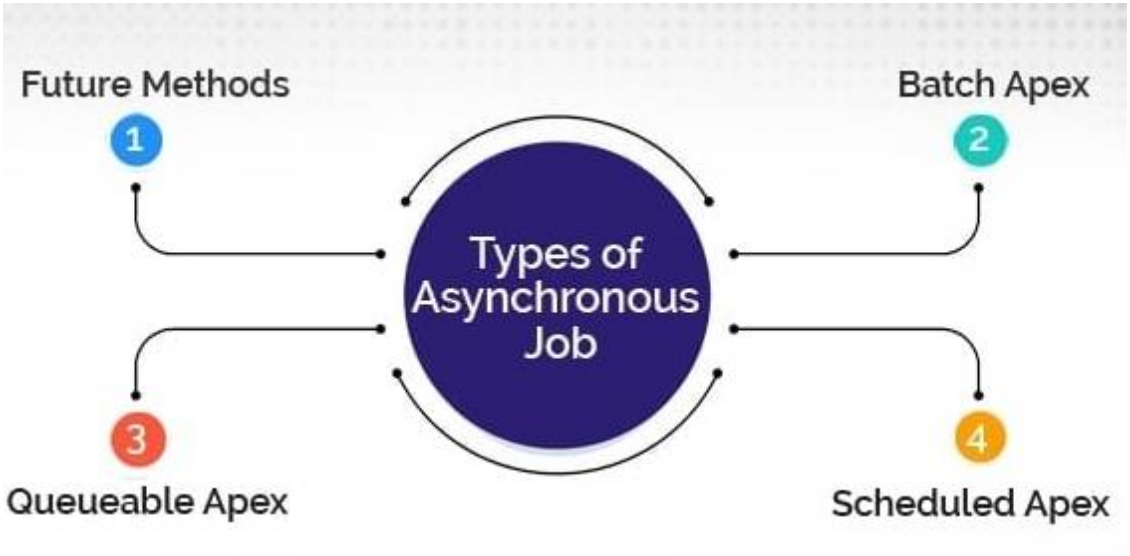
BY SMRITI SHARAN

🚀 YOUR COMPLETE INTERVIEW GUIDE TO MASTERING ASYNCHRONOUS APEX ⚡



BY SMRITI SHARAN

Welcome to sfdcAmplified!

Hi, my name is Smriti Sharan. I am an avid blogger and youtuber. I break down complex concepts with fun relatable real-time examples so that learning is fun. Wow!!!

You Can connect with me on:

LinkedIn

Subscribe to my YouTube Channel

Follow my Blog

BY SMRITI SHARAN

What is Asynchronous Apex?

An asynchronous process is a process or function that executes a task “in the background” without the user having to wait for the task to finish. Asynchronous Apex is used to run processes in a separate thread.

Imagine you're at home, and you have a load of laundry to do. You also want to cook dinner.

If you had to wait for the washing machine to finish before you could start cooking, it would be inefficient. You'd waste a lot of time just waiting around. Instead, what you can do is put your clothes in the washing machine and start the wash cycle. While the machine is doing its job in the background, you can head to the kitchen and start cooking dinner.

BY SMRITI SHARAN

One of the main benefits of running asynchronous Apex is higher governor and execution limits.

Description	Synchronous Limit	Asynchronous Limit
Total number of SOQL queries issued[1]	100	200
Total number of records retrieved by SOQL queries	50,000	50,000
Total number of records retrieved by `Database.getQueryLocator`	10,000	10,000
Total number of SOSL queries issued	20	20
Total number of records retrieved by a single SOSL query	2,000	2,000
Total number of DML statements issued[2]	150	150

What are types of Asynchronous apex?

Asynchronous Apex features their name and use cases are mentioned below:-

Future Methods: it is a basic asynchronous feature, when we make a web call out or when we want to prevent the mixed DML error.

Batch Apex: To do bulk processing of records or for the jobs that require larger query result for example processes like Database maintenance jobs

BY SMRITI SHARAN

Schedule Apex: This feature is used to schedule the invocation of an apex class at a specific time, this can be a recurring event or a one-time task.

Queueable Apex: When one task is dependent on completion of another task we make use of this asynchronous feature, Also job chaining and complex type of jobs are achieved using this feature

BY SMRITI SHARAN

Batch Apex

BY SMRITI SHARAN

Batch class in salesforce is used to run large jobs (think thousands or millions of records!) that would exceed normal processing limits.

To use batch Apex, write an Apex class that implements Database.Batchable interface and make your class global

Implement the following 3 methods

- start()
- execute()
- finish()

The default batch size is 200.

Understanding Methods

Start: This method is called at the starting of a batch job to collect the data on which the batch job will be operating. It breaks the data or record into batches

Syntax:

global void execute(Database.BatchableContext BC, list<sobject<) {}

Execute: This method executes after the Start method, and it does the actual processing for each batch, separately.

BY SMRITI SHARAN

Syntax:

global void execute(Database.BatchableContext BC, list<sobject<) {}

Finish: This method will be called at last. Since this method is called in the end, it is responsible to perform post-processing operations such as sending an email. When this process is called all batches are already executed.

Syntax:

global void finish(Database.BatchableContext BC) {}

Note:

If your code accesses external objects and is used in batch Apex, use Iterable<sObject> instead of Database.QueryLocator.

```
global class MyBatchClass implements Database.Batchable<sObject> {
    global (Database.QueryLocator | Iterable<sObject>)
start(Database.BatchableContext bc) {
        // collect the batches of records or objects to be passed to
execute
    }
    global void execute(Database.BatchableContext bc, List<P>
records){
        // process each batch of records
    }
    global void finish(Database.BatchableContext bc){
        // execute any post-processing operations
```

BY SMRITI SHARAN

```
    }
}
```

Invoking a Batch Class

Database.executeBatch(new BatchApexExample(),100);

Monitoring Batch Apex

To monitor or stop the execution of the batch Apex job, from Setup, enter Apex Jobs in the Quick Find box, then select Apex Jobs.

Batch Apex Chaining

With the onset of API version 29.0 and after, a batch class can be chained to another batch class.

Chaining of batch apex kickstarts the execution of the chained batch class once the execution of the base batch class is finished. At maximum, only 5 Batch jobs can be chained to one another.

```
global database.querylocator start(Database.BatchableContext BC)
{
//start method logic here
}
global void execute(Database.BatchableContext BC, List<sObject> scope)
```

BY SMRITI SHARAN

```
{
//start method logic here

}
global void finish(Database.BatchableContext BC)
{
//Batch Chaining Step Starts here
AccountBatch accBatch = new AccountBatch ();
Id batchProcessId = Database.executeBatch(accBatch);

//finish method logic here
}
```

Why do we use batch apex?

Batch Apex is used to run large jobs (think thousands or millions of records!) that would exceed normal processing limits.

What interface will you use for batch apex?

It is Database.Batchable interface

Why use Batch Apex in Salesforce instead of the normal Apex?

There are various reasons why Batch Apex is better than normal Apex.

- SOQL queries: Normal Apex uses 100 records per cycle to execute SOQL queries. Whereas, Batch Apex does the same in 200 records per cycle.

BY SMRITI SHARAN

- Retrieval of SOQL queries: Normal Apex can retrieve 50,000 SOQL queries but, in Batch Apex, 50,000,000 SOQL queries can be retrieved.
- Heap size: Normal Apex has a heap size of 6 MB; whereas, Batch Apex has a heap size of 12 MB.
- Errors: When executing bulk records, Normal Apex classes are more vulnerable to encountering errors as compared to Batch Apex

Advantages:

- Every transaction starts with a new set of governor limits, making it easier to ensure that your code stays within the governor execution limits.
- If one batch fails to process successfully, all other successful batch transactions aren't rolled back.

How many active batches(running parallel) can be allowed at a time?

Salesforce by default allows 5 active batches running at a time and other batches will be in queue for running

How can we schedule batch class to run in future only once from the specified minutes from current time.

BY SMRITI SHARAN

public static String scheduleBatch(Database.Batchable batchable, String jobName, Integer minutesFromNow)

or

public static String scheduleBatch(Database.Batchable batchable, String jobName, Integer minutesFromNow,Integer batchSize)

How to calculate the batch size if we are making any call out from the batch class execute method?

Basically in salesforce we have limitation of 100 call outs for a transaction. When it comes to batch class each execute method call will be considered as one transaction. So,in this case your batch size must be calculated in such way

Batch size = (No of callouts allowed for single transaction /total number of call outs for each record) – 1;

Batch size = (100 /total number of call outs for each record) – 1;

How can we schedule a batch class to run every 10 or 5 or 2 mins.

Salesforce haven't provided wild card entries (* every,- range,/,?) in place of minutes and seconds.So,if you want schedule a class to run every 10 minutes you have to

BY SMRITI SHARAN

schedule same class 6 times (60min/6) with diff batch name ,for 5 mins 12 times and for 2 mins 30 times..As the time interval keeps going down schedule the same class goes up which is an irritating thing to monitor or even it's very hard to handle these many jobs.

For this we can simply do this in the code so that at any given point of time you will have only schedule job running for the same.

```
global class CaseCreationonSchedular implements Schedulable
{
public void execute(SchedulableContext scon)
{
System.abortJob(scon.getTriggerId()); // abort already running schedule job
Decimal nextInterval = System.Label.nextIntervalTime; // config to decide next interval 2,5 or 10 mins etc..
System.schedule('CaseCreationonBatch – '+String.valueOf(DateTime.now()), '0 '+DateTime.now().addMinutes(Integer.valueOf(nextInterval)).minute()+' */1 ? * *', this); //Schedule the class to run with specified time

Database.executeBatch(new YourBatchClassName()); // Call you batch class
```

BY SMRITI SHARAN

}

}

Why to use Batch class as we already having data loader to process the bulk data.

