Database Index Costs in the Cloud-Based Multitenant Architecture on the Salesforce Platform

Stanisław Zań

Abstract

How in the multitenant cloud architecture of the Salesforce platform, different field configurations available to administrators impact database index usage, query costs, and data manipulation language operations.
Results show that while ”External ID” and ”Unique” settings consistently create indexes with varying performance effects, the ”Required” flag unexpectedly affects query costs in a meaningful manner. Results highlight the need for case-by-case tuning using the Salesforce Query Planner.

Introduction

Database tuning is one of the techniques used to achieve better performance and lower response time in the running software. In relational-based SQL databases, it requires from tuning person a deep understanding of the underlying database engine and its optimizers, as well as the end users' use cases and what operations they perform in the application.
Salesforce is a Custom Relationship Management (CRM) application, running in a cloud environment using a multitenant architecture, which means that servers' resources are shared between multiple running users at the same time.
The database that the end-user operates with does not have a traditional structure with tables representing real-world entities, but it's being virtualized on demand, based on the stored metadata, data and pivot tables. This approach makes the traditional performance-tuning method not suitable to implement on the Salesforce instance.
The aim of this paper is to look into how field's different configuration options, available for the Administrator in the Setup panel, can affect database indexes, their influence on the query costs, and Data Manipulation Language (DML) operations' markup.

Methodology

Tests were conducted using 1.5 million rows of randomly generated data using Faker's library.
Standard Account object with private sharing settings and no additional functionalities enabled was used.
Fields of different datatypes and configurations were tested, clearing the database between each of the configuration tests.
Insertion and deletion operations were performed using Salesforce's Bulk API 2.0, and their time was taken from Setup's Bulk Data Jobs view. Indexes performance were retrieved from the Salesforce Query Planner using a dedicated REST endpoint on the Salesforce platform.
Query Costs were retrieved for different types of queries, as proposed in the Database Tuning book.
Salesforce Production Trial instance was used, received by requesting Feedback Management Free Trial and deleting all preinstalled customizations.
The database server type was running on the Amazon Web Services (AWS) infrastructure, called Hyperforce.

