ASYNCHRONOUS APEX

1. Use Future Methods

Challenge

```
public class AccountProcessor
{
    @future
    public static void countContacts(List<Id> accountIds)
    {
        List<Account> accounts = [SELECT Id, (SELECT Id FROM Contacts) FROM Account
                                WHERE Id IN :accountIds];
        for (Account acc : accounts)
        {
            acc.Number_Of_Contacts__c = acc.Contacts.size();
        }
        update accounts;
    }
}
@isTest
public class AccountProcessorTest
{
    @isTest
    public static void countContactsTest()
    {
        Account a = new Account();
        a.Name = 'Test Account';
        Insert a;

        Contact c = New Contact();
        c.FirstName ='Sreelal';
        c.LastName ='T';
        c.AccountId = a.Id;
```

```
                Insert c;

                List<Id> accountIds = new List<ID>();

                accountIds.add(a.id);

                Test.startTest();

                AccountProcessor.countContacts(accountIds);

                Test.stopTest();

                Account acc = [SELECT Number_of_Contacts__c FROM Account WHERE id = :a.id LIMIT 1];

                System.assertEquals ( Integer.valueOf(acc.Number_Of_Contacts__c) ,1);

        }

}
```

2. Use Batch Apex

Challenge

```
public class LeadProcessor implements Database.Batchable <SObject>

{

    public Database.QueryLocator start(Database.BatchableContext bc)

    {

        String Query='SELECT id,LeadSource FROM Lead';

        return Database.getQueryLocator(Query);

    }

    public void execute(Database.BatchableContext bc, List<Lead> scope)

    {

        for(Lead l: scope)

        {

            l.LeadSource='DreamForce';

        }

        update scope;

    }

    public void finish(Database.BatchableContext bc)

    {

        Id job= bc.getJobId();

         System.debug(job);

    }

}
```

```
@isTest
private class LeadProcessorTest
{
        @isTest
        private static void testBatchClass()
        {
                // Load test data
                List<Lead> leads = new List<Lead>();
                for (Integer i=0; i<200; i++)
                {
                        leads.add(new Lead(LastName='Sreelal', Company='Cognizant'));
                }
                insert leads;

                // Perform the test
                Test.startTest();
                LeadProcessor lp = new LeadProcessor();
                Id batchId = Database.executeBatch(lp, 200);
                Test.stopTest();

                // Check the result
                List<Lead> updatedLeads = [SELECT Id FROM Lead WHERE LeadSource = 'Dreamforce'];
                System.assertEquals(200, updatedLeads.size());
        }
}
```

3. Control Processes with Queueable Apex

Challenge

```
public class AddPrimaryContact implements Queueable

{

        private Contact con;

        private String state;


        public AddPrimaryContact(Contact con, String state)

        {

                this.con = con;

                this.state = state;

        }


        public void execute(QueueableContext context)

        {

                List<Account> accounts  = [SELECT ID, Name, (SELECT id, FirstName, LastName FROM Contacts)
                FROM Account WHERE BillingState = :state LIMIT 200 ];

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

                for (Account acc : accounts)

                {

                        Contact c = con.clone();

                        c.AccountId = acc.Id;

                }


                if (primaryContacts.size() > 0)

                {

                        insert primaryContacts;

                }

        }

}
```

```
@isTest
public class AddPrimaryContactTest
{
        static testmethod void testQueueable()
        {
                List<Account> testAccounts = new List<Account>();
                for (Integer i = 0; i < 50; i++)
                {
                        testAccounts.add(new Account(Name='Account'+i, BillingState='CA'));
                }

                for (Integer j = 0; j < 50; j++)
                {
                        testAccounts.add(new Account(Name='Account'+j, BillingState='NY'));
                }
                insert testAccounts;

        Contact testContact = new Contact(FirstName = 'Sreelal', LastName = 'T');
        insert testContact;

        AddPrimaryContact apc = new AddPrimaryContact(testContact, 'CA');
        Test.startTest();
        System.enqueueJob(apc);
        Test.stopTest();

        System.assertEquals(50, [SELECT count() FROM Contact WHERE AccountId IN (SELECT Id FROM Account
        WHERE BillingState = 'CA')]);
        }
}
```

4. Schedule Jobs Using the Apex Scheduler

Challenge

```
public class DailyLeadProcessor implements Schedulable
{
	public void execute(SchedulableContext ctx)
	{
		List<Lead> leads = [SELECT Id, LeadSource FROM Lead WHERE LeadSource = null LIMIT 200];
		for ( Lead l : leads)
		{
			l.LeadSource = 'Dreamforce';
		}
		update leads;
	}
}


@isTest
private class DailyLeadProcessorTest
{
	private static String CRON_EXP = '0 0 0 ? * * *';
	@isTest
	private static void testSchedulableClass()
	{
		// Load test data
		List<Lead> leads = new List<Lead>();
		for (Integer i=0; i<500; i++)
		{
			if ( i < 250 )
			{
				leads.add(new Lead(LastName='LN', Company='Salesforce'));
			}
```

```
                else
                {
                        leads.add(new Lead(LastName='LN', Company='Salesforce',
                        LeadSource='Other'));
                }
        }
        insert leads;

        // Perform the test
        Test.startTest();
        String jobId = System.schedule('Process Leads', CRON_EXP, new DailyLeadProcessor());
        Test.stopTest();

        // Check the result
        List<Lead> updatedLeads = [SELECT Id, LeadSource FROM Lead WHERE LeadSource =
        'Dreamforce'];
        System.assertEquals(200, updatedLeads.size());
    }
}
```