|  |  |
| --- | --- |
| Nama | : Afina Putri Dayanti |
| NIM | : 825200049 |
| Jurusan | : Sistem Informasi |
| Mata Kuliah | : Database Design and Management (Praktikum) |

**Vocabulary**

Identify the vocabulary word for each definition below:

|  |  |  |
| --- | --- | --- |
| Compound Trigger |  | is a single trigger that can include actions for each of the four possible timing points |
| INSTEAD OF Trigger |  | a trigger which replaces a DML statement on a complex view with DML statements on the tables on which the view is based |
| Conditional Predicates |  | predefined Boolean variables INSERTING, DELETING and UPDATING which can be tested in a trigger body to take different code paths depending on which DML statement caused the trigger to fire |
| :OLD and :NEW Qualifiers |  | enables a row trigger to access column values in the table row currently being modified by the triggering statement |
| DML row trigger |  | a DML trigger which fires once for each row affected by the triggering DML statement |

**Try It / Solve It**

1. Retrieve the code for the AFTER INSERT trigger you created in the previous practice, question 2B. If you have lost the code, here it is again:

CREATE OR REPLACE TRIGGER emp\_audit\_trigg

AFTER INSERT ON employees

BEGIN

INSERT INTO audit\_table (action) VALUES ('Inserting');

END;

Answer :

|  |
| --- |
| create or replace trigger log\_audit\_table  after insert on emp1  begin  insert into audit\_table(action, user\_name, last\_change\_date)  values ('Inserting',user, systimestamp );  end; |

1. Modify this trigger so that a DELETE on the EMPLOYEES table will fire the same trigger. Use the conditional predicates so an INSERT adds a row to the AUDIT\_EMP table with ‘Inserted’ for the action column and a DELETE adds a row with ‘Deleted’ in the action column. Save the script and test your trigger by inserting an employee row and then deleting the same row, querying the AUDIT\_EMP table each time.

Answer :

|  |
| --- |
| create or replace trigger log\_audit\_table  after insert or delete on emp1  begin  if inserting then  insert into audit\_table(action, user\_name, last\_change\_date)  values ('Inserting',user, systimestamp );  elsif deleting then  insert into audit\_table(action, user\_name, last\_change\_date)  values ('Deleting',user, systimestamp );  end if;  end; |

|  |
| --- |
| ***Insert***  insert into emp1  values (58, 'Afina', 'Putri', 'afnpd03@gmail.com', '085772610027', TO\_DATE(SYSDATE), 'AD\_PRES', 3200, null, null, null, null);  select \* from audit\_table; |

|  |
| --- |
| ***Delete***  delete from emp1  where employee\_id = 58;  select \* from audit\_table; |

1. Add a new column called emp\_id to the AUDIT\_EMP table. This column will contain the employee id of the worker whose record was inserted or deleted. Modify your trigger to be a row trigger so it will fire once for each row affected. The INSERTs into the AUDIT\_EMP table should now include the employee id of the affected employee. INSERT and DELETE one or more employees. Query the AUDIT\_EMP table to see the audit trail.

Answer :

|  |
| --- |
| alter table audit\_table add emp\_id number(3);  create or replace trigger log\_audit\_table  after insert or delete on emp1 for each row  begin  if inserting then  insert into audit\_table(action, user\_name, last\_change\_date, emp\_id)  values ('Inserting', user, systimestamp, :new.employee\_id);  elsif deleting then  insert into audit\_table(action, user\_name, last\_change\_date, emp\_id)  values ('Deleting', user, systimestamp,:old.employee\_id);  end if;  end; |

1. To practice using INSTEAD OF triggers, complete the following steps.
2. Execute the following statement to create a table called DEPT\_COUNT that keeps track of how many employees are in each department.

