Item 377
git_comments:

1. * * This method will initialize the thread pool size to be used for creating the * max number of threads for a job
2. * * this variable is to enable/disable prefetch of data during merge sort while * reading data from sort temp files
3. * * This method will delete the local data load folder location after data load is complete * * @param loadModel * @param segmentName
4. * * This method will get the store location for the given path, segment id and partition id * * @param carbonStorePath * @param dbName * @param tableName * @param partitionCount * @param segmentId
5. form local store location
6. * * This method will get the store location for the given path, segment id and partition id * * @return data directory path
7. * * This method will read the records from sort temp file and keep it in a buffer * * @param numberOfRecords * @return * @throws CarbonSortKeyAndGroupByException
8. notify the consumer thread when index at which object is to be inserted becomes equal to current index from where data has to be picked for writing
9. insert the object in array according to sequence number
10. * * array of blocklet data holder objects
11. * * a private class that will hold the data for blocklets
12. * * This method will close writer execution service and get the node holders and * add them to node holder list * * @param service the service to shutdown * @throws CarbonDataWriterException
13. * * Computes a result, or throws an exception if unable to do so. * * @return computed result * @throws Exception if unable to compute a result
14. * * a private class which will take each blocklet in order and write to a file
15. * * This method will iterate through future task list and check if any exception * occurred during the thread execution * * @param taskList * @throws CarbonDataWriterException
16. * * This method will reset the block processing count
17. * * Producer which will process data equivalent to 1 blocklet size
18. * * integer that will be incremented for every new blocklet submitted to producer for processing * the data and decremented every time consumer fetches the blocklet for writing
19. * * Consumer class will get one blocklet data at a time and submit for writing
20. * * index from which data node holder object needs to be picked for writing
21. * * flag to check whether the producer has completed processing for holder * object which is required to be picked form an index
22. if entry count reaches to leaf node size then we are ready to write this to leaf node file and update the intermediate files
23. * * number of cores configured
24. * * semaphore which will used for managing node holder objects
25. wait until all blocklets have been finished writing
26. * * data directory location in carbon store path
27. * * This class will hold the holder objects and manage producer and consumer for reading * and writing the blocklet data
28. reset current index when it reaches length of node holder array
29. * * counter that incremented for every job submitted to data writer thread
30. * * flag to check whether all blocklets have been finished writing
31. * * @param nodeHolder * @param index
32. * * @return a node holder object * @throws InterruptedException if consumer thread is interrupted
33. still some data is present in stores if entryCount is more
34. if node holder is null means producer thread processing the data which has to be inserted at this current index has not completed yet
35. * * carbon data directory path
36. * * executorService
37. * * complete path along with file name which needs to be copied to * carbon store path
38. * * This method will copy the given file to carbon store location * * @param localFileName local file name with full path * @throws CarbonDataWriterException

39. * * This method will rename carbon data file from in progress status to normal * * @throws CarbonDataWriterException
40. open channel for new data file
41. * * file size at any given point
42. * * data directory location in carbon store path
43. * * This method will read the local carbon data file and write to carbon data file in HDFS * * @param carbonStoreFilePath * @param localFilePath * @param bufferSize * @param blockSize * @throws IOException
44. * * This method will return max of block size and file size * * @param blockSize * @param fileSize * @return
45. * * Computes a result, or throws an exception if unable to do so. * * @return computed result * @throws Exception if unable to compute a result
46. block size should be exactly divisible by 512 which is maintained by HDFS as bytes per checksum, dfs.bytes-per-checksum=512 must divide block size
47. rename carbon data file from in progress status to actual
48. * * This method will close the executor service which is used for copying carbon * data files to carbon store path * * @throws CarbonDataWriterException
49. * * If node holder flag is enabled the object will be added to list * and all the blocklets will be return together. If disabled then this * method will itself will call for writing the fact data * * @param holder

git_commits:

1. **summary:** [Issue-324] Data loading performance optimization (#444)
   message: [Issue-324] Data loading performance optimization (#444) 1. Enabled prefetch - code modifications done to make prefetch work according to new code 2. Moved mdkey processing code to a different thread 3. Moved copying of file from local to hdfs as soon as file is completed writing

github_issues:

github_issues_comments:

github_pulls:

1. **title:** [CARBONDATA -541] Tescases for dictionary subpackage in processing added
   body:

github_pulls_comments:

1. Build Failed with Spark 1.5.2, Please check CI
   http://136.243.101.176:8080/job/ApacheCarbonPRBuilder/228/

github_pulls_reviews:

jira_issues:

jira_issues_comments: