LastModifiedDate Vs SystemModStamp

Sudipta Deb Friday, February 02, 2018

In Salesforce, when we create or update a record, there are so many things happen in the background. In this post, I am going to share my knowledge regarding two important fields present in each Salesforce record. I will start with a basic explanation regarding these fields and then at the end, I will explain how these fields are going to impact your query performance.

Nice to check out:

- Salesforce Spring 21 Release Feature
- Salesforce Certification Notes
- Salesforce Release Video Tutorials
- Salesforce Lightning Flow Video Tutorials

Let's start with basic explanation -

LastModifiedDate:

This is a standard field present in each Salesforce record. This field will be updated with current timestamp when the record is created or edited by Salesforce user. This field also has a very unique feature i.e. this field can be imported with some previous date. Let's say you want to load data from your old legacy system, but you want to preserve the dates from your legacy system. In this situation, you can load the records from your legacy system into Salesforce by updating the LastModifiedDate field with the previous date.

SystemModStamp:

This is a read-only field. This field will be updated with current timestamp when the record is updated by the user. But the important difference is that this field will also be updated when the record is updated by Automated System Process. So in mathematical term -

LastModifiedDate <= SystemModStamp This is possible.
LastModifiedDate > SystemModStamp This is not possible.

How LastModifiedDate and SystemModStamp affect SOQL performance:

LastModifiedDate field is not indexed, but SystemModStamp field is indexed. So when SOQL query uses LastModifiedDate field in the where clause, Salesforce's Query Optimizer will internally use SystemModStamp to improve the performance. But when LastModifiedDate is used to determine the upper boundary of a date range, then Salesforce's Query Optimizer will

not be able to use SystemModStamp. Confused???
Let's understand the concept with below examples -

Consider one account record - ABC Technologies

Record created by user on Jan 31st, 2018	LastModifiedDate: Jan 31st, 2018 SystemModStamp: Jan 31st, 2018
Record updated by Automated Process on Feb 2nd, 2018	LastModifiedDate: Jan 31st, 2018 SystemModStamp: Feb 2nd, 2018

Query: *Select Id From Account Where **LastModifiedDate** > 2018-01-30T00:00:00Z*
Can Salesforce's Query Optimizer replace LastModifiedDate with SystemModStamp? Answer is **YES**
Salesforce's Query Optimizer will change the query to -
*Select Id From Account Where **SystemModStamp** > 2018-01-30T00:00:00Z* to improve the performance and definitely we will get out ABC Technologies back from the SOQL.

Now consider another situation where the query is:
Select Id From Account Where LastModifiedDate < 2018-02-01T00:00:00Z.
Can Salesforce's Query Optimizer replace LastModifiedDate with SystemModStamp? Answer is **NO**
Why?
With original query (*Select Id From Account Where **LastModifiedDate** < 2018-02-01T00:00:00Z*) we are expecting ABC Technologies will come out, but if Salesforce's Query Optimizer replaces LastModifiedDate with SystemModStamp, then the query will become *Select Id From Account Where **SystemModStamp** < 2018-02-01T00:00:00Z*. So definitely with this query, we will not get our ABC Technologies back.

That is why Salesforce's Query Optimizer will not be able to use SystemModStamp instead of LastModifiedDate when LastModifiedDate is used to determine with the upper boundary of a date range, otherwise, the records in between these two dates(SystemModStamp and LastModifiedDate) will be missed from the query result.