THE ALIFATAL SYNDROM

```
AliCDBEntry* AliCDBStorage::Get(const AliCDBId& query)
```

STORYLINE

- Since rev 38594 *(Feb. 2nd, 2010, «Changing AliInfo to AliFatal if the entry for a specific query is not found.»)* we're finding more and more places where this AliFatal is hurting us...

MCHVIEW

- GUI to display our «raw» calibrations and derived quantities like the statusmap

- User-driven. Users are not robots. They can ask for non-existing OCDB objects. That should not make the program crash...

RECO

- Case in point : MUON/Calib/RejectList
- If everything is OK, this object was not there at all in raw OCDBs
- Now we must provide a dummy (empty) object just to avoid a crash
- Plus it's now more difficult to know whether or not we ran with a *meaningfull* RejectList (we now have to open the object itself to see if it's empty or not...)

AMORE

- So far, we've tried our best to use the same QA code online and offline.
- But not everything that's done offline can be done online. Our code was coded to deal with that very fact
- For instance, MUON/Calib/OccupancyMap, which is computed online by a DA, can *not*, by definition, be available when AMORE is running on this same run...
- Was not a problem : our code was applying occupancy thresholds *if and only if* occupancy map was available.
- Not cutting on occupancy does not make the code crash, it may make it slower simply
- Now we are asked to provide a default object... Would need to change that object (adding a flag or whatever) to know that it is a fake (don't want to cut on values that were not computed, and how do I insure the consistency between real cuts and fake occupancies ?).
- Once we have a default object to get online running, offline reco will always be happy, even if the DA did not provide a meaningfull occupancy map... How bad is that ? Kind of defeat why the AliFatal was put in in the first place...

PROPOSAL

- Provide a way to disable the Get's AliFatal, e.g.
  - AliCDBManager::SetGetBehavior(Bool_t failIfAbsent=kTRUE);