

### Program 1

Write a code in PL/SQL to develop a trigger that enforces referential integrity by preventing the deletion of a parent record if child records exist.

create or replace trigger prevent-parent-delete  
before delete on departments  
for each row

declare  
v\_count number;

begin

select count(\*) into v\_count

from employees

where department\_id = :old.department\_id;

if v\_count > 0 then

raise\_application\_error(-20001,

'cannot delete department: Employees exist  
in this department');

end if;

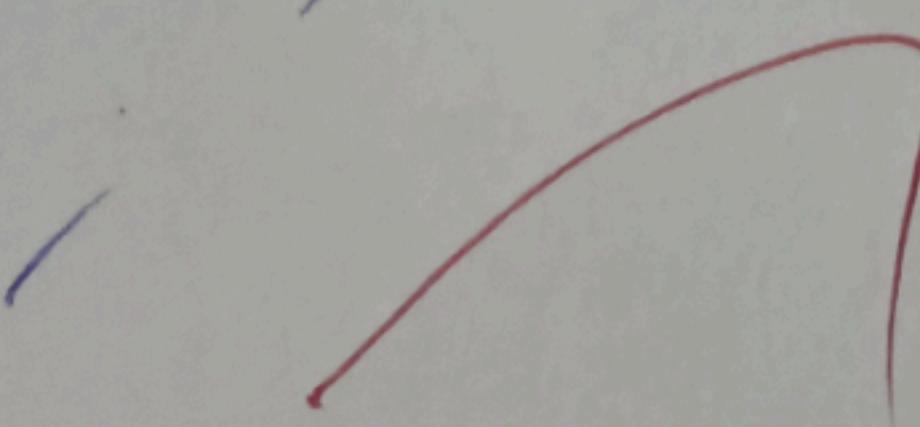
end;



## Program 2

Write a code in PL/SQL to create a trigger that checks for duplicate values in a specific column and raises an exception if found.

```
create or replace trigger prevent-duplicate-
email
before insert or update on employees
for each row
declare
    v_count number;
begin
    select count(*) into v_count
    from employees
    where email = New_email;
    if v_count > 0 then
        raise_application_error (-20002)
        ('Duplicate email detected! Each employee
        must have a unique email');
    end if;
end;
```



### Program 3

Write a code in PL/SQL to create a trigger that restricts the insertion of new rows if the total of a column's values exceeds a certain threshold.

create or replace trigger salary - limit - check

before insert on employees

for each row

declare

v-limit constant number := 100000;

begin  
select nvl(sum(salary), 0) + :new.salary  
into v-total-salary  
from employees;

if v-total-salary > v-limit then

raise application\_error(-20003,

'Insertion rejected: Total salary (

of ' || v-limit || ' exceeded');

end if;

end;



#### Program 4

Write a code in PL/SQL to design a trigger that captures changes made to specific columns and logs them in an audit table.

```
create table emp trigger employee-audit-
trigger
after update of salary, job-id on employees
for each row
begin
if :old.salary != New.salary then
insert into employee-audit(
employee-id, column-name, old-value, new-value,
changed-by, changed-on) values (:old.employee-id,
(salary), to_char(:old.salary) to char
(' ', new.salary),
user, sysdate
);
end if;
if :old.job-id != new.job-id then
insert into employee-(id, 'job-id',
:old.job-id: new.job-id,
user, sysdate
);
end if;
end /
```

## Program 5

Write a code in PL/SQL to implement a trigger that records user activity (inserts, updates, deletes) in an audit log for a given set of tables.

create or replace trigger employee\_activity\_audit  
after insert or update or delete on employee  
for each row

declare

v\_action varchar2(10);

begin

if inserting then

v\_action := 'insert';

else if updating then

v\_action := 'update';

else if deleting then

v\_action := 'delete';

end if;

insert into audit\_log (

table\_name, operation, user\_name, operation\_date, primary  
value) values (

'employees', v\_action, user, sysdate;

end

when inserting then dbms\_output.put\_line('New Employee inserted');

when updating then dbms\_output.put\_line('Employee updated');

end

end;

194