Undocumented Error Behaviour

- Syscalls, including fsync(), have basically no real-world applicable error documentation
- Before 4.13 errors may not have been reported **at all**
- There's still no documentation explaining whether fsync() can be retried, etc

  https://wiki.postgresql.org/wiki/Fsync_Errors

Existing durability operations weirdly documented

- sync_file_range():

Warning

This system call is extremely dangerous and should not be used in portable programs. None of these operations writes out the file's metadata. Therefore, unless the application is strictly performing overwrites of already-instantiated disk blocks, there are no guarantees that the data will be available after a crash. There is no user interface to know if a write is purely an overwrite. On filesystems using copy-on-write semantics (e.g., btrfs) an overwrite of existing allocated blocks is impossible. When writing into preallocated space, many filesystems also require calls into the block allocator, which this system call does not sync out to disk. This system call does not flush disk write caches and thus does not provide any data integrity on systems with volatile disk write caches.

Error behavior inconsistent between fs/block/kernel version

- reads after IO errors due to buffered write yield different results for filesystems (pre-write contents, post-write contents)
- block devices can cause ENOSPC, even if FS is not full (thin provisioning), no way to differentiate
- types of storage influences retry behavior, and error behavior
- errors are not actually that uncommon on larger scale, and especially with network storage

No Guidance / Documentation how to achieve Durability

- Whatever a database does, some kernel dev will suggest that it’s the wrong approach
- Lack of documentation leads to proliferation of userspace approaches, and differing assumption of kernel side guarantees
- Most applications, including databases, are broken for some filesystem
- creation: fsync(fname); fsync(dirname);?
- atomic rename: fsync(oldfile); fsync(olddir); rename(); fsync(newfile); fsync(newdir); ?
- posix very imprecise / open to interpretation

Error Reporting Facility

- EIO/ENOSPC etc doesn’t allow to determine cause of errors
- No way (?) to query all IO errors for device
  - databases want to refuse writes / fail over upon error
  - regexes over logs only approach