When and where to trigger the compact?

How to save the deleted cells?

How to select the compacted mob files?

How to handle the mob file across regions?

1 Overview

The mob compaction is executed after the major compaction in each store.

In each major compaction, two files are generated, one is the new store file, the other is the del file with all deleted cells (not the delete markers, but all the deleted cells).

Since in the minor compaction, there're also cells deleted, we need to customize the minor compaction for the mob column to retain all the deleted cells and only delete them in the major compaction.

After the major compaction is finished, we should select the compacted mob files, and do the mob compaction. The candidate mob files should be:

1. Invalid mob files which have too many deleted cells.
2. Small mob files.
3. HFileLink, all the mob hfile links should be rewritten to new mob files.

2 How to find the candidate files?

Each store (which contains the mob reference cells) will do the mob compaction after the major compaction. The candidates should be the mob files belong to the current store. So now the question is how to find the mob files belongs to the current store.

Currently the prefix of each mob file is the MD5 of the start key of each region, so we could scan all the existing mob files including the mob files and del mob files, find all the files that have the same start key with the current store. They're the candidates.

3 How to find the invalid mob files?

After each hbase compaction, there's a new del mob flle generated. There might be del mob files generated in previous major compaction, but the number of the del files is supposed to be small. It means there might be more than one del file for a region when the mob compaction is started.

1. Find all the del files owned by the current region in the found files in Chapter#2.
    a. If the del files owned by a region is more than three (we could make it configurable), merge them to one and remove the old ones. Open a scanner to this new del file then.
    b. If the number is less than 3, we could open scanners to these del files.
2. Scan the del files to find the files that contain these deleted cells, if deletedCellsSizeInOneMobFile/fileSize > ratio, this mob file is invalid.

3. Open scanners to the del files and the invalid file, rewrite existing cells to a new mob file.
4. Archive the invalid file.
5. When all the invalid files are done, archive the del files.

4 How to find the small files?

Sort the candidates files got in chapter#2, if the fileSize < threshold, this mob file is a small one.

Select all or part of them, merge them into a bigger one, and archive the selected small files at last.

5 How to handle split cases?

A normal mob file, even a del mob file might cross multiple regions. If these files are valid or big enough, we don't touch them. But if they're invalid or too small, how to handle them in mob compaction?

When a file across regions needs to be compacted, we should create reference files for it in directories of daughter regions. Each reference file is handled in the mob compaction of each daughter region.

1. Region#1 selects three mob files to compact (they has the same prefix with the start key of Region#1), all of them cross regions Region#1 and Region#2, Region#1 creates three reference files in a directory of Region#1, and create other three reference files in a directory of Region#2.
2. Region#1 starts to compact the reference files and rewrite the parent files to a new file. When the compaction is finished, it deletes its references files and checks whether the selected files across regions are still referenced by Region#2.
    a. If yes, leave the selected mob files there.
    b. If no, archive selected mob files.
3. Region#2 will handle its reference files in the mob compaction of itself, and archive the selected mob files if they're no longer referenced by other regions.