**Trigger Scenario**

**1)Account trigger to set account rating as ‘Hot’ if account industry is ‘Banking’ or ‘Healthcare’**

trigger AccountTrigger on Account (before insert,before update) {  
     
     
    if((Trigger.isBefore) && ((Trigger.isInsert) || (Trigger.isUpdate))){  
        for(Account acc:Trigger.new){  
            if(acc.Industry=='Banking' || acc.Industry=='Healthcare'){  
                acc.Rating='Hot';  
            }  
        }  
    }  
  
}

**2)trigger on contact which will create Account record whenever contact is created without an account.**

trigger ContactTrigger on Contact (after insert) {  
   List<Account> accList=new List<Account>();  
   if((Trigger.isAfter) && (Trigger.isInsert)){  
    for(Contact con:Trigger.new){  
        if(con.AccountId==null){  
        Account acc=new Account();  
        acc.Name=con.LastName;  
        acc.Phone=con.Phone;  
        accList.add(acc);  
    }  
   }  
  
   }  
    if(!accList.isEmpty()){  
        insert accList;  
    }  
     
}

**3)Create a Trigger on Contact Object, to make the Email and Phone Fields should be Mandatory upon creating or updating a Contact Record.**

trigger ContactTrigger on Contact (before Insert,Before update,after insert) {

if((Trigger.isBefore) && ((Trigger.isInsert) || (Trigger.isUpdate))){

for(Contact con:Trigger.new){

if(con.Phone==null || con.Phone==''){

con.Phone.addError('Please enter Phone number');

}

if(con.Email==null || con.Email==''){

con.Email.addError('Please enter Email Id');

}

}

}

}

**4)Create a Trigger on Account Object, To assign the Annualrevenue value as below..**

**Industry Name Annualrevenue**

**----------------------------------------------------------------**

**Banking ----> 5000000**

**Finance ----> 4000000**

**Insurance ----> 3500000**

**Healthcare ----> 2500000**

**Else ----> 500000**

if((Trigger.isBefore) && ((Trigger.isInsert) || (Trigger.isUpdate))){

for(Account acc:Trigger.new){

if(acc.Industry=='Banking'){

acc.AnnualRevenue=5000000;

}

else if(acc.Industry=='Finance'){

acc.AnnualRevenue=4000000;

}

else if(acc.Industry=='Insurance'){

acc.AnnualRevenue=3500000;

}

else if(acc.Industry=='Healthcare'){

acc.AnnualRevenue=2500000;

}

else{

acc.AnnualRevenue=500000;

}

}

}

**5)Create a Trigger, to make sure Each Contact Should be get associated with an Account.**

if((Trigger.isBefore) && ((Trigger.isInsert) || (Trigger.isUpdate))){

for(Contact con:Trigger.new){

if(con.AccountId==null){

con.AccountId.addError('Please select Account Name');

}

}

}

**6)Create a Trigger To Prevent the deletion of an Active Account Record.**

trigger AccountTriggerExample6 on Account (before delete) {

if((Trigger.isBefore) && (Trigger.isDelete)){

for(Account acc:Trigger.old){

if(acc.Active\_\_c=='Yes'){

acc.addError('You cannot delete an active Active');

}

}

}

}

**7)Write an apex Trigger, to prevent the Duplicate Candidate Records inside the object.**

trigger CandidateTriggerEx7 on Candidate\_\_c (before insert) {

for(Candidate\_\_c cand:trigger.new){

integer recordsCount=[select count() from Candidate\_\_c where name=:cand.name];

if(recordsCount>0){

cand.addError('Duplicate Candidate record, already a candidate record with same name exists');

}

}

}

**8)Write a Trigger, to Prevent the Deletion of the Associated contacts upon removing the Account record.**

trigger AccountTriggerEx8 on Account (before delete) {

if(Trigger.isBefore && Trigger.IsDelete){

List<Contact> contList=[select Id,AccountId from Contact where AccountId=:Trigger.oldMap.KeySet()];

if(!contList.isEmpty()){

for(Contact con:contList){

con.AccountId=null;

}

update contList;

}

}

}

**9)Write an Apex Trigger, to synchronize the Contact Record's Phone and Fax value with the associated Account Record's Phone and Fax Numbers.**

trigger AccountTriggerEx9 on Account (after insert,after update) {

for(Account acc:Trigger.new){

List<Contact> conList=[Select Id,AccountId,Phone,fax from Contact where AccountId =:acc.Id];

for(Contact con:conList){

con.Phone=acc.Phone;

con.Fax=acc.Fax;

}

update conList;

}

}

trigger AccountTriggerEx9 on Account (after update) {

List<Account> accList=[select Id,Phone,Fax,(select Id,Phone,Fax from Contacts) from Account where id in:trigger.newmap.keyset()];

List<Contact> conListToUpdate=new List<Contact>();

if(!accList.isEmpty()){

for(Account acc:accList){

if(!acc.contacts.isEmpty()){

for(Contact con:acc.contacts){

con.Phone=acc.Phone;

con.Fax=acc.Fax;

conListToUpdate.add(con);

}

}

}

}

if(!conListToUpdate.isEmpty()){

update conListToUpdate;

}

}

**10) When lead is created or updated then check if the email of lead is already there in existing contacts. If email already exist then throw error.**

trigger LeadTriggerEx10 on Lead (before insert,before update) {

Map<String,Contact> mapOfContact=new Map<String,Contact>();

list<Contact> conList=[select id,email from Contact];

for(Contact con:conList){

mapOfContact.put(con.email,con);

}

for(Lead l:trigger.new){

if((l.email!=null) && (trigger.isInsert || (l.email!=trigger.oldmap.get(l.id).email)) ){

if(mapOfContact.containsKey(l.email)){

l.email.addError('Email already Exists');

}

}

}

}