Nouveautés de PostgreSQL 9.2

Table des matières

PostgreSQL 9.2

1 Au menu

- Performances
- Réplication
- Administration
- Fonctionnalités utilisateur

Cette présentation est un coup d'œil rapide sur les nouveautés de la version 9.2. Nous avons découpé cela en plusieurs parties. Tout d'abord, les performances car un énorme travail a été réalisé sur ce domaine, et plus précisément sur la scalabilité. La réplication a eu droit à quelques améliorations qui vont couvrir les quelques manques qui restaient. Les utilisateurs et les administrateurs n'ont pas été oubliés : ils auront le plaisir de découvrir beaucoup de nouveautés qui vont faciliter leur travail de tous les jours.

2 Performances

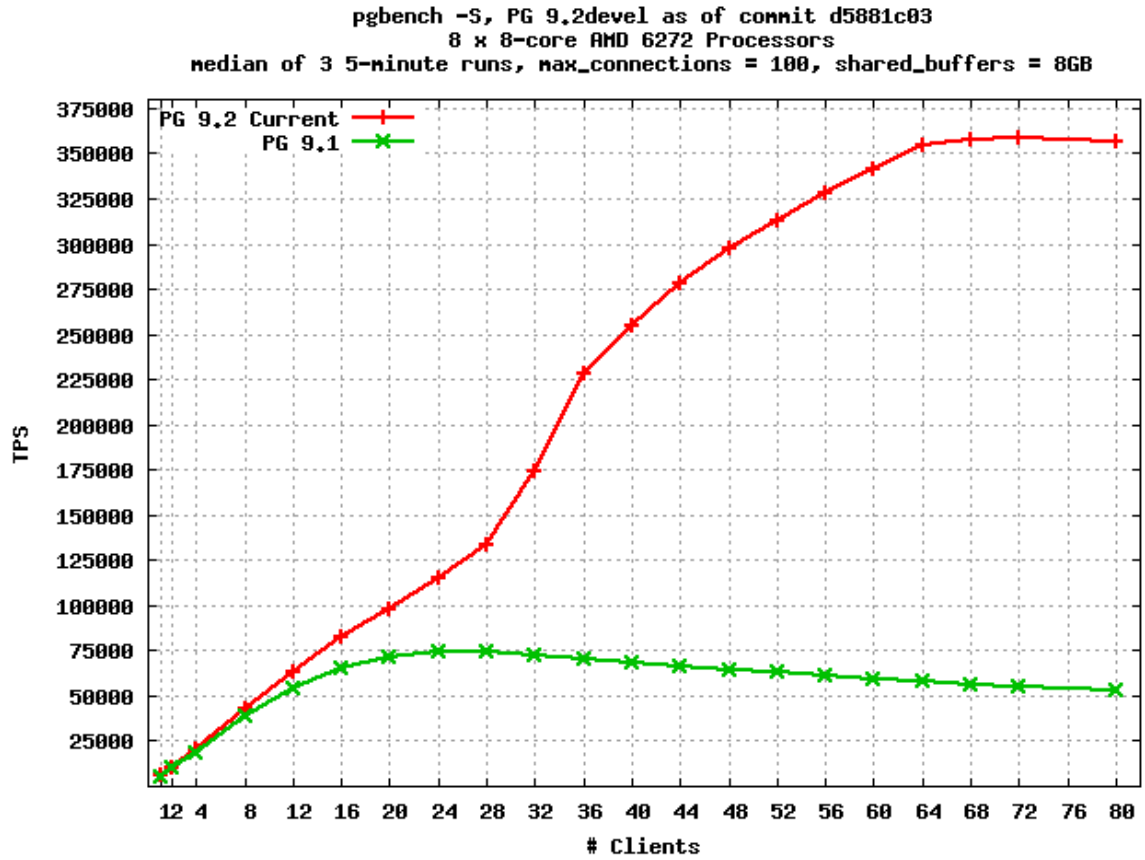
- Scalabilité des lectures
- Scalabilité des écritures
- Parcours d'index
- Améliorations de l'EXPLAIN