Agree with this point if and only if the data needs to be updated is static or predefined or which can be done through excel.

We will choose batch class if we have to perform some custom calculations at run time or if you want to run some complex logic which can't be driven by excel sheets in those cases, we have to go with batch classes.

Examples:

Do some relationship quires to update the data.

Make a call out to get some information related to each record.

How many times the execute method will be executed to process the 1234 records.

It depends on your batch size what you have configured at the time of calling the batch class from schedule class.

BY SMRITI SHARAN

Execution method count = Total No Of Records/Batch Size (Any decimal ceil it to upper value)

If you haven't set any batch size then – 1234/200 = 6.17 = 7 times execute method will be called

What is the maximum size of a batch that we can set up ?

2000

What is the minimum batch size we can set up is?

1

What is the default size of batch if we haven't configured at the time of execution?

200

What is Apex Flex Queue?

At a time salesforce allolws 5 batches to be running or to be in queued state.So,if you have consumed all these 5 limits and if system has received one or more batch execution request all these waiting batch will be stored in this Apex Flex Queue.

BY SMRITI SHARAN

What is the maximum number of batch classes allowed in Apex Flex Queue for execution?

100

How can we do chaining of batch jobs?

we can do this by simply calling the chained batch/second batch class from the finish method of the first batch class.

Why to call only from the finish method why not from execute?because the execute method will gets invoked multiple times based on the volume of the records and batch size.So,if you're calling it from execute method then the chaining class will get called multiple times which is not an suggested way of doing.

What is the difference between queryLocator object and Iterable used in batch apex?

QueryLocator object, the governor limit for the total number of records retrieved by SOQL queries is bypassed and you can query up to 50 million records. However, with an Iterable, the governor limit for the total number of records retrieved by SOQL queries is still enforced.

What are the parameters passed in the execute method?

BY SMRITI SHARAN

This method takes the following:

A reference to the Database.BatchableContext object.

A list of sObjects, such as List<sObject>, or a list of parameterized types. If you are using a Database.QueryLocator, use the returned list.

How to invoke batch Class?

To invoke a batch class, simply instantiate it and then call Database.executeBatch with the instance:

You can also optionally pass a second scope parameter to specify the number of records that should be passed into the execute method for each batch

What is the state of batch apex?

Batch Apex is typically stateless. Each execution of a batch Apex job is considered a discrete transaction. For example, a batch Apex job that contains 1,000 records and uses the default batch size is considered five transactions of 200 records each.

What is the use of Database.Stateful?

If you specify Database.Stateful in the class definition, you can maintain state across all transactions. When using Database.Stateful, only instance member variables retain

BY SMRITI SHARAN

their values between transactions. Maintaining state is useful for counting or summarizing records as they're processed. For example, we'll be updating contact records in our batch job and want to keep track of the total records affected so we can include it in the notification email.

When to use batch apex instead of Queueable Apex?

Only you should use Batch Apex if you have more than one batch of records. If you don't have enough records to run more than one batch, you should use Queueable Apex.

How to monitor Batch job?

Go to Setup->Apex Job page.

How to use HTTP Callouts in batch class?

To use HTTP Callouts in batch class we need to use Database.allowcallouts in interface.

How to schedule a batch?

Using schedule apex we can schedule a batch

If a batch is having 200 records and 1 record fails what will happen?

BY SMRITI SHARAN

If any record fails all 200 record will fail but next batch will get executed

Can we call the batch into another batch apex?

Yes, we can call from the finish method.

Can we call batch apex into another batch in execute method?

Only in batch class finish method, We can call another batch class. If you will call another batch class from batch class execute and start method, then Salesforce will throw below runtime error.

System.AsyncException: Database.executeBatch cannot be called from a batch start, batch execute, or future method.

Can we call the batch apex from triggers in salesforce?

Yes, it is possible. We can call a batch apex from trigger but we should always keep in mind that we should not call batch apex from trigger each time as this will exceeds the governor limit this is because of the reason that we can only have 5 apex jobs queued or executing at a time.

Can we call a webservice callout from batch apex?

BY SMRITI SHARAN

To make a Webservice callout in batch Apex, we have to implement Database.AllowsCallouts interface.

How many times start,execute,finish methods will execute in batch apex?

Start method,finish method one time, execute method it depends on requirement. Based on the batch size and data retrieved in Start method.

What is the Batch executions limit per day?

The maximum number of batch executions is 250,000 per 24 hours.

Can we call the future method in batch class?

No,we can't call.

Can I call Queueable from a batch?

Yes, But you're limited to just one System.enqueueJob call per execute in the Database.Batchable class. Salesforce has imposed this limitation to prevent explosive execution.

How to test batch apex?

BY SMRITI SHARAN

Code is run between test.startTest and test.stopTest. Any asynchronous code included within Test.startTest and Test.stopTest is executed synchronously after Test.stopTest.

How many records we can insert while testing batch apex?

We have to make sure that the number of records inserted is less than or equal to the batch size of 200 because test methods can execute only one batch. We must also ensure that the Iterable returned by the start method matches the batch size.

What is apex Flex Queue?

The Apex Flex queue enables you to submit up to 100 batch jobs for execution. Any jobs that are submitted for execution are in holding status and are placed in the Apex Flex queue. Up to 100 batch jobs can be in the holding status.

Can you change order of job in Apex Flex Queue?

Jobs are processed first-in first-out—in the order in which they're submitted. You can look at the current queue order and shuffle the queue, so that you could move an important job to the front, or less important ones to the back.

```
Boolean isSuccess = System.FlexQueue.moveBeforeJob(jobToMoveId,
jobInQueueId);
```

BY SMRITI SHARAN

How many job can run concurrently?

The system can process up to five queued or active jobs simultaneously for each organization

Explain status of jobs in Apex Flex Queue?

Holding : Job has been submitted and is held in the Apex flex queue until system resources become available to queue the job for processing.

Queued : Job is awaiting execution.

Preparing : The start method of the job has been invoked. This status can last a few minutes depending on the size of the batch of records.

Processing: Job is being processed.

Aborted : Job aborted by a user.

Completed : Job completed with or without failures.

Failed : Job experienced a system failure.

Let's say, I have 150 Batch jobs to execute, Will I be able to queue them in one go?

Once you run Database.executeBatch, the Batch jobs will be placed in the Apex flex queue and its status becomes Holding. The Apex flex queue has the maximum number of 100 jobs, Database.executeBatch throws a LimitException and doesn't add the job to the queue. So atmost 100 jobs can be added in one go.

BY SMRITI SHARAN

Also, if Apex flex queue is not enabled, the Job status becomes Queued, Since the concurrent limit of the queued or active batch is 5, so atmost 5 batch jobs can be added in one go.

Can I Use FOR UPDATE in SOQL using Database.QueryLocator?

No, We can't. It will throw an exception stating that "Locking is implied for each batch execution and therefore FOR UPDATE should not be specified"

let's make this really simple!

Imagine you and your friends are coloring pictures. But there's a rule: only one person can color one picture at a time. This is to make sure nobody draws over someone else's art.

BY SMRITI SHARAN

In Salesforce, "records" are like these pictures. When you use "FOR UPDATE," it's like you're saying, "I'm coloring this one, nobody touch it!" — ensuring no one else can color your picture until you're done.

Here's how you'd use "FOR UPDATE" in a SOQL query:

List<Account> accounts = [SELECT Id, Name FROM Account WHERE Name = 'SpecificAccountName' FOR UPDATE];

This query selects all Account records with the Name 'SpecificAccountName' and locks them in the current transaction. This means that if another piece of code (another user or process) tries to modify the 'SpecificAccountName' Account records while your code is running, it will have to wait until your transaction is finished; this ensures that your operations on the record don't conflict with others.

Now, when Salesforce does big jobs (like coloring lots and lots of pictures), called "batch processes," it automatically makes sure that nobody else can color the picture you're working on.

BY SMRITI SHARAN

It's as if there's a teacher watching over everyone, making sure that once you've started coloring, no one else grabs your picture.

So, if you try to say "FOR UPDATE" (or, "nobody touch it!") while already in this big, organized coloring activity, Salesforce is like, "Don't worry! I got it covered! The teacher is already making sure nobody else can take your picture."

That's why you can't use "FOR UPDATE" in these big jobs. It's like double claiming a coloring picture when the teacher already has rules to make sure your picture is safe. Simple!

Can I query related records using Database.QueryLocator?

Yes, You can do subquery for related records, but with a relationship subquery, the

BY SMRITI SHARAN

batch job processing becomes slower. A better strategy is to perform the subquery separately, from within the execute method, which allows the batch job to run faster.

let's break it down even more simply!

Think of "Database.QueryLocator" like being at a fruit market. Your job is to pick apples and then find matching baskets for them.

1. The long way: You pick one apple, then immediately hunt for a basket that fits it. Then you go back for another apple and again search for a new basket. This back-and-forth takes a lot of time because you're focusing on one apple at a time.

The query directly fetches baskets (child records) along with each apple (parent record) using a subquery, which can be slower if there are many records.

'SELECT Id, Name, (SELECT Id, Name FROM Baskets__r) FROM Apple__c'

BY SMRITI SHARAN

2. The quick way: First, you pick all the apples you need. After you've got your apples, you then start finding baskets for all of them at once. This way, you're not running around as much, and you finish your job faster.

This approach first fetches all apples (parent records) and then fetches their related baskets (child records) in the execute method, which is more efficient in batch processing.

// This gets all the apples

'SELECT Id, Name FROM Apple__c'

// Then, for the apples we have, this gets all their baskets

'SELECT Id, Name, Apple__c FROM Basket__c WHERE Apple__c IN :apples'

BY SMRITI SHARAN

So, with Salesforce's "Database.QueryLocator," you can choose to collect your main items (apples) and their related items (baskets) all at once, but it's slower. The smarter strategy is to gather all your main items first, then look for the related items. It saves time and energy!

Can you write a batch class blueprint?

```
global class batchExample implements Database.Batchable<sObject> {
global (Database.QueryLocator | Iterable<sObject>)
start(Database.BatchableContext bc) {

// collect the batches of records or objects to be passed to execute
}

global void execute(Database.BatchableContext bc, List<sObject>
records){
// process each batch of records
}

global void finish(Database.BatchableContext bc){
// execute any post-processing operations
}

}
```

Difference between query locator and iteratable?

Iterable and querylocator are both used for apex batches.

QueryLocator can be used when your data that needs to be executed in batch can be fetched using query. And it has limit of 5million reccords.

BY SMRITI SHARAN

Wherein When records cannot be filtered by SOQL and scope is based on some cusotm business logic, then we go for iterable. And it has limit of 50K records

Let's say Record A has to be processed before Record B, but Record B came in the first Batch and Record A came in the second batch. The batch picks records which are unprocessed every time it runs. How will you control the processing Order?

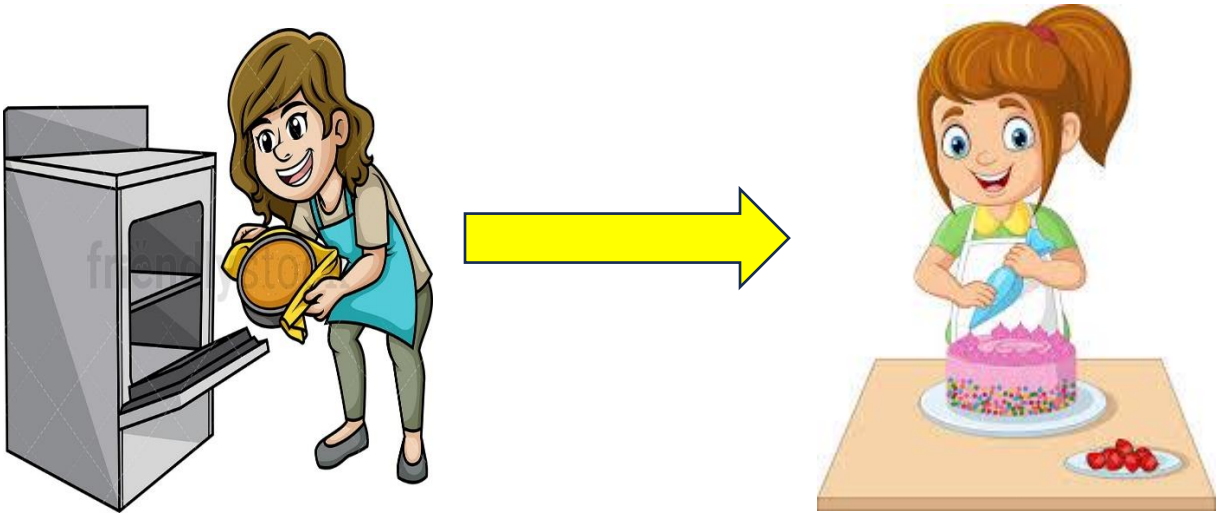
The Processing order can't be controlled, but we can bypass the record B processing before Record A. We can implement Database.STATEFUL and use one class variable to track whether Record A has processed or not. If not processed and Record B has come, don't process Record B. After all the execution completes, Record A has already been processed so Run the batch again, to process Record B.

Let's understand in an easier way!

Imagine you're baking a cake (Record A) and frosting it (Record B). But you can't frost the cake until it's baked. Now, suppose you accidentally try to frost the cake before baking it. We need a system to ensure things happen in the right order!

STEP 1: BAKE THE CAKE **STEP 2: FROZE THE CAKE**

BY SMRITI SHARAN



Here's how you can manage it, like Salesforce's `Database.STATEFUL`:

1. You have a checklist that remembers your progress (using a variable in the code).

2. If you try to frost the cake before baking, you check the list, see it's not ready, and wait.

3. Once you bake the cake, you mark it on your checklist.

4. Now, you can frost the cake because your checklist confirms that the cake is baked.

Pseudo Code:

```
// Our 'checklist' to track if the cake is baked
```

BY SMRITI SHARAN

```
    Boolean isCakeBaked = false;
    public void execute(Database.BatchableContext BC, List<sObject>
scope) {
        for (sObject item : scope) {
            // Trying to frost the cake
            if (item == RecordB) {
                // The cake isn't baked yet, can't frost!
                if (!isCakeBaked) {
                    continue; // Skip, don't frost now.
                }
            }
            // Baking the cake
            else if (item == RecordA) {
                bakeTheCake();
          // Cake is baked, checklist updated!
                isCakeBaked = true.
            }
        }
    }
```

In this simplified scenario, `Database.STATEFUL` ensures you don't frost the cake before it's baked 🎂☑

If you didn't use `Database.Stateful` in your batch class while needing to remember whether "the cake is baked" between different batches (or transactions), you'd face a problem: the system wouldn't remember the state of your "cake" from one batch of records to the next.

1.No Memory: Without `Database.Stateful`, each batch (group of records) processed by your batch Apex class is independent. It's like each time you start working with a new

BY SMRITI SHARAN

group of records, you forget what you did with the previous group. In our scenario, it's as if every time you go to check on a cake, you forget whether or not you've already baked it.

2. Can't Track Progress: Since you can't remember if the cake was baked in a previous batch, you might try to frost a cake that isn't baked. Your process gets confused because it doesn't know what's already happened.

3. Inconsistent Results: This leads to unpredictable outcomes. Some cakes might get frosted without being baked, or some might never get frosted at all because the system thought they weren't baked yet.

Here's what that problem looks like:

```
// This variable WON'T keep its value between batches without
Database.Stateful
    Boolean isCakeBaked = false;

    public void execute(Database.BatchableContext BC, List<sObject>
scope) {
        for (sObject item : scope) {
            // Trying to frost the cake
            if (item == RecordB) {
                // We don't know if the cake is baked!.
                // Might frost an unbaked cake or skip a baked one.
            }
            // Baking the cake
            else if (item == RecordA) {
                bakeTheCake();
```

BY SMRITI SHARAN

```
                isCakeBaked = true; // We'll forget this immediately
in the next batch.
            }
        }
    }
```

Without `Database.Stateful`, the `isCakeBaked` variable resets to its default value (false) in each new batch, so you can't properly track whether the cake has been baked before trying to frost it. It leads to chaos ! 🎂🙈 Using `Database.Stateful` allows the class to remember the state (like whether a cake is baked) across multiple batches of records, ensuring each cake is perfectly baked and frosted. Yay!!!

Let’s say, we have run an apex batch to process 1000 records, and It is running with batch size 200. Now, while doing DML on 395th record, an error occurred, What will happen in that case?

In batches, If the first transaction succeeds but the second fails, the database updates made in the first transaction are not rolled back.
Since the batch size is 200, so the first batch will be processed completely, and all data will be committed to DB. In seconds batch, if we are committing records using normal DML statements like insert, update than the whole batch will be rollbacked. So records 201 to 400 will not be processed.

BY SMRITI SHARAN

Q. How can you stop a Batch job?

The Database.executeBatch and System.scheduleBatch method returns an ID that can be used in System.abortJob method.

Q. Write batch apex program to get phone no. in the Contact object with the phone no. in the Corresponding Account object. Where Contact is a child of Account.

```
global class ContactUpdate implements Database.Batchable<sObject>
{
global Database.QueryLocator start(Database.BatchableContaxt bc)
{
String query = 'select id, phone, Account.phone from Contact';

return Database.getQueryLoactor(query);
}
global void execute(Database.BatchableContext bc, List<Contact> scope)
{
for(contact con: scope)
{
con.phone = con.Account.phone;
Conlist.add(con);
}
update conlist;
}
global void finish(Database.Batchable Context bc)
{

}
}
```

Understand Code Line by Line

BY SMRITI SHARAN

global class ContactUpdate implements Database.Batchable<sObject>

This line declares a global class named ContactUpdate that implements the Database.Batchable<sObject> interface, allowing it to execute batch jobs, which are capable of processing multiple records.

sObject is a generic object type that this batch job is designed to process. sObject is a generic object type that this batch job is designed to process. In this context, it's going to be `Contact` records.

global Database.QueryLocator start(Database.BatchableContext bc)

- The `start` method is the first stage of the batch process. It's responsible for collecting the records that need to be processed by the batch job. This method returns a 'Database.QueryLocator' object. The 'Database.BatchableContext' parameter is a way for the platform to pass in runtime information about the batch job, like the job's ID.

String query = 'select id, phone, Account.phone from Contact';

his block of code inside the start method is creating a SOQL query string that fetches certain fields from Contact records.

return Database.getQueryLocator(query);

BY SMRITI SHARAN

This line executes the SOQL query we stored in the query variable and returns a Database.QueryLocator object. This object is used by Salesforce to handle and divide the queried records into different batches for processing.

global void execute(Database.BatchableContext bc, List<Contact> scope)

The execute method is called for each batch of records. The records that were collected in the start method are now processed in groups referred to as "batches." The scope parameter represents a list of Contact records that are being processed in the current batch.

for(Contact con : scope)

This line starts a loop that goes through each Contact record in the current batch

con.phone = con.Account.phone;

For each `Contact` in our loop, this line updates the `Contact`'s phone field to be the same as the phone field of its related `Account`.

