Improve Performance of SOQL Queries using a Custom Index

Knowledge Article Number 000325247

Description

SOQL queries must be selective, particularly for queries inside triggers for the best performance. To avoid long execution times, non-selective SOQL queries may be terminated by the system. Developers will receive an error message when a non-selective query in a trigger executes against an object that contains more than 200,000 records. Learn how you can avoid the error, and make the query selective below.

Resolution

Important Note:

While every effort has been made to ensure this article explains how the Salesforce SOQL query optimizer works at a high level, its behavior may change without notice to accommodate additional performance enhancements. Developers are advised to use the **Query Plan view** (available since Summer '14) in the Developer Console, to be able to tune non-selective queries based on the information provided by the optimizer at the time the query plan for the affected SOQL query is generated.

Additionally, indexes can add cost to DML and/or in some cases Query Optimizer needs to do things differently based on indexes which may impact performance.

Selective SOQL Query Criteria

A query is selective when one of the query filters is on an indexed field and the query filter reduces the resulting number of rows below a system-defined threshold. The performance of the SOQL query improves when two or more filters used in the WHERE clause meet the mentioned conditions.

The selectivity threshold is 10% of the records for the first million records and less than 5% of the records after the first million records, up to a maximum of 333,000 records. In some circumstances, for example with a query filter that is an indexed standard field, the threshold may be higher. Also, the selectivity threshold is subject to change.

Custom Index Considerations for Selective SOQL Queries

The following fields are indexed by default:

- Primary keys (ID, Name, and Owner fields).

- Foreign keys (lookup or master-detail relationship fields).
- Audit dates (such as SystemModStamp).
- Custom fields marked as External ID or Unique.

Salesforce Support can add Custom Indexes upon request for customers

When creating a case with Support, be sure to include the SOQL query which has the field to be indexed as a filter in the WHERE clause. Also be sure to include bind values if any.
Review the Checklist in **Make SOQL query selective** before creating a Support case.
You can also determine which fields can be indexed by referring to **How to make SOQL query selective** and doing Query Plan analysis in Developer Console as mentioned here: **Query Plan Tool**.
Custom index on checkbox field values which occur least frequently would provide the most benefit for performance, and would generally indicate a selective query.

A Custom Index can't be created on these types of fields:

- Multi-select Picklists.
- Currency fields in a Multicurrency Organization.
- Long text fields.
- Binary fields (fields of type blob, file, or encrypted text).

Note: New data types, typically complex ones, may be added to Salesforce and fields of these types may not allow custom indexing.

Typically, a custom index won't be used in these cases:

- The value(s) queried for exceeds the system-defined threshold mentioned above.
- The filter operator is a negative operator such as NOT EQUAL TO (or !=), NOT CONTAINS, and NOT STARTS WITH.
- The CONTAINS operator is used in the filter and the number of rows to be scanned exceeds 333,000. This is because the CONTAINS operator requires a full scan of the index. Note that this threshold is subject to change.
- When comparing with an empty value (Name != '').

Note: There are other complex scenarios in which Custom Indexes won't be used. Contact your Salesforce representative if your scenario isn't covered by these cases or if you need further assistance with non-selective queries.

Examples of Selective SOQL Queries

To better understand whether a query on a large object is selective or not, let's analyze some queries. For these queries, we will assume there are more than 200,000 records (including soft-deleted records, that is, deleted records that are still in the Recycle Bin) for the Account sObject.

Query 1:

```
SELECT Id FROM Account WHERE Id IN (<list of account IDs>)
```

The WHERE clause is on an indexed field (Id). If SELECT COUNT() FROM Account WHERE Id IN (<list of account IDs>) returns fewer records than the selectivity threshold, the index on Id is used. This will typically be the case since the list of IDs only contains a small amount of records.

Query 2:

```
SELECT Id FROM Account WHERE Name != ''
```

Since Account is a large object even though Name is indexed (primary key), this filter returns most of the records, making the query non-selective.

Query 3:

```
SELECT Id FROM Account WHERE Name != '' AND CustomField__c = 'ValueA'
```

Here we have to see if each filter, when considered individually, is selective. As we saw in the previous example the first filter isn't selective. So let's focus on the second one. If the count of records returned by SELECT COUNT() FROM Account WHERE CustomField__c = 'ValueA' is lower than the selectivity threshold, and CustomField__c is indexed, the query is selective.

Query 4:

```
SELECT Id FROM Account WHERE FormulaField__c = 'ValueA'
```

The following rules have to be true in order to index a Formula Field:

- The formula contains fields from a single object only (not relationship fields).
- The formula field doesn't reference any non-deterministic functions (e.g. SYSDATE).
- The formula field doesn't reference any non-supported fields for including in indexes. This list isn't documented anywhere specifically (there are lots of special cases), but in Spring 12(176), createdById was non-supported, but in Summer 12 (178), it is supported. Same story for CreatedDate.
- The formula field doesn't contain references to Primary Keys (e.g Id)
- The formula field does not use TEXT(<picklist-field>) function
- If the formula references any Lookup fields, the field must not have the option "What to do if the lookup record is deleted?" set to "Clear the value of this field."

If one of the condition above is false, the "Add index" does not display on the field.

Note: The performance of the SOQL query will be analyzed by Salesforce Support, if it passes the criteria, then the index will be done.