La scalabilité, c'est gagner en performances après avoir amélioré le matériel. Autrement dit, plus on met de CPU, de RAM, etc, et plus on a de performances. Le logiciel doit pouvoir utiliser toute cette puissance ajoutée. Il doit pouvoir profiter des nouveaux processeurs, de la mémoire augmentée, etc. Avec PostgreSQL 9.1, la scalabilité était sévèrement limitée par plusieurs facteurs. L'un des plus importants était la contention au niveau des verrous, y compris sur de petits systèmes avec huit cœurs, voire moins. PostgreSQL 9.2 améliore cela en gagnant linéairement en performance jusqu'à 32 cœurs. La courbe faiblit un peu au-dessus, mais reste bien au-dessus de ce que pouvait faire la version 9.1.

2.1 Scalabilité des lectures

- Les lectures ont une bonne scalabilité
- Une exception: plusieurs sessions accèdent à la même table
  - contention au niveau du gestionnaire de verrous
- Plusieurs améliorations
  - acquisition rapide des verrous dans certains cas
  - réduction de la surcharge dû à l'acquisition des verrous
  - différentes autres petites optimisations

2.2 Scalabilité des lectures - graphique

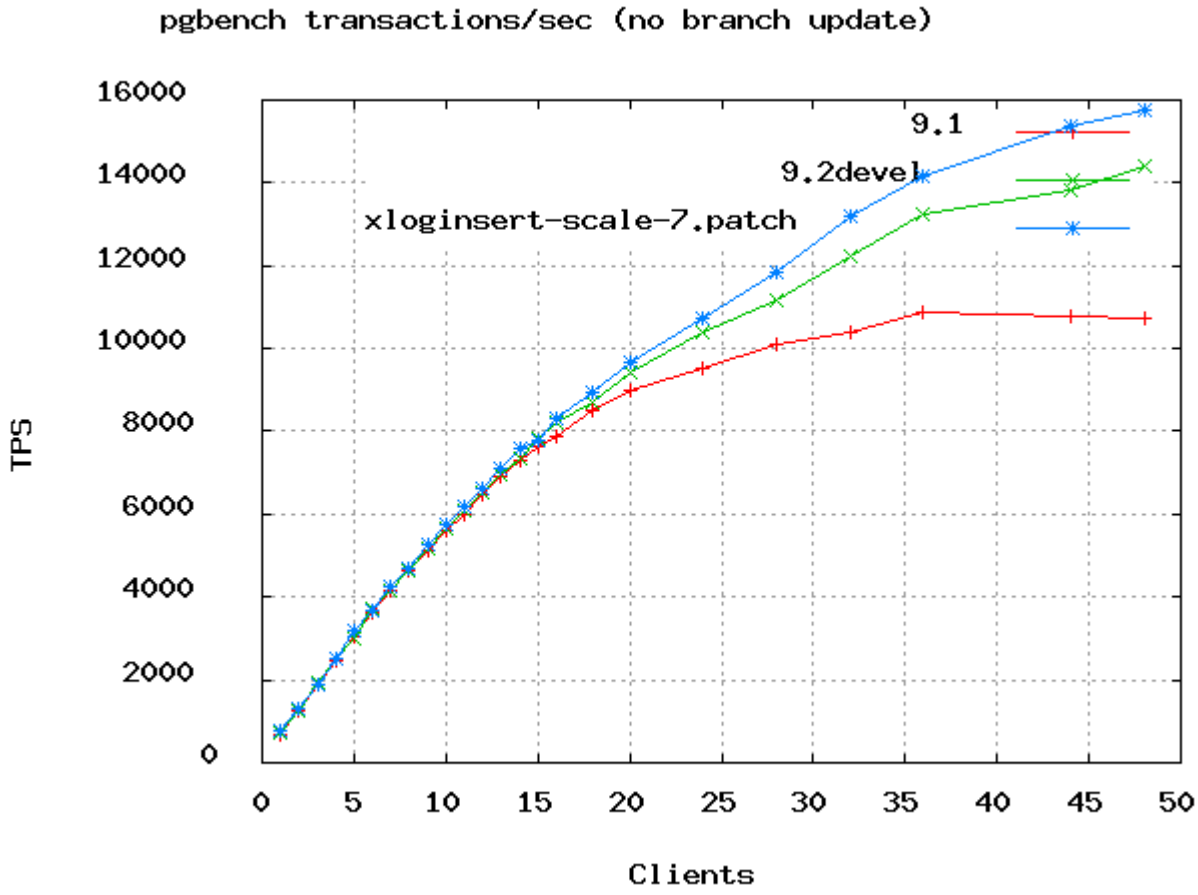


2.3 Scalabilité des écritures

- Meilleur groupement des écritures dans les journaux de transactions
- Réduction de la contention des verrous sur l'écriture dans les journaux de transactions
- Réduction de l'activité des CHECKPOINT

Le plus gros travail a eu lieu sur la scalabilité au niveau du vidage des journaux de transactions. Ce travail a permis une réduction des contentions sur les verrous quand plusieurs processus serveurs tentent simultanément de vider le cache des journaux de transactions

2.4 Scalabilité des écritures - graphique

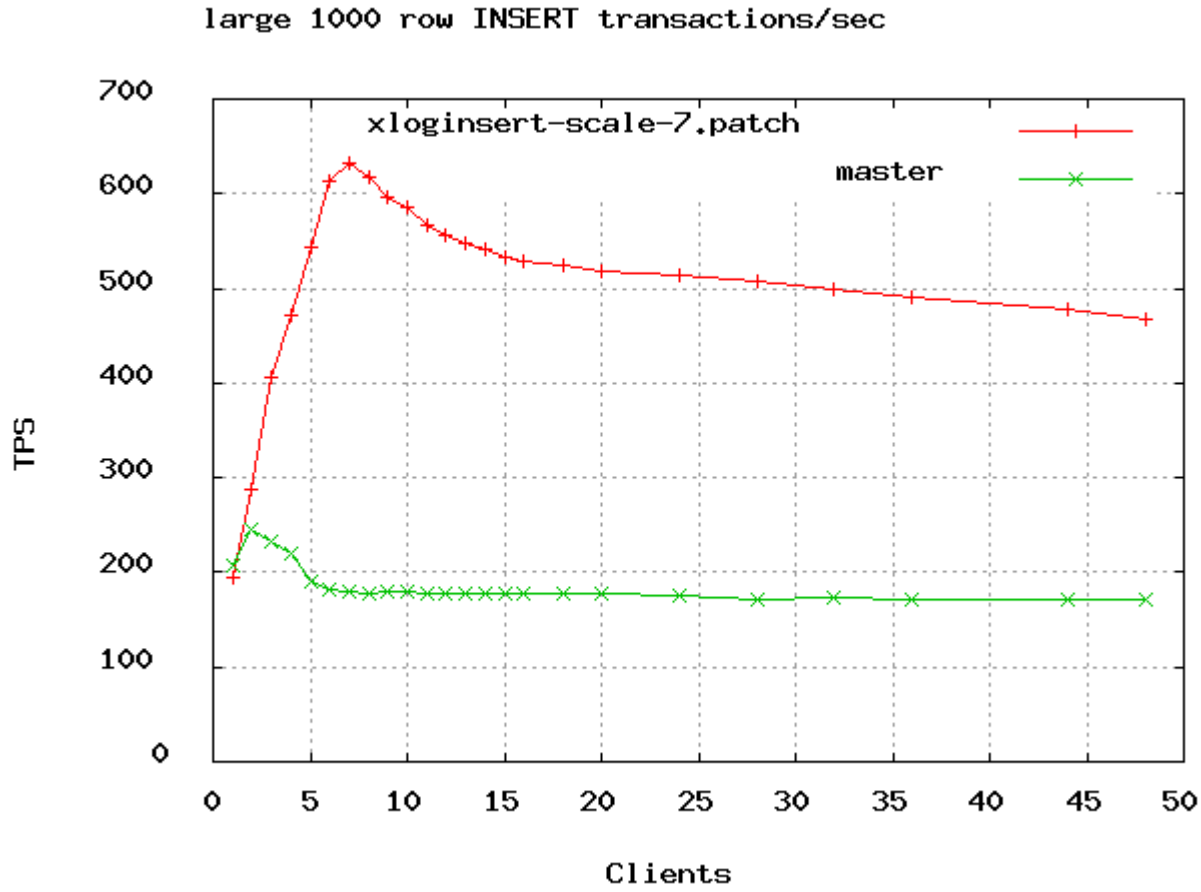


2.5 Chargement en masse

- Les chargements massifs avec COPY offrent une bonne scalabilité
  - … sauf s'il faut les journaliser
- Amélioration
  - copier les données en batch de plusieurs lignes
  - réduit le volume des journaux
  - réduit les verrous sur les pages
- Optimisation désactivée en cas de triggers

Le chargement de données en masse avec COPY offre une bonne montée en puissance sauf lorsqu'il faut journaliser l'opération, ce qui se révèle être fréquemment le cas. Le nouveau patch intègre les lignes d'un COPY en plusieurs batchs, ce qui permet de réduire le volume dans les journaux de transactions. Les verrous sur les pages disques sont aussi réduits. La réduction du trafic des journaux de transactions permet d'exécution plusieurs requêtes COPY en même temps sans perdre en performances.

2.6 Chargement en masse - graphique



2.7 Parcours d'index

- En 9.1
  - Tout accès à l'index nécessite un accès à la table…
  - … pour déterminer si la ligne est visible pour la transaction en cours
  - Nœud « Index Scan »
- En 9.2
  - Si toutes les colonnes nécessaires sont présentes dans l'index…
  - … et si la page est connue pour ne contenir que des lignes visibles à toutes les transactions…
  - … alors l'accès à la table n'est pas utile
  - Nœud « Index Only Scan »

Pour les versions 9.1 et antérieures de PostgreSQL, tout accès à un index nécessite un accès à la table pour vérifier si les lignes trouvées dans l'index sont visibles dans la table. La version 9.2 permet d'éviter l'accès à la table dans certaines conditions :

