Q1: You are the DBA for university database and has relation Instructor. For authorization reasons, you also define views EmployeeNames (with *instructor name* as the only attribute) and DeptInfo with fields *dept* and *avgsalary*. The latter lists the average salary for each department. Using create a view for the instructor .

Q2: create table sailorr(sid int,sname varchar(30),rating int,location varchar(30),age int);

insert into sailorr values(101,'ajay',3,'dadar',14);

insert into sailorr values(102,'sham',14,'chembur',13);

insert into sailorr values(103,'ram',15,'andheri',10);

insert into sailorr values(104,'shah',10,'vashi',11);

insert into sailorr values(105,'jay',8,'govandi',12);

select \* from sailor

1. Create a view to contain the listing of all sailors having rating greater than 7
2. Define rule to insert records using view in the sailors relation
3. Define rule to update records using view in the sailors relation
4. Define rule to delete records using view in the sailors relation

Syntax:

create rule in\_v as on insert to sailorvw do instead insert into sailorr values(new.sid,new.sname,new.rating,NULL,new.age)

Q2.Triggers :

select \* from instructor

create table instrcutor\_audit(id char(5) NOT NULL,entry\_date TEXT NOT NULL);

create function auditlogfunc() returns trigger as $example$

begin

insert into instrcutor\_audit(id,entry\_date) values(new.id,current\_timestamp);

return NEW;

end

$example$ LANGUAGE plpgsql

drop function auditlogfunc()

select \* from instructor

create trigger example\_trigger after insert on instructor for each row execute procedure auditlogfunc();

INSERT into instructor values('20111','Aparna','Music',80000);

select \* from instrcutor\_audit

create table instructor\_sal\_raise( id char(5),name char(20),dept\_name char(20),salary numeric(8,2),changed\_on timestamp(6) NOT NULL)

create function log\_sal\_change() returns trigger as $example\_table$

BEGIN

IF NEW.salary <> OLD.salary THEN

insert into instructor\_sal\_raise(id,name,dept\_name,salary,changed\_on) values(OLD.id,OLD.name,OLD.dept\_name,OLD.salary,now());

END IF;

RETURN NEW;

END;

$example\_table$ LANGUAGE plpgsql;

create trigger last\_sal\_changes

before update on instructor

for each row

execute procedure log\_sal\_change()

select \* from instructor;

update instructor set salary=30000 where id='10101'

select \* from instructor\_sal\_raise

select \* from pg\_trigger