Conlist.add(con);

- This line would add the modified `Contact` record to a list named Conlist.

update conlist;

BY SMRITI SHARAN

This line updates the database with changes made to the conlist.

global void finish(Database.BatchableContext bc)

The finish method is the last stage of the batch process. It's used for any post-processing operations, like sending a notification email when the batch job is done. It accepts a `Database.BatchableContext` parameter, which, like in the `start` method, contains information about the runtime context of the batch job.

Create an Apex Class to invoke the batch apex job?

```
public class TestMyBatch
{
public string cname{get; set;}
public pageReference show()
{
customerBatch mybatch = new customerBatch (cname);
Id id = Database.executeBatch (mybatch, 400);
system.debug ('My Job id' + id);
}
}
```

Id id = Database.executeBatch(mybatch, 400)

This line calls the static method executeBatch of Salesforce's Database class, which starts the execution of the batch class instance mybatch. The number 400 specifies the batch size, i.e., how many records the batch process should handle simultaneously. The

BY SMRITI SHARAN

method returns the job ID of the batch process, which is a unique identifier for the batch job in the system.

system.debug('My Job id' + id)

This message will show the job ID of the batch job that was started.

Call we call future method in batch class?

Methods declared as the future aren't allowed in the classes that implement Database.Batchable interface.

Methods declared as future can't be called from Batch Apex class.

Batch class and future method are designed for a different kind of task. Batch Apex is like a heavy-duty used for big tasks (processing lots of data), while future methods are more like a screwdriver, good for smaller, quick fixes (simple, one-time operations- a method). They work differently, so they can't always be used together.

How to use Aggregate queries in Batch Apex

Aggregate queries don't work in Batch Apex because aggregate queries doesn't support queryMore(). They run into the error '**Aggregate query does not support queryMore(), use LIMIT to restrict the results to a single batch'**

BY SMRITI SHARAN

Let’s Understand in simpler way!

Standard Query:

Imagine you have a machine that can give you candies. You say, "Give me candies," and it gives you 200.

String query = 'SELECT Id, Name FROM Candy';

Aggregate Query (Problematic):

Now, you want a special count of candies grouped by color (an aggregate query). You say, "Give me the count of candies by color," and it tries to give you ALL of them at once.

String aggQuery = 'SELECT Color, COUNT(Id) FROM Candy GROUP BY Color'

BY SMRITI SHARAN

Why It's a Problem: **Batch Apex is designed to process data in smaller groups**, like eating candies a few at a time. But aggregate queries don't work that way. They **want to give you ALL the counts at once**, not just a few at a time.

Way to fix this error:

1. Create an Apex class implements Iterator<AggregateResult>.
2. Create an Apex class implements Iterable<AggregateResult>.
3. Implementing to Database.Batchable<AggregateResult>, and Using Iterable at start execution in Batch Apex.

What is the difference between database.batchable & database.batchablecontext bc?

Database.Batchable (Interface):

- Think of this as a blueprint that your Apex class needs to follow.
- When you implement Database.Batchable, you're telling Salesforce that your class is ready to be used for batch processing.
- It defines the methods your class must have, like start(), execute(), and finish().

Database.BatchableContext (Context Variable):

- It stores runtime information about the batch job, like the job ID and other details.

BY SMRITI SHARAN

- You can use this to access and work with information related to the current batch job.

Which platform event can fire from batch ?

The BatchApexErrorEvent object represents a platform event associated with a batch Apex class.It is possible to fire platform events from batch apex. So whenever any error or exception occurs, you can fire platform events which can be handled by different subscriber. Batch class needs to implement “Database.RaisesPlatformEvents” interface in order to fire platform event.

```
global class SK_AccountProcessBatch implements
Database.Batchable<sObject>,Database.RaisesPlatformEvents{
//batch logic
}
```

What if you change the name of Execute method to Execute1 in the batch class? Will the batch job still run?

Go ahead and change Execute to Excute1 and try saving the class.

Output

Class batchUpdateAccountsContacts must implement the method: void Database.Batchable<SObject>.execute(Database.BatchableContext, List<SObject>)

BY SMRITI SHARAN

Finding

It won let you save the batch class as it says class must implement execute method.

Is there a way in which I can call a future method from a batch Job?

Calling a future method is not allowed in the Execute method, but a web service can be called. A web service can also call an @future method. So, we can define a web service having a future method invocation and call the web service from the execute method of Batch Job.

what is batchable context? why we need it?

1. database.batchable is an interface.

2. database.batchableContext is a context variable which store the runtime information eg jobId

It gives the context of the batch class. Represents the parameter type of a batch job method and contains the batch job ID.

What is use of batch job Id which we get from database.batchableContext?

It gets information about the job

```
AsyncApexJob jobInfo = [SELECT Status, NumberOfErrors
FROM AsyncApexJob WHERE Id = :jobID];
```

BY SMRITI SHARAN

How to know it is batch job?

We have to create an global apex class which extends Database.Batchable Interface because of which the salesforce compiler will know.

Can I write method other than start, execute, finish?

No we cannot write other method. Whatever method is there is Database.batchable interface only that we can call.

Can we keep start, finish, execute method?

As the class implements batchable interface, we need to define the methods.

Difference between query locator and iteratable?

Iterable return types have no hard maximum scope size, unlike QueryLocator, which will not allow more than 2,000 records per execute call.

If logic is complex and not filtered through query then we will use iteratable.

Why batch class is global?

global makes a class usable by code outside of a managed package.

How to execute batch job?

batchAccountUpdate b = new batchAccountUpdate();

database.executeBatch(b);

BY SMRITI SHARAN

What is database.stateful?

It is an interface which maintains state of variables.

eg- send failure of number of records failed to salesforce admin over email

insert vs database.insert?

If we use the DML statement (insert), then in bulk operation if error occurs, the execution will stop and Apex code throws an error which can be handled in try catch block.

If DML database methods (Database.insert) used, then if error occurs the remaining records will be inserted / updated means partial DML operation will be done.

database.insert vs Stateful?

Database.insert – It show failure for records where complete batch is failed and wont pick the records where some records are passed for a particular batch and therefore we need to use database.stateful

What we can do from finish method?

We can do post processing logic

Is finish asynchronous?

Finish method is sync because it does not go in queue. Only execute method is async.

Let’s break down and simplify further!

BY SMRITI SHARAN

Execute method: The execute method is asynchronous. It processes records in batches and may make use of the Salesforce job queue. Records are processed in the background, and the batch job is subject to Salesforce's asynchronous processing limits and queuing mechanism.

Finish method: The finish method, on the other hand, is synchronous. It doesn't go into a queue, and it runs immediately after the execute method has completed processing all records.

What we do to do callout from batch?

1. Use Database.allowcallout interface on class level
2. Do callout from execute method.

Can we do callout from start method?

We can do from start, but it will hit the limit so best is to do from execute. The execute method processes records in batches, which means you can make callouts for each batch of records, allowing you to stay within Salesforce's governor limits.

Additionally, if a callout fails in the execute method, you can implement error handling and retry logic for that specific batch of records, which is more challenging to do in the start method.

BY SMRITI SHARAN

Can we call queuable from batch?

We can call because it is a job and can be queued. It can be called from finish.

Can I call a batch from another batch?

We can call from finish.

Can we get records without querying a batch?

We will do using iterator

from Batch A we will call batch B

In Batch B constructor will pass success records

In this case we dont need to query but we will use iterator

BATCH APEX SCENARIOS

What is the upper limit of scope parameter if batch class returns iterable?

If the start method of the batch class returns an iterable, the scope parameter value has no upper limit. However, if you use a high number, you can run into other limits.

Count the “Customer – Direct” account records processed by the batch class?

Batch Apex is stateless by default. That means for each execution of your execute method, you receive a fresh copy of your object. All fields of the class are initialized,

BY SMRITI SHARAN

static and instance. If your batch process needs information that is shared across transactions, one approach is to make the Batch Apex class itself stateful by implementing the Database.Stateful interface.

```
global class AccountBatchApex implements Database.Batchable<sObject>,
Database.Stateful{

    global integer numberofDirectCustomers = 0;

    global Database.QueryLocator start(Database.BatchableContext bc){

        String soqlQuery = 'SELECT Name, AccountNumber, Type From
Account';

        return Database.getQueryLocator(soqlQuery);

    }



    global void execute(Database.BatchableContext bc, List<Account>
scope){



        for (Account acc : scope){

            if(acc.Type.equals('Customer - Direct')){

                numberofDirectCustomers++;

            }

        }

    }
```

BY SMRITI SHARAN

```
    global void finish(Database.BatchableContext bc){

    }

}
```

Give example of Batch Apex class for deleting records?

```
public class BatchDelete implements Database.Batchable<sObject> {
       public String query;
                  public Database.QueryLocator
start(Database.BatchableContext BC){
       return Database.getQueryLocator(query);
       }
       public void execute(Database.BatchableContext BC, List<sObject>
scope){
       delete scope;
       DataBase.emptyRecycleBin(scope);
       }

       public void finish(Database.BatchableContext BC){
       }
       }
```

Update account description, number of employees, contact last name using batch apex. Get the failure record ids in the email. Also schedule the job for every Monday ?

