

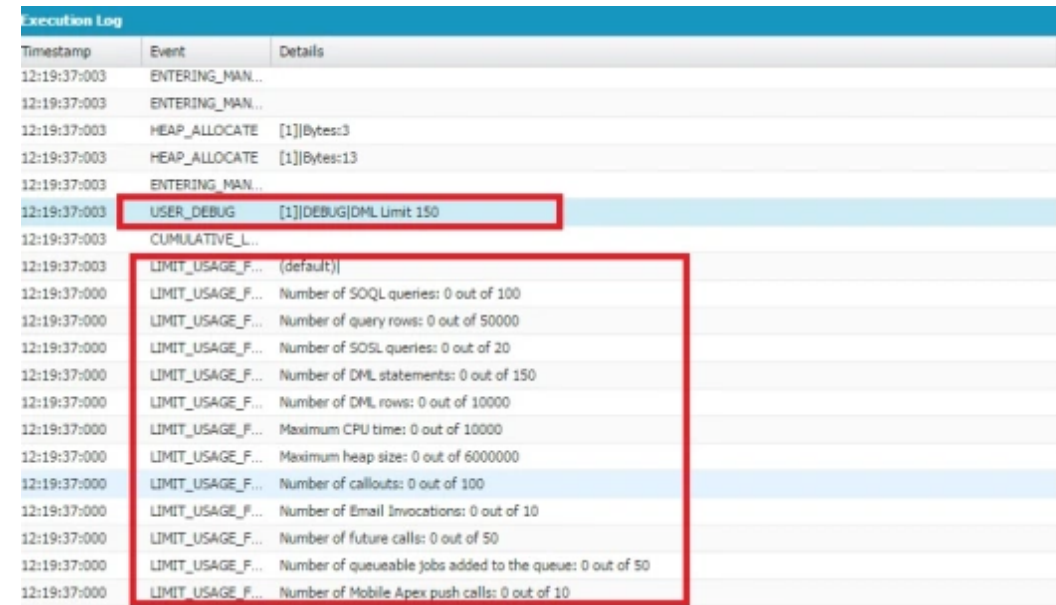
Reduce number of DML Statements

Posted on [August 20, 2016](#) by [Dilip Singh](#) in [Salesforce](#)

One of the most common governor limit which every developer encounter is the number of DML Statements and there are many more as well like SOQL limit which I will be explaining in my next blog.

Identify the number of DML Statements allowed per transaction?

```
1 | system.debug('DML Limit ' + Limits.getLimitDMLStatements());
```



The screenshot shows the Salesforce Execution Log. A red box highlights the 'USER_DEBUG' event with the message '[1] (DEBUG) DML Limit 150'. Another red box highlights the 'LIMIT_USAGE_F...' events, which show various governor limits. The 'LIMIT_USAGE_F...' event for 'Number of DML statements' shows '0 out of 150'.

Timestamp	Event	Details
12:19:37:003	ENTERING_MAN...	
12:19:37:003	ENTERING_MAN...	
12:19:37:003	HEAP_ALLOCATE	[1] Bytes:3
12:19:37:003	HEAP_ALLOCATE	[1] Bytes:13
12:19:37:003	ENTERING_MAN...	
12:19:37:003	USER_DEBUG	[1] (DEBUG) DML Limit 150
12:19:37:003	CUMULATIVE_L...	
12:19:37:000	LIMIT_USAGE_F...	(default)
12:19:37:000	LIMIT_USAGE_F...	Number of SOQL queries: 0 out of 100
12:19:37:000	LIMIT_USAGE_F...	Number of query rows: 0 out of 50000
12:19:37:000	LIMIT_USAGE_F...	Number of SOSL queries: 0 out of 20
12:19:37:000	LIMIT_USAGE_F...	Number of DML statements: 0 out of 150
12:19:37:000	LIMIT_USAGE_F...	Number of DML rows: 0 out of 10000
12:19:37:000	LIMIT_USAGE_F...	Maximum CPU time: 0 out of 10000
12:19:37:000	LIMIT_USAGE_F...	Maximum heap size: 0 out of 6000000
12:19:37:000	LIMIT_USAGE_F...	Number of callouts: 0 out of 100
12:19:37:000	LIMIT_USAGE_F...	Number of Email Invocations: 0 out of 10
12:19:37:000	LIMIT_USAGE_F...	Number of future calls: 0 out of 50
12:19:37:000	LIMIT_USAGE_F...	Number of queueable jobs added to the queue: 0 out of 50
12:19:37:000	LIMIT_USAGE_F...	Number of Mobile Apex push calls: 0 out of 10

