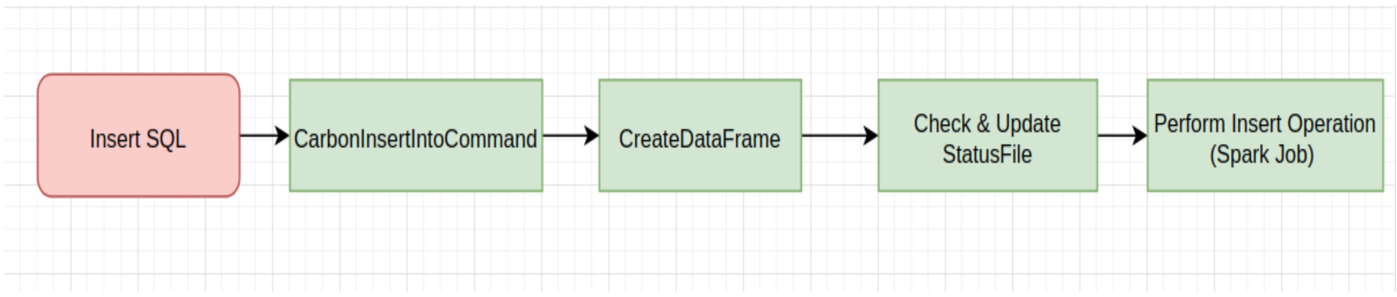
Simple insert performance improvement

Author: Akshay Kumar N
Date: 02-02-2021
Version: V1

Background:

As Carbon is closely integrated with spark, insert operations in carbon are done using spark API. This inturn fires spark jobs, which adds various overheads like task serialization cost, extra memory consumption, total job execution time in remote nodes, shuffle etc.

The **current flow** is as below


Insert SQL
CarbonInsertIntoCommand
CreateDataFrame
Check & Update StatusFile
Perform Insert Operation (Spark Job)

1. Plan level changes and creating a dataframe
2. Check and update the status files
3. Perform insertion, which basically performs an insert with the existing dataframe.

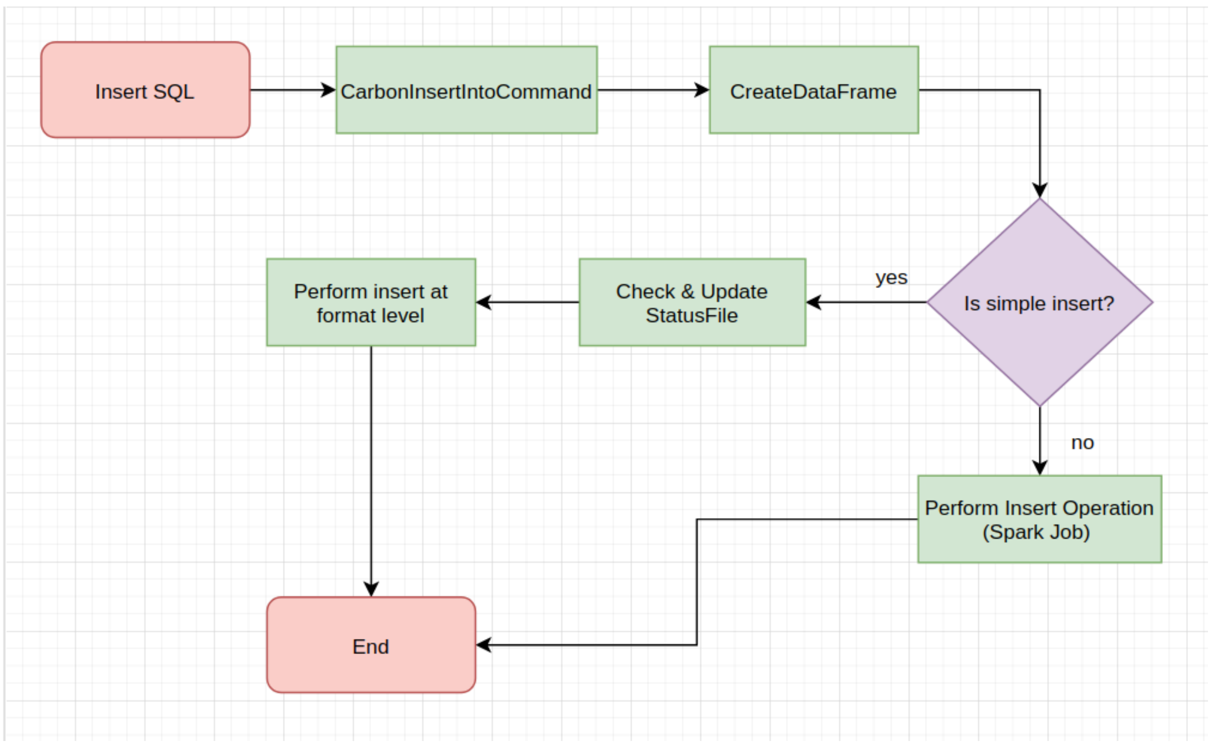
Here, in step 3 spark Jobs are created to perform these functions and add overhead as mentioned above. The existing logic is best for huge insertions. But for simple insert operations, performance will be slow.
Simple insert example -
INSERT INTO test_table SELECT 1,2

Proposed Solution:

SDK is implemented at the carbon file format level, so we can reuse the same in case of simple insert operations and avoid spark by using simple java code.

Implementation:

The new modified flow for simpler inserts is as below.



Insert SQL
CarbonInsertIntoCommand
CreateDataFrame
Is simple insert?
yes
Check & Update StatusFile
Perform insert at format level
no
Perform Insert Operation (Spark Job)
End

1. Check if it is a simple insert
2. If yes, flow goes to a separate class which handles simple insert, creation of RDD which basically creates the tasks based on the segment number and then each task makes a call to the SDK to perform insert.
3. Existing logic will check and update the table status files.

   How to get the segmentMetadataInfo which we store in the segment file?
   In the current flow, min-max is updated and written to the segment file with the help of an accumulator sent from the driver to the executor. But now we will have to read the footer of the data files and update the segment files, as we are not using spark.