Item 248
git_comments:

git_commits:

1. **summary:** Simpler and safer db open/closing in view group servers
   message: Simpler and safer db open/closing in view group servers This makes the opening and closing of databases in the view group server to be more friendly with the db reference counting system, avoiding more potential db file leaking after compaction, as we currently open a database in one process and use it on another process (view compactor, view updater). This significantly reduces the chances of failure when compacting very large views as discussed in COUCHDB-994. This relates to COUCHDB-926 and COUCHDB-994. git-svn-id: https://svn.apache.org/repos/asf/couchdb/trunk@1138796 13f79535-47bb-0310-9956-ffa450edef68
   label: code-design

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
2. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
3. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
4. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
5. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
6. **summary:** Crash after compacting large views

description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.

7. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
8. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
9. **summary:** Crash after compacting large views
   description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
10. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
11. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
12. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
13. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
14. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
15. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.

16. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
17. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
18. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
19. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
20. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
21. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
    label: code-design
22. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
23. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
24. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
25. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the

compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.

26. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
27. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
28. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
29. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
30. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
31. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
    label: code-design
32. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
33. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
34. **summary:** Crash after compacting large views
    description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.
35. **summary:** Crash after compacting large views

description: The database has over 9 million records. Several of the views are relatively dense in that they emit a key for most documents. The views are successfully created initially but with relatively large sizes from 20 to 95G. When attempting to compact them, the server will crash upon completion of the compaction. This does not occur with the released 1.0.1 version but does with the 1.0.x svn version. I'll attach example logs. Unfortunately they are level error and may not have enough information.

jira_issues_comments:

1. error logs
2. I had a chance to delve into this and I think there's a very real bug here. The problem is two-fold: 1) The header for the compacted index is written much later than it could be. 2) We don't try to use the .compact index storage if the primary storage is missing. We can fix the first problem if we simply write a header in the view compactor process before we send the compact_done message to the view group. The current system doesn't write the header until the view group processes the 'delayed_commit' message that it sends to itself when it switches the file over. If the group process closes at any point in the interim we're going to reset. The second problem seems a bit tricky on the face of it, but I think it will work out OK. View group compaction, unlike database compaction, is not all that easy to resume. We don't have access to detailed sequence numbers for each piece of the tree; all we have is a single current_seq for the entire group. But that's alright. I think all we need to do is implement the fix for #1, and then change the view group process to check for the presence of a .compact file if the primary storage is missing. Then one of 3 things can happen 1) no .compact file, so we create a new file and index from scratch 2) .compact file is partially written, but if we seek backwards to find the header it will still have current_seq = 0, so we'll index from scratch 3) .compact file is fully written and has a valid current_seq. We rename it and have a successful recovery. The second case can potentially block the view group for a long period of time as it scans backwards through the file. The third case is obviously the one we have to be the most careful about, particularly when it comes to database deletion. It looks to me like couch_view:do_reset_indexes/2 takes care of .compact files as well as primary storage, so I don't think adding recovery changed our behavior at all there.
3. Raising for 1.2
4. We are seeing the same problem with CouchDB release 1.0.2 on top of Erlang R14B and 64bit RHEL 5.x. This is how it happened: 1) We installed a new version of a design doc, which of course forces a full view index update. This took 30h to complete. 2) Since the brand new view was 200GB (should be 1/4 of that or less), a couple of days later we triggered view compaction. This also took 30h to complete. It might also be worth mentioning here that we have a constant flow of updates/inserts to the database during the compaction, but it's only something like 1000 updates/inserts per *minute*. 3) Right after the view compaction was complete, couchdb restarted itself and started the full view index update from scratch. To me our logs from the incident look very similar to the ones posted by Bob Clary. Although I cannot post you the full logs from our client's production system, find below the beginning of the two consecutive (but clearly separate) log messages: First: [Tue, 10 May 2011 03:50:28 GMT] [error] [<0.153.0>] ** Generic server <0.153.0> terminating ** Last message in was {'$gen_cast', {compact_done, {group, <<66,241,166,10,26,2,78,192,180,77,204,69,249,98,70, 23>>, Second: [Tue, 10 May 2011 03:50:30 GMT] [error] [<0.153.0>] {error_report,<0.31.0>, {<0.153.0>,crash_report, [[{initial_call, {couch_view_group,init,['Argument__1']}}, {pid,<0.153.0>}, {registered_name,[]}, {error_info, {exit, {timeout,{gen_server,call,[<0.156.0>,{drop,<0.153.0>}]}}, [{gen_server,terminate,6}, {proc_lib,init_p_do_apply,3}]}}, {ancestors, [couch_view,couch_secondary_services,couch_server_sup, <0.32.0>]}, {messages, [{'EXIT',<0.32379.11>,
5. Hi, I believe this might have been fixed in the branch 1.0.x (upcoming 1.0.3 release). Do you have a chance to test this branch?
6. Filipe, I'm pretty sure this is still an open issue. It looks like Jaako's view group crashed with a timeout trying to decrement the ref counter, and we still use a 5 second timeout there. But more importantly, the root cause of the view indexing reset after a crash has not been addressed.
7. Adam, I think it might have been. Since the ref counter call is made after the db was compacted (by analysing the stack trace) - this is no longer done after: https://github.com/apache/couchdb/commit/f18f4e31c5e7bb242d6683a0ad9e5ae3b9b1acd1 The 5 seconds timeout is another issue, that I happened to change on trunk today - i've set it to infinity, I've been using this timeouts for some weeks now on local branches and it seems to work ok - http://svn.apache.org/viewvc?view=revision&revision=1104168
8. Hi Filipe, did you happen to read my long analysis of the bug earlier in this ticket? We don't write the header in the compacted view index file before triggering the switchover, so _any_ crash in the time

before the header is eventually written will result in a complete reset. I don't believe your patch addresses this (though I do think it was a good thing to do in its own right).

