Have you ever performed **roll-up** or aggregate of more than 50000 records? If your answer is NO, then this blog post will be helpful for you. In this blog post, **we will implement how to get roll-up of more than 50000 Contacts on Account.**

If you want to get roll up of more than 50,000 records then you can not use aggregate query as it limits to 2000 records only. If you want to use the custom calculation then there is limit that you can only query 50,000 records in one transaction. So the workaround is we can use **@ReadOnly Annotation.**

What is @ReadOnly Annotation?

- The **@ReadOnly** annotation allows you to **perform unrestricted queries** against the Lightning Platform database.

Where we can use @ReadOnly Annotation?

- The @ReadOnly annotation is available for **Web services and the Schedulable interface**. To use the @ReadOnly annotation, the top level request must be in the schedule execution or the Web service invocation. For example, if a Visualforce page calls a Web service that contains the @ReadOnly annotation, the request fails because Visualforce is the top level request, not the Web service. Visualforce pages can call controller methods with the @ReadOnly annotation, and those methods will run with the same relaxed restrictions.

GROUP BY ROLLUP Clause

- The **GROUP BY ROLLUP** clause is used in a SOQL query to **add subtotals for aggregated data** in query results. This action enables the query to **calculate subtotals** so that you **don't have to maintain that logic** in your code.

Let's implement using code

```
/*
 * Author Name: Priyanka Dadhe
 * Source     : sfdcsurf.blogspot.com
 * Date       : 12th January, 2020
 * Class Name : ContactRollupSchedulable
 * Description: To get Roll-Up of more than 50000 Contacts on Account.
*/
global class ContactRollupSchedulable implements Schedulable {
  @ReadOnly
  global void execute (SchedulableContext ctx){
  Map<Id,Decimal>                 = new Map<Id,Decimal>();
  for(AggregateResult aggResult : [SELECT  SUM(Amount__c) totalAmt, accountId FROM Contact GROUP BY ROLLUP (accountId)]){
                    .   (         .         ,(Decimal)         .   ('totalAmt'));
  }
  RollUpUpdateBatch          = new RollUpUpdateBatch(               );
  Database.executeBatch(accBatch);
 }
}
```

```
/*
 * Author Name: Priyanka Dadhe
 * Source     : sfdcsurf.blogspot.com
 * Date       : 12th January, 2020
 * Class Name : RollUpUpdateBatch
 * Description: To get Roll-Up of more than 50000 Contacts on Account.
*/
global  RollUpUpdateBatch implements Database.Batchable<sObject>{
        Map<String,Decimal>                     = new Map<String,Decimal>();
  global RollUpUpdateBatch(Map<Id,Decimal> accIdToconAmtCountMap){
    this.                     =                     ;
  }
        Database.QueryLocator      (Database.BatchableContext    ){
    String query = 'SELECT Id,total_con_amount__c FROM Account WHERE Id IN :accIdToconAmtCountMap.keySet()';
    return Database.               (     );
  }
        void        (Database.BatchableContext     , List<Account>        ){
   for(Account acc : accList){
    if(!                    .       () &&                     .          (    .Id)){
      acc.total_con_amount__c = accIdToconAmtCountMap.get(acc.Id);
    }
   }
                ;
  }
        void       (Database.BatchableContext    ){
  }
}
```