

```

1) trigger AcceptAllClosedCase on Case (before insert) {

    List newCases = new List();

    User u = [Select ID from User where Name = 'NeenOpal User'];

    for (Case a : Trigger.new) {

        a.Status = 'Closed';

        a.Ownerid = u.ID;

        newCases.add(a);

    }

    update newCases;

}

2)

trigger CreateRenewalOpps on Opportunity (after update) {

// Create a Map to store all renewal opps for bulk inserting

    List< Opportunity> renewals = new List< Opportunity>();

    for (Opportunity opp : Trigger.new) {

        opportunity oldvalue= trigger.oldmap.get(opp.id);

// Only create renewal opps for closed won deals

        if (opp.StageName.contains('Closed') && oldvalue.StageName!= opp.StageName) {

            Opportunity renewal = new Opportunity();

            renewal.Name = opp.Name + 'Renewal';

            renewal.AccountId=opp.AccountId ;

            renewal.CloseDate = opp.CloseDate + 365; // Add a year

            renewal.StageName = 'Open';

            renewal.Recordtypeid =

Schema.SObjectType.Opportunity.getRecordTypeInfosByName().get('renewal').getRecordTypeId();

            renewal.Ownerid= opp.Ownerid;

        }

    }

}

```

```

        renewals.add( renewal);

    }

}

insert renewals;

}

3)

trigger numberofopportunity on Opportunity (after insert,after delete) {
set<id>accid=new set<id>();

list<Opportunity>opportunitylist=new list<Opportunity>();

list<Opportunity>listopp=new list<Opportunity>();

list<account>acclist=new list<account>();

list<account>listacc=new list<account>();

map<id,integer>mapOpp=new map<id,integer>();

if(trigger.isinsert){

for(Opportunity opp:trigger.new){

accid.add(opp.accountid);

}

}

if (trigger.isdelete){

for(Opportunity opp:trigger.old){

accid.add(opp.accountid);

}

}

acclist=[SELECT id,name FROM account WHERE id in:accid];

```

```

opportunitylist=[SELECT id,name,accountid,StageName FROM Opportunity WHERE accountid
in:accid];

for(account acc:acclist){

    listopp.clear();

    for(Opportunity o:opportunitylist){

        if(o.accountid==acc.id && (o.StageName !='Closed Won') && (o.StageName!='Closed Lost')){

            listopp.add(o);

            mapOpp.put(o.accountid,listopp.size());

        }

    }

}

if(acclist.size()>0){

    for(Account a:acclist){

        if(mapOpp.get(a.id)==null)

            a.No_of_Opportunity__c=0;

        else

            a.No_of_Opportunity__c=mapOpp.get(a.id);

        listacc.add(a);

    }

}

if(listacc.size()>0)

update listacc;

}

4 )

trigger CaseOrigin on Case (before insert) {

```

```

        for(case c : trigger.new){
            if(c.origin == 'phone'){
                c.status = 'New';
                c.priority = 'High';
            }
        }
    }

trigger insertTasks on User (after insert) {
// get list of users needed to assign
List<User> userList = [SELECT Id FROM User where ProfileId = 'xxxxxx' ];
List<Objectives__c> objList = [SELECT Id FROM Objectives__c];

//loop thru users in the users list created above.
for(User newUser : Trigger.new){
//create new objectives
Objectives__c obj = new Objectives__c();
obj.OwnerId = userList.get(0).Id;
obj.Status__c = 'Not Started';
}

if(userList.size() > 0)
{
    insert objList;
}
}

```

5)

```

public class wrapper {
    public void callwrapper(){
List<wrapperclass> lst=new List<wrapperclass>();
contact cnn=[select id,firstname from contact limit 1];
for(Account acc_new:[select id,name from Account]){
    wrapperclass obj=new wrapperclass(acc_new);
    lst.add(obj);
}
    wrapperclass obj1=new wrapperclass(cnn);
    lst.add(obj1);
    system.debug(lst.size());
}

public class wrapperclass{
    public string name;
    public wrapperclass(Account x1){
        name=x1.name;
    }
    public wrapperclass(contact x1){
        name=x1.firstname;
    }
}

```

}
 }