- toutes les colonnes nécessaires doivent être présentes dans l'index ;
- les enregistrements correspondants de l'index pointent vers des pages dont il est connu qu'elles ne contiennent que des lignes visibles pour toutes les transactions.

Si ces deux conditions sont vérifiées, PostgreSQL n'aura pas besoin de lire partiellement la table à la recherche des informations de visibilité. Le nouveau nœud du plan d'exécution s'appelle « Index Only Scan ».

L'information de visibilité des lignes dans les blocs est enregistrée dans un fichier de suffixe `_vm` (`vm` est l'acronyme de « Visibility Map »). Il existe un fichier de ce type par table depuis la version 8.4 mais ce fichier pouvait être corrompu en cas de crash. Maintenant qu'il est lui-aussi journalisé, il est utilisable pour des tâches aussi importantes que ce nouveau type de parcours.

2.8 Améliorations de l'EXPLAIN

- Tracer les pages disque modifiées/écrites
- Indiquer le nombre de lignes filtrées
- Ne pas afficher le chronométrage

Dans la recherche d'optimisation sur des requêtes, la commande `EXPLAIN` est souvent utilisée. Aussi intéressante que puisse être sa sortie, certains manques font cruellement défaut.

Depuis la version 9.0, `EXPLAIN` est capable d'indiquer le nombre de pages disque lus dans le cache et en dehors du cache, ainsi que quelques autres informations. Par contre, `EXPLAIN`

n'indiquait pas le nombre de lignes modifiées/écrites. La 9.2 comble ce manque.

