use analyticsdb;

create table course(

course\_id int,

course\_desc varchar(50),

course\_mentor varchar(60),

course\_price int,

course\_discount int,

create\_date date);

alter table course add column(user\_update varchar(50));

drop table course;

create table course\_update(

course\_mentor varchar(50),

course\_price\_update int,

course\_discount\_update int

);

drop table course\_update;

-- Trigger

delimiter $$

create trigger course\_before\_insert

before insert

on course for each row

begin

declare user\_val varchar(50);

set new.create\_date = sysdate();

select user() into user\_val;

set new.user\_update= user\_val;

end; $$

delimiter $$

create trigger course\_before\_insert

before insert

on course for each row

begin

set new.create\_date = sysdate();

set new.user\_update= user();

end; $$

drop trigger course\_before\_insert2;

insert into course (course\_id,course\_desc, course\_mentor,course\_price,course\_discount) values(102,'Data Science','Krish Naik',2700,15);

select \* from course order by course\_id;

drop table ref\_course;

create table ref\_course(

course\_id int,

record\_insert\_date date,

record\_insert\_user varchar(50)

);

drop trigger course\_before\_insert;

Drop trigger course\_ref\_insert;

delimiter $$

create trigger course\_ref\_insert

before insert

on course for each row

begin

declare course\_id\_val int;

select new.course\_id into course\_id\_val;

set new.create\_date = sysdate();

set new.user\_update= user();

insert into ref\_course values(course\_id\_val, sysdate(),user(),'New Record Inserted');

end; $$

insert into course (course\_id,course\_desc, course\_mentor,course\_price,course\_discount) values(105,'Power BI','Jayant',5000,7);

select \* from course;

select \* from ref\_course;

alter table ref\_course add column(update\_reason varchar(50));

-- In case of Deletion, Comment should be deletion done on record

delimiter $$

create trigger course\_ref\_insert\_on\_delete

before delete

on course for each row

begin

declare course\_id\_val int;

select old.course\_id into course\_id\_val;

insert into ref\_course values(course\_id\_val, sysdate(),user(),'Deletion of Record');

end; $$

delete from course where course\_id = 104;

select \* from course;

select \* from ref\_course;

create table test1(a varchar(20), date1 date, empid int);

create table test2(a varchar(20), date1 date, empid int);

create table test3(a varchar(20), date1 date, empid int);

drop trigger before\_insert;

delimiter ##

create trigger before\_insert

before insert on test1 for each row

begin

insert into test2 values("abc",sysdate(),111);

insert into test3 values("abc",sysdate(),111);

end;##

select \* from test1;

select \* from test2;

delete from test3;

select \* from test3;

insert into test1 values("Rank",sysdate(),147);

update test1

set a = "Peter";

-- UPDATE TRIGGER

delimiter $$

create trigger update\_triggger

after update on test1 for each row

begin

update test2 set empid = 100;

insert into test3 values("abc",sysdate(),100);

delete from test3 where empid = 111;

end;

$$

-- DELETE TRIGGER

delimiter ^^

create trigger after\_delete

after delete on test1 for each row

begin

insert into test3 values('after delete operation', sysdate(),1111);

end;

delimiter ^^

create trigger before\_delete

before delete on test1 for each row

begin

insert into test3 values('after delete operation', sysdate(),1111);

end;

delimiter ^^

create trigger before\_delete

before delete on test1 for each row

begin

insert into test2 values('after delete ', sysdate(),1111);

end;

delete from test1 where empid = 145;

select \* from test1;

select \* from test2;

delete from test3;

select \* from test3;

----

drop table test11;

create table test11(

c11 varchar(50),

c21 date,

c31 timestamp,

c41 int);

drop table test12;

create table test12(

c12 varchar(50),

c22 date,

c32 timestamp,

c42 int );

drop table test13;

create table test13(

c13 varchar(50),

c23 date,

c33 timestamp,

c43 int );

delimiter //

create trigger to\_delete\_others\_before\_observation

before delete on test11 for each row

begin

insert into test12(c12,c22,c32,c42) values(old.c11, old.c21,old.c31,old.c41);

end; //

select CURRENT\_TIMESTAMP();

select current\_time();

insert into test11 values("sudh" , sysdate(),current\_time(),435456);

select \* from test11

delete from test11 where c11 = 'sudh'

select \* from test12

delimiter //

create trigger to\_delete\_others\_after\_observation

after delete on test11 for each row

begin

insert into test12(c12,c22,c32,c42) values(old.c11, old.c21,old.c31,old.c41);

end; //

insert into test11 values("sudh" , sysdate(),current\_time(),435456);

insert into test11 values("sudhanshu" , sysdate(),current\_time(),435456);

select \* from test11;

select \* from test12;

delete from test11 where c11 = 'sudhanshu';

-- Below Update trigger will update the values in test11 with new values but will update test12 with old values of test11.

delimiter //

create trigger to\_update\_others

after update on test11 for each row

begin

insert into test12(c12,c22,c32,c42) values(old.c11, old.c21,old.c31,old.c41);

end; //

select \* from test11;

insert into test11 values("Ravi Dubey" , sysdate(),current\_time(),435456);

select \* from test11;

select \* from test12;

update test11 set c11 = "after update" where c11 = "Ravi Dubey";

-- Below Update trigger will update the values in test11 with new values but will update test12 with old values of test11.

delimiter //

create trigger to\_update\_before

before update on test11 for each row

begin

insert into test12(c12,c22,c32,c42) values(old.c11, old.c21,old.c31,old.c41);

insert into test12(c12,c22,c32,c42) values(new.c11, new.c21,new.c31,new.c41);

end; //

select \* from test11;

insert into test11 values("Astha" , sysdate(),current\_time(),10081986);

select \* from test11;

select \* from test12;Top of Form

Bottom of Form

Top of Form

Bottom of Form

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | update test11 set c11 = "Astha Dubey" , c41= 28112012 where c11 = "Astha"; |
|  |  |

# CASE Statement

|  |  |
| --- | --- |
|  |  |
|  | select \* from ineuron\_Students; |
|  |  |
|  | select \*, |
|  | case |
|  |  |
|  | when student\_batch =' FSDS' then "Datascience" |
|  | else "Non-DataScience" |
|  | end as statement, |
|  | case |
|  |  |
|  | when student\_stream = 'ECE' then "Electronics" |
|  | when student\_stream = 'me' then "Mechanical Engineering" |
|  | when student\_stream = 'ee' then "Electronics Engineering" |
|  | ELSE "Non Computers" |
|  | end as student\_stream\_importance |
|  | from ineuron\_Students; |
|  |  |
|  |  |
|  | update ineuron\_Students |
|  | set student\_mail\_id = |
|  | case |
|  | when student\_stream = 'cs' then 'computers@gmail.com' |
|  | when student\_stream = 'CI' then 'ComputersInfotech@gmail.com' |
|  | else student\_mail\_id |
|  | end; |
|  |  |
|  | select \* from ineuron\_Students; |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |