Table des matières

Le projet PostgreSQL

Cet article, écrit par Guillaume Lelarge, a été publié dans le hors-série 44 du magazine GNU/Linux Magazine France, hors-série dédié à PostgreSQL. Il est disponible maintenant sous licence Creative Commons.

Un historique rapide

Ce projet a réellement débuté en 1996 quand Bruce Momjian et Marc Fournier ont repris un certain nombre de patchs qui traînaient pour les intégrer dans ce qui s'appelait alors Postgres95. Leur collaboration a permis de mettre en place une plateforme (web, cvs, listes de discussion) et de s'assurer de la sortie de versions officielles.

La première version, labellisée PostgreSQL 1.0, sort en 1996. Un saut est fait en 1997 pour la version 6.0. À partir de ce moment, les développeurs sortiront une version majeure tous les ans.

Ce n'est qu'à partir de la version 8.0 que PostgreSQL commence à disposer de fonctionnalités très intéressantes en entreprises, notamment pour les grosses bases de données. Les tablespaces et la technologie PITR en sont les deux meilleurs exemples. C'est aussi la première version disposant d'exécutables natifs sous PostgreSQL. Paraît ensuite la 8.1 (gestion des rôles, ajout de la technologie 2PC, intégration de l'autovacuum, meilleure gestion du partitionnement) fin 2005, puis la 8.2 (support de LDAP pour l'authentification, ajout du mode Warm Standby) fin 2006.

Le groupe devient de plus en plus rodé. Bien que le projet PostgreSQL n'ait pas de plan de développement précis, les nouvelles fonctionnalités affluent. Les développeurs envoient leur patchs sur la liste pgsql-hackers, les anciens relisent et appliquent les patchs si possible mais toujours après vérification. La « core team », constituée de sept personnes présents sur pratiquement tous les continents, s'assure de sorties régulières des versions mineures et majeures. Les versions mineures ne contiennent que des correctifs de bugs et de failles de sécurité. Il y a donc très peu de surprises lors de la mise en production de ces versions. Quant aux versions majeures, elles sont là pour fournir les nouvelles fonctionnalités. Et question nouvelles fonctionnalités, le rythme des patchs augmente en permanence.

La 8.3 sort le 4 février 2008. Cette version apporte quantité de nouvelles fonctionnalités, comme la recherche plein texte, des nouveaux types (XML, ENUM et UUID), les traces au format CSV, une nouvelle méthode d'authentification (GSSAPI) et bien d'autres choses.

Une nouvelle organisation du développement se met en place pour la version 8.4 : les commits fest. Pendant deux mois, les développeurs travaillent sur leurs patchs puis les envoient sur la liste pgsql-hackers. Une fois ce délai écoulé, des développeurs prennent les patchs en attente un par un, les relisent, les testent et les vérifient tous. Les patchs réussissant les tests sont immédiatement appliqués sur les sources du CVS. Les autres sont repoussés à la prochaine commit fest, à condition que l'auteur ait revu les différents problèmes constatés par les développeurs ayant relu ce patch (le problème pouvant aller du gros bug à un simple manque de documentation de la nouvelle fonctionnalité). Du coup, la version 8.4 dispose d'encore plus de nouvelles fonctionnalités : fonctions de fenêtrage, requêtes CTE, paramètres par défaut pour les fonctions, restauration parallélisée d'une sauvegarde, droits sur les colonnes, etc. Néanmoins, des problèmes dans l'organisation du développement sont toujours présents. Le système de commit fest n'a pas été aussi efficace que ce qui avait été espéré et la sortie a été repoussée de plusieurs mois en attente de deux patchs très intéressants mais qui ont dû être reportés. Pour le développement de la 8.5, un nouveau système de commit fest voit le jour. Les développements ont lieu sur un mois, la période de commit fest est étendue à l'autre mois. Cela se passe plutôt bien pour l'instant, mais il faudra attendre la fin du développement de la 8.5 pour connaître réellement l'efficacité de cette nouvelle méthode.

Bien que la dernière version stable est la 8.4, des versions mineures sont toujours proposées de la version 7.4 à la 8.4. À chaque découverte d'un bug, ce bug est testé dans toutes ces versions. Chaque version impactée est corrigée si la modification n'implique pas de trop gros changements dans le code. En effet, une modification très importante des sources a potentiellement plus de possibilités d'ajouter un nouveau bug que de supprimer l'ancien. Néanmoins, dans la très grande majorité des cas, les bugs sont corrigés pour toutes les versions impactées.

Avoir autant de versions mises à jour (six à ce jour) peut devenir un problème pour l'utilisateur qui ne sait plus forcément quelle version installer. Il est très souvent préférable d'utiliser la dernière version stable de PostgreSQL : plus de fonctionnalités, plus de performances, moins de bugs. Dans un certain nombre de cas limité, une nouvelle installation devra se faire sur une ancienne version de PostgreSQL. Fréquemment, le problème qui se pose est qu'on veut utiliser un outil qui se sert de ce moteur de bases de données comme dépôt de données et que cet outil impose une certaine (vieille) version de PostgreSQL. Si c'est le cas, réfléchissez bien avant de valider ce produit, regardez la concurrence, il y a certainement mieux ailleurs.

Les versions dérivées (comme PostgresPlus d'EnterpriseDB ou Greenplum) n'améliorent pas la situation. Ces versions, souvent commerciales, apportent peu au contenu de PostgreSQL, et en enlèvent une grosse partie : la relation avec la communauté. De préférence, utilisez la version PostgreSQL de la communauté. Seule l'évolution de cette dernière est garantie par la communauté.

Concepts de base

Tout ceci est très bien mais encore faut-il qu'une telle rigueur dans le développement soit suivie des faits.

PostgreSQL est conçu de façon à être utilisable et modifiable par tout le monde. C'est pour cela que, dès le départ, les développeurs ont choisi la licence BSD. Cette licence assure à ces utilisateurs la possibilité de pouvoir lire et modifier le code si besoin est. Cela a permis l'essor de sociétés comme EnterpriseDB et Greenplum. Cela a aussi permis à des sociétés comme Yahoo! et NTT de développer leur version de PostgreSQL. La première a ainsi amélioré les capacités de PostgreSQL pour des bases très conséquentes (plusieurs péta-octets) alors que la seconde a ajouté un système de réplication synchrone.

PostgreSQL a pour principal sujet de préoccupation la sécurité des données des utilisateurs. Les fonctionnalités et les performances sont intéressantes en soi, mais elles ne le sont qu'à partir du moment où vous êtes certains de retrouver vos données telles que vous les avez enregistré.

Pour cela, PostgreSQL utilise des journaux de transactions. L'idée est de placer toutes les modifications effectuées par les requêtes SQL des utilisateurs dans des journaux, avant de les enregistrer sur les vrais fichiers de données. On peut comparer cela à la journalisation des systèmes de fichiers. L'inconvénient est que chaque écriture sera doublée, donc cela implique des contre-performances. L'autre inconvénient, c'est que PostgreSQL, pour s'assurer de l'écriture sur disque, impose au système d'exploitation d'écrire son cache disque sur le disque (une opération appelée fsync). Là-aussi, c'est source de nombreuses lenteurs. Ne rejettez pas PostgreSQL pour autant. Tous ces inconvénients sont aussi la force majeure de PostgreSQL. Oui, cela implique des écritures disque qui sont source de lenteur, mais oui, vos données sont sécurisées. Le moindre crash du serveur n'impliquera pas de perte de données. Quant aux lenteurs, il existe plusieurs moyens de les diminuer. Le premier est conceptuel : les journaux de transactions sont écrits de façon séquentielle, seul un fichier est concerné, y compris si la mise à jour d'une ligne implique de modifier la table et les index. Le deuxième concerne l'architecture du serveur. Il est généralement préférable, pour un serveur effectuant beaucoup de modification de données, de placer les journaux de transactions sur un autre système disque que celui des fichiers de données.

PostgreSQL est aussi conçu pour être utilisé par de nombreux utilisateurs en simultané. Pour cela, PostgreSQL utilise une technologie appelée MVCC. Cet acronyme signifie MultiVersion Concurrency Control. Ce système permet de poser un nombre minimal de verrous. De cette façon, une lecture ne bloque pas une autre lecture. De même, une écriture ne bloque pas une lecture et, inversement, une lecture ne bloque pas une écriture. En effet, une écriture ne se fait jamais sur la même ligne. Par exemple, un UPDATE ne va pas mettre à jour la ligne, elle va enregistrer le fait qu'une ligne a été mise à jour (on va donc se trouver avec une ancienne version de ligne) et elle va insérer une nouvelle ligne contenant les nouvelles informations (cette nouvelle ligne sera en fait la nouvelle version de l'ancienne ligne). Enfin, une écriture ne bloquera pas une écriture si elles ne concernent pas la même ligne. L'inconvénient de ce système est la fragmentation des tables que cela implique. Du coup, tout un système est créé autour pour éviter une grosse fragmentation, mais aussi pour défragmenter une table.

L'autre idée de base de PostgreSQL est de se conformer le plus strictement possible aux standards SQL. Évidemment, l'implantation de certaines fonctionnalités nécessite d'aller plus loin que ce que proposent les standards. Dans ce cas, PostgreSQL essaie de trouver une syntaxe proche des autres SGBD possédant cette fonctionnalité et indique très précisément dans sa documentation qu'il s'agit d'une extension au standard.

Les fonctionnalités essentielles pour les utilisateurs

Au niveau de PostgreSQL, un utilisateur peut définir tout un ensemble d'objets :

- bases de données;
- schémas (groupes logiques d'objets dans une base);
- tables;
- index (complet, partiel, fonctionnel);
- vues;
- procédures stockées;
- opérateurs;
- types de données;
- séquences;
- triggers.

Il peut aussi définir des contraintes : clé primaire, clé étrangère, contrainte CHECK, contrainte UNIQUE, contrainte NOT NULL. Il peut ajouter des langages pour les procédures stockées.

Au niveau administration, il peut aussi ajouter des tablespaces, des utilisateurs, des groupes. Des droits sont disponibles pour tous les objets de la base. Les requêtes peuvent aller du plus simple au plus complexe. Les jointures sont disponibles depuis longtemps, les sous-requêtes aussi. Les fonctions d'agrégat, de fenêtrage et même les requêtes récursives sont supportées (depuis la 8.4 pour les deux derniers).

L'un des gros points forts de PostgreSQL est sa gestion des transactions. Elle respecte complètement ACID. Ce dernier est un acronyme signifiant:

- A pour atomicité (les transactions sont en tout ou rien, toutes les modifications des requêtes d'une transaction sont visibles ou aucune des modifications ne l'est).
- C pour cohérence (une transaction permet d'amener le système d'un état stable à un autre, l'exemple type étant le transfert d'une somme d'un compte en banque à un autre pour lequel il ne faut pas que les utilisateurs puissent voir le débit d'un compte si le crédit de l'autre n'est toujours pas visible).
- I pour isolation (les modifications réalisées par une transaction ne sont pas visibles par les autres transactions).
- D pour durabilité (une fois que PostgreSQL a indiqué le succès de la validation de la transaction suite à l'instruction COMMIT, on sait que les données sont enregistrées sur disque).

Écosystème

PostgreSQL est le moteur de base de données. Il suit le concept Unix : « il ne fait que ça, mais il le fait bien ». Si on veut aller plus loin, il faudra s'intéresser à d'autres projets.

Pour les outils d'administration, les deux applications officielles de la communauté sont pgAdmin, un client lourd disponible sur Linux, Windows et Mac OS X, ainsi que phpPgAdmin, une application web écrite en PHP. pgAdmin est codé en C++ et utilise wxWidgets pour la gestion de l'interface. Il dispose d'au moins une version majeure à chaque version majeure de PostgreSQL, pour proposer une gestion graphique des nouveautés de la dernière version. Il sait gérer les versions 7.4 à 8.4 de PostgreSQL, il n'y a donc aucune raison de rester avec une ancienne version. La dernière sortie est la 1.10. Quand à phpPgAdmin, son nom indique clairement qu'il est écrit en PHP. Ceux qui ont déjà utilisé phpMyAdmin (outil identique, mais pour MySQL) ne seront pas perdus. Même si l'interface n'est pas identique, elle y ressemble suffisamment. Les versions sont moins nombreuses que pgAdmin et surtout moins

synchronisée avec les sorties de PostgreSQL. C'est dommage car c'est un outil extrêmement pratique. La dernière version est la version 4.2.2, sortie le 18 décembre 2008.

Au niveau réplication, il existe un grand nombre d'outils. Seuls quelques-uns sont vraiment utilisables en production. On notera par exemple Slony, Londiste, Bucardo qui font l'objet d'articles dans ce hors-série.

Deux poolers de connexions sont disponibles : pgPool, un espèce de couteau suisse capable en plus de faire de la répartition de charge et de la réplication, ainsi que le très spécialisé pgbouncer. Ils sont aussi l'objet d'articles dans ce hors-série.