Il est souvent préconisé de remplacer un parcours de table par un parcours d'index. Cependant, cela n'a un intérêt que si le filtre réalisé ne récupère que très peu de lignes. Plus exactement, cela se révèle utile si le filtre en rejete beaucoup. La commande EXPLAIN de la version 9.2 donne cette information directement dans le plan d'exécution :

```
demo_9_2=# EXPLAIN ANALYZE SELECT * FROM t1 WHERE c1 < 500;
                                                 QUERY PLAN
-------------------------------------------------------------------------------------------------------------
 Seq Scan on t1  (cost=0.00..180.00 rows=500 width=14) (actual time=0.005..0.671 rows=499 loops=1)
   Filter: (c1 < 500)
   Rows Removed by Filter: 9501
 Total runtime: 0.698 ms
(4 lignes)
```

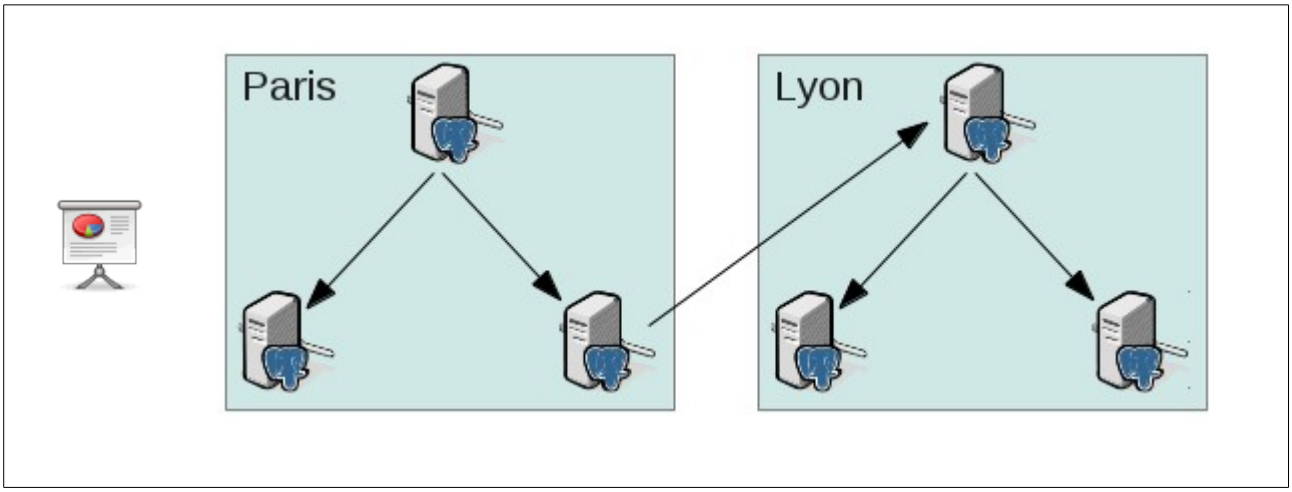
Sur certains matériels et sur certains systèmes d'exploitation, récupérer la date et l'heure système prend beaucoup de temps. Cela peut devenir un très gros problème lors de l'exécution d'un EXPLAIN ANALYZE car cette commande chronomètre chaque nœud. EXPLAIN permet en 9.2 de conserver toutes les informations habituelles sauf ce chronométrage. Pour cela, il suffit de désactiver l'option TIMING.

3 Réplication

- Réplication en cascade
- Sauvegarde d'un esclave
- Outil pg_receivexlog
- Nouveau mode de réplication

La réplication est un thème maintenant habituel de chaque nouvelle version majeure. Il restait quelques fonctionnalités à ajouter et c'est ce que fait la 9.2.

3.1 Réplication en cascade



En 9.0 et en 9.1, seul le maître est capable de pousser les informations de réplication à d'autres serveurs. À partir de la version 9.2, un esclave peut aussi avoir ce rôle-là. Cette fonctionnalité est appelée la réplication en cascade. Elle existe depuis bien longtemps dans Slony, et depuis quelques semaines dans Londiste. C'était un manque important dans la réplication interne, et la 9.2 comble aussi ce manque.

Le but de cette fonctionnalité est de faire en sorte que le maître ne soit pas chargé de l'envoi des informations de réplication à tous les esclaves. Cela permet d'éviter une consommation très importante de la bande passante réseau.

