

Consistency Check in OpenLSD

Author: Frédéric Brégier under LGPL

CONSISTENCY CHECK IN OPENLSD.....	2
THE CHECK OF CONSISTENCY IS DONE AS THE FOLLOWING:	2
USAGE OF CONSISTENCY CHECK CONSIDERATIONS	3

Website: <http://openlsd.free.fr/en/OpenLSD.html>

Consistency check in OpenLSD

To check consistency of files, there is one way (very very non efficient) that uses one client that connects to one OpenLSD Server and asks for each reference in the database the existence or not of each document using the Exist function with the MD5 key from the database. Obviously, it can not be used in a huge storage situation.

So there is another way that must be executed on the OpenLSD Server host itself and that use Database optimisation to run quickly as possible such consistency check. There are two ways:

- One using a reference date where only new documents inserted after this date are tested. Only imported documents can be tested, deletion is out of scope. This method can be used frequently in production. Optionally, a second date can be used to limit new documents before this date. This option allows suspending a short time all insertions/deletions, for instance 1 minute before and after launching the consistency check. When using this option, only one Legacy at a time can be checked (LSDCheckInDbThreadedDualLimit).
- One using no reference date, so testing everything, even deleted documents. Obviously this method is longer than the previous one but is more accurate. It should only be used in case of (suspicion of) crash.

Both methods can have repair options:

- repairing nothing,
- repairing only database reference (by default, delete DB entry if no file in OpenLSD in reference exist ; in ML mode the option -optfix allows to re-import file if it does not exist from a valid OpenLSD Server as source),
- repairing only OpenLSD files (delete file in OpenLSD if no reference in DB exist),
- repairing only on update MD5 key in database from OpenLSD files,
- or repairing everything.

I strongly recommend running first check without any repair options on, to check if there is any problem.

Both methods can check only existence of files (faster, about 10 time faster), or also it can check MD5 key too.

The check of consistency is done as the following:

- 1) First, for each Legacy (sequentially) or for the specified Legacy, the list of existing files is created, computing or not the MD5 key accordingly to the options.
 - a. One version uses a full java version.
 - b. Another one uses a C coded version that is about 3 times faster than Java version. However, this can change from one system to another, so one should make some tests before choosing one implementation.The final choice can be done at runtime.
This listing is done using parallel discovery (multiple threads/processes) to enable faster resolution. Each listing in parallel is stored in one file and contains LSD indexes and MD5 key (if any).
If the first date option is allowed, only newer directories/files are testing.
If the second date option is allowed, only directories/files between the two dates are testing.
- 2) This file is then stored in a specific temporary table in the database using efficient solution when they exist (SQL loader for instance).

- 3) When all information for one Legacy is stored in the database, a computation is done to get the non-intersection of existing reference from/to the database and to/from the OpenLSD Legacy accordingly to the options (MD5 key or not, reference date or not) and to get the reference that are in both containers but with different MD5 key.
- 4) According to the options (repair, listing), repairs are done or not (database update/delete, OpenLSD direct files delete, OpenLSD re-import from another valid source OpenLSD Server in ML mode), and list of found problems. A resume is always given.
- 5) After all corrections are done (if any), a quick correction on database is done (compute of Storage size according to the referenced documents). This function can also be used alone without the consistency check.
- 6) If the repair option is given, the reference date for the given Legacy is updated. One option (**‘-update’**) can be passed to force the update of the date in case of no repair option is passed and if no error is found.

To check in context of ML support, one similar function exists to be executed on “clone” Legacies, except that no database delete or update can be done, only Legacy repairs can be done (deletion of false files).

Usage of consistency check considerations

Consistency check can be done with OpenLSD Server running or not. However, one should take care about that false response can be given on imported or deleted documents while the check and the server are running at the same time, both for past files and newly files considering that timestamps can differ between the storage timestamp and the database timestamp. So it is preferred to stop the OpenLSD Server while running the consistency check and to surely not run in repair mode the consistency check while the OpenLSD Server is running.

In order to limit the OpenLSD Server stop duration, one can use `LSDCheckInDbThreadedDualLimit`, which allows to only testing new files between the last successful check and the start of this new check. Doing this will allow the production to stop about 2 minutes, 1 minute before the check is started and 1 minute after the check is started, given 2 minutes of delay for the check to be fully consistent. You have to take care about date synchronisation between the OpenLSD Server (storage server) and the Database Server since both date must be consistent (at least under 1 minute of delay).

Consistency check without repair options can run during import or delete operations using the `DualLimit` version since no repair will be done. But using a consistency check with repair options (whatever the version `DualLimit` or ML subversions) during import or delete operations can lead to unpredictable behaviour like:

- One file is found as not present in database and so will be deleted,
- In the same time, one file is imported in the same place by an importer,
- Therefore the correct imported file can be deleted since the delete operation has a delay of operation and can occur not on the true file which needs to be deleted but on the correct file which was imported after the first pass of check.

ML versions and not ML versions have a very different behaviour on repair:

- When a document exists in DB but not in LSD:
 - In not ML version, the DB reference will be deleted since there is no copy left of this physical document.
 - In ML version, the LSD document will be imported from the specified valid source.

- When a document exists in DB but with a different MD5 in LSD:
 - In not ML version, the DB reference is updated from the MD5 computed with the physical document.
 - In ML version, the LSD document will be re-imported (replaced) from the specified valid source.
- When a document exists in LSD but not in DB:
 - Both in not ML version and in ML version, the physical file will be deleted since there is no information on the physical document in the database.

As a resume, Consistency check can be done frequently using no repair option, whatever ML or not ML versions, using DualLimit if one wants to limit the time of import or delete interruption, or using no DualLimit if one can interrupt the import or delete operations during the full time of Consistency Check. Probably in these cases, the option ‘-nomd5’ should be used in order to speed up this operation.

When a Consistency check reports an error, then one should totally interrupt the import or delete operations and then runs the Consistency Check again (with or without the ‘-nomd5’ option, with or without the DualLimit option) with repair option on (according to the error found).

If a global Consistency check should be done, the ‘-nodate’ option should be used with a not DualLimit version. This disables the date as limit (both low and up limit) so as to check everything. If it is known that the consistency default is limited to operations after the previous successful check, then one can use the standard Consistency check without the ‘-nodate’ option in order to check after the previous date of valid check, but still not a DualLimit version.