Efficient Selective SOQL queries

Many of us would have faced issues writing SOQL queries when the D/B returns huge amount of data. Developers receive an error message when a non-selective query in a trigger executes against an object that contains more than 100,000 records.

A similar situation had arisen in our project where we were facing the **"TimeOut Issue"** since our query was returning million of records (business requirement). Due to this, the performance was badly hit and the system used to time out owing to Salesforce threshold standards. To overcome this situation, we made our query selective enough to limit the searchable records and got the required fields indexed by **Salesforce Support team**.

For good performance, SOQL queries must be selective (particularly for queries inside of triggers). To avoid long execution times, non-selective SOQL queries may be terminated by the system.

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 LastModifiedDate), and custom fields marked as External ID or Unique.
- Salesforce.com Support can add custom indexes on request for customers.
- A **custom index can't be created on these types of fields**: formula fields, multi-select picklists, currency fields in a multicurrency organization, long text fields, and binary fields (fields of type blob, file, or encrypted text.) Note that 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 != '')**

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 100,000 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.

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 (crossing the threshold), making the query non-selective.

Query 3:

SELECT Id FROM Account WHERE Name != '' AND CustomField__c = 'AnyValue'

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

Query 4:

SELECT Id FROM Account WHERE FormulaField__c = 'AnyValue'

Since a formula field can't be custom indexed, the query won't be selective, regardless of how many records have actually 'AnyValue'. Remember that filtering on a formula field should be avoided, especially when querying on large objects, since the formula needs to be evaluated for every Account record on the fly.