Livre Blanc DALIBO #01

Migrer d'Oracle à PostgreSQL

19.04

DALIBO
L'expertise PostgreSQL

Dalibo SCOP

https ://wwww.dalibo.com/

Migrer d'Oracle à PostgreSQL

Livre Blanc DALIBO #01

TITRE : Migrer d'Oracle à PostgreSQL
SOUS-TITRE : Livre Blanc DALIBO #01

REVISION : 19.04

Remerciements : Ce livre blanc est le fruit d'un travail collectif. Nous remercions chaleureusement ici toutes les personnes qui ont contribué directement ou indirectement à cet ouvrage, notamment : Jean-Paul Argudo, Philippe Beaudoin, Damien Clochard et Léo Cossic.

À propos de DALIBO :

DALIBO est le spécialiste français de PostgreSQL. Nous proposons du support, de la formation et du conseil depuis 2005.

Retrouvez toutes nos livres blancs sur https ://dalibo.com/

Chers lectrices & lecteurs,

Nos livres blancs sont issus de plus de 12 ans d'études, d'expérience de terrain et de passion pour les logiciels libres. Pour Dalibo, l'utilisation de PostgreSQL n'est pas une marque d'opportunisme commercial, mais l'expression d'un engagement de longue date. Le choix de l'Open Source est aussi le choix de l'implication dans la communauté du logiciel.

Au-delà du contenu technique en lui-même, notre intention est de transmettre les valeurs qui animent et unissent les développeurs de PostgreSQL depuis toujours : partage, ouverture, transparence, créativité, dynamisme… Le but premier de nos livres blancs est de vous aider à mieux exploiter toute la puissance de PostgreSQL mais nous espérons également qu'ils vous inciteront à devenir un membre actif de la communauté en partageant à votre tour le savoir-faire que vous aurez acquis avec nous.

Nous mettons un point d'honneur à maintenir nos productions à jour, avec des informations précises et des exemples détaillés. Toutefois malgré nos efforts et nos multiples relectures, il est probable que ce document contienne des oublis, des coquilles, des imprécisions ou des erreurs. Si vous constatez un souci, n'hésitez pas à le signaler via l'adresse contact@dalibo.com !

Table des Matières

LA TRANSITION VERS POSTGRESQL EST UNE CRÉATION DE VALEUR

L'histoire récente de l'informatique regorge d'exemples de "duels" entre logiciels libres et logiciels propriétaires : Linux face à Windows, Firefox versus Internet Explorer, etc. Parmi toutes ces oppositions emblématiques, c'est dans le domaine des bases de données que le contraste est le plus frappant ! D'un coté, Oracle avec sa position dominante et ses pratiques commerciales agressives. De l'autre, PostgreSQL avec sa communauté décentralisée et son modèle économique basé sur le partage.

Toutefois, malgré ces divergences évidentes, les deux SGBD sont extrêmement proches sur le plan technique et partagent énormément de concepts communs : transactions, vues matérialisées, procédures stockées, fonctions analytiques, etc. Au point que PostgreSQL est généralement présenté comme l'alternative n°1 à Oracle. Dès lors, migrer d'Oracle à PostgreSQL n'a rien d'un séisme technologique : au contraire, les DBA Oracle peuvent maîtriser sans difficultés un parc d'instances PostgreSQL.

Au final, le réel changement de paradigme se situe au niveau de la "relation fournisseur". Dans la communauté PostgreSQL, il n'y a pas d'éditeur unique, pas de vente liée et pas de "vendor lock-in".

Au contraire, PostgreSQL offre aux entreprises des libertés nouvelles notamment : choisir un mode de virtualisation sans être pénalisé, pouvoir changer de prestataire technique sans remettre en cause le choix d'architecture ou encore allouer plus de CPU à une instance sans mettre son budget en péril, déplacer ses données dans le cloud ou les ré-internaliser.

Ces libertés nouvelles ont évidemment d'autres implications : au premier regard, la richesse de l'écosystème PostgreSQL peut apparaître comme une complexité. Pour chaque domaine du métier de DBA (sauvegarde, tuning, haute-disponibilité, etc.) la communauté PostgreSQL propose des dizaines d'outils open-source. Difficile de faire son choix lorsque l'on vient du monde Oracle où la stack applicative est monolithique !

Autre enjeu majeur de la migration : la maîtriser des coûts cachés. Car bien sûr "logiciel libre" ne veut pas dire "logiciel gratuit" et là encore le mode de calcul du coût de possession d'une instance PostgreSQL est radicalement différent du modèle Oracle.

Depuis plus de 10 ans, Dalibo assiste des sociétés en France et en Europe dans leur transition vers l'Open-source. De cette expérience unique, nous avons tiré le bilan et défini une méthode en 5 étapes pour réussir le passage à PostgreSQL. Nous proposons cette méthode dans ce livre blanc, non pas comme une recette qu'il faudrait suivre à la lettre, mais comme une feuille de route qui aidera le lecteur dans sa démarche pour moderniser

son infrastructure et reprendre le contrôle de ses données. Car au-delà des réductions des coûts de licence, la transition vers PostgreSQL est une véritable dynamique d'innovation et de création de valeurs au cœur des systèmes d'informations.

PARTIE 1 - L'ÉTAT DE L'ART

Conçu dans les années 80 à l'université de Berkeley et publié sous licence open-source en 1995, PostgreSQL est le Système de Gestion de Bases de Données Relationnel (SGBDR) open-source de référence. Depuis plus de 20 ans, ce projet a traversé les époques et a assis sa réputation sur des principes simples et forts : intégrité des données, stabilité, ouverture, performances et respect des standards.

À tel point qu'aujourd'hui, PostgreSQL (également appelé "Postgres") est désormais considéré comme l'alternative numéro 1 à Oracle Database. L'État français lui-même préconise PostgreSQL dans son Socle Logiciels Libres (SILL) et, en 2015, Jacques Marzin, alors Directeur de la Direction Interministérielle des Systèmes d'Information et de Communication (DISIC) déclarait :

« L'équipe de pilotage du SILL publie désormais des guides d'accompagnement basés sur des retours d'expérience. Le premier concerne la migration vers le logiciel de gestion de bases de données Postgres. »

> **Pour aller plus loin :**
>
> https ://dali.bo/sill https ://dali.bo/disic

FONCTIONNALITÉS

Aux fondements de PostgreSQL se trouvent 4 valeurs essentielles qui définissent ce logiciel et en font le SGBD open-source de référence.

L'INTÉGRITÉ DES DONNÉES ET LA STABILITÉ

Depuis la fin des années 2000 et l'apparition des bases NoSQL, les fondements des bases de données sont remis en cause… Pour obtenir plus de performances, les bases NoSQL s'affranchissent partiellement des 4 principes d'ACID (Atomicité, Cohérence, Isolation et Durabilité) qui sont pourtant essentiels pour garantir la sécurité des données et des transactions. À l'inverse, PostgreSQL avance et améliore ses performances chaque année sans faire de compromis sur l'intégrité des données. Grâce à cette exigence et ses 20 ans d'existence, PostgreSQL a acquis une solide réputation de stabilité, de pérennité et de robustesse.

LA SÉCURITÉ

La protection des données est un enjeu majeur des années à venir. Dans ce contexte, PostgreSQL propose un arsenal de défenses à la fois puissant et varié. Plusieurs méthodes d'authentification sont disponibles (LDAP, GSSAPI, SCRAM,...) et l'accès aux données peut être filtré de manière fine avec les permissions par colonnes et les politiques Row-Level Security (RLS). Face aux attaques extérieures, il est possible de chiffrer les communications avec des certificats SSL, de chiffrer les données avec l'extension pg_crypto et il est possible de renforcer la sécurité en activant SE-Postgres, l'équivalent de SE-linux.

L'EXTENSIBILITÉ

Créé initialement dans un laboratoire à l'université de Berkeley (voir encadré) et donc conçu comme un objet, PostgreSQL est une base de données extensible dans le sens où il est très simple d'ajouter des objets spécifiques : procédures stockées, opérateurs, extensions... Autant d'éléments qui permettent d'implémenter rapidement de nouvelles fonctionnalités. Le logiciel PostGIS (voir plus loin) en est une illustration emblématique.

LE RESPECT DES STANDARDS

PostgreSQL implémente la majeure partie des préconisations du standard SQL :2016 et à ce titre il est l'un des SGBD le plus proche de la norme.

> **PostgreSQL, pourquoi ce nom ?**
>
> Aussi imprononçable soit-il le nom "PostgreSQL" contient depuis son origine un état d'esprit de dépassement et d'innovation. En 1986, Michael Stonebraker, le fondateur du logiciel Ingres (et de nombreux autres SGBD), décida de lancer une nouvelle génération qu'il nomme Postgres pour "Post-Ingres". Le suffixe SQL fut ajouté au milieu des années 90, lorsque le support du standard SQL fut implémenté.

5 CHANGEMENTS DE PARADIGMES

Au niveau technique, PostgreSQL est le SGBD le plus proche d'Oracle tout en proposant un modèle économique radicalement différent. Choisir PostgreSQL ce n'est pas seulement faire un choix technique, c'est surtout opter pour un virage à 180° dans la relation client-fournisseurs.

PAS DE LICENCES PAYANTES

Opter pour PostgreSQL, c'est tout d'abord quitter le monde des licences propriétaires pour une licence open-source, ici de type BSD. Cette licence donne le droit d'utiliser gratuitement PostgreSQL sans restrictions et sans redevance. Pour autant, cela ne signifie pas que le coût de possession de PostgreSQL est nul : Logiciel Libre ne veut pas dire logiciel gratuit. Il existe un TCO incompressible qui devra être évalué (voir plus loin). Cependant, d'une manière générale, il est largement inférieur au coût de possession des SGBD propriétaires.

PAS DE FREINS À LA VIRTUALISATION

Comme pour la plupart des SGBD propriétaires, le coût de possession des bases Oracle est généralement directement indexé sur le nombre de processeurs des serveurs hébergeant les bases. Pire, dans les environnements virtualisés de type VMware, le tarif reste indexé sur le nombre de processeurs physiques et non pas sur le nombre de processeurs virtuels. Un mode de calcul volontairement pénalisant pour les sociétés qui modernisent leur parc en virtualisant les environnements de production. Avec PostgreSQL, ces freins n'existent pas. PostgreSQL fonctionne parfaitement sur des serveurs physiques ou virtuels, sans que cela n'impacte les coûts de possession. Ce n'est d'ailleurs pas un hasard si de grands acteurs du Cloud Computing ont opté pour PostgreSQL comme base de données par défaut.

> **Pour aller plus loin :**
>
> https ://dali.bo/gaFZ5

PAS DE VENDOR LOCK-IN

La "révolution PostgreSQL" réside aussi dans le choix et la liberté rendus aux utilisateurs. Depuis plus de 20 ans, PostgreSQL est développé par une communauté d'entreprises, d'universitaires et d'utilisateurs répartis dans le monde entier. Comme pour Linux, il n'y a pas d'éditeur unique contrôlant le projet. Dès lors, pour le support, la formation et les autres services associés, les utilisateurs ont le choix entre différents acteurs de différentes tailles et entre différents niveaux de prestation. A tout moment, une société peut changer de prestataire, mais garder ses bases PostgreSQL. En dissociant choix technologique et choix du prestataire, PostgreSQL permet aux entreprises de changer de prestataire de service sans pour autant remettre en cause les fondements de leurs Systèmes d'Information.

PAS DE VENTES LIÉES

Le projet PostgreSQL est totalement indépendant et fonctionne sur tous les environnements de productions modernes. Dès lors, avec le choix de PostgreSQL, chacun est libre de choisir la solution de stockage, le constructeur matériel, le système d'exploitation, les serveurs d'applications et le langage de développement qui conviennent en fonction de son contexte.

PAS D'AUDITS INTRUSIFS

Profitant de leurs positions de quasi-monopoles, les grands éditeurs de bases de données ont fait évoluer leurs pratiques pour mettre une pression plus forte sur leurs utilisateurs, en imposant notamment des audits de parcs pour recenser les instances déployées et revaloriser les contrats de licences. Oracle est la tête de pont de ce mouvement avec des batailles juridiques qui font rage avec plusieurs de leurs grands clients. Pour la majorité des entreprises, le risque d'un audit Oracle est devenu un enjeu majeur dans leur politique de gestion de coûts de licence. À l'inverse, par son modèle décentralisé, PostgreSQL élimine de facto toute menace de ce type. Les entreprises sont libres de déployer autant d'instances que nécessaire et ne risquent pas une explosion de leurs coûts de possession.

> **Plus d'informations sur :**
>
> https ://dali.bo/Vc4DX

POSTGRESQL, C'EST AUSSI...

DES INNOVATIONS RÉGULIÈRES

Depuis 10 ans, la communauté PostgreSQL a adopté un cycle de vie à la fois clair et extrêmement dynamique. Chaque année, une nouvelle version majeure est publiée et introduit des nouvelles fonctionnalités et des améliorations de performances. Ce rythme exceptionnel fait de PostgreSQL le SGBD le plus innovant et le plus évolutif du marché.

Pour preuve, suite à l'arrivée sur le marché des bases NoSQL, la communauté Postgres s'est mise en ordre de marche pour intégrer et développer les fonctionnalités de ces bases d'un nouveau type (format JSON, stockage clé-valeur, etc.)

Le même mouvement est visible dans de nombreux domaines : pour les bases de données Géographiques (SIG), PostgreSQL est devenu la référence de l'industrie. Pour l'interopérabilité des données, c'est également le SGBD le plus puissant au point que l'on parle désormais de PostgreSQL comme d'une plate forme d'intégration de données.

> **Plus d'informations sur :**
>
> https ://dali.bo/3wy8m

Les Foreign Data Wrappers : vers une intégration de données directe et sans ETL

Basés sur la norme SQL/MED, les Foreign Data Wrappers (FDW) sont des connecteurs externes qui permettent à PostgreSQL de lire et écrire des données sur tout type de stockage externe (base Oracle, annuaire LDAP, cluster Hadoop, fichier CSV, web services, etc.) Dès lors, il devient possible de fédérer des données hétérogènes sans avoir à utiliser un outil d'intégration spécifique (ETL). Le chargement des données se fait directement par la base PostgreSQL elle-même et la transformation est spécifiée en langage SQL.

SQL + NoSQL : le meilleur des deux mondes

L'opposition entre bases SQL et NoSQL est dépassée ! L'avenir appartient aux bases hybrides, capables de gérer à la fois les données relationnelles et les données peu structurées. Des bases capables de faire de justes compromis entre constance, disponibilité et performances. PostgreSQL est à la pointe de ce mouvement : avec le format JSON binaire, le stockage clef-valeur hstore, les index GIN, le langage PL/V8, Postgres n'a rien à envier aux moteurs NoSQL tels que Mongodb ou Cassandra.

PostGIS : La référence pour les données géo-localisées

PostGIS est la "cartouche spatiale" : une extension qui ajoute des outils et types de données géographiques supplémentaires pour qualifier et manipuler les données. Créé en 2004, ce projet est aujourd'hui la référence incontournable dans le monde des SIG : Mappy, l'IGN ou encore le projet Open Street Map reposent sur PostGIS et donc PostgreSQL.

UN PROJET EN PLEIN ESSOR

Difficile de prédire les prochaines avancées de PostgreSQL, la nature décentralisée de ce projet rend obsolète la notion de "roadmap" telle qu'on l'entend chez les éditeurs de logiciels propriétaires. On peut toutefois entrevoir 2 grands domaines d'innovations majeurs dans un proche avenir :

Compilation à la volée

Même si le coût des ressources machines a toujours tendance à baisser, le temps processeur et celui des utilisateurs est précieux et tout gain dans ce domaine est bon à prendre. L'introduction dans le code de PostgreSQL de techniques de compilation à la volée (JIT) va dans ce sens. Il s'agit de rendre plus performante l'exécution des requêtes complexes, notamment dans les phases de calcul. Une première étape s'est matérialisée dans la version 11. D'autres devraient suivre.

> **Pour en savoir plus :**
>
> https ://dali.bo/brFB8

PARTITIONNEMENT EXTERNE (``SCALE-OUT'')

Face à l'explosion des volumes de données, les stratégies sont multiples. Le partitionnement "classique" est déjà possible avec PostgreSQL, et dans ses dernières versions son administration a été largement améliorée. Il permet de "découper" une table de plusieurs To et de répartir les données sur de multiples partitions selon un critère sélectif. Le partitionnement externe étend cette possibilité en permettant d'exporter de manière transparente ces tables filles sur d'autres instances. On pourra ainsi mettre en place des stratégies de scale-out et répartir les données sur différents serveurs.

> **Pour en savoir plus :**
>
> https ://dali.bo/Lqzal

PARTIE 2 - CINQ ÉTAPES POUR RÉUSSIR LA TRANSITION

Les bases de données sont des pièces essentielles des Systèmes d'Information modernes. Remplacer une brique aussi importante nécessite de la méthode et un plan de route précis. Les étapes exposées ici sont issues de 10 ans d'expérience dans ce domaine.

Fig. 1 : Processus de transition

L'ordre proposé est donné à titre indicatif et peut bien sûr être adapté à votre contexte. À vous de trouver la formule gagnante pour passer à PostgreSQL en toute sérénité

ÉTAPE 1 - ÉTUDE

À l'origine de toute migration réussie, il y a une analyse précise des enjeux, des différences techniques et de l'effort à produire.

Commençons par dire que toutes les bases ne sont pas candidates à une migration. Par exemple, les bases des progiciels dont l'éditeur ne supporte pas PostgreSQL resteront dans leur environnement. (Au passage, nous vous recommandons d'informer vos éditeurs de progiciel de votre stratégie SGBDR, pour les inciter à porter leurs solutions sur PostgreSQL si ce n'est pas encore le cas.) De la même manière, une base de données dont l'application est en fin de vie n'aura logiquement pas vocation à être migrée.

Parmi les bases potentiellement candidates, certaines sont éligibles à la migration vers PostgreSQL sans problèmes. D'autres demandent plus d'attention. On classe généralement les instances Oracle en 3 catégories selon leur degré de compatibilité avec PostgreSQL :

- Type A : les bases qui peuvent être migrées via un processus automatique, à l'aide d'un outil de migration dédié tel qu'Ora2PG,
- Type B : les bases qui peuvent être transférées de manière semi- automatique ; dans ces cas-là, la majeure partie de l'effort de transition est prise en charge par un processus automatique (comme pour les bases Type A) mais il reste quelques modifications "manuelles" qui devront être réalisées par un DBA maîtrisant à la fois Oracle et PostgreSQL ; le temps nécessaire pour ces opérations est généralement inférieur à 10 jours/homme,

- Type C : les bases qui ne peuvent pas être migrées de manière automatique. Il s'agit principalement de bases Oracle contenant un grand volume de procédures stockées PL/SQL qui implique un travail de réécriture plus important.

La proportion des différents types de base dépend de différents facteurs. Pour une société qui utilisent massivement des ORM, les bases de Type A seront prédominantes. À l'inverse si une société possède beaucoup d'applications métier qui reposent sur des procédures stockées alors les bases de Type B et C seront nombreuses.

Ci-dessous un exemple "classique" d'inventaire de parc :

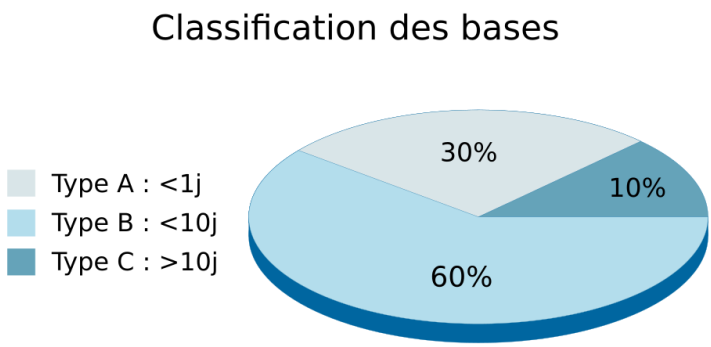

Fig. 2 : Typologie d'un parc d'instance

COMMENT CONDUIRE UN INVENTAIRE DE PARC ORACLE ?

Développé par Gilles DAROLD, salarié de Dalibo, Ora2PG est un outil open-source de migration d'Oracle vers PostgreSQL qui prend en charge la migration des structures (tables, index, vues,...) , des données et d'une large partie du code des procédures stockées.

En 2011, via un partenariat avec le Ministère des Finances et Atos, DALIBO a développé une fonction d'évaluation qui analyse une instance Oracle et en estime la charge de migration.

L'étude préalable doit aussi s'intéresser aux clients qui accèdent à la base de données, car ceux-ci peuvent nécessiter également des adaptations. Le cas le plus favorable est celui des applications qui utilisent un ORM. Un simple changement de paramétrage permettra à l'application d'accéder à PostgreSQL. Pour les applications dont les accès sont codés en SQL, il est probable que des retouches soient nécessaires. L'ampleur de la tâche sera lié

bien sûr au nombre de requêtes codées, mais aussi au degré de respect des normes SQL par les développeurs. Dans certains cas, les API d'accès doivent également évoluer.

COMMENT ÉVALUER UN GRAND PARC DE APPLICATIONS/BASES ORACLE ?

Cette étude des traitements peut s'avérer assez lourde à mener. Aussi, lorsque le nombre de bases de données à examiner est important, nous préconisons une démarche en 3 étapes :

- Recensement : il s'agit de lister les bases/applications Oracle du SI, en collectant pour chacune un petit nombre d'information ; une première sélection permet alors d'exclure de cette première liste les bases qui ne sont à l'évidence pas éligibles,
- Examen des données : la fonction d'audit d'Ora2Pg est passée sur la sélection des bases opérée à l'étape précédente ; ceci permet d'enrichir la liste initiale avec des estimations de difficultés et de charges de migration des données, puis d'opérer une deuxième sélection,
- Examen des traitements : on examine enfin les traitements sur le sous-ensemble sélectionné.

Comment prioriser les bases à migrer ?

Un grand nombre de critères peuvent être pris en compte pour ensuite prioriser les bases à migrer. On peut par exemple commencer par : * les bases techniquement les plus simples, * les bases/applications les moins coûteuses à migrer, * les projets dont le ROI est le plus fort. Bien sûr, il faut pouvoir aussi tenir compte des « roadmaps » applicatives.

Notons que la criticité n'est généralement pas un critère qui entre en ligne de compte dans la priorisation. De nombreuses applications critiques utilisent PostgreSQL.

Le choix de la première base est important. Pour crédibiliser le programme global, la première migration doit être à la fois significative et … absolument réussie. Il faut donc choisir une cible non triviale mais pas trop complexe.

Lorsque les parcs à migrer sont importants, les programmes s'étalent sur plusieurs années. Il peut alors être judicieux d'ordonner les migrations de telle sorte que les gains économiques obtenus sur les premières financent les suivantes.

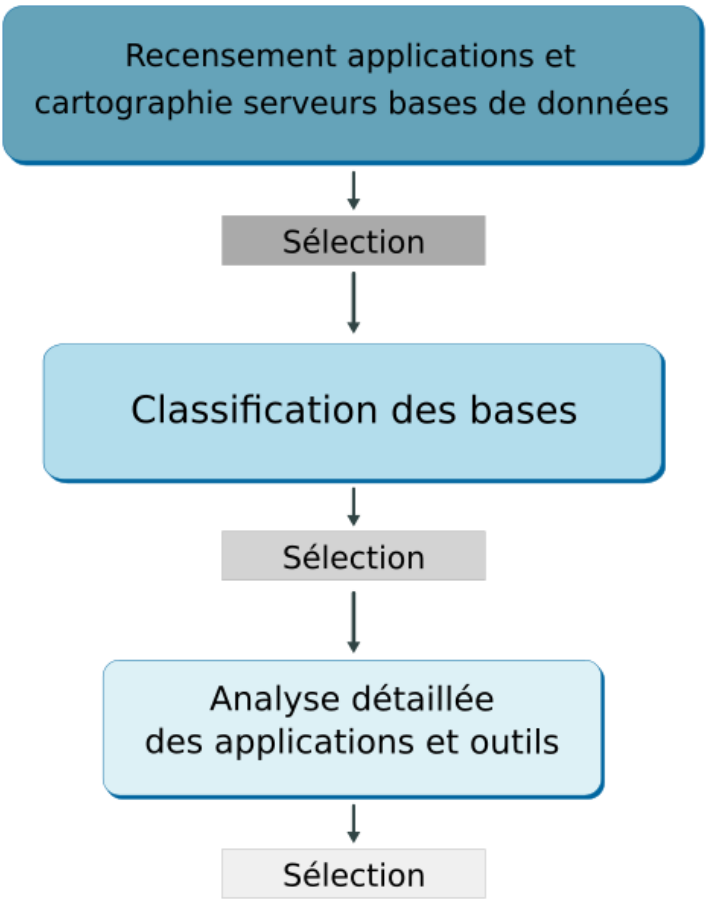

Fig. 3 : Etapes d'évaluation d'un grand parc Oracle

ÉTAPE 2 - FORMATION

L'élément clef de la migration est le transfert des compétences des équipes DBA en place. Le succès d'une transition est étroitement lié à la montée en compétence des équipes en charge de l'exploitation du logiciel.

Les similarités entre Oracle et PostgreSQL sont importantes et des administrateurs Oracle peuvent rapidement monter en compétences sur PostgreSQL. Toutefois le changement peut engendrer des inquiétudes et des réticences. Certains administrateurs de SGBD ayant une longue et riche expérience avec Oracle peuvent vivre la transition comme une certaine dévalorisation de leur travail et une perte de leurs acquis techniques.

Les sessions de formation sont des moments importants pour évoquer ces réticences et désamorcer les a priori. Les équipes techniques peuvent librement poser toutes leurs questions et travailler avec le formateur pour comprendre les variations de concepts, les similarités et les nouvelles possibilités.

La construction d'un savoir-faire autour de PostgreSQL s'appuie nécessairement sur les connaissances des autres SGBD. Il est donc important de choisir un organisme de formation spécialisé et un formateur qui connaîtra PostgreSQL mais également Oracle, pour aider les équipes techniques à acquérir la double-compétence.

Mais il ne faut pas non plus oublier les équipes de développement et les architectes. Non seulement ils se trouveront concernés par le changement, même si c'est dans une moindre mesure que les DBA, mais ils peuvent aussi jouer un rôle de prescripteur dans le choix du SGBD.

QUELLE CERTIFICATION ?

PostgreSQL n'étant pas contrôlé par un éditeur unique, il n'existe donc pas de certification "officielle" pour valider l'acquisition d'un enseignement…

Néanmoins quelques sociétés de la communauté proposent leur propre programme de validation des compétences. Parmi ces sociétés, Dalibo est la seule à délivrer une certification en français qui s'adresse à toute personne exerçant une activité professionnelle impliquant PostgreSQL, ou à toute entreprise souhaitant voir les compétences de ses salariés ou collaborateurs certifiées.

Deux niveaux sont accessibles : "Dalibo Essential Postgres Professionnal" ou "Dalibo Advanced Postgres Professionnal", afin de valoriser les connaissances autour de PostgreSQL, que l'on soit DBA confirmé ou expert dans le domaine.

Bien qu'indépendante de la communauté, la certification Dalibo est le premier programme

de certification PostgreSQL francophone, basée sur les 13 années d'expériences de Dalibo autour de PostgreSQL.

ÉTAPE 3 - SOCLE

PostgreSQL est un logiciel sophistiqué et aux usages variés. Il existe de multiples manières de l'exploiter en production. Et pour chaque fonction, il existe souvent plusieurs outils « satellites » pour enrichir le logiciel de base.

Avant même de déployer une base de données en production, il est important de bâtir un socle PostgreSQL solide. Non seulement il s'agit de mettre en place tout ce qui concourra à une exploitation fiable, maîtrisée et performante. Mais cela permettra ensuite de déployer le SGBD pour tous les projets à venir. L'homogénéisation du parc qui sera ainsi constitué au fil du temps est un gage de qualité et de maîtrise des coûts sur le long terme.

Concrètement, bâtir un socle recouvre les actions suivantes : * définir l'architecture technique dans laquelle la base de données sera exploitée (hébergement physique des données, mode de déploiement, haute disponibilité, politique de sauvegarde, protocole de supervision, sécurité,...), * tester et sélectionner les logiciels « satellites » nécessaires, * développer et tester les procédures de déploiement et d'administration, * rédiger le corpus documentaire indispensable à une bonne exploitation (dossier d'architecture, guide d'installation, manuel d'exploitation, ...).

Il s'agit donc d'un investissement non négligeable. Mais de la qualité de ce travail dépendra la qualité du « service PostgreSQL » rendu.

Il est possible de trouver sur le marché des « socles packagés » recouvrant tout ou partie du besoin.

On pourra se référer au Livre Blanc « Industrialiser PostgreSQL » pour plus de détail sur ce sujet.

QUELS OUTILS ?

Pour chaque outil de la sphère Oracle, il existe des alternatives open-source. Dans le cadre d'une transition vers PostgreSQL, il est important d'établir une liste des logiciels utilisés et leur remplacement. Voici une série d'équivalences donnée à titre d'exemple :

- Rman : Barman ou Pitrery
- Automatic Workload Repository : PoWA
- Grid Control : OPM
- Active Data Guard : Hot Standby

- Database Resident Connection Pool : PgBouncer
- DB LINK : Foreign Data Wrappers

ÉTAPE 4 - MIGRATION

Changer de SGBD n'est bien sûr pas une opération anodine. L'opération doit donc être conduite comme un projet en soi. L'ampleur du projet, et donc les moyens à y consacrer, peut être très variable. Mais tous les projets de migration partagent néanmoins des caractéristiques communes. Hormis les habituelles chantiers transverses de pilotage et d'accompagnement du changement, on peut distinguer les phases suivantes.