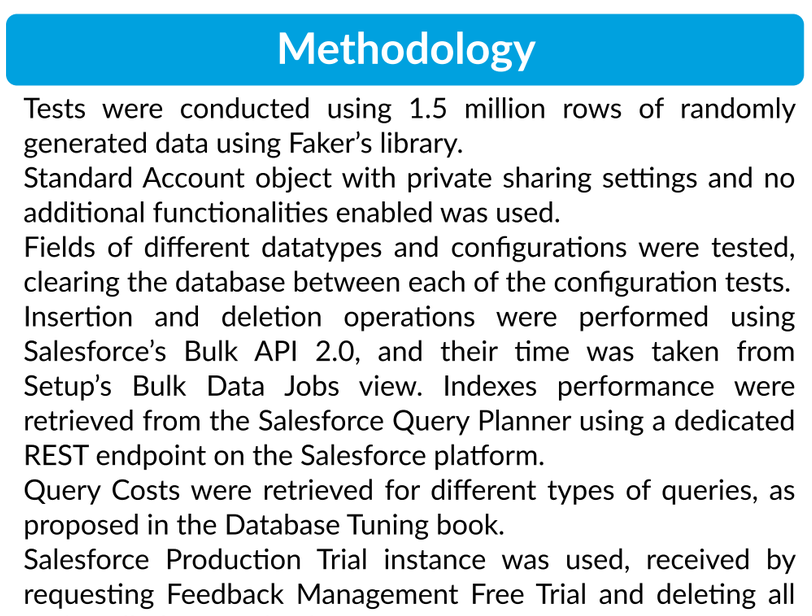

Figure 2 - Metadata architecture in the Salesforce Platform

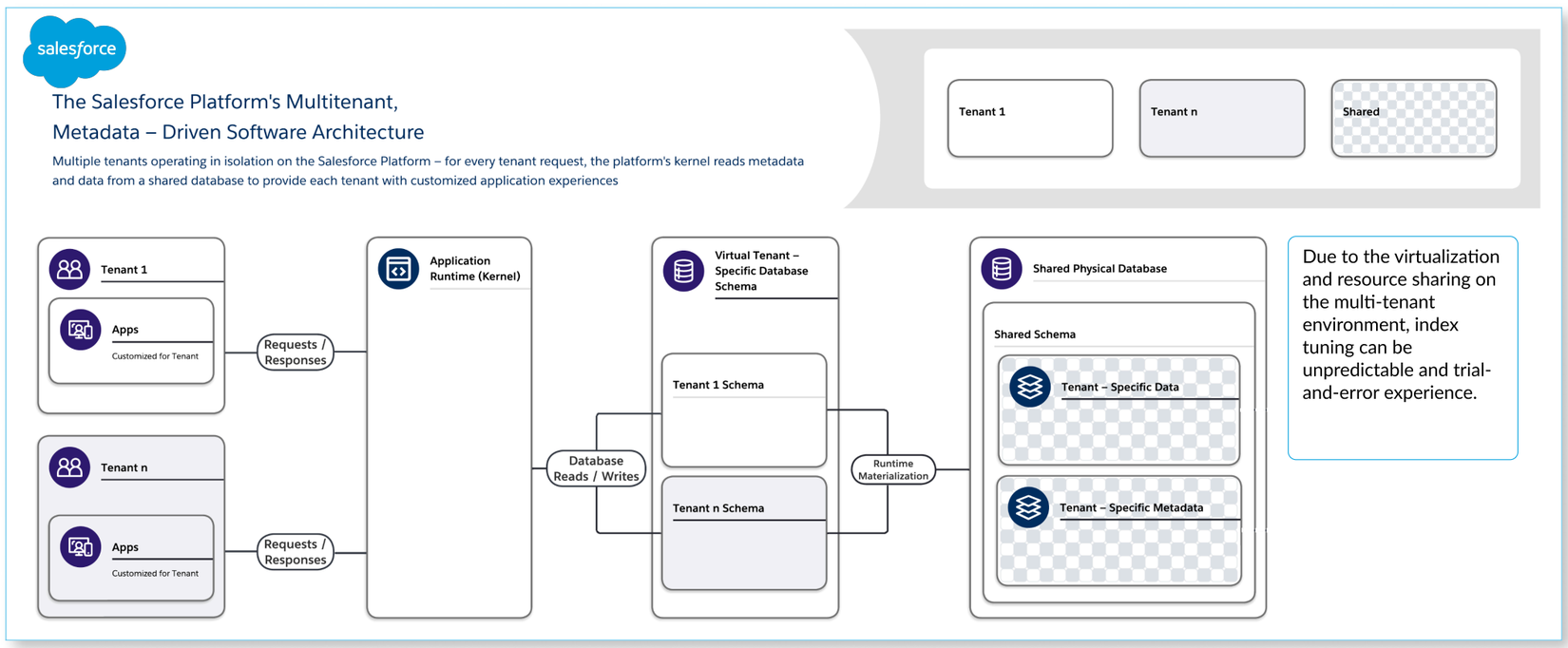

Figure 1 - Multitenant and virtualization schema of Salesforce Platform

Results

A. DML Times

Bulk API 2.0 does not allow using insert operation, providing upsert instead, however without existing database and present ID in the input file, upsert works exactly as insert and it will be named as it later on the paper.
Inserting and deleting records using Bulk API 2.0 showed significant differences in execution time depending on the field configuration. Fields marked as Required in most cases resulted in slower DML operations. This was especially noticeable for fields of type Text and Email, where required validation added a hidden computational cost.
On the other hand, configurations involving External ID and Unique options often performed better than expected, with some indexed fields showing shorter insertion times than non-indexed ones. This contradicts the common assumption that indexing introduces overhead for write operations.

Table I
The most significant comparison of insertion times between indexed and not-indexed configurations on Standard Account Object

Field Api Name	*Insertion* Diff to No Index
Email_required__c	-11.66%
Number_short_externalId_nulls__c	-9.36%
Number_short_unique_external__c	-6.39%
Email_unique_external__c	-6.28%
Number_short_all__c	-5.99%
Number_short_unique__c	-5.63%
Email_externalId__c	-5.14%
Email_unique__c	-5.09%
Text_autonumber_unique_extId_case_sensi__c	5.32%
Number_short_unique_required__c	6.56%
Text_email_unique_case_insensitive__c	6.57%
Text_autonumber_required_externalId__c	7.10%
Text_autonumber_unique_req_case_insen__c	8.20%
Text_unique_required_insen__c	8.34%
Text_autonumber_unique_req_case_sensi__c	11.68%
Text_unique_required_sensi__c	12.56%
Text_email_required__c	49.02%
Text_email_required_externalId__c	51.93%
Text_email_unique_required_insen__c	52.42%
Text_email_all_case_insensitive__c	53.51%
Text_email_unique_required_sensi__c	53.85%
Text_email_all_case_sensitive__c	54.04%