3.2 Sauvegarde d'un esclave

- Comme un esclave peut envoyer les données de réplication, il peut servir à faire une sauvegarde en ligne
- `pg_basebackup` enfin utilisable sur les esclaves

3.3 Outil pg_receivexlog

- Nouveau outil utilisant le protocole de réplication
- Enregistre les informations de réplication dans des fichiers
- Permet l'équivalent d'un « Log Shipping » beaucoup plus proche du maître
  - Pas besoin d'attendre le remplissage d'un journal complet
  - Récupération des informations au fil du temps

`pg_receive_xlog` est un outil se connectant à un serveur PostgreSQL (maître ou esclave) pour

récupérer les informations de réplication. Les informations sont concaténées dans un fichier équivalent à un journal de transactions.

Par rapport à du « Log Shipping » standard, cette méthode est bien préférable. Il n'est pas nécessaire de remplir un journal complet avant de l'archiver. Les informations sont envoyées et enregistrées au fil du temps. Un serveur restauré avec ces fichiers sera généralement plus à jour qu'un serveur restauré avec des journaux archivés.

3.4 Nouveau mode de réplication

- Réplication en mémoire seulement sur l'esclave
  - paramètre `synchronous_commit` à `remote_write`
- Avantage
  - meilleures performances
- Inconvénient
  - durabilité moindre

Certains utilisateurs souhaitent gagner en performance sur la réplication. Un nouveau mode de réplication leur est proposé en configurant le paramètre `remote_write` à la valeur `synchronous_commit`. Avec cette configuration, l'esclave indique au maître qu'il a bien reçu les données lorsque celles-ci sont en mémoire et non pas sur disque. Le risque inhérent à ce comportement est que, en cas de crash simultané du maître et de l'esclave, et de la corruption des données du serveur maître, l'esclave disponible pourrait ne pas avoir toutes les données validées (toutes les données restées en mémoire et non enregistrées sur disque). C'est un risque que certains utilisateurs sont prêt à prendre pour gagner en performance.

4 Administration

- Commandes DDL
- Supervision

Un des points faibles de PostgreSQL reste la supervision. Quelques nouveautés apparaissent sur ce front. Quant à l'administration, un gros travail a eu lieu sur les commandes DDL, qui va permettre de faciliter la vie des codeurs de scripts SQL.

4.1 Commandes DDL

- Clause `IF EXISTS` pour les commandes ALTER
- Moins de réécriture de table pour `ALTER TABLE`
- Contraintes CHECK non valides à l'ajout
- Renommage d'objets enfin géré pour les domaines et les FDW

Les commandes ALTER peuvent enfin utiliser la clause `IF EXISTS`. C'est très intéressant pour l'écriture de scripts SQL. Cela permet d'éviter un échec lors de la mise à jour d'une structure de base.

L'un des gros soucis avec `ALTER TABLE` vient du fait qu'il nécessite fréquemment de réécrire toute la table. Cela génère un grand nombre d'écritures disques qu'il serait mieux d'éviter. La version 9.2 optimise certains comportement de cette commande. Par exemple, certains `ALTER TABLE .. ALTER TYPE` peuvent fonctionner sans réécrire la table.

Il est possible d'ajouter des contraintes CHECK sur une table, sans vérifier préalablement le contenu existant de la table. Toute nouvelle insertion ou modification vérifiera la contrainte, mais les anciennes données ne sont pas vérifiées. Elles le seront à la demande de l'administrateur. Cela permet de placer des contraintes, et de faire la vérification à un moment de moindre activité (la nuit par exemple).

Curieusement, les domaines et les `foreign data wrappers` n'étaient pas renommables. C'est enfin possible en 9.2.

4.2 Supervision

- Module `pg_stat_statements` : normalisation des requêtes
- Traces
  - autovacuum (pages disques, taux d'I/O)
- Statistiques
  - Nouveau paramètre `track_io_timing`
  - dans `pg_stat_database`, compteur de `deadlocks` et compteurs de fichiers temporaires
  - dans `pg_stat_bgwriter`, durée des `CHECKPOINT`

Le module `pg_stat_statements` a subi de nombreux changements. Le plus important est qu'à l'instar de pgfouine et pgbadger, `pg_stat_statements` sait faire de la normalisation de requêtes. Les statistiques rapportées sont donc plus intéressantes car il ne fait plus attention aux valeurs litérales.

L'autovacuum trace un peu plus son activité (utilisation des pages disques, taux moyen de

lecture et d'écriture).

Au niveau des statistiques, on remarque l'apparition d'un compteur de `deadlocks`, ainsi qu'un compteur de nombre et de taille des fichiers temporaires dans la vue `pg_stat_database`. La vue `pg_stat_bgwriter` donne aussi la durée totale des écritures réalisées par des `CHECKPOINT` ainsi que la durée des demandes de synchronisation effectuées par les `CHECKPOINT`.

5 Fonctionnalités utilisateurs

- Intervalles de valeurs
- Indexage SP-GiST
- Type JSON

Les utilisateurs ne sont pas oubliés. De nombreuses nouvelles fonctionnalités ciblent les utilisateurs, comme les intervalles de valeurs et le nouveau type `json`.

5.1 Intervalles de valeurs

- Intervalles de valeurs
  - date : `[ 2012-04-10, 2012-04-12 )`
  - texte: `( Abbe, Babel ]`
  - point: `( 375.453, 374.441 )`
- Recherche et tri possibles

Il est maintenant possible de déclarer des intervalles de valeurs de plusieurs types : date, entier, texte, point, etc. Il est même possible d'en ajouter. Les intervalles ont des bornes incluantes ou excluantes. Les tests, recherches, tris sont possibles sur ce type de données. Un index améliore les performances comme pour tout autre type de données.

5.2 Type JSON

- Même idée que le type `xml`
  - validation du type
- Fonctions
  - `array_to_json`
  - `row_to_json`
- Permet de récupérer le résultat de requêtes au format JSON
  - directement interprétable par de nombreux langages (comme JavaScript)

`json` est un format particulier pouvant stocker des données. Comme le type `xml` introduit en 8.4, la validité de la valeur est vérifiée à chaque insertion ou modification. Le plus intéressant réside certainement dans le fait qu'il est possible de récupérer le résultat d'une requête ou le contenu d'une table dans un champ textuel codé suivant le format JSON. De nombreux langages (généralement ciblant le web) comprennent directement le format `json`. Cela facilite ainsi la discussion entre un serveur PostgreSQL et un langage comme JavaScript.

5.3 Indexage SP-GiST

- Basé sur les « Space-Partitioning Trees »
- Plus performant en lecture et en écriture que GiST
- PostGIS les utilisera avec sa version 2.1

PostgreSQL disposait déjà d'un grand nombre d'algorithmes pour ces index : Btree, Hash, GiST, GIN. Un nouveau fait son apparition en version 9.2.

Les index SP-GiST sont des index similaires aux index GiST de part leur flexibilité, mais permettent de construire des arbres différents (suffixés, non balancés, décomposés dans l'espace, …). Cela permet des gains de performances dans certains cas, la recherche dans l'index étant dépendante uniquement de la taille de la requête. Mais c'est surtout au niveau de leur création que les performances sont visibles.

6 Autres

- Amélioration de l'optimiseur
- Plus d'options pour `pg_dump/pg_restore`
- Plus de fonctionnalités pour `psql`
- Plus de corrections et de nouveautés pour `pg_upgrade`
- Plus de sécurité avec sepgsql
- Moins de réveil des CPU (moins de consommation électrique)
- Snapshots exportables

PostgreSQL 9.2 dispose de plein d'autres nouvelles fonctionnalités ou améliorations. Il est difficile de toutes les indiquer dans cette présentation. Nous vous encourageons à lire les articles associés à cette version.

7 Vous voulez plus de technique ?

- Workshop 9.2
- Prochaines dates
  - 2 novembre
  - 7 décembre
  - ou chez vous ?

Alors venez aux workshops proposés par Dalibo. Un workshop se déroule sur une journée, accueille un maximum de cinq personnes. Le but est de découvrir techniquement la 9.2 et surtout de manipuler. De nombreux TP sont proposés.

Notez que si vous êtes plusieurs de la même entreprise, nous sommes disposés à venir faire cette journée chez vous, gratuitement.