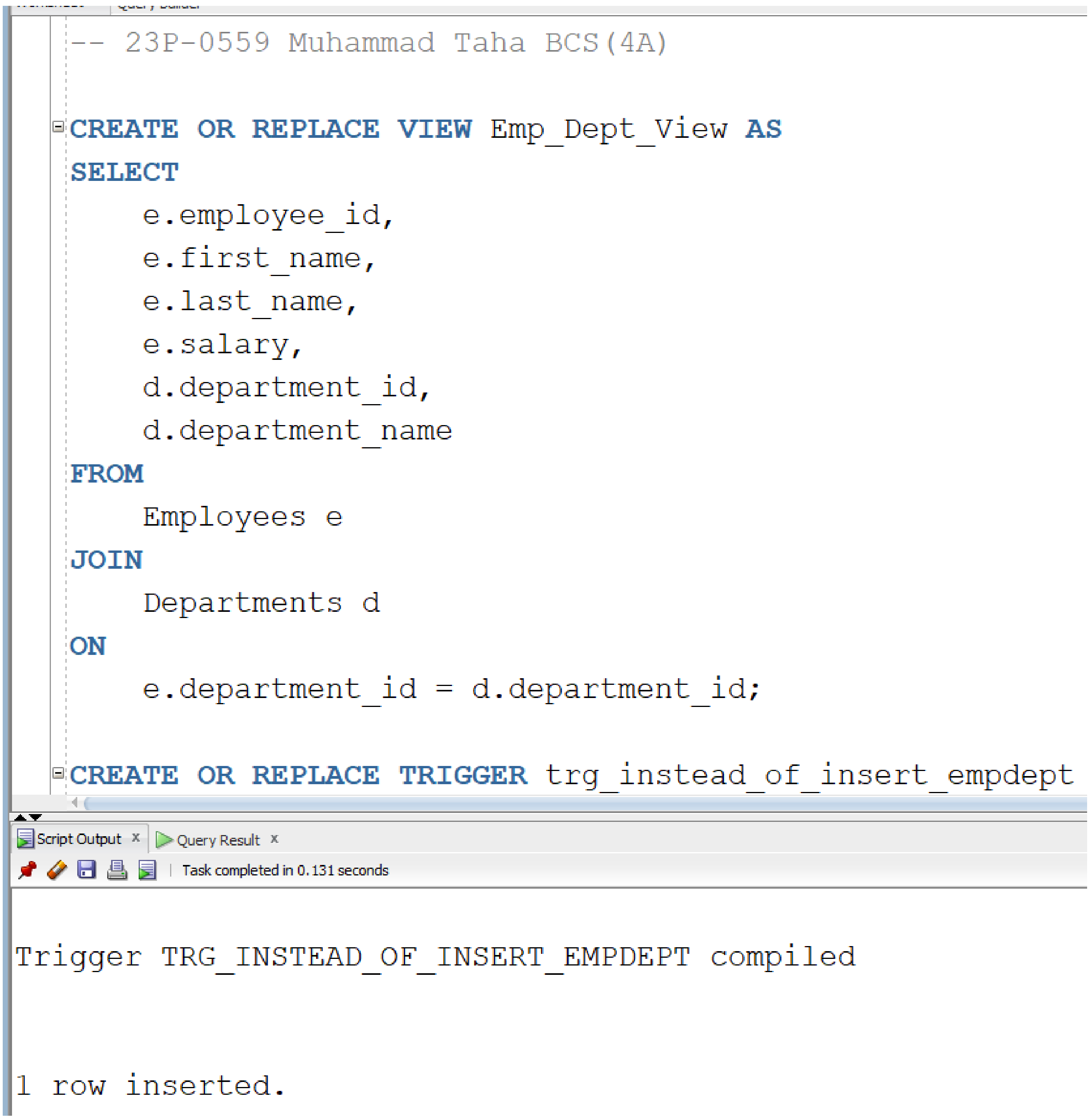
```
Trigger TRG_LOG_LOGIN_ATTEMPTS compiled


Table USER_ACTIVITY_LOG created.


Global temporary TABLE created.


Trigger TRG_TRACK_LOGIN compiled


Trigger TRG_TRACK_LOGOUT compiled
```

Instead of Trigger Task:

Create a view that joins Employees and Departments, and write an INSTEAD OF INSERT trigger that correctly distributes new data into both the Employees and Departments tables.

```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE VIEW Emp_Dept_View AS
SELECT
   e.employee_id,
   e.first_name,
   e.last_name,
   e.salary,
   d.department_id,
   d.department_name
FROM
   Employees e
JOIN
   Departments d
ON
   e.department_id = d.department_id;

CREATE OR REPLACE TRIGGER trg_instead_of_insert_empdept
INSTEAD OF INSERT ON Emp_Dept_View
FOR EACH ROW
DECLARE
   v_count NUMBER;
BEGIN
   -- Check if the department already exists in the Departments table
   SELECT COUNT(*) INTO v_count
   FROM Departments
   WHERE department_id = :NEW.department_id;

   -- If the department doesn't exist, insert it
   IF v_count = 0 THEN
      INSERT INTO Departments (department_id, department_name)
      VALUES (:NEW.department_id, :NEW.department_name);
   END IF;

   -- Insert into the Employees table with default values for missing columns
   INSERT INTO Employees (employee_id, first_name, last_name, salary, department_id, email, hire_date,
job_id, commission_pct, manager_id)
   VALUES (:NEW.employee_id, :NEW.first_name, :NEW.last_name, :NEW.salary, :NEW.department_id,
         'DEFAULT_EMAIL@example.com', SYSDATE, 'IT_PROG', 0, NULL);  -- Using 'IT_PROG' as
default JOB_ID
END;
/
```

INSERT INTO Emp_Dept_View (employee_id, first_name, last_name, salary, department_id, department_name)
VALUES (1901, 'Muhammad', 'Taha', 90000, 60, 'IT');



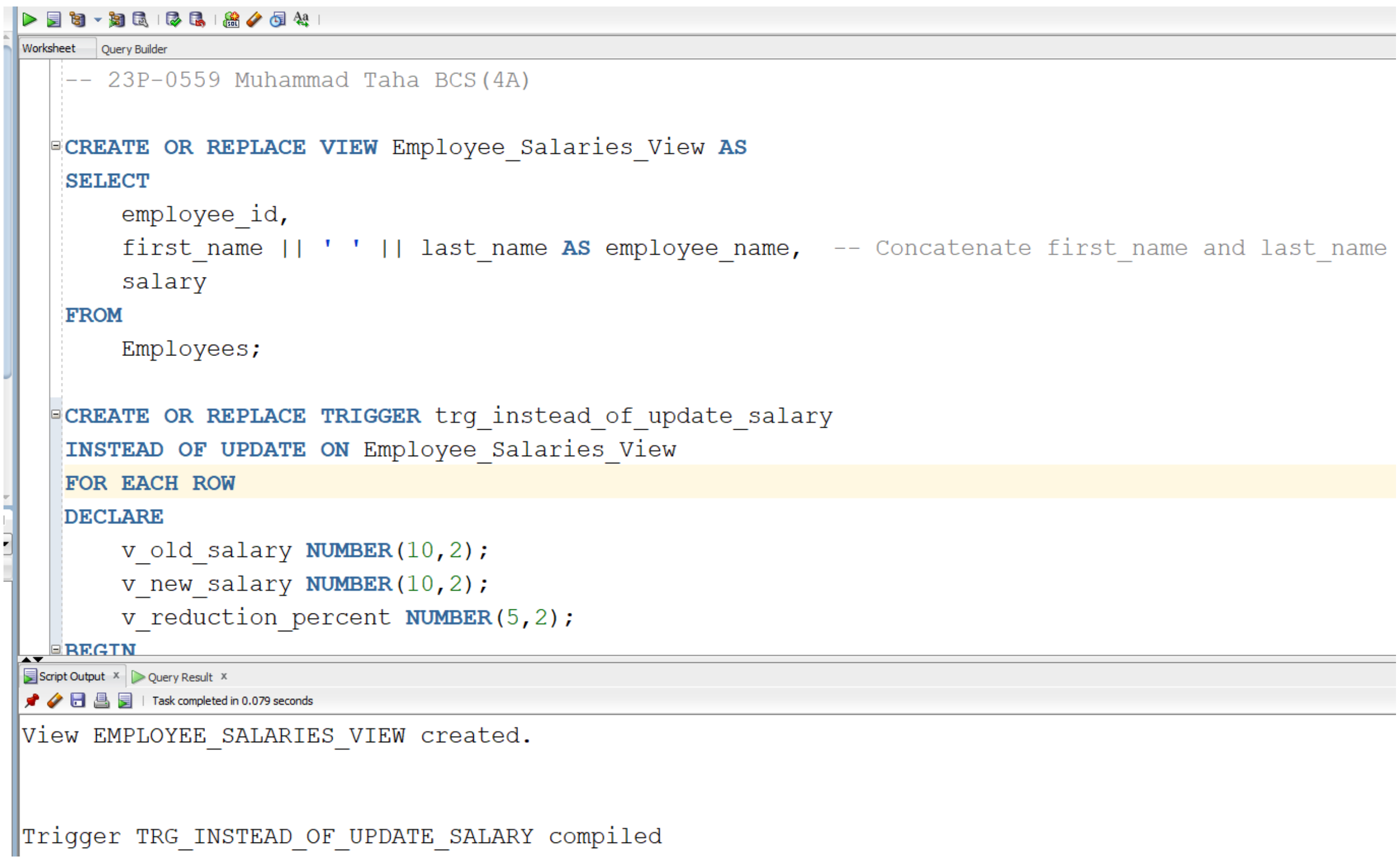
```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE VIEW Emp_Dept_View AS
SELECT
    e.employee_id,
    e.first_name,
    e.last_name,
    e.salary,
    d.department_id,
    d.department_name
FROM
    Employees e
JOIN
    Departments d
ON
    e.department_id = d.department_id;

CREATE OR REPLACE TRIGGER trg_instead_of_insert_empdept
```