BY SMRITI SHARAN

```
global class batchUpdateAccountsContacts implements Database.Batchable
<sObject>,Database.Stateful,Schedulable {
    global batchUpdateAccountsContacts(){
    }
    Set<id> successRecord = new Set<id>();
    Set<id> failRecord = new Set<id>();

 global Database.QueryLocator start(Database.BatchableContext info){
        String SOQL='Select id,name,NumberOfEmployees,
description,(select id, name from contacts) from Account';
       return Database.getQueryLocator(SOQL);
   }
   global void execute(Database.BatchableContext info, List<Account>
scope){
       List<Account> accsToUpdate = new List<Account>();
       List<Contact> cUpdate = new List<Contact>();
       for(Account a : scope)
          {
           a.description ='Test';
           a.NumberOfEmployees = 70;
           accsToUpdate.add(a);
           for (Contact c:a.contacts){
               c.lastname = 'test+a';
               cUpdate.add(c);
           }
       }
       Database.SaveResult[] srList = Database.update(accsToUpdate,
false);
       Database.SaveResult[] srList1 = Database.update(cUpdate,
false);


       for (Database.SaveResult sr : srList) {
            if (sr.isSuccess()) {
                // Operation was successful, so get the ID of the
record that was processed
                successRecord.add(sr.getId());
            }
```

BY SMRITI SHARAN

```
    else {
        for(Database.Error err : sr.getErrors()) {
        }
            failRecord.add(sr.getId());
    }
  }

        for (Database.SaveResult sr : srList1) {
            if (sr.isSuccess()) {
                successRecord.add(sr.getId());
            }
    else {
        for(Database.Error err : sr.getErrors()) {
        }
          failRecord.add(sr.getId());
    }
  }


   }
   global void finish(Database.BatchableContext info){
   // Get the ID of the AsyncApexJob representing this batch job
   // from Database.BatchableContext.
   // Query the AsyncApexJob object to retrieve the current job's
information.
   AsyncApexJob a = [SELECT Id, Status, NumberOfErrors,
JobItemsProcessed,
      TotalJobItems, CreatedBy.Email FROM AsyncApexJob WHERE Id =
:info.getJobId()];

   // Send an email to the Apex job's submitter notifying of job
completion.
   Messaging.SingleEmailMessage mail = new
Messaging.SingleEmailMessage();

   String[] toAddresses = new String[] {a.CreatedBy.Email};
   mail.setToAddresses(toAddresses);
   mail.setSubject('Account and contact update' + a.Status);
   mail.setPlainTextBody
```

BY SMRITI SHARAN

```
   ('The batch Apex job processed ' + a.TotalJobItems +
   ' batches with '+ a.NumberOfErrors + '
failures.'+successRecord+'successRecordids: '+ 'failRecordids: '+
failRecord);
   Messaging.sendEmail(new Messaging.SingleEmailMessage[] { mail });
   }


    global void execute(SchedulableContext SC){
        database.executeBatch(new batchUpdateAccountsContacts(),100);
       //for cron expression
        // String cronexpression = '0 0 0 ? * * *'
       // System.schedule('Testing', cronexpression, testobj);
    }
}
```

Explain Batch Apex With Webservice Callout

```
global class AccountBatchApex implements Database.Batchable<sObject>,
Database.AllowsCallouts{

    global Database.QueryLocator start(Database.BatchableContext bc){
        String soqlQuery = 'SELECT Name, AccountNumber, Type From
Account';
        return Database.getQueryLocator(soqlQuery);
    }

    global void execute(Database.BatchableContext bc, List<Account>
scope){

        for (Account acc : scope){
            if(acc.Type.equals('Customer - Direct')){
                try{
                    HttpRequest request = new HttpRequest();
                    HttpResponse response = new HttpResponse();
                    Http http = new Http();
```

BY SMRITI SHARAN

```
                    String username = 'YourUsername';
                    String password = 'YourPassword';
                    Blob headerValue = Blob.valueOf(username + ':' +
password);
                    String authorizationHeader = 'BASIC ' +
EncodingUtil.base64Encode(headerValue);

                    request.setHeader('Authorization',
authorizationHeader);
                    request.setHeader('Content-Type',
'application/json');
                    request.setEndpoint('Your Endpoint URL');
                    request.setMethod('POST');
                    request.setBody('Information to Send');
                    response = http.send(request);
                    if (response.getStatusCode() == 200) {
                        String jsonResponse = response.getBody();
                        System.debug('Response-' + jsonResponse);
                    }
                }
                catch(Exception){
                    System.debug('Error-' + e.getMessage());
                }
            }
        }
    }

    global void finish(Database.BatchableContext bc){

    }
}
```

Create three account records, one with missing required information. Once the code is executed then the two records are saved to the database. The one record which is not saved should print error ?

BY SMRITI SHARAN

```
Account[] accts = new List<Account>{
    new Account(Name='Account1'),
    new Account(),
    new Account(Name='Account3')
        };
   Database.SaveResult[] sr = Database.insert(accts, false);


       for (Database.SaveResult sr : sr) {
            if (sr.isSuccess()) {
                // Operation was successful, so get the ID of the
record that was processed
                System.debug('Successfully inserted account. Account
ID: ' + sr.getId());
            }

    else {
        for(Database.Error err : sr.getErrors()) {
            system.debug('err'+err);
            System.debug('The following error has occurred.');
            System.debug(err.getStatusCode() + ': ' +
err.getMessage());
            System.debug('Account fields that affected this error: ' +
err.getFields());
        }
    }
```

Write a test class for batch apex for inserting account and related contacts?

```
@isTest
private class testbatchUpdateAccountContacts {
    @testSetup
    static void setup() {
```

BY SMRITI SHARAN

```
        List<Account> accounts = new List<Account>();
        List<Contact> contacts = new List<Contact>();

        // insert 200 accounts
        for (Integer i=0;i<200;i++) {

            accounts.add(new Account(name='Account '+i,
                description ='Test', NumberOfEmployees=70));
        }
        insert accounts;
        // find the account just inserted. add contact for each
        for (Account account : [select id from account]) {
            contacts.add(new Contact(firstname='first',
                lastname='Test', accountId=account.id));
        }
        insert contacts;
    }
        static testmethod void test() {
        Test.startTest();
        batchUpdateAccountsContacts tb = new
batchUpdateAccountsContacts();
        Id batchId = Database.executeBatch(tb);
        Test.stopTest();
            Integer con=[select count() from contact];
        // after the testing stops, assert records were inserted
properly
        System.assertEquals(200, [select count() from contact]);
        System.assertEquals(200, [select count() from account]);
        }
}
```

Explaining code line by line

@isTest annotation

It is used to define classes and methods that only contain code used for testing our application.

BY SMRITI SHARAN

```
@isTest
```

Advantage of writing a Test Class

The advantage of creating a separate class for testing is that classes defined with is Test don’t count against our organization limit of 6 MB for all Apex code.

```
private class testbatchUpdateAccountContacts {}
```

@testSetup annotation

Methods that are annotated with @testSetup are used to create test records once and then access them in every test method in the test class. I have created a Test setup method so as to create common test data so that I do not need to re-create records for each test method.

```
@testSetup
```

In the below method I create 200 account and related contact records and then insert them in the list (accounts) and (contacts) respectively.

BY SMRITI SHARAN

```
static void setup() {
        List<Account> accounts = new List<Account>();
        List<Contact> contacts = new List<Contact>();

        // insert 200 accounts
        for (Integer i=0;i<200;i++) {

            accounts.add(new Account(name='Account '+i,
                description ='Test', NumberOfEmployees=70));
        }
        insert accounts;
        // find the account just inserted. add contact for each
        for (Account account : [select id from account]) {
            contacts.add(new Contact(firstname='first',
                lastname='Test', accountId=account.id));
        }
        insert contacts;
    }
```

Test Method Name

A test method is defined as static testMethod void testMethodName()

```
        static testmethod void test() {}
```

Here I have executed the batch class between Test.startTest and

Test.stopTest so as to

BY SMRITI SHARAN

ensure that any asynchronous transactions finish executing before Test.stopTest() exits.

startTest()

Marks the point in our test code when our test actually begins. Each test method is allowed to call this method only once.

```
Test.startTest();
```

stopTest()

Marks the point in our test code when our test ends. Each test method is allowed to call this method only once. Any code that executes after the stopTest method is assigned the original limits that were in effect before startTest was called.

```
Test.stopTest();
```

Database.ExecuteBatch

BY SMRITI SHARAN

Create instance of the batch class and execute the class by writing Database.executeBatch and pass instance of the class inside it.

```
batchUpdateAccountsContacts tb = new batchUpdateAccountsContacts();
        Id batchId = Database.executeBatch(tb);
```

System.AssertEquals

It accepts three parameters; the first two are the variables that will be tested for equality or inequality, and the third (optional) one is used to display a message in case the condition fails.

Here I am checking the count of contact and account are equal to 200 by using system.assertEquals()

```
System.assertEquals(200, [select count() from contact]);
 System.assertEquals(200, [select count() from account]);
```

Best Practices

- We should always test batch class with good amount of data (minimum 200 records) in our test class.

BY SMRITI SHARAN

- We must execute the batch class between Test.StartTest() and Test.StopTest().

BY SMRITI SHARAN

Future

Method

BY SMRITI SHARAN

Future method a set of codes that runs in the background. We can take them.

Example:

brushTeeth(){}

eatBreakfast(){}

clothesInLaundry(){}

studySalesforce(){}

BY SMRITI SHARAN

If we want any method to run in future, put @future annotation. This way Salesforce understands that this method needs to run asynchronously.

```
@future
public static void clothesInLaundry(){
}
```

Void – As it runs in future, it does not return any value to you currently.

The specified parameters must be primitive data types, arrays of primitive data types, or collections of primitive data types; future methods can't take objects as arguments.

Let's understand in depth.

Primitive Values:

- Primitive data types, such as `Integer`, `Boolean`, or `String`, represent simple values with no internal complexity.

BY SMRITI SHARAN

- These values are typically immutable, meaning they can't change after they are assigned. For example, once you set an Integer to 5, it will always be 5.

- Because of their immutability and simplicity, Salesforce can reliably pass primitive values to a future method without concerns about changes occurring between the time of invocation and execution.

Objects and Complexity:

- Objects in Salesforce can be complex structures with various fields and relationships.

