## NIRAJ NANDISH

## 191IT234 | IT252 | 28/03/2021

# IT252

# Database Management System

## Assignment VIII

1.

create table STUDENT\_MARKS(

STUDENT\_ID int not null primary key,

NAME varchar(30),

SUB1 int default 0,

SUB2 int default 0,

SUB3 int default 0,

SUB4 int default 0,

SUB5 int default 0,

TOTAL int default 0,

PER\_MARKS decimal(5,2) default 0.00,

GRADE varchar(20) default ''

);

insert into STUDENT\_MARKS (STUDENT\_ID, NAME) values

(1, 'Steven King'),

(2, 'Neena Kochhar'),

(3, 'Lex De Haan'),

(4, 'Alexander Hunold');

delimiter $$

create trigger upd\_stud\_tmarks

before update

on STUDENT\_MARKS for each row

begin

set new.TOTAL = new.SUB1 + new.SUB2 + new.SUB3 + new.SUB4 + new.SUB5;

set new.PER\_MARKS = new.TOTAL/5;

if new.PER\_MARKS >= 90 then

set new.GRADE = "EXCELLENT";

elseif new.PER\_MARKS >= 75 and new.PER\_MARKS < 90 then

set new.GRADE = "VERY GOOD";

elseif new.PER\_MARKS >= 60 and new.PER\_MARKS < 75 then

set new.GRADE = "GOOD";

elseif new.PER\_MARKS >= 40 and new.PER\_MARKS < 60 then

set new.GRADE = "AVERAGE";

elseif new.PER\_MARKS < 40 then

set new.GRADE = "NOT PROMOTED";

end if;

end $$

delimiter ;

update STUDENT\_MARKS set SUB1 = 54, SUB2 = 69, SUB3 = 89, SUB4 = 87, SUB5 = 59 where STUDENT\_ID = 1

select \* from STUDENT\_MARKS;

Text

Description automatically generated

2.

delimiter $$

create trigger blog\_after\_insert

after insert

on blog for each row

begin

if new.deleted then

insert into audit (blog\_id, changetype) values (new.id, 'DELETE');

else

insert into audit (blog\_id, changetype) values (new.id, 'NEW');

end if;

end $$

create trigger blog\_after\_update

after update

on blog for each row

begin

if new.deleted then

insert into audit (blog\_id, changetype) values (new.id, 'DELETE');

else

insert into audit (blog\_id, changetype) values (new.id, 'EDIT');

end if;

end $$

delimiter ;

insert into blog values (1, 'Article One', 'Initial text', 0);

select \* from audit;

update blog set content = 'Edited text' where id = 1;

Text

Description automatically generatedselect \* from audit;

3.

create table Customer\_Bank(

Account\_No int not null,

Customer\_Name varchar(50),

Address varchar(75),

Branch\_Code int,

Type\_of\_Transaction varchar(30),

Balance\_Amount decimal(8,2)

);

delimiter $$

create trigger type before update on Customer\_Bank for each row

begin

declare amount int default 0;

declare type\_type varchar(50);

select new.Balance\_Amount into amount;

select new.Type\_of\_Transaction into type\_type;

if type\_type = 'Credit' then

set new.Balance\_Amount = amount + old.Balance\_Amount;

elseif type\_type = 'Debit' and amount<=50000 and amount<old.Balance\_Amount then

set new.Balance\_Amount = old.Balance\_Amount - amount;

end if;

end $$

delimiter ;

insert into Customer\_Bank values (1, 'Niraj', 'Mysore', 12345, 'Credit', 50000);

update Customer\_Bank set Balance\_Amount = 65000 where Account\_No = 1;

select \* from Customer\_Bank;

insert into Customer\_Bank values (2, 'Tom', 'UK', 2345, 'Credit', 100000);

update Customer\_Bank set Balance\_Amount = 30000 where Account\_No = 2;

select \* from Customer\_Bank;

Text

Description automatically generatedText

Description automatically generated

Text

Description automatically generated

4.

delimiter $$

create procedure updatestuff()

begin

declare suba int;

declare subb int;

declare subc int;

declare subd int;

declare sube int;

declare id\_no int;

declare total\_marks int;

declare flag int default 0;

declare percentage decimal(19,2);

declare grade\_student varchar(50);

declare marks cursor for select STUDENT\_ID,SUB1,SUB2,SUB3,SUB4,SUB5 from STUDENT\_MARKS;

declare continue handler for not found set flag = 1;

open marks;

getMarks : loop

fetch marks into id\_no,suba,subb,subc,subd,sube;

set total\_marks = suba + subb + subc + subd + sube ;

if flag = 1 then

leave getMarks;

end if;

set percentage = total\_marks/5;

if percentage >= 90 then set grade\_student = 'EXCELLENT';

elseif percentage >= 75 then set grade\_student = 'VERY GOOD' ;

elseif percentage >=60 then set grade\_student = 'GOOD' ;

elseif percentage >= 40 then set grade\_student = 'AVERAGE' ;

else set grade\_student = 'NOT PROMOTED';

end if;

update STUDENT\_MARKS set TOTAL = total\_marks,PER\_MARKS = percentage,

GRADE = grade\_student where STUDENT\_ID = id\_no;

end loop getMarks;

close marks;

end $$

delimiter ;

Text

Description automatically generated

A picture containing text, monitor, black, scoreboard

Description automatically generated

5.

delimiter $$

create procedure add\_level()

begin

declare flag int default 0;

declare acc\_no int;

declare amount double;

declare cal\_level varchar(50);

declare curno cursor for select Account\_no from Customer\_Bank;

declare curamt cursor for select Balance\_Amount from Customer\_Bank;

declare continue handler for not found set flag = 1;

open curno; open curamt;

changelevel: loop

fetch curamt into amount;

fetch curno into acc\_no;

if flag=1 then

leave changelevel;

end if;

if amount>=100000 then set cal\_level = 'PLATINUM';

elseif amount>=50000 and amount<100000 then

set cal\_level='GOLD';

elseif amount<50000 then set cal\_level='SILVER';

end if;

update Customer\_Bank set Customer\_Level=cal\_level

where Account\_no = acc\_no;

end loop changelevel;

close curno;

close curamt;

end $$

delimiter ;

Text

Description automatically generated