Postgres-XC - Aperçu

- Solution Open Source multi-maître synchrone pour PostgreSQL
- NTT & EntrepriseDB

A propos de l'auteur

- Auteur: Julien Rouhaud
- Société: DALIBO
- Date: Octobre 2012
- URL: www.dalibo.org

Sommaire

- Architecture générale
- Mise en place
- Haute disponibilité et performances

Architecture générale

- 3 types de daemons
  - GTM
  - Coordinator
  - Datanode

GTM

- Global Transaction Manager
- Gère les identifiants de transaction
- Très gourmand en bande passante
  - Réseau gigabit à faible latence
  - GTM en STANDBY
  - GTM_proxy

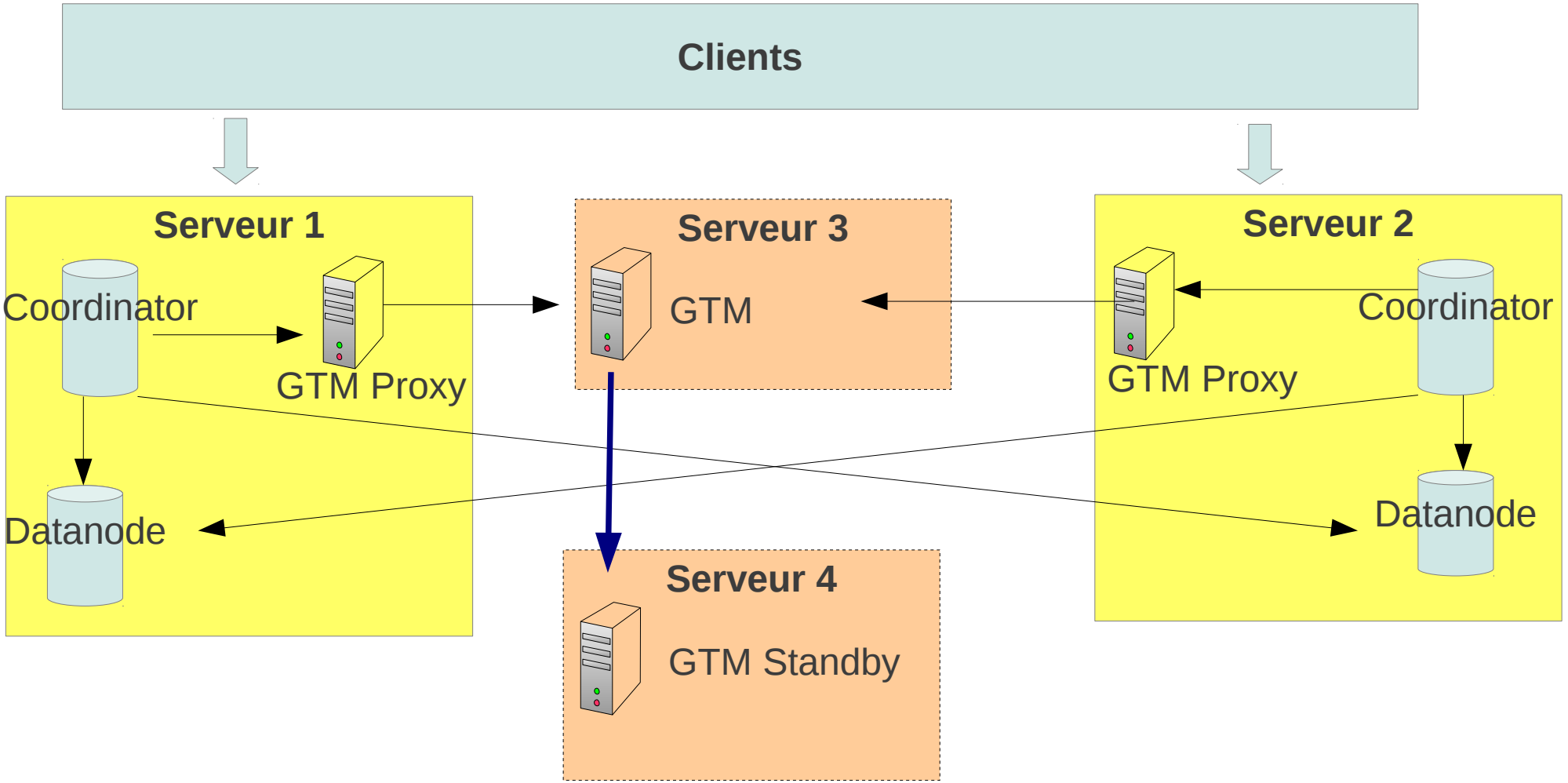
Coordinator

- Point d'entrée pour les applications
- Décompose et transmet les requêtes aux datanodes
- Stocke le catalogue
- Agrège les données si besoin

Datanode

- Très proche d'un serveur PostgreSQL
- Gère les requêtes envoyées par le Coordinator
- Stocke les données

Architecture générale



Compilation et installation

- Similaire à PostgreSQL
  - `./configure --prefix ...`
- Quelques binaires spécifiques :
  - initgtm
  - gtm_ctl

Initialisation et lancement du GTM

- `initgtm -Z gtm -D /.../gtm`
- gtm.conf
  - port par défaut 6666
  - nodename
  - Startup: ACT