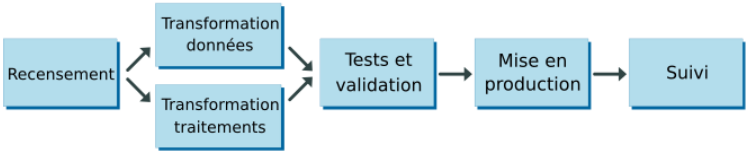

Fig. 4 : Structure du projet de migration

La phase de **recensement** est très rarement coûteuse. Mais il est important de ne pas l'oublier, car elle peut éviter des désagréments lors de la bascule en production. Il s'agit en effet de lister de manière exhaustives les composants qui sont impactés par la migration. Si on pense immédiatement à la base de données elle-même et l'application principale qui l'utilise, il ne faut pas oublier tous les outils périphériques, utilisés par une équipe ou une autre, notamment pour les développements, les tests ou la production. Cette phase est l'occasion de formaliser clairement les relations qui peuvent exister avec d'autres bases de données du Système d'Information.

La phase de **préparation des données** est paradoxalement l'étape la plus triviale du processus de transition.

Il nous faut ici distinguer le contenant (les structures) et le contenu même des données.

Pour ce qui est des **structures**, grâce aux outillages disponibles, l'obtention de la version PostgreSQL du DDL de la base de données est quasi automatique. L'opération peut tout de même nécessiter quelques travaux manuels complémentaires, par exemple pour ajuster la syntaxe SQL des vues ou des triggers, retranscrire des DATABASE LINK avec le module ora_fdw, ou encore transformer les SYNONYMs en vues.

Pour ce qui est des **contenus** de données, là encore les outillages disponibles permettent

de traiter facilement l'opération. Sa durée dépend essentiellement de la quantité de données à migrer. Pour autant, il est parfois nécessaire de dépenser un peu plus d'énergie, par exemple lorsqu'il faut changer l'encodage des données, ou traiter des structures binaires complexes, ou lorsque les contraintes de disponibilité du « service données » ne sont à priori pas compatibles avec le volume des données à migrer.

La phase de **préparation des traitements** a un poids très variable dans le projet. Dans les cas les plus favorables, il n'y a même rien à faire. Les adaptations requises portent sur quatre domaines : * les traitements embarqués dans la base de données, c'est à dire les **procédures et fonctions en PL/SQL**, qu'elles soient ou non intégrées dans des packages. Même si les outillages arrivent à adapter une large part du code, il reste toujours plus ou moins de travaux à conduire manuellement. De la quantité de code embarqué dépend la charge de travail associée. * les **API** d'accès aux données utilisées dans les applications clients. Souvent, les API d'accès aux données sont standardisées. C'est le cas par exemple pour JDBC dans le monde Java. Le code n'est alors pas impacté. Mais parfois, il peut être nécessaire de modifier les méthodes d'accès codées. Toutefois, lorsque ceci s'avère nécessaire, un traitement de masse permet le plus souvent de limiter ce coût d'adaptation. * le code SQL inclus dans les applications clients. Il s'agit là de procéder aux adaptation de syntaxe des requêtes, les 2 langages étant différents sur certains points. Cette tâche sera d'autant plus coûteuse que des syntaxes spécifiques non normalisées ont été utilisées dans le code source. Naturellement, lorsqu'un ORM (comme Hibernate) ou une couche d'abstraction (AdoDB par exemple), sont utilisés, aucune adaptation n'est requise. * les **scripts**. Les scripts qui font appel aux utilitaires Oracle doivent aussi être adaptés. Les équivalents existent (psql pour remplacer isql par exemple). Mais les directives doivent souvent être reprises.

Une fois les données et traitements préparés, la phase de **test** va permettre de s'assurer du bon fonctionnement de l'ensemble des composants cible. Naturellement la charge de travail induite dépendra de la criticité de l'application concernée, et de l'existence de moyens de test automatisés. Sur les gros projets, cette phase de test représente souvent la