Asynchronous Apex

1. Use Future Methods

Account Processor Class

```
public class AccountProcessor {
   @future
   public static void countContacts(List<ID> accIDs)
   {
      List<Account> accList = [Select ID, (Select Id From Contacts) From Account
WHERE ID IN : accIDS];
      for(Account acc : accList)
      {
         acc.Number_of_Contacts__c = acc.Contacts.size();
      }
      if(!accList.isEmpty())
      {
         update accList;
      }
   }

}
```

Account Processor Test Class

```
@isTest
public class AccountProcessorTest {
   @isTest private static void countContacttest()
   {
      Integer i;
      List<Account> accList = new List<Account>();
      for(i=0;i<250;i++)
      {
         accList.add(new Account(Name ='Test'+i));
      }

      insert accList;

      List<Contact> conList = new List<Contact>();
      List<Id> accIDs = new List<ID>();
      for (Account acc : accList)
```

```
        {
            conList.add(new Contact(FirstName ='Test', LastName= acc.Name,
AccountId=acc.ID));
            accIDS.add(acc.Id);
        }

        insert conList;

        Test.startTest();
        AccountProcessor.countContacts(accIDS);
        Test.stopTest();

        List<Account> accs = [Select Id, Number_of_Contacts__c from Account];
        System.assertEquals(1, accs[0].Number_of_Contacts__c);
    }

}
```

2. Use Batch Apex

Lead Processor Class

```
public class LeadProcessor implements Database.Batchable<sObject>,
Database.stateful
{
    public integer recordCount=0;
    public Database.QueryLocator start(Database.BatchableContext bc)
    {
        return Database.getQueryLocator([Select ID, Name From Lead]);
    }

    public void execute(Database.BatchableContext bc, List<Lead> lad)
    {
```

```
        for(Lead ldd : lad)
        {
            ldd.LeadSource ='DreamForce';
        }
        update lad;
        recordCount = recordCount + lad.size();
    }

    public void finish(Database.BatchableContext bc)
    {
        System.debug('total record processed ' + recordCount);
    }

}
```

Lead Processor Test Class

```
@isTest
public class LeadProcessorTest {
    @isTest public static void test1()
    {
        List<Lead> lope = new List<Lead>();
        for(integer i=0;i<200;i++)
        {
            lope.add(new Lead (LastName = 'Test'+i, Company ='Gate'+i,
LeadSource='Web',Status='Closed-Converted'));
        }
        insert lope;
```

```
        Test.startTest();
        LeadProcessor leaping = new LeadProcessor();
        ID batchID = Database.executeBatch(leaping,200);
        Test.stopTest();

        List<Lead> countLead = [Select Id from Lead WHERE
LeadSource='Dreamforce'];
        System.assertEquals(200, countLead.size());
        System.debug(countLead.size());
    }

}
```

3. Control Processes with Queueable Apex

Add Primary Contact Class

```
public class AddPrimaryContact implements Queueable{
    Contact con;
    String state;

    public AddPrimaryContact(Contact con, String state)
    {
        this.con= con;
        this.state= state;
    }
    public void execute(QueueableContext Context)
    {
        List<Account> lstofAccs =[Select ID FROM Account WHERE BillingState =
:state Limit 200];
        List<Contact> lstofConts = new List<Contact>();
        for(Account acc : lstofAccs)
        {
        Contact conInst = con.clone(false,false,false,false);
```

```
        conInst.AccountId = acc.ID;

          lstofConts.add(conInst);

          }
        Insert lstofConts;
    }
}
```

Add Primary Contact Test Class

```
@isTest
public class AddPrimaryContactTest {
    @testSetup static void setup()
    {
        List<Account> lstofAcct = new List<Account>();
        for(Integer i=1;i<100;i++)
        {
            if(i<=50)
                lstofAcct.add(new Account(name='AC'+i, BillingState='NY'));
            else
                lstofAcct.add(new Account(name='AC'+i, BillingState='CA'));
        }
        Insert lstofAcct;
    }

    @isTest static void testAddPrimaryContact(){
    Contact con = new Contact(LastName='TestCont');
        AddPrimaryContact addPCIns = new AddPrimaryContact(con,'CA');

        Test.startTest();
        System.enqueueJob(addPCIns);
        Test.stopTest();

        System.assertEquals(50, [select count() from Contact]);
    }

}
```

4. Schedule Jobs using apex scheduler

Daily Lead Processor Class

```
public class DailyLeadProcessor implements Schedulable{
    public void execute(SchedulableContext ctx)
    {
        List<Lead> leadList = [Select Id, Name FROM Lead WHERE LeadSource = NULL LIMIT 200];
        if(!leadList.isEmpty())
        {
        for(Lead led : leadList)
        {
            led.LeadSource = 'Dreamforce';
        }
        update leadList;
        }

    }

}
```

Daily Lead Processor Test Class

```
@isTest
public class DailyLeadProcessorTest {
    @isTest static void test1()
    {
        List<Lead> leadList = new List<Lead>();
```

```
        for(integer i=0;i<400;i++)
        {
            leadList.add(new Lead(LastName='LeadTest'+i, Company='TestCompany', Status='Open -
Not Contacted'));



        }
        Insert leadlist;


    Test.startTest();
    DailyLeadProcessor d3 = new DailyLeadProcessor();
    String sch = '20 30 8 10 2 ?';
    String jobID = system.schedule('TestingLeadJOb', sch, d3);
    Test.stopTest();


    List<Lead> countLead = [Select Id, LastName From Lead WHERE LeadSource = 'Dreamforce'];
        System.assertEquals(200, countLead.size());
    }
}
```