Batch Apex - Iterable

Syntax : `global (Database.QueryLocator | Iterable) start(Database.BatchableContext bc) {}`

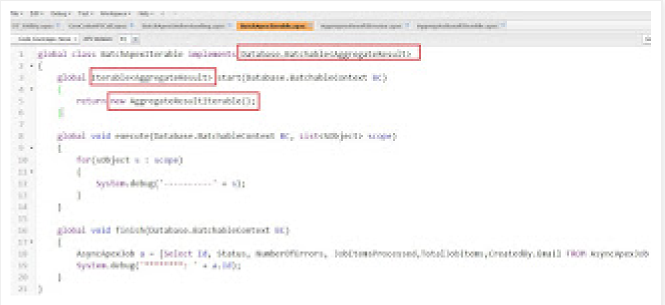
When do we use Iterable ?

When some thing is not achieved by direct SOQL .

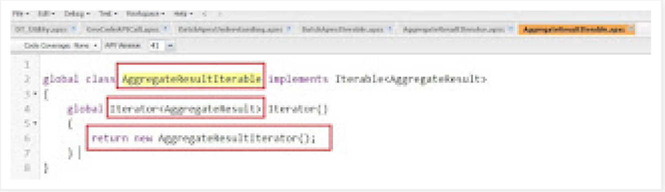
1. Aggregate queries in Batch Apex
2. Querying on multiple object
3. To process CSV / XML files

Refer SF Knowledge Article:

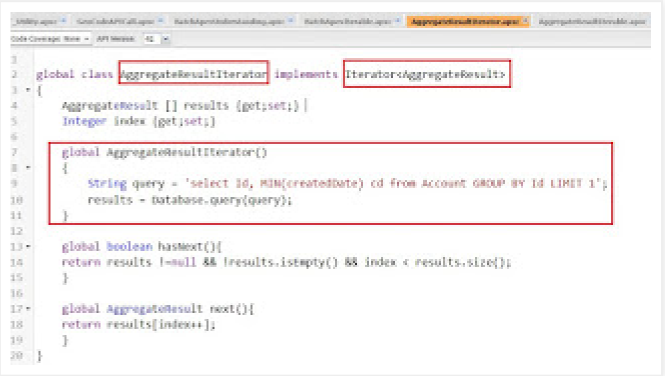
Sample Code :



```
global class BatchApexIterable implements Database.Batchable<AggregateResult>
{
    global Iterable<AggregateResult> start(Database.BatchableContext BC)
    {
        return new AggregateResultIterable();
    }

    global void execute(Database.BatchableContext BC, List<SObject> scope)
    {
        for(sObject s : scope)
        {
            System.debug('----------' + s);
        }
    }

    global void finish(Database.BatchableContext BC)
    {
        AsyncApexJob a = [Select Id, Status, NumberOfErrors, JobItemsProcessed,TotalJobItems,CreatedBy.Email FROM AsyncApexJob
        System.debug('********: ' + a.Id);
    }
}
```



```

global class AggregateResultIterable implements Iterable<AggregateResult>
{
    global Iterator<AggregateResult> Iterator()
    {
        return new AggregateResultIterator();
    }
}
```



```

global class AggregateResultIterator implements Iterator<AggregateResult>
{
    AggregateResult [] results {get;set;}
    Integer index {get;set;}

    global AggregateResultIterator()
    {
        String query = 'select Id, MIN(createdDate) cd from Account GROUP BY Id LIMIT 1';
        results = Database.query(query);
    }

    global boolean hasNext(){
    return results !=null && !results.isEmpty() && index < results.size();
    }

    global AggregateResult next(){
    return results[index++];
    }
}
```