- An object's data can change at any time due to user interactions or other processes. For example, a record's fields can be updated, related records can be created or deleted, and triggers can fire, all of which might modify the object's state.

- When you pass an object to a future method, it's possible that the object's data may change before the future method runs. This can lead to

BY SMRITI SHARAN

unexpected behavior or inconsistencies if the future method relies on specific data within the object.

In essence, allowing objects as arguments in future methods would introduce potential data integrity and consistency issues**. By restricting future methods to primitive data types, Salesforce ensures that the data passed to future methods is stable and won't change unexpectedly** between the time of invocation and execution.

How to call future method?

ClassName.clothesInLaundry();

Scenarios when we use future method?

Scenario 1: Whenever we want our code to be run in background.

Scenario 2: You must have faced DML exception. This exception occurs when we insert setup and non-setup objects in one transaction.

Setup: User, Group, Profile,Layout, Email Template

BY SMRITI SHARAN

Non Setup: All other standard and custom objects (Account, Custom Objects)

In same context, we cannot insert User/Profile and Account in same object. Future method helps in this situation.

Scenario 3: We cannot perform callout from trigger. Use future method to put callout in future method to make a callout.

What are Limitations of Future method?

1. You cannot call one future method from another method
2. Can have only primate as parameters
3. Order of Invocation is not respected
4. Cannot invoke future method from another future method

Why do we use Future Methods?

You can call a future method for executing long-running operations, such as callouts to external Web services or any operation you'd like to run in its own thread, on its own time.

BY SMRITI SHARAN

Can you write the syntax of a future method?

Methods with the future annotation must be static methods and can only return a void type.

```
global class FutureClass
{
@future
public static void myFutureMethod()
{
// Perform some operations
}
}
```

What kinds of parameters are supported in Future methods?

You can pass primitive data types, arrays of primitive data types, or collections of primitive data types as parameters. *No sObjects OR objects as arguments can be passed.*

Why sobject type parameters are not supported in Future methods?

The reason why sObjects can’t be passed as arguments to future methods is that the sObject might change between the time you call the method and

BY SMRITI SHARAN

the time it executes. In this case, the future method will get the old sObject values and might overwrite them.

What could be the workaround for sobject types?

To work with sObjects, pass the sObject ID instead (or collection of IDs) and use the ID to perform a query for the most up-to-date record.

```
global class FutureMethodRecordProcessing
{
@future
public static void processRecords(List<ID> recordIds)
{
// Get those records based on the IDs
// Process records
}
}
```

How can I perform Callouts from Future methods?

We need to add a parameter **callout=true** in **@future.**

```
global class FutureMethodExample
{
@future(callout=true)
public static void doCallouts(String name)
{
```

BY SMRITI SHARAN

```
// Perform a callout to an external service
}
}
```

How Can I call a future method?

You can invoke future methods the same way you invoke any other method.

```
FutureMethodExample.doCallouts('Account');
```

Can I write a future call in Trigger?

Yes, you can.

Consider a case, I have Written a future call in the Account's trigger update operation. and I have a batch job running on Account records and does DML on them. Will the future call be invoked after DML?

The Problem:

When the batch job updates the Account records, it triggers the Account update operation, which means your trigger also fires.

BY SMRITI SHARAN

Both the batch job and the trigger run in the same "execution context" or "transaction."

Now, here's the issue: Salesforce doesn't allow you to call a future method from within another future method or a batch method. It's like a rule they've set to avoid potential conflicts and infinite loops.

What Happens:

So, when the trigger, which includes a future method, is called because of the batch job's updates, it tries to invoke that future method.

Salesforce detects this and throws an exception with a message that says, "Future method cannot be called from a future or batch method."

The Result:

Your process breaks, and the Account updates don't happen as expected because of this exception.

Solutions:

BY SMRITI SHARAN

- To avoid this issue, you might consider:
- Refactoring your code to remove the need for a future method in the trigger.
- Implementing the logic from the future method directly in your batch job, if possible.
- Exploring alternatives like using queueable Apex or different trigger contexts, depending on your specific requirements.

How can avoid this Exception condition, without using try-catch?

We can update the trigger logic to leverage the *System.isFuture()* and *System.isBatch()* calls so that the future method invocation is not made if the current execution context is future or batch.

In Any case, I can't call a future method from a batch Job?

Calling a future method is not allowed in the Execute method, but a web service can be called. A web service can also call an @future method.

Sample Code

BY SMRITI SHARAN

Webservice Class containing future method.

```
public class MyWebService {
    @future
    public static void myFutureMethod(String param) {
        // Your future method logic here
    }

    webservice static void myWebServiceMethod(String param) {
        // Call the future method from the web service
        myFutureMethod(param);
    }
```

Within BatchClass execute method webservice class with webservice

method (future method) is called

```
public class MyBatchClass implements Database.Batchable<sObject> {
    public Database.QueryLocator start(Database.BatchableContext
context) {
        // Query for records to process
        return Database.getQueryLocator([SELECT Id, Name FROM
MyObject__c WHERE ...]);
    }

    public void execute(Database.BatchableContext context, List<sObject>
scope) {
        // Process your records here

        // Call the web service method, which indirectly invokes the @future
method
```

BY SMRITI SHARAN

```
        String param = 'SomeValue';
        MyWebService.myWebServiceMethod(param);
    }

    public void finish(Database.BatchableContext context) {
        // Batch finish logic
    }
}
```

How Many Future methods can be defined in a Class?

Any number of. There are no restrictions as such.

Take the below case,

```
global class FutureMethodExample
{
@future(callout=true)
public static void doCallouts(String name)
{
// Perform a callout to an external service
doCallouts2(name);}

@future
public static void doCallouts2(String name)
{
// Perform a callout to an external service
```

BY SMRITI SHARAN

```
}}
```

How many future calls will the above code invoke?

This code is an invalid code as *a future method can't invoke another future method.*

If I want to call a future method from a future method, what could be the solution?

Workaround could be calling **a web service** that has future invocation.

What are the other use cases of using a future method?

We can also use future methods to isolate DML operations on different sObject types to prevent the **mixed DML** error.

What is a Mixed DML error?

There are 2 kinds of sObjects in salesforce.

1. Non-Setup: Account, opportunity, etc
2. Setup: User, groups, etc

BY SMRITI SHARAN

In a single transaction Or execution context, If you are performing DML on both kinds, the system doesn't allow it and throws an exception called Mixed DML exception, stating that a transaction cannot have Mixture of DML operation(Setup and Non-Setup)

How Future method helps in avoiding Mixed DML errors?

We can shift DML operations of a particular kind in the Future scope. Since both the DML operations are isolated from each other, the transaction doesn't fail.

For example:

```
public class MixedDMLFuture {
public static void useFutureMethod() {
// First DML operation
Account a = new Account(Name='Acme');
insert a;

// This next operation (insert a user with a role)
// can't be mixed with the previous insert unless
// it is within a future method.
// Call future method to insert a user with a role.
Util.insertUserWithRole(
'abc.com', 'Test',
```

BY SMRITI SHARAN

```
'acb.com', 'Test2');
}
}
```

Once I call a future method, how can I trace its execution?

Salesforce uses a queue-based framework to handle asynchronous processes from such sources as future methods and batch Apex. So, we can check the apex jobs if it has run or not.

But if it in the queue, and resources are not available, It won't show up on Apex jobs Page, So we can poll **AsyncApexJob *object to get its status.***

However, **future methods don't return an ID,** so We can't trace it directly. We can use another filter such as MethodName, or JobType, to find the required job.

How to test a future method?

To test methods defined with the future annotation, call the class containing the method in a startTest(), stopTest() code block. All asynchronous calls

BY SMRITI SHARAN

made after the startTest method are collected by the system. When stopTest is executed, all asynchronous processes are run synchronously.

Sample Code

```
@isTest
public class MyFutureMethodTest {
    @isTest
    static void testMyFutureMethod() {
        // Create test data
        Account testAccount = new Account(Name = 'Test Account');
        insert testAccount;

        // Start the test context to collect asynchronous calls
        Test.startTest();

        // Call the future method
        MyFutureClass.myFutureMethod(testAccount.Id);

        // Stop the test context to run asynchronous calls synchronously
        Test.stopTest();

        // Perform assertions or checks to verify the expected behavior
        // For example, check if the future method has completed its work
        System.assertEquals('Expected Result', testAccount.SomeField);
}
```

What are some limitations of future methods?

Some of the limitations:

BY SMRITI SHARAN

1. It is not a good option to process large numbers of records.
2. Only primitive data types supported.
3. Tracing a future job is also typical.
4. Can’t call future from batch and future contexts, 1 call from queueable context is allowed.

Is it possible to call future method from apex scheduler or not?

Yes, it is possible to call future method from apex scheduler

```
//Scheduled Apex
public class DemoScheduler1 implements Schedulable{
    public void execute(SchedulableContext sc){
        system.debug('*******Going to call future method ');
        DemoAsynchronousTest.futureMethodCallFromScheduler();
    }
}
//apex class containing future method
public class DemoAsynchronousTest{
    @future
    public static void futureMethodCallFromScheduler(){
        system.debug('******futureMethodCallFromScheduler get called');
    }

}
```

Why future method is static and void?

BY SMRITI SHARAN

Future methods will run in the future. You don't want your synchronous code waiting an unknown period for an asynchronous bit of code to finish working. By only returning void, you can't have code that waits for a result.

The future method is static so that variables with this method is associated to the class and not the instance and **you can access them without instantiating the class.**

Can we call future method from process builder?

To call Future methods from Process Builder, call the future method from the invocable method.

From which places we can call future method?

- Trigger
- Apex Class
- Schedulable Class

Explain How to avoid Mixed DML Error?

BY SMRITI SHARAN

```
public class Util {

    @future

    public static void insertUserWithRole(

        String uname, String al, String em, String lname) {

        Profile p = [SELECT Id FROM Profile WHERE Name='Standard
User'];

        UserRole r = [SELECT Id FROM UserRole WHERE Name='COO'];

        // Create new user with a non-null user role ID

        User u = new User(alias = al, email=em,

            emailencodingkey='UTF-8', lastname=lname,

            languagelocalekey='en_US',

            localesidkey='en_US', profileid = p.Id, userroleid = r.Id,

            timezonesidkey='America/Los_Angeles',

            username=uname);

        insert u;

    }

}
public class MixedDMLFuture {
```

BY SMRITI SHARAN

```
    public static void useFutureMethod() {
        // First DML operation
        Account a = new Account(Name='Acme');
        insert a;

        // This next operation (insert a user with a role)
        // can't be mixed with the previous insert unless
        // it is within a future method.
        // Call future method to insert a user with a role.
        Util.insertUserWithRole(
            'mruiz@awcomputing.com', 'mruiz',
            'mruiz@awcomputing.com', 'Ruiz');
    }
}
```

Can we pass the wrapper to future method?

You can pass wrapper, but for that, you'll need to serialize/deserialize that parameter. You can convert the Wrapper to a String which is a primitive.

BY SMRITI SHARAN

Once converted into a String you can them pass that string as a parameter to the future method in consideration.

BY SMRITI SHARAN

Queueable

Apex

BY SMRITI SHARAN

What is the advantage of Queueable Apex over future methods?

Queueable Apex allows you to submit jobs for asynchronous processing like future methods with the following additional benefits:

- Non-primitive types: Your Queueable class can contain member variables of non-primitive data types, such as sObjects or custom Apex types.
- Monitoring: When you submit your job by invoking the System.enqueueJob method, the method returns the ID of the AsyncApexJob record. You can use this ID to identify your job and monitor its progress, either through the Salesforce user interface in the Apex Jobs page, or programmatically by querying your record from AsyncApexJob.

```
// You can use jobId to monitor the progress of your job
AsyncApexJob jobInfo = [SELECT Id, Status, NumberOfErrors
                        FROM AsyncApexJob
                        WHERE Id = :jobId];
```

BY SMRITI SHARAN

- Chaining jobs: You can chain one job to another job by starting a second job from a running job. Chaining jobs is useful if you need to do some sequential processing.

What is the interface used for Queueable Apex?

Queueable interface

What are methods used in Queueable Apex Class?

Execute method.

When will we use future methods instead of Queueable?

You use future methods instead of queueable is when your functionality is sometimes executed synchronously, and sometimes asynchronously. It's much easier to refactor a method in this manner than converting to a queueable class. This is handy when you discover that part of your existing code needs to be moved to async execution. You can simply create a similar future method that wraps your synchronous method.

How many jobs can you chain from executing a job?

BY SMRITI SHARAN

You can add only one job from an executing job, which means that only one child's job can exist for each parent job.

How many jobs can you queue in a single transaction?

You can add up to 50 jobs to the queue with System.enqueueJob in a single transaction.

How can I use this Job Id to trace the Job?

Just perform a SOQL query on AsyncApexJob by filtering on the job ID.

AsyncApexJob jobInfo = [SELECT Status,NumberOfErrors FROM AsyncApexJob WHERE Id=:jobID];

Can I do callouts from a Queueable Job?

Yes, you have to implement the Database.AllowsCallouts interface to do callouts from Queueable Jobs.

If I have written more than one System.enqueueJob call, what will happen?

BY SMRITI SHARAN

System will throw LimitException stating “Too many queueable jobs added to the queue: N”

I have a use case to call more than one Queueable Jobs from a Batch apex, how can I achieve it?

Since we can’t call more than one Queueable Job from each execution Context, we can go for scheduling the Queueable Jobs.

How to test Queueable Apex?

To ensure that the queueable process runs within the test method, the job is submitted to the queue between the Test.startTest and Test.stopTest block. The system executes all asynchronous processes started in a test method synchronously after the Test.startTest statement. Next, the test method verifies the results of the queueable job by querying the account records that the job updated.

What are the Limitations of Queueable Jobs?

BY SMRITI SHARAN

50 jobs can be added to the queue with System.enqueueJob() method in a single transaction

Maximum depth of chain job is 5 i.e., 4 child jobs and initial parent jobs for Developer and Trail organizations but there is no limit in other editions

Can you write a blueprint of Queueable Job?

Create a class, implement the Queueable interface, and override the execute method.

```
public class QueueableApexExample implements Queueable {
    public void execute(QueueableContext context) {
        //some process
    }
}
```

What is the return type of the id of Queueable apex in test class?

The ID of a queueable Apex job isn't returned in test context—System.enqueueJob returns null in a running test

What is QueueableContext?

BY SMRITI SHARAN

It is an interface that is implemented internally by Apex, and contains the job ID. Once you queue for the Queueable Job, the Job Id will be returned to you, by apex through QueueableContext's *getJobId()* method**.**

How can I queue Queueable Job?

Using *System.enqueueJob* Method.

```
ID jobID = System.enqueueJob(new QueueableApexExample());
```

I have 200 records to be processed using Queueable Apex, How Can I divide the execution Context for every 100 records?

Similar to future jobs, queueable jobs don't process batches, so you can't divide the execution Context. It will process all 200 records, in a single execution Context.

How Chaining works in Queueable Apex?

BY SMRITI SHARAN

```
public class A implements Queueable {
    public void execute(QueueableContext context) {
        // Chain this job to next job by submitting the next job
        System.enqueueJob(new B());
    }
}public class B implements Queueable {
    public void execute(QueueableContext context) {
        // Chain this job to next job by submitting the next job
        System.enqueueJob(new C());
    }
}public class C implements Queueable {
    public void execute(QueueableContext context) {
        // Chain this job to next job by submitting the next job
        System.enqueueJob(new D());
    }
}
```

Can I chain a job that has implemented *allowsCallouts*from a Job that doesn't have?

Yes**,** callouts are also allowed in chained queueable jobs.

How to test Chaining?

You can't chain queueable jobs in an Apex test. So you have to write separate test cases for each chained queueable job. Also, while chaining

BY SMRITI SHARAN

the jobs, add a check of Test.isRunningTest() before calling the enqueueJob.

Test.isRunningTest() Returns true if the currently executing code was called by code contained in a test method, false otherwise.

```
public class A implements Queueable {
    public void execute(QueueableContext context) {
// Test to make sure that Unit test are not running before chaining
// the call. We don't want to chain another job during a Unit test run.

        if(!Test.isRunningTest()){
          System.enqueueJob(new B());
       }
    }
}
```

What is Transaction Finalizers?

Transaction Finalizers feature enables you to attach actions, using the System.Finalizer interface, to asynchronous Apex jobs that use the Queueable framework.

Before Transaction Finalizers, there was no direct way for you to specify actions to be taken when asynchronous jobs succeeded or failed. With

BY SMRITI SHARAN

transaction finalizers, you can **attach a post-action sequence to a Queueable job** and take relevant actions based on the job execution result.

A Queueable job that failed due to an unhandled exception can be successively re-enqueued five times by a Transaction.

Example: We try to make a callout to an external platform, because of network issue, if the callout fails, how do we make sure that the callout is made again and re-enqueued?

We need to get the JobId after the callout is made, then check the status of the Job, if it failed then we need to re-enqueue it manually.

Salesforce team launched called **Finalizers** which will make the whole process of re-enqueueing the failed queueable job so easy.

Step 1 – Implement a class that implements the System.Finalizer interface.

Step 2 – Attach a finalizer within a queueable job's execute method by invoking the System.attachFinalizer method with an argument of the

BY SMRITI SHARAN

instantiated class that implements the finalizer interface. Only one finalizer can be attached to any queueable job.

```
public class UdpateFinalizer implements Finalizer {

    public void execute(FinalizerContext ctx) {
          //FinalizerContext is a class that provides us access to 4 methods.
        //getRequestId()
        //getAsyncApexJobId()
        //getResult()
        //getException()

        String reqId = ctx.getRequestId();
        String jobId = Id.valueOf(ctx.getAsyncApexJobId());

        if (ctx.getResult() == ParentJobResult.SUCCESS) {
            System.debug('Parent Queueable (job id: ' + jobId + '): completed
successfully!');

        } else { // Queueable failed
            //provide a counter & it should not exceed 5 times
            //re-enqueue the job
        }
    }

}
public class SampleAsyncJob implements Queueable {

    public void execute(QueueableContext ctx) {

        UdpateFinalizer f = new UdpateFinalizer();
        System.attachFinalizer(f);
```

BY SMRITI SHARAN

```
        //Business logic
    }

}
```

Differences between Future and Queueable Apex?

Future Method	Queueable Job
1. Future will never use to work on SObjects or object types. 2. When using the future method we cannot monitor the jobs which are in process. 3. The future method cannot be called inside the future or batch class.	1. Queueable Jobs can contain the member variable as SObjects or custom Apex Types. 2. When using queueable jobs it will make the AsyncApexJob which we can monitor like Scheduled jobs. 3. Queueable Apex can be called from the future and batch class.

BY SMRITI SHARAN

Future Method	Queueable Job
4. The future method will never be queued.	4. Using Queueable Apex will chain up to queueable jobs and in Developer Edition it is only 5 Jobs.