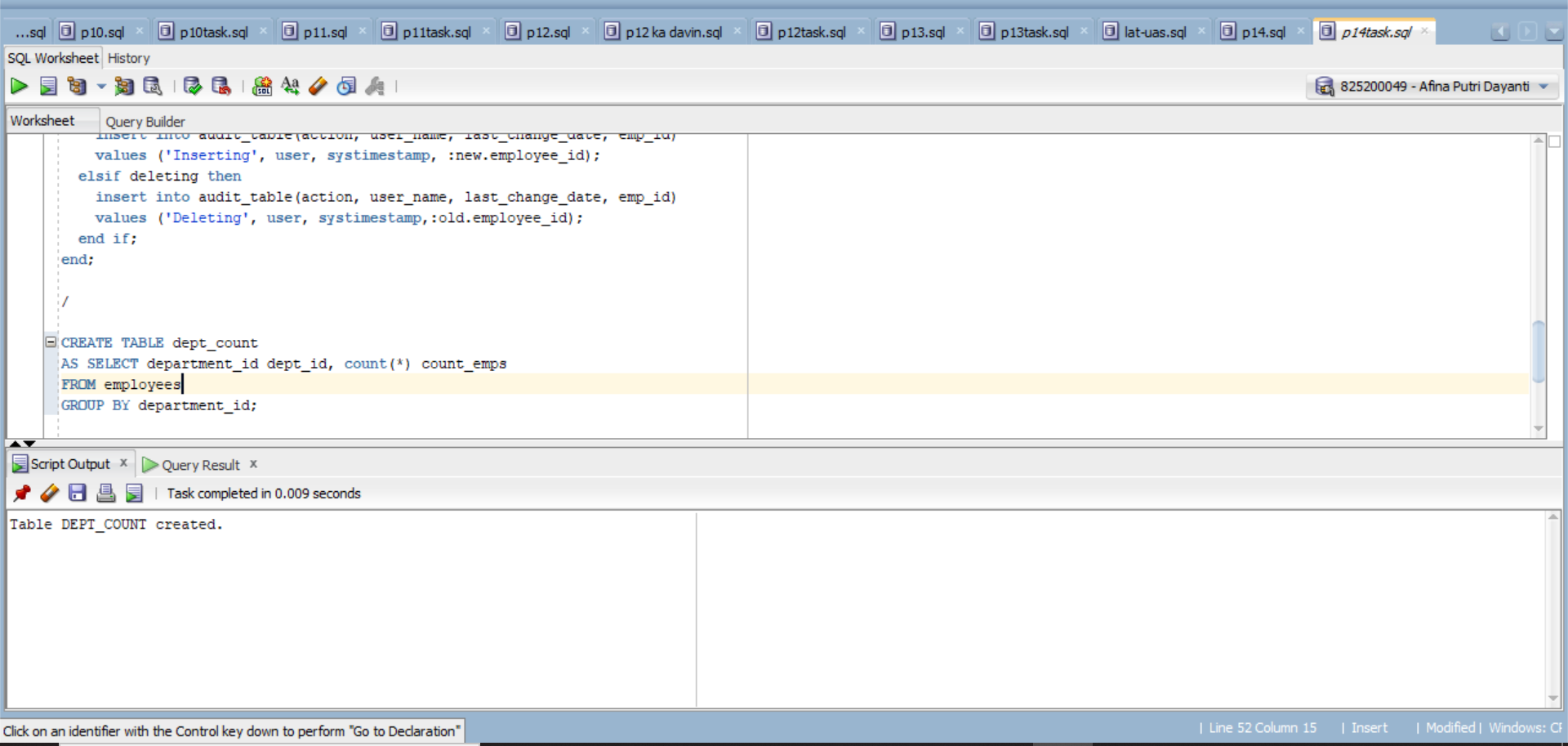
CREATE TABLE dept\_count

AS SELECT department\_id dept\_id, count(\*) count\_emps

FROM employees

GROUP BY department\_id;

Answer :



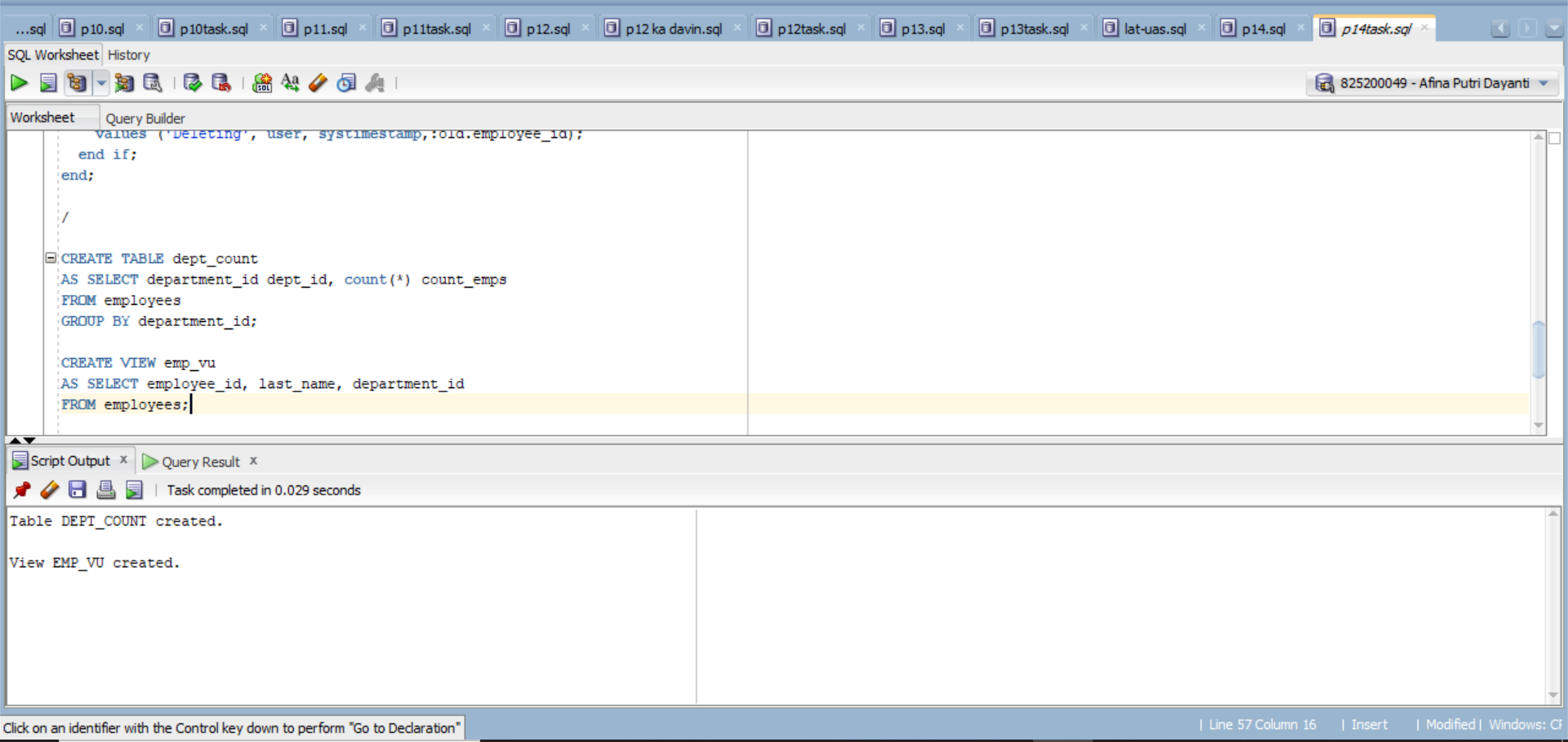
1. Execute the following statement to create a view of the EMPLOYEES table called EMP\_VU.

