# BATCH CLASS

Link: https://intellipaat.com/blog/tutorial/salesforce-tutorial/salesforce-batch-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.

**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.

## Batch Class In Salesforce :

When using a Batch class in Salesforce, the batchable interface needs to be implemented first. It has the following three methods:

**1. 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. In some cases, the 'QueryLocator' method is used to operate with the simple SOQL query to generate the scope of objects inside a batch job.

**Syntex** : global void start(Database.BatchableContext BC, list<sobject<) {}

**2. Execute**:

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

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

**3. 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.

**Syntex** : global void finish(Database.BatchableContext BC) {}

## Batch Class Notes :

Existing Record ya update Record and delete Record k liye use kiya jata h.
Insert k liye nhi use hota h.

Start → this is to initiate the batch process

Execute → this is the main process for batch

Finish → this is used to end the batch

## Database.Stateful in Batch Apex in Salesforce?

**Answer**
Batch Apex is stateless. Each execution of a batch Apex job is considered a discrete transaction. If we implements **Database.Stateful** we can maintained state across transactions. Using **Database.Stateful** only instance variable holds values static members does not hold values. If we want to count records as batch proceeds maintaining state is important as after one transaction new transaction will start and members will loose their values.
**[To maintain variable value inside the Batch class, Database.Stateful is used.]**

**More Details :**

https://www.sfdc-lightning.com/2021/05/Database.Stateful-in-Batch-Apex-in-Salesforce.html

---

## Batch Class

Batch Apex is used to run large jobs (think thousands or millions of records!) that would exceed normal processing limits. Using Batch Apex, you can process records asynchronously in batches (hence the name, “Batch Apex”) to stay within platform limits. If you have a lot of records to process, for example, data cleansing or archiving, Batch Apex is probably your best solution.

| Type | Overview | Common Scenarios |
| --- | --- | --- |
| Future Methods | Run in their own thread, and do not start until resources are available. | Web service callout. |
| Batch Apex | Run large jobs that would exceed normal processing limits. | Data cleansing or archiving of records. |
| Queueable Apex | Similar to future methods, but provide additional job chaining and allow more complex data types to be used. | Performing sequential processing operations with external Web services. |
| Scheduled Apex | Schedule Apex to run at a specified time. | Daily or weekly tasks. |