9. Adam, I was thinking the issue is related to database compaction, not view compaction. Either way, when compact_done is received, there's one less couch_db:close call (for the old group's db), therefore it might help this particular issue. I
10. I believe view compaction is also an issue to be handled.
11. I ran into the same problem today on a CouchDB 1.1.0 (Erlang R14B03). It was reproducible at least three times in a row. I've attached the logs of the first incident in 2011-06-11-couchdb.log
12. I've applied the patch from http://svn.apache.org/viewvc?view=revision&revision=1104168 onto CouchDB 1.1.0 and tried to compact again. The result was that the compaction finnishes, an error is reported in the log (2011-06-13-couchdb.log) and the .view files are not replaced with their corresponding .compact.view files.
13. Thanks Stefan. If you apply as well this patch http://friendpaste.com/51pU8KPuysD9fT3ncChipc I believe you'll no longer see errors/stack traces in the log. Do you think you can test it? :) Thanks again
14. Stefan, try this patch instead (couchdb-994-db-open-logic.patch). This is something I've been wanting to do for some time now, to avoid opening the databases in one process and then pass them to another, as this goes a bit against the ref counting mechanism. This is a complement to Adam's observations, and together with previous patch (infinity timeout for the ref counter calls), should eliminate this view compaction issue. Adam, do you think you can give a quick look to this patch? Thank you both
15. The general idea looks right, but it doesn't address the core problem in this ticket, namely that a well-timed crash can reset the view index. I'll take a closer look at the particulars of the patch later today.
16. Filipe, thanks for providing the patch! I just wanted to test it, but it seems it doesn't apply on CouchDB 1.1.0. git apply --check couchdb-994-db-open-logic.patch error: patch failed: src/couchdb/couch_view_compactor.erl:36 error: src/couchdb/couch_view_compactor.erl: patch does not apply error: patch failed: src/couchdb/couch_view_group.erl:386 error: src/couchdb/couch_view_group.erl: patch does not apply Could you maybe provide a rebased patch for CouchDB 1.1.0?
17. Stefan, here's a patch for 1.1.x (couchdb-994-db-open-logic-11x.patch). Adam, right, I didn't try to fix any of those 2 issues. Leaving them for you :) Thank you both again
18. I'm still seeing errors in the logs and compaction still fails with complete .compact-files. I've attached the new logs in 2011-06-15-couchdb.log
19. Another compaction job just produced another error. Logs are attached in 2011-06-15-2-couchdb.log
20. Thanks Stefan, we will look into it
21. **body:** Stefan, I think it's failing for you because you duplicates in your view, which relates to issue COUCHDB-999. The exit reason you get in the log is likely to be related. Unfortunately it's not very readable. Can you also apply this one line patch http://friendpaste.com/3rFccubDlgZNuFQ7hfpLNr which will make the exit reason message more readable? If it tells you compaction failed because there are duplicates, the only solution is to recreate the index (COUCHDB-999). thanks again
    label: code-design
22. It seems you are right about the duplicate problem (see 2011-06-16-couchdb.log). I'll recreate the index files and report back.
23. After recreating the index files, compaction worked without any problems so far. Thanks!
24. should also be applied to 1.1.x
25. It would be great to have a fix for 1.0.x as well - this is hitting us hard in production...
26. I had the same problem with one of my databases (CouchDB 1.0.1). I was able to work around the problem doing the following: 0. compaction failed 1. sudo /etc/init.d/couchdb stop 2. mv /var/lib/couchdb/mydb.couch.compact /tmp 3. sudo /etc/init.d/couchdb start 4. run compaction again (now it works)
27. Here's a patch to address issue #1. It ensures that the compacted view group has a valid header before handing it over to the couch_view_group process. This means that the index will not be reset if the server crashes after the file has been renamed but before the next delayed_commit message arrives. It does not address the possibility of using a .compact view index if the main index file has been deleted, which could happen if the server crashes in between the delete and the rename.
28. I again has a compaction crash, this time with a current 1.1.x branch. It seems the view indexer and the compaction have been running in parallel. As soon as the indexer finished, the compaction finished too and crashed. The current seq was around 90M, the the view probably contained <10 rows.
29. +1 to Adam's patch, lgtm.
30. Did you consider adding an fsync bracket around that? Other than that, +1

31. **body:** @davisp Probably not a bad idea, though of course we never fsync view group files before writing the header in normal operations :/ I think that's a topic for a separate ticket.
    label: code-design
32. @Adam Fair point. I'll open one now so I don't forget.
33. I believe this is resolved. The couch_db:close(nil) path has been eliminated and we know write a header before flipping the view file, which largely mitigates the category of problem itself.