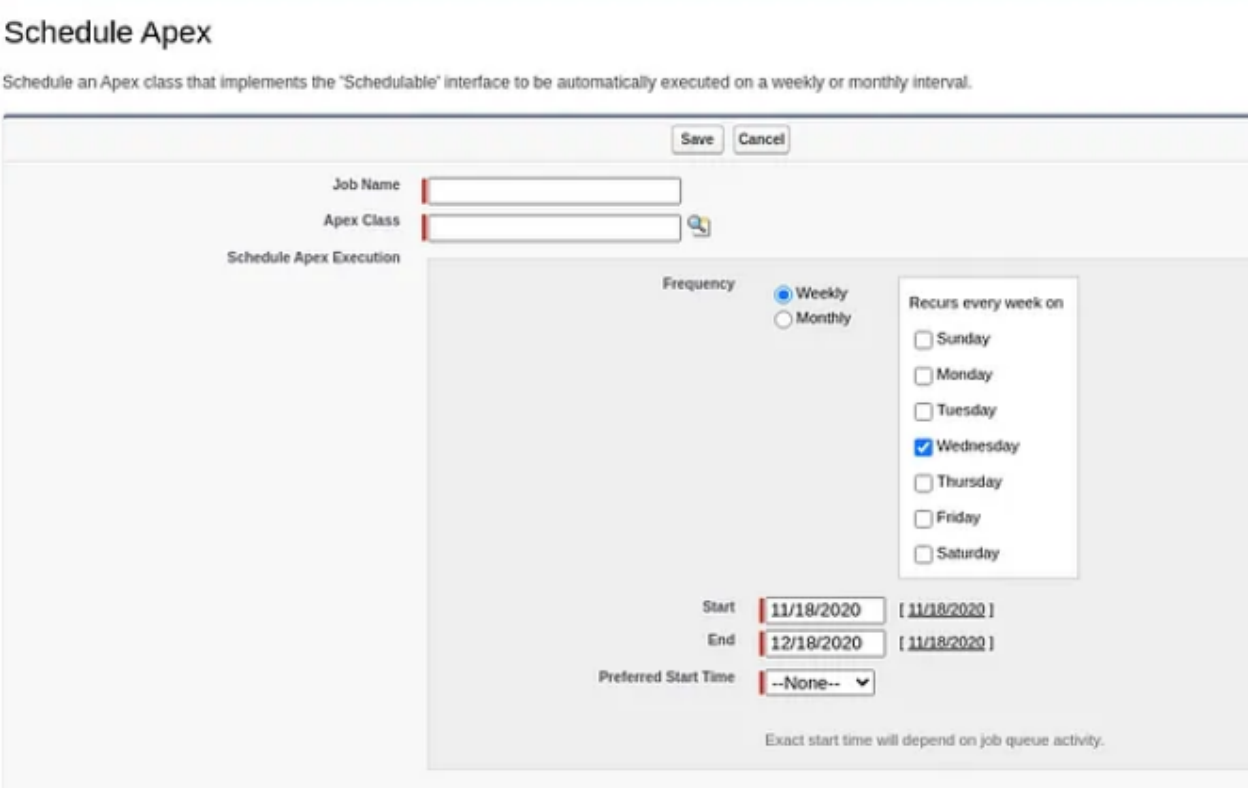
BY SMRITI SHARAN

Schedulable Apex

BY SMRITI SHARAN

What is an apex Scheduler?

The Apex Scheduler lets you delay execution so that you can run Apex classes at a specified time. This is ideal for daily or weekly maintenance tasks using Batch Apex.



What is the interface used for Schedulable Apex?

Schedulable interface

BY SMRITI SHARAN

What are the methods used with a Schedulable interface?

The only method this interface contains is the execute method.

What is the parameter of the execute method ?

The parameter of this method is a SchedulableContext object.

What happens after the class is scheduled?

After a class has been scheduled, a CronTrigger object is created that represents the scheduled job. It provides a getTriggerId method that returns the ID of a CronTrigger API object.

What are the arguments of the System.Schedule method?

The System.Schedule method takes three arguments:

- Name for the job
- CRON expression used to represent the time and date the job is scheduled to run
- Name of the class.

BY SMRITI SHARAN

How to Test Scheduled Apex?

To test Scheduled Apex you must ensure that the scheduled job is finished before testing against the results. To do this, use startTest and stopTest around the System.schedule method, to ensure processing finishes before continuing your test.

What is the governor limit of Scheduled Apex?

You can only have 100 scheduled Apex jobs at one time and there are a maximum number of scheduled Apex executions per a 24-hour period.

Can we make a callout from Scheduled Apex?

Synchronous Web service callouts are **not supported** from scheduled Apex.

How to monitor Scheduled Jobs?

After an Apex job has been scheduled, you can obtain more information about it by running a SOQL query on CronTrigger.

BY SMRITI SHARAN

CronTrigger job = [SELECT Id, CronJobDetail.Id, CronJobDetail.Name,

CronJobDetail.JobType FROM CronTrigger ORDER BY CreatedDate

DESC LIMIT 1];

Q. Explain code to schedule batch Apex to run at regular intervals?

```
global class SampleBatchScheduler implements Schedulable {

    // Execute at regular intervals
    global void execute(SchedulableContext ctx){
      String soql = 'SELECT Id, Name FROM Account';
      SampleBatch batch = new SampleBatch(soql);
      Database.executebatch(batch, 200);
    }
}
```

What is System. Schedule ?

Once you are implemented schedulable interface

use system.schedulable method to execute the class.

system.schedule() method takes 3 parameters :

1. Name of the job

2. An expression that is used to represent the time and date of the

BY SMRITI SHARAN

operation.

3. The object of the class which you want to execute.

An expression is written in the form of ‘Seconds, minutes, hours, day of the month, month day of the week, optional year.’

‘Seconds’ : 0-60

‘Min’ : 0-60

‘Hours’ : 0-24

‘Day-Months’ : 1-31

‘Month’ : 1-12

‘Day-Week’ : 1-7

‘Optional Year’ : —

What is return type of system.schedule?

System.schedule method returns the job ID in string format.

```
String jobID = system.schedule('Merge Job', sch, m);
```

Explain how to use System.schedule ?

BY SMRITI SHARAN

```
class MySchedule implements Schedulable
{
 public void execute(SchedulableContext BC)
 {
  //---------//
 }
 MySchedule mysc = New MySchedule();
 String str = '0 0 10 * 3 2';

// MyJob is Name, str is Time Format, mysc is Object

System.schedule('MyJob', str, mysc);
}
```

Write an expression to schedule an operation on Jan Tuesday 12:30 ?
‘0 30 12 ? 1 TUES’
‘0 30 12 ? 1 3’
‘Seconds’ : 0
‘Min’ : 30
‘Hours’ : 12
‘Day-Months’ : ?
‘Month’ : 1
‘Day-Week’ : 3
‘Optional Year’ : —

Write the expression to schedule an operation on every day of the SEP at 12:30 PM.
‘0 30 12 * SEP ?’
‘0 30 12 * 9 ?’
‘Seconds’ : 0
‘Min’ : 30
‘Hours’ : 12

BY SMRITI SHARAN

‘Day-Months’ : *
‘Month’ : 9
‘Day-Week’ : ?
‘Optional Year’ : —

Expression to schedule on every hour on 11th SEP.
‘0 30 * 11 9 ?’
‘Seconds’ : 0
‘Min’ : 30
‘Hours’ : *
‘Day-Months’ : 11
‘Month’ : 9
‘Day-Week’ : ?
‘Optional Year’ : —

Want to schedule batch job at one time only and not again. How to do it?

System.scheduleBatch() is used to run a schedule a batch job only once for a future time. This method has got 3 parameters.

param 1 : Instance of a class that implements Database.Batchable interface.

param 2 : Job name.

param 3 : Time interval after which the job should start executing.

param 4 : It’s an optional parameter which will define the no. of that

BY SMRITI SHARAN

processed at a time. The system.scheduleBatch() returns the scheduled job Id.

We can use the job Id to abort the job.

This method returns the **scheduled job ID** also called **CronTrigger ID**.

```
String cronID = System.scheduleBatch(reassign, 'job example', 1);

CronTrigger ct = [SELECT Id, TimesTriggered, NextFireTime
                  FROM CronTrigger WHERE Id = :cronID];
```

This method is available only for batch classes and doesn't require the implementation of the Schedulable interface. This makes it easy to schedule a batch job for one execution.

How to get count of Apex Scheduled Job programmatically?

You can programmatically query the CronTrigger and CronJobDetail objects to get the count of Apex scheduled jobs.

If there are one or more active scheduled jobs for an Apex class, can you update the class, or any classes referenced in Salesforce UI?

BY SMRITI SHARAN

If there are one or more active scheduled jobs for an Apex class, you cannot update the class or any classes referenced by this class through the Salesforce user interface. However, you can enable deployments to update the class with active scheduled jobs by using the Metadata API

Does Apex Scheduler run in system mode?

The scheduler runs as system—all classes are executed, whether or not the user has permission to execute the class.

How to Call batch apex from schedulable class?

Create instance of batchClass and then pass the instance in database.executebatch

```
batchable b = new batchable();
database.executebatch(b);
```

An easier way to schedule a batch job is to call the System.scheduleBatch method without having to implement the Schedulable interface.

BY SMRITI SHARAN

How to get Job name and job type for Scheduled jobs?

You can get job's name and the job's type from the CronJobDetail record associated with the CronTrigger record.

Callout is not supported in Scheduled Apex so what is the alternative?

Synchronous Web service callouts are not supported from scheduled Apex. To be able to make callouts, make an asynchronous callout by placing the callout in a method annotated with @future(callout=true) and call this method from scheduled Apex. However, if your scheduled Apex executes a batch job, callouts are supported from the batch class.

What are limitations of Scheduled Apex?

1. We can schedule only 100 jobs at a time.

2. Max no. of apex schedule jobs in 24 hours is 2,50,000 number of jobs (can change with salesforce updates).