Workaround for 100 DML and 100SOQL limit during creation of data on refreshed developer sandbox

July 2, 2014 bartoszborowiec Leave a comment

Always after you refresh your sandbox you have the same issue. You have to create data on it. Usually you have to create more data than DML limits in single transaction allows, therefore you have to execute many scripts, remember about order of these scripts e.t.c. But there is a simple solution. **Batches**. In a single step of the salesforce batch you can have 200 SOQL and 150 DML. Quite Good. But remember that you can have as many steps as you need. And when you use Stateful batch you can transfer data between steps. Of course when you are using batches, you should iterate on SObjects, but you can create these SObjects in start method of your batch:) Is is a really nice 'Hack', isn't it?

Here you can download working solution.

How does it work:

- **MultiStepScriptTemplate.cls** – abstract class that implements all common functionalities for this solution
- **Batch_STEP__c.object** – help SObjecs that are created in start method of batch. Our bath iterates on them
- **MultiStepScriptImplementation.cls** – class that implemented given script to create a data. Every step of our script should be implemented in dedicated inner class that extends MultiStepScriptTemplate.ScriptStep class. Method initScriptSteps should create a list with all steps of script in order given by developer. Every step of script has an access to public variables declared in MultiStepScriptImplementation class. This access is ensured by inner variable **caller** from MultiStepScriptTemplate. This variable should be cast to class that implements script(for example MultiStepScriptImplementation)
- **BB_C__c.object** – just dummy object that is used to prove that you can execute as many **DMLs** as you need when you create data on your sandbox

This script is executed from realforce explorer or developer console in following way:

```
1  MultiStepScriptImplementation multiStepScript = new MultiStep
2  multiStepScript.startScript();
```