- `gtm_ctl -Z gtm -D /.../gtm start`

GTM Standby

- Même initialisation
- gtm.conf
  - startup = **STANDBY**
  - active_host
  - active_port
  - synchronous_backup

Initialisation et lancement GTM_proxy

- initgtm -Z gtm_proxy -D /.../gtm_proxy
- gtm_proxy.conf
  - port
  - listen_addresses
  - nodename
  - gtm_host
  - gtm_port

Initialisation et lancement des Coordinator

- initdb -D /.../coord --nodename nom_coord
- port = 5432 (point d'entrée)
- pooler_port
- gtm_host
- gtm_port
- enforce_two_phase_commit = on
- pg_ctl -D /.../coord -Z coordinator start

Initialisation et lancement des Datanode

- initdb -D /.../node --nodename Node
- port = 15432
- shared_buffers …
- pg_ctl -D /.../node -Z datanode start

Mise à jour du cluster

- CREATE NODE nom WITH (

  TYPE='datanode',

  HOST='ip serveur',

  PORT=5432);
- CREATE NODE nom WITH (

  TYPE='coordinator',

  HOST='ip serveur',

  PORT=15432);
- SELECT pgxc_pool_reload();

Haute disponibilité et performances

- GTM Standby
- Réplication / distribution

Promotion GTM standby

- gtm_ctl -Z gtm -D /.../gtm_standby promote
- Besoin de notifier chaque GTM_proxy
- gtm_ctl -Z gtm_proxy

  -D /.../gtm_proxy

  -o '-s ip_nv_gtm -t port_nv_gtm'