CREATE VIEW emp\_vu

AS SELECT employee\_id, last\_name, department\_id

FROM employees;

Answer :



1. Create an INSTEAD OF row trigger on EMP\_VU that increases the current count for a department by 1 if a new employee is added and subtracts 1 from the count for a department if an employee is deleted.

Answer :

|  |
| --- |
| create or replace trigger count\_trigger  instead of insert or delete on emp\_vu  begin  if inserting then  update dept\_count set count\_emps = count\_emps + 1  where dept\_id = :new.department\_id;  elsif deleting then  update dept\_count set count\_emps = count\_emps - 1  where dept\_id = :old.department\_id;  end if;  end; |

1. Look at the counts for all departments in DEPT\_COUNT. Test to see if your trigger fires correctly by inserting a row into EMP\_VU. Look at the count for the department of the new employee. Delete a row from EMP\_VU. Look at the count for the department where the employee was just deleted.

Answer :

|  |
| --- |
| ***Before Insert***  select \* from dept\_count where dept\_id = 90;    ***After Insert***  insert into emp\_vu values(60, 'Afina', 90);  select \* from dept\_count where dept\_id = 90; |

|  |
| --- |
| ***Before Delete***  select \* from dept\_count where dept\_id = 90;    ***After Delete***  delete from emp\_vu where employee\_id = 100;  select \* from dept\_count where dept\_id = 90; |

1. In this question, you will create a compound trigger. Once again, you will use the AUDIT\_TABLE you created in a previous exercise. If you have lost that table, below is the code to recreate it.

CREATE TABLE audit\_table

(action VARCHAR2(50),

user\_name VARCHAR2(30) DEFAULT USER,

last\_change\_date TIMESTAMP DEFAULT SYSTIMESTAMP,

emp\_id NUMBER(6));

1. Create a compound trigger emp\_audit\_trigg on the EMPLOYEES table for the following events: when updating the salary column of the EMPLOYEES table, enter the value ‘Updating’ into the action column of the AUDIT\_TABLE before the change occurs. Next, once the action is complete, change the action to ‘Update complete; old salary was (old\_sal); new salary is (new\_sal)’ where old\_sal is the original salary before the UPDATE, and new\_sal is the new salary.

Answer :

|  |
| --- |
| create or replace trigger log\_audit\_table  for update of salary on emp1 compound trigger  log varchar2(200);  before each row is begin  insert into audit\_table(action, user\_name, last\_change\_date, emp\_id)  values ('Updating', user, systimestamp, :new.employee\_id);  end before each row;  after each row is begin  log := 'Update complete; old salary ' || to\_char(:old.salary) || '; new salary ' || to\_char(:new.salary);  insert into audit\_table(action, user\_name, last\_change\_date, emp\_id)  values (log, user, systimestamp, :new.employee\_id);  end after each row;  end log\_audit\_table; |

1. Test your trigger by updating the salary of employee\_id = 124 to be 1200, then querying the AUDIT\_TABLE to see that it contains a new row.

|  |
| --- |
| update emp1  set salary = 1200  where employee\_id = 124;  select \* from audit\_table; |