• Create a table emp with fields eno, ename, department, salary

• Insert atleast 5 records into given table.

=> Table created:

```
CREATE TABLE emp1 (
  eno NUMBER,
  ename VARCHAR2(50),
  department VARCHAR2(50),
  salary NUMBER
);
```

=> Inserted Values:

```
INSERT INTO emp1 VALUES (1,'Alice','IT',50000);
INSERT INTO emp1 VALUES (2, 'Bob', 'HR', 6000);
INSERT INTO emp1 VALUES (3, 'Charlie', 'IT', 7000);
INSERT INTO emp1 VALUES (4, 'David', 'Finance', 8000);
INSERT INTO emp1 VALUES (5, 'Eve', 'IT', 9000);
```

1)1. Write a database trigger store the deleted data of EMP table in EMPDEL table.

```
CREATE TABLE EMPDEL (
  eno NUMBER,
  ename VARCHAR2(50),
  department VARCHAR2(50),
  salary NUMBER
);

CREATE OR REPLACE TRIGGER EMPDEL
BEFORE DELETE ON emp1
FOR EACH ROW
```

```
begin
  INSERT INTO EMPDEL (eno,ename,department,salary) VALUES
(:OLD.eno,:OLD.ename,:OLD.department,:OLD.salary);
end;
/
```

Test:

```
SQL> select * from emp1;

      ENO ENAME
---------- --------------------------------------------------
DEPARTMENT                                          SALARY
-------------------------------------------------- ----------
         2 Bob
HR                                                    6000

         3 Charlie
IT                                                    7000

         4 David
Finance                                               8000


      ENO ENAME
---------- --------------------------------------------------
DEPARTMENT                                          SALARY
-------------------------------------------------- ----------
         5 Eve
IT                                                    9000
```

SQL> delete from emp1 where eno=2;

1 row deleted.

SQL> select * from empdel;

ENO ENAME

---------- --

DEPARTMENT SALARY

-- ----------

2 Bob

HR 6000

2)2. Write a database trigger to update salary in employee table and it shows salary difference before updating data.

```
CREATE OR REPLACE TRIGGER emp_salary_trigger
BEFORE UPDATE OF salary ON emp1
FOR EACH ROW
DECLARE
   salary_diff NUMBER;
BEGIN
   salary_diff := :NEW.salary - :OLD.salary;
   DBMS_OUTPUT.PUT_LINE('Salary difference: ' || salary_diff);
END;
/
```

test:

```
INSERT INTO emp1 VALUES (1,'Alice','IT',50000);
```

```
SQL> UPDATE emp SET salary = 5500 WHERE eno = 1; -- Assuming eno 1 is one of the existing rows
  2
SQL> UPDATE emp1 SET salary = 5500 WHERE eno = 1;
Salary difference: -44500

1 row updated.
```

3) Write a trigger to store eid,department,salary to new table before inserting a new record into emp table.

```
CREATE TABLE new_emp (
    eno NUMBER,
    department VARCHAR2(50),
    salary NUMBER
);

CREATE OR REPLACE TRIGGER emp_insert_trigger
BEFORE INSERT ON emp1
FOR EACH ROW
BEGIN
    INSERT INTO new_emp (eno, department, salary)
    VALUES (:NEW.eno, :NEW.department, :NEW.salary);
END;
/
```

test/check:

```
INSERT INTO emp1 (eno, ename, department, salary) VALUES (101, 'John Doe', 'IT', 5000);
```

```
SELECT * FROM new_emp WHERE eno = 101;
```

4. Write a trigger to store employee details into new table who is working in ‘IT’ department.

```
CREATE OR REPLACE TRIGGER emp_it_trigger
BEFORE INSERT ON emp1
FOR EACH ROW
BEGIN
   IF :NEW.department = 'IT' THEN
      INSERT INTO new_emp (eno, department, salary)
      VALUES (:NEW.eno, :NEW.department, :NEW.salary);
   END IF;
END;

INSERT INTO emp1 VALUES (102,'Ashish','IT',9000);
INSERT INTO emp1 VALUES (103,'Harshil','IT',19000);

SQL> select * from new_emp;

     ENO DEPARTMENT                                SALARY
---------- -------------------------------------------------- ----------
     101 IT                                        5000
     102 IT                                        9000
     102 IT                                        9000

SQL> INSERT INTO emp1 VALUES (103,'Harshil','IT',19000);
```

1 row created.

SQL> select * from new_emp;

ENO	DEPARTMENT	SALARY
101	IT	5000
102	IT	9000
102	IT	9000
103	IT	19000
103	IT	19000

5. Write a trigger which will store eno,old_salary,new_salary into new_emp table before update the salary in emp table.

```
CREATE OR REPLACE TRIGGER emp_salary_update_trigger
BEFORE UPDATE OF salary ON emp1
FOR EACH ROW
BEGIN

   INSERT INTO new_emp (eno, department, salary)
   VALUES (:OLD.eno, :OLD.department, :OLD.salary);

   INSERT INTO new_emp (eno, department, salary)
   VALUES (:OLD.eno, :OLD.department, :NEW.salary);
```

```
END;
/

UPDATE emp1 SET salary = 7000 WHERE eno = 4;
SQL>  select * from new_emp;

      ENO DEPARTMENT                                          SALARY
---------- -------------------------------------------------- ----------
       101 IT                                                   5000
       102 IT                                                   9000
       102 IT                                                   9000
       103 IT                                                  19000
       103 IT                                                  19000
         1 IT                                                  7000
         1 IT                                                  7000
         4 Finance                                             8000
         4 Finance                                             7000
```