  reconnect

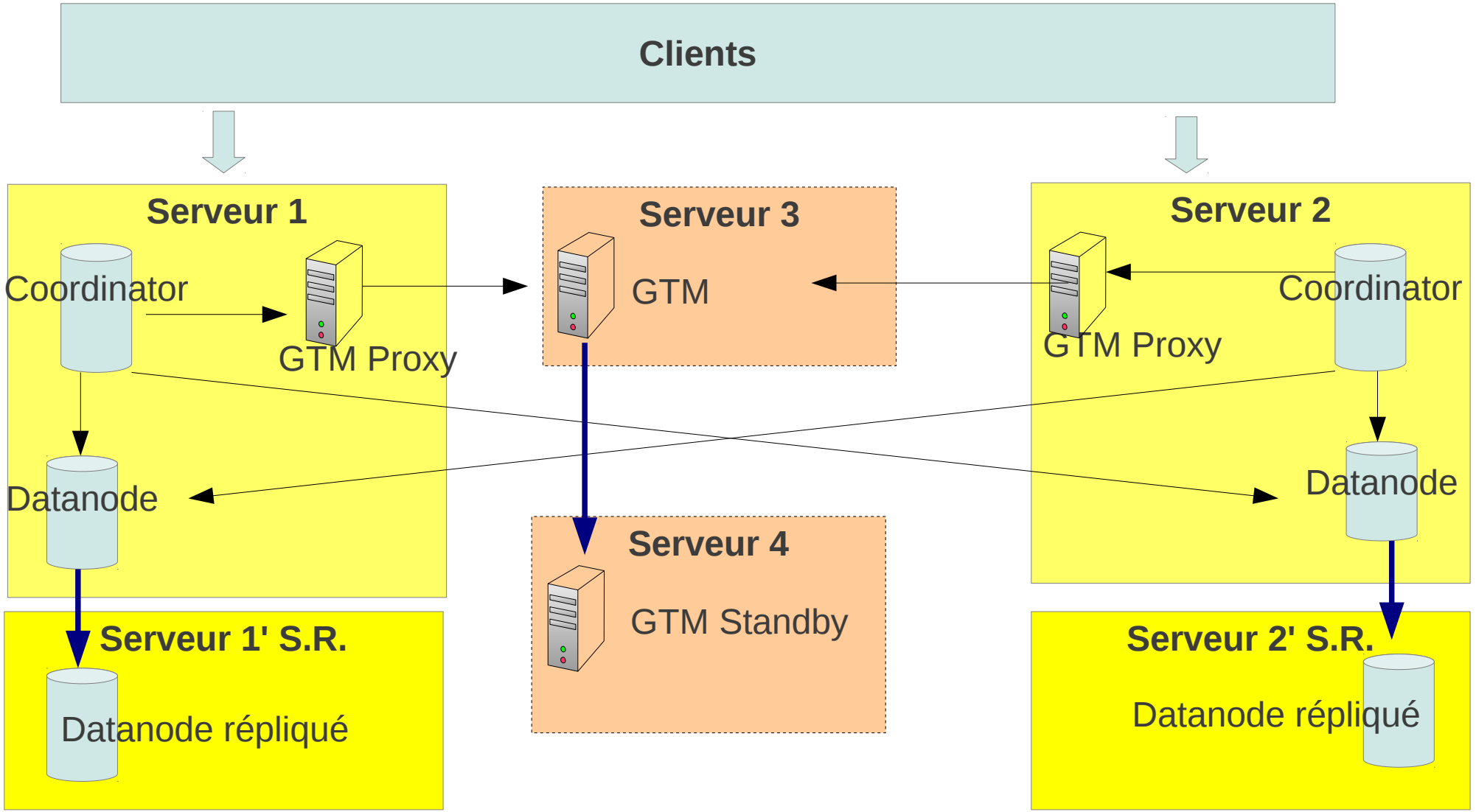
Stockage des données - 1

- Tables distribuées
  - [CREATE | ALTER] TABLE nom DISTRIBUTE BY
    - ROUND ROBIN
    - [HASH | MODULO] nom_colonne
  - Chaque nœud n'a qu'une partie des données
  - Réplication PostgreSQL des Datanodes

Stockage des données - 2

- Tables répliquées
  - [CREATE | ALTER] TABLE nom DISTRIBUTE BY REPLICATION
  - Toutes les données sont sur tous les nœuds
  - Pas de gains en écriture
  - Permet une jointure locale des tables distribuées

Architecture générale répliquée



Pour aller plus loin

- Site officiel http://postgres-xc.sourceforge.net/
  - Documentation
  - Wiki
  - Mailing list
- http://michael.otacoo.com/tag/postgres-xc/

Conclusions

- Une solution encore jeune mais prometteuse
- Complexe à administrer
- Encore quelques limitations
- En attente de PostgreSQL 9.2

Des questions ?