150 DML statements for a transaction is more than enough, but it may hit the limit if code is not properly written.

```
1 | List<Account> acclist = new List<Account>();
2 | acclist.add(new Account(Name = 'Account 1'));
3 | acclist.add(new Account(Name = 'Account 2'));
4 |
5 | List<CustomObject__c> customObjlist = new List<CustomObject__c>();
6 | customObjlist.add(new CustomObject__c(Name = 'Custom Rec 1'));
7 | customObjlist.add(new CustomObject__c(Name = 'Custom Rec 2'));
8 |
9 | insert acclist;
10 | insert customObjlist;
```

Here, if we see the above code consumes 2 DML statements – one for each object which is good but what if the number of different objects increases?

We can use sObject list which reduces it even further:

sObject is a generic base object where every object is an extension of sObject in APEX.

```
1 | List<sObject> insertSObjectlist = new List<sObject>();
2 |
3 | insertSObjectlist.add(new Account(Name = 'Account 1'));
4 | insertSObjectlist.add(new Account(Name = 'Account 2'));
5 |
6 | insertSObjectlist.add(new CustomObject__c(Name = 'Custom Rec 1'));
7 | insertSObjectlist.add(new CustomObject__c(Name = 'Custom Rec 2'));
8 |
9 | insert insertSObjectlist;
```

12:39:13:263	LIMIT_USAGE_F...	singhforce
12:39:13:000	LIMIT_USAGE_F...	Number of SOQL queries: 0 out of 100
12:39:13:000	LIMIT_USAGE_F...	Number of query rows: 0 out of 50000
12:39:13:000	LIMIT_USAGE_F...	Number of SOSL queries: 0 out of 20
12:39:13:000	LIMIT_USAGE_F...	Number of DML statements: 1 out of 150
12:39:13:000	LIMIT_USAGE_F...	Number of DML rows: 4 out of 10000
12:39:13:000	LIMIT_USAGE_F...	Maximum CPU time: 0 out of 10000
12:39:13:000	LIMIT_USAGE_F...	Maximum heap size: 0 out of 6000000
12:39:13:000	LIMIT_USAGE_F...	Number of callouts: 0 out of 100
12:39:13:000	LIMIT_USAGE_F...	Number of Email Invocations: 0 out of 10
12:39:13:000	LIMIT_USAGE_F...	Number of future calls: 0 out of 50
12:39:13:000	LIMIT_USAGE_F...	Number of queueable jobs added to the queue: 0 out of 50
12:39:13:000	LIMIT_USAGE_F...	Number of Mobile Apex push calls: 0 out of 10

Now, using sObject list the earlier 2 DML statements are now reduced to 1, likewise we can use sObjectlist for all other types of DML.

Here is another limit, we cannot add 'n' number of objects to generic sObject list. At max, **we can have a mix of 10 different objects**

sample code for Update, Upsert and Delete using sObject list:

For Update:

```

1  List<sObject> updateSObjectlist = new List<sObject>();
2
3  updateSObjectlist.add(new Account(id= '0012800000pNrFL', Name = 'Account Rec 1'));
4  updateSObjectlist.add(new Account(id= '0012800000pNrFM', Name = 'Account Rec 2'));
5
6  updateSObjectlist.add(new CustomObject__c(id= 'a042800000RHfnU', Name = 'Custom Rec 1'));
7  updateSObjectlist.add(new CustomObject__c(id= 'a042800000RHfnV', Name = 'Custom Rec 2'));
8
9  update updateSObjectlist;
```

For Upsert:

Currently, Salesforce doesn't support Upsert on generic sObject list.

Here is [idea](#), you can vote and salesforce may support soon which saves us in writing extra code for splitting up the list into insert and update list.

For Delete:

```

1  List<sObject> deleteSObjectlist = new List<sObject>();
2
3  deleteSObjectlist.add(new Account(id= '0012800000pNrFL'));
4  deleteSObjectlist.add(new Account(id= '0012800000pNrFM'));
5
6  deleteSObjectlist.add(new CustomObject__c(id= 'a042800000RHfnU'));
7  deleteSObjectlist.add(new CustomObject__c(id= 'a042800000RHfnV'));
8
9  delete deleteSObjectlist;
```