Pour le monitoring, il faudra compter avec Munin et ses plugins PostgreSQL (http://muninpgplugins.projects.postgresql.org/) ainsi que Nagios et son script check_postgres.pl (http://bucardo.org/check_postgres/).

Le dernier outil particulièrement intéressant à montrer est PostGIS, la couche spatiale de PostgreSQL. Cet outil dispose de types géographiques et géométriques qui lui permettent de stocker des données géographiques et d'interagir avec. La version 1.4.0 vient tout juste de sortir pour ajouter le support de PostgreSQL 8.4, résoudre quelques problèmes de performances et ajouter plusieurs fonctions très intéressantes.

Une communauté internationale

Nous n'avons toujours pas parlé de l'intérêt principal de PostgreSQL. Sa communauté !

Lorsqu'on parle de logiciels libres, c'est pourtant l'un des points majeurs. La communauté PostgreSQL est internationale. Elle est tout particulièrement présente aux États-Unis, au Japon et en Europe (France, Italie, Angleterre, Allemagne). Les autres pays ne sont pas forcément en reste. Par exemple, la Russie dispose de deux développeurs PostgreSQL très aguerris.

Il y a peu de forums web. La plupart des discussions se font par mail. Les listes de discussion hébergées sur les serveurs de la communauté sont très peu nombreux, les abonnés aussi. Il est fréquent d'avoir une réponse à sa question très rapidement. Tom Lane, l'un des principaux développeurs de PostgreSQL, répond très souvent aux différentes questions, y compris celles des plus novices.

Le canal IRC est moins utilisé par les développeurs. Néanmoins, beaucoup d'utilisateurs sont là pour aider les autres.

Enfin, dernièrement, de nombreuses manifestations ont eu lieu un peu partout dans le monde (en Italie, au Brésil, aux États-Unis, au Japon, en Pologne, en Autriche, etc.) pour des conférences sur PostgreSQL. L'association française remet le couvert cette année lors du PGDay européen.

La communauté française

Ce n'est qu'en 2003 (soit 7 ans après le début de PostgreSQL) qu'apparaît la liste de discussion pgsql-fr-generale. Elle doit son existence à François Suter, qui a été longtemps le contact francophone de PostgreSQL. C'est sur cette liste qu'a débuté le projet de traduction du manuel de PostgreSQL, suite à une conférence de Peter Eisentraut que j'avais été suivre lors des RMLL 2003 à Metz. Il s'agissait à l'époque de la version 7.4.

En 2004, Jean-Paul Argudo rencontre Bruce Momjian. Ce dernier était venu faire une conférence sur PostgreSQL dans la session SGBD présidée justement par Jean-Paul pendant le salon Solutions Linux. Des discussions entre Jean-Paul Argudo et Bruce Momjian va naître le site postgresqlfr.org.

En 2005, toujours lors du salon Solutions Linux, l'association PostgreSQLfr est créée. Jan Wieck, hacker de PostgreSQL et créateur de Slony, invité par Jean-Paul Argudo à passer les quelques jours du salon sur le stand de PostgreSQLfr, assiste à la première assemblée de l'association.

L'association a pour but de promouvoir PostgreSQL dans les pays francophones. La traduction de la documentation est le premier projet. Rapidement, l'activité de l'association se restreint à donner des moyens pour la promotion de PostgreSQL: mise à disposition d'un serveur, d'un site web, d'un forum, achat de goodies pour les salons, réalisation de plaquettes toujours pour les salons. Elle contacte aussi les hackers PostgreSQL pour qu'ils viennent participer au stand.

Solutions Linux, quelque soit l'année, est toujours un succès pour cette association. En effet, PostgreSQL est surtout utilisé en entreprises et Solutions Linux est un salon mixant entreprises et associations. Il est donc logique que le stand soit souvent rempli à craquer pendant Solutions Linux, alors qu'il est généralement vide pendant les RMLL ou les JDLL (événements principalement associatifs).

À cet époque, le seul réel projet de PostgreSQLfr reste malgré tout la traduction de la documentation. Mais elle le fait particulièrement bien. Le jour même de la sortie d'une nouvelle version, la documentation traduite est disponible. PostgreSQL gère des correctifs pour les versions allant de la 7.4 à la 8.4, la traduction suit toutes ces versions. Cela étant dit, la communauté francophone n'en reste pas là. En 2006 apparaît le premier (et seul à ce jour, en dehors d'une seconde édition) livre écrit par un français sur PostgreSQL. Sébastien Lardière fournit ainsi une excellente introduction à l'installation et à l'utilisation d'un serveur PostgreSQL.