Table II
The most significant comparison of deletion times between indexed and not-indexed configurations on Standard Account Object

Field Api Name	*Deletion* Diff to No Index
Text_email_required__c	-21.03%
Text_email_all_case_insensitive__c	-18.39%
Text_email_all_case_sensitive__c	-18.31%
Text_email_required_externalId__c	-17.81%
Text_email_unique_required_sensi__c	-17.77%
Text_email_unique_required_insen__c	-17.58%
Email_required__c	-5.55%
Text_unique_case_insensitive__c	5.01%
Email_unique_required__c	5.18%
Text_unique_required_sensi__c	5.43%
Number_externalId_nulls__c	5.56%
Text_unique_externalId_sensi__c	6.06%
Email_externalId_nulls__c	6.22%
Text_autonumber_unique_extId_case_sensi__c	7.27%
Text_unique_required_insen__c	8.10%
Text_autonumber_unique_req_case_sensi__c	10.58%

Table III
Example result with the comparison of query cost between setting up a Required configuration on not indexed field on Standard Account Object

Query	Email, No Config	Email, Required	Diff
Query all	0.6773	1.0640	57.1%
Point Query	0.6773	0.9255	36.6%
Point Query on not existing records	0.6773	0.9255	36.6%
Multipoint Query	0.6773	0.9255	36.6%
Range Query on < 'rebeccakrueger@gmail.com'	0.6773	1.0640	57.1%
Range Query on > 'margarethenson@gmail.com'	0.6773	1.0799	59.4%
Range Query on < 'margarethenson@gmail.com'	0.6773	1.0640	57.1%
Range Query on > 'margarethenson@gmail.com' *AND* < 'rebeccakrueger@gmail.com'	0.6773	0.9849	45.4%

B. External ID, or Unique, or both

Field configurations using the External ID or Unique options provided an indexing benefit, as observed through the Query Plan Operation Type and costs. Fields with either configuration in most cases appeared with the lower cost units in *SELECT* queries, confirming that an index was indeed in use. In all cases, External ID and Unique results were different from each other. The combination of both of them yielded another result, which could be higher or lower than a single configuration of them. On every datatype used, different combinations of External ID and Unique were most optimal. In text fields, two different types of Unique are available - Case Sensitive, and Case Insensitive. During testing, both of them resulted in different query costs and DML times. The behaviors above confirm that these configurations are treated differently by the database's internal engine and query optimizer, and no general rule emerged during the testing. Each case in the field configuration should be tuned with Query Planner, considering all possible combinations, to get the most performative results.

C. Required fields

Fields configured as Required did not generate an index, which follows the Salesforce's documentation. However, the Required flag had a measurable impact on performance, on both queries and in the DML operations. In most cases, adding the Required configuration to the field resulted in higher query costs, but it is not a strict rule.
Thus, while the Required setting should be used to improve and maintain the data integrity, it impacts the query and DML performance, introducing unexpected build-up in high-volume scenarios.

Conclusion

This paper discusses three field configurations and their effect on database index costs in the Salesforce platform.
The ”Required” configuration does not create an index, but it affects data manipulation time and query costs, making it worse in most of the cases. The negative effect of the field's requirements on the *SELECT* statement is an unexpected result, because it should have either a positive effect, or no effect at all.
Both ”External ID” and ”Unique” field configurations create indexes, but their query costs and DML performance are not the same. Mixing two of the configurations resulted in another outcome, different from the alone configurations. It implies that these indexes are treated differently by Salesforce's database engine. The exact influence or pattern on the performance could not be determined during the testing.
Besides, some indexed configurations had better insertion times than the non-indexed ones, which is another unexpected result. Following the above, it is hard to predict the effect of the field's configuration on Salesforce's platform performance, therefore any index tuning should be investigated with the Query Plan tool, and all the possible combinations should be considered to achieve the best performance.

Table IV
Recommended Index on Standard Object type - Unique, External ID or both. Required configuration was not accounted.

Field	Recommendation on field's configuration
Email	Unique + External ID
Text	Unique Insensitive + External ID
Number Short	Unique + External ID
Number Long	Unique

Add me at LinkedIn!