Data Skew in Salesforce

By Shubham Jha March 11th, 2019

Data is getting generated at an explosive pace nowadays and we are running out of storage solutions in order to manage that data. Researches by multiple magazines and portals suggest that 90 percent of the total data in the world was created in the last two years only. This pace continues to increase day by day and we are slowly approaching a state where we would not be able to deal with this data. **Salesforce** is no exception in this case where organization instances are having a large number of records related to the business process needs. When this plethora of data is not managed properly, we slowly approach a state which is termed as Data Skew.

What is Data Skew?

Data Skew generally refers to a condition where data is distributed unevenly in a large data set. In Salesforce, data skew occurs when more than 10000 child object records are related to a single parent object record, or more than 10000 records of any object are owned by a single Salesforce user. This skewness leads to major performance hits and long-running processes which are something that one should avoid.

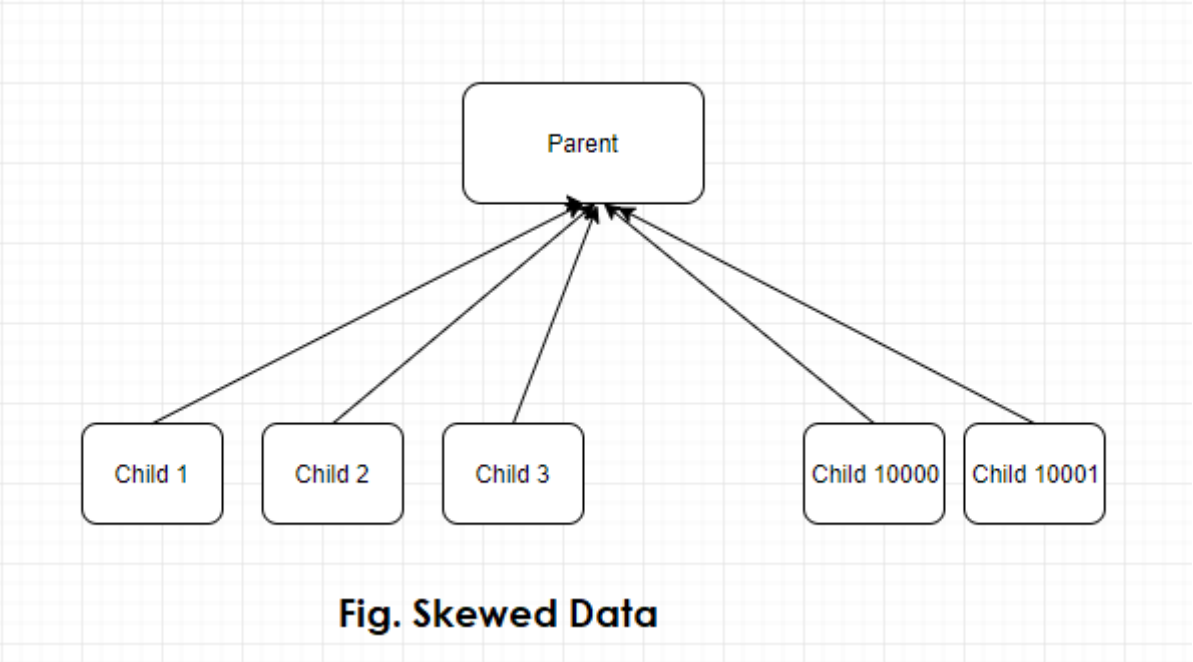

Fig. Skewed Data

Types of Data Skew in Salesforce

Three types of data skew exist in Salesforce which is as follows:

1. Account Skew
2. Ownership Skew
3. Lookup Skew

1. Account Skew
This type of Salesforce data skew comes into existence when you have a large number of child records present under a single account record. This is a very common scenario as it is quite tempting to place all your unwanted or unassigned records under an account named Miscellaneous or Unassigned. As easy and correct as it may look, it can cause major issues such as record locking and sharing performances. This is mainly because certain standard objects like Opportunity and Account, have special data relationships which maintain record access under private sharing models. The problems that you will face in a state of Account skew are:

- **Record Locking:** When we are performing an update operation on a large number of child records in separate threads, the system locks the child being updated as well as the parent record in order to maintain database integrity for each update. Hence, the parent record might be locked by one thread while some other thread is trying to update the same.
- **Sharing Problems:** When we have many child associations with a single parent record, a simple change in sharing setting might lead to a chain of time-consuming processes. Even a meager change like updating the owner of a parent record may lead to all the sharing rules on the child records being recalculated as well as the recalculation of the role hierarchy.

Possible Way for Avoiding Account Skew:
There is only one way to avoid Account skew, that is by the distribution of such child records across multiple accounts rather than accumulation on a single record. Having an even distribution of child records across parent accounts fool proofs our organization against performance hits due to account skew.
2. Ownership Skew
Ownership data skew is another type of date skew which is very common in Salesforce. This issue occurs when more than 10000 records are owned by a single Salesforce user. Since every record inside Salesforce needs to have an owner, it is quite common in organizations to make a default owner or queue, to which all the unassigned or unused records go to. It is a preferred solution for many organizations in such use cases, but little do they know that though this might work for small data sets, this will fail when we are dealing with large data. This increases the probability of performance issues whenever some change to the sharing settings or some similar operation occurs. For example, if a user owns a large number of records and he/she is moved around in the role hierarchy, then the sharing rules for all the records owned by that user will be reevaluated and that will result in a long-running operation.
Possible Ways for Avoiding Ownership Skew:

- The best way to avoid this kind of skew will be even distribution of such records among multiple users rather than having a single user for all.
- If you are compelled to stay put with this solution, then the performance impacts can be reduced by not assigning the user (record owner) to a role.

- If the owner must have a role, then try to keep the user on top of the role hierarchy. This will avoid the user being passed around the role hierarchy.
- Make sure that the user is not a member of any public group which is acting as the source for a sharing rule.

3. Lookup Skew

Lookup skew is similar to Account skew but can affect a broader number of objects. This happens when a large number of records are associated with a single record in the lookup object. Since lookup fields can exist on standard as well as custom fields, lookup skew problems can arise on any custom object in the organization. This happens regardless of whether that lookup exists on a single object or across multiple objects.

Possible Ways for Avoiding Lookup Skew:

- One method is to distribute the skew across multiple lookup fields. The main cause of the problem is that a large number of records are lookup to the same record. By providing additional lookup values to distribute the skew, record lock exceptions can be minimized or even eliminated.
- Remove unnecessary workflow rules or process builders on the objects in order to reduce the record saving time. Also, make sure that the synchronous apex code and triggers are well optimized.
- In case the number of lookup values is low and definite, you can use picklist values to represent the lookup values rather than using lookup fields.

Conclusion

Data plays a crucial role in the business architecture of large organizations and hence these problems are very common. By taking a few steps while designing our architecture, the data skew problems can be avoided. Having distributed data is still the best bet for getting rid of these skews and their repercussions.