Task completed in 0.131 seconds

```
Trigger TRG_INSTEAD_OF_INSERT_EMPDEPT compiled


1 row inserted.
```

Create a view that shows employee salaries, and write an INSTEAD OF UPDATE trigger to prevent any salary updates that reduce the employee's salary by more than 20%.

```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE VIEW Employee_Salaries_View AS
SELECT
   employee_id,
   first_name || ' ' || last_name AS employee_name,  -- Concatenate first_name and last_name
   salary
FROM
   Employees;

CREATE OR REPLACE TRIGGER trg_instead_of_update_salary
INSTEAD OF UPDATE ON Employee_Salaries_View
FOR EACH ROW
DECLARE
   v_old_salary NUMBER(10,2);
   v_new_salary NUMBER(10,2);
   v_reduction_percent NUMBER(5,2);
BEGIN
   -- Assign old and new salary values
   v_old_salary := :OLD.salary;
   v_new_salary := :NEW.salary;

   -- Check if the new salary is less than the old salary (i.e., reduction)
   IF v_new_salary < v_old_salary THEN
      -- Calculate the percentage reduction
      v_reduction_percent := (v_old_salary - v_new_salary) / v_old_salary * 100;

      -- If reduction exceeds 20%, raise an exception (block the update)
      IF v_reduction_percent > 20 THEN
         RAISE_APPLICATION_ERROR(-20001, 'Error: Salary reduction exceeds 20%. Update not
allowed.');
      ELSE
         -- If the reduction is within the limit, perform the update
         UPDATE Employees
         SET salary = v_new_salary
         WHERE employee_id = :OLD.employee_id;
      END IF;
   ELSE
      -- If the salary is increased or remains the same, update the salary normally
      UPDATE Employees
      SET salary = v_new_salary
      WHERE employee_id = :OLD.employee_id;
   END IF;
END;
```



```
-- 23P-0559 Muhammad Taha BCS(4A)

CREATE OR REPLACE VIEW Employee_Salaries_View AS
SELECT
    employee_id,
    first_name || ' ' || last_name AS employee_name,  -- Concatenate first_name and last_name
    salary
FROM
    Employees;

CREATE OR REPLACE TRIGGER trg_instead_of_update_salary
INSTEAD OF UPDATE ON Employee_Salaries_View
FOR EACH ROW
DECLARE
    v_old_salary NUMBER(10,2);
    v_new_salary NUMBER(10,2);
    v_reduction_percent NUMBER(5,2);
BEGIN
```

```
View EMPLOYEE_SALARIES_VIEW created.


Trigger TRG_INSTEAD_OF_UPDATE_SALARY compiled
```