Avoiding, Detecting, and Recovering From Corruption

- Robert Haas | PGCONF.IN 2020

Overview

- Definition
- Causes
- Best Practices for Avoidance and Detection
- Signs of Trouble
- Recovery

Definition of Corruption

- Database queries behave in a manner that is unexpected given the SQL statements previously executed.
- For example, if we store some data into a table using an INSERT statement, we expect that a subsequent SELECT statement will return the same data that we stored.
  - If we get back different data, that's corruption.
  - If we get an ERROR, that's corruption.

Causes of Corruption

- Bad Hardware
  - For example, bad disk or bad memory.
- Bad Software
  - For example, bugs in PostgreSQL, kernel, filesystem, backup tool, etc.
- User Error
  - For example, faulty backup and recovery procedures.

Best Practices

- Backup and Restore
- Configuration
- Storage
- Administration

Best Practices: Backup and Restore

- Take backups regularly.
  - If you can take a backup, all of your data is still readable.
  - Also, if you have corruption later, you have the option of restoring from your backup.
- Make sure that you can restore your backups.
  - Otherwise, they're not very useful.
- Verify that the restored backups look OK.
  - Otherwise, they're not very useful.
- Use a good backup tool, not a home-grown script.
  - Writing a correct backup script is not straightforward and it is easy to get things wrong.

Best Practices: Configuration

- Don’t use fsync=off
  - An OS crash while fsync=off can corrupt the database.
  - Turning fsync=off “temporarily” during initial loading and forgetting to turn it back on is a common mistake.
- Don’t use full_page_writes=off
  - In theory, some filesystems don’t need full page writes.
  - In practice, they all do.
- Use a safe value for wal_sync_method
  - On MacOS X, use fsync_writethrough.
  - On Windows, use fsync or fsync_writethrough, or disable write caching.
  - Check pg_test_fsync output – your chosen method should not be “unrealistically” fast.

Best Practices: Configuration (2)

- Run with checksums enabled (initdb -k).
  - This won’t prevent your database from becoming corrupted, but the corruption is more likely to be automatically detected and produce errors that you can notice.

Best Practices: Storage

- Use local storage, not a network filesystem.
  - There are fewer ways for a directly connected disk to fail.
  - It may be possible to set up NFS or iSCSI reliably, but many setups seem to be unreliable.
- Use RAID, but don’t rely on it not to fail.
  - You still need backups.
- Exercise caution when choosing a filesystem.
  - ext4 and xfs seem to be more reliable than other Linux filesystems.
- Monitor PostgreSQL and OS logs for storage-related errors.

Best Practices: Administration

- Do not manually modify the data files in any way.
  - In particular, do not remove files manually – not from pg_wal or pg_xlog, not from pg_xact or pg_clog, not from anywhere.
- Do not run anti-virus software.
  - If you must run it, exclude the PostgreSQL data directory – but that may not be good enough.
- Don’t remove postmaster.pid.
  - This could permit multiple copies of PostgreSQL to run at the same time, which will corrupt data.
- Consider performing “plug testing.”

Signs of Trouble

- Monitor the PostgreSQL logs for strange errors!
- Database corruption won't necessarily cause errors, but serious corruption often does.
- A wide variety of errors are possible depending on the exact nature of the problem.
- Be on the lookout for errors which seem to be complaining about things internal to the database system rather than user-facing things.
- Database corruption can sometimes cause crashes, wrong answers, or infinite loops; these problems can be tricky to diagnose.

Some Messages Suggesting Corruption

- ERROR: could not access status of transaction 3881522
  DETAIL: could not open file “pg_xact/0003”: No such file or directory
- ERROR: could not read block 123 in file “base/45678/90123”: read only 0 of 8192 bytes
- ERROR: failed to re-find parent key in index "xxx_idx" for split pages 123/456
- ERROR: cache lookup failed for relation 123456
- ERROR: found multixact 123 from before relminmxid 456
- PANIC: could not locate a valid checkpoint record
  LOG: startup process (pid 12345) was terminated by signal 6: Aborted

Corruption Recovery: Alternatives

- If the problem is only with a standby, rebuild the standby.
- If the problem is with the master, consider restoring from backup or failing over to an unaffected standby.
- Consider whether you can rebuild the data from some other source.
- Only if none of that is possible, attempt to recover the data from the corrupted database.

Corruption Recovery: Disclaimer

- Please proceed with extreme caution when attempting to recover a corrupted database.
- Many PostgreSQL companies have experts on staff who can help. Consider hiring one!
- Like all my talks, the advice in this talk is based on my experiences and is provided in the hope that it may be useful, but without any guarantee.
- If you try any of the techniques mentioned in this talk, you do so ***at your own risk*** and ***you may lose or further corrupt your data***.

Corruption Recovery: General Approach

- If you must try to recover data from a corrupted database, take a backup first.
  - What if your attempt at recovering data makes things worse?
- Your goal should be to use pg_dump to back up the contents of the database and then restore them into a new database created by a new initdb.
  - If you just hack on the existing, corrupted database until it seems to run again, there's a good chance of future trouble.

Corruption When You Can't Start DB

- pg_resetwal (or pg_resetxlog) can often allow a corrupted database to start.
- It's still corrupted … maybe moreso … but at least it starts.
- Use of pg_resetwal is indicated when the database won't start because necessary WAL files have been lost or corrupted and cannot be recovered.
- It is also indicated if pg_control has been lost and cannot be recovered.
- If the database is so badly corrupted that there's no hope of starting it, pg_filedump can recover data from raw files without a running server.

Corruption Recovery If Dump Fails

- If pg_dump is failing due to a problem with a specific object, consider dropping it – especially if it is an index or a temporary table.
- If pg_dump is failing due to a problem with a specific index, and you can't or don't want to drop it, consider REINDEX.
- If dumping the whole database is failing, consider trying to dump individual tables or schemas.
- If dump of a particular table is failing, try to extract at least some of the data using SELECT .. WHERE. Filtering on ctid can be helpful.
- EnterpriseDB publishes pg_catcheck utility that can help diagnose catalog corruption.

Corruption Recovery If Restore Fails

- Typically, restore failures are due to foreign key or unique constraint violations.
- A human being will need to decide what to do.

Thanks

- Any Questions?