Solutions Linux 2007, certainement le meilleur salon pour l'association PostgreSQLfr. Un grand nombre de hackers ont fait le voyage jusqu'à Paris. Magnus Hagander, responsable notamment du portage vers Windows, donne même une conférence et discute avec de nombreux visiteurs. Devrim Gunduz (paquets Red Hat, traduction grecque, etc.) est là-aussi. Il en profite pour discuter avec Atsuchi Mitani, développeur de PGCluster. Heikki Linnakangas et Greg Stark, deux hackers PostgreSQL et employés d'EnterpriseDB, sont aussi présents le dernier jour. Greg Stark parlant un peu français en profite pour discuter avec les visiteurs et pour faire l'interface avec Heikki. Bref, un moment inoubliable pour la communauté française.

Un autre projet de traduction commence fin 2008 : le manuel de Slony. Coordonné par Damien Clochard, cet effort supplémentaire devrait bientôt porter ses fruits… dès que le rédacteur de cet article prendra le temps de relire le dernier fichier. Rappelons aussi que la lettre d'information hebdomadaire de PostgreSQL est traduite toutes les semaines par Nicolas Bougain.

Toujours en 2008, suivant l'exemple du groupe italien, l'association a voulu tester la création complète d'une journée de conférences sur PostgreSQL. Elle a eu lieu le 4 octobre, à Toulouse et a remporté un franc succès. Plus de 60 personnes se sont déplacées pour assister aux conférences qui allaient du retour d'expériences à des sujets bien plus techniques. Il y a aussi eu les RMLL 2008. L'association y a envoyé autant de membres que possible. Un stand était disponible, une conférence a été tenue, ainsi que plusieurs ateliers pratiques.

2008 a aussi été l'année d'un gros changement dans les sites web de l'association. Drupal a été abandonné au profit de différents logiciels: Dotclear pour le blog, Fluxbb pour le forum, dokuwiki pour le wiki de l'association, planetplanet pour l'agrégation des blogs, etc. Ils proposent tous la même interface, ce qui rend l'ensemble très cohérent. Enfin, c'est aussi cette année qu'a commencé une série d'articles sur PostgreSQL pour GNU/Linux Magazine France.

Notons que la communauté française ne se borne pas à traduire des manuels, à écrire des articles ou à participer à des événements. Elle devient de plus en plus active dans le développement (dans le sens codage) de PostgreSQL. On peut noter dans les nouveaux projets Guillaume Smet avec le script PHP pgfouine (qui fera partie d'un chapitre dans l'article sur la configuration de PostgreSQL), Stéphane Schildknecht avec les scripts bash slony1-ctl (dont nous

reparlerons dans l'article sur Slony1), Guillaume Lelarge avec le script PHP pgsnap, Cédric Villemain avec les addons Perl pour Munin et la bibliothèque pgfincore et Dimitri Fontaine avec le script Python pgloader et la bibliothèque prefix. Guillaume Lelarge travaille aussi beaucoup sur pgAdmin, il fait même partie de l'équipe officielle de développement. Cédric Villemain contribue beaucoup au script check_postgres.pl (qui est discuté dans plusieurs articles de ce hors-série). Quant à Guillaume de Rorthais, son travail avec Robert Treat sur phpPgAdmin a permis de bien améliorer cet outil PHP. L'année 2008 a vu les premiers patchs de la communauté française pour PostgreSQL. La 8.4 contient deux patchs de Guillaume Lelarge alors que la 8.5 en contient un de Damien Clochard et un autre de Guillaume Smet.

Quant à 2009, ce sera l'année du pgday européen. Il aura lieu à Paris les 6 et 7 novembre. Vous pourrez y rencontrer des développeurs, des contributeurs mais aussi des utilisateurs de PostgreSQL. La mise en place est assez complexe mais le résultat devrait être à la hauteur. En tout, l'association francophone comprend actuellement une quarantaine de membres dont quelques sociétés (par ordre alphabétique Benchmark Group, CS, Dalibo, OpenWide et Sylëam). Son objectif reste toujours de faciliter la promotion de PostgreSQL dans les pays francophones.

Conclusion

J'espère que ce petit état de lieux de PostgreSQL ne vous a pas repoussé. Qu'au contraire, il vous a alléché. Si c'est bien le cas, alors commençons par le commencement : l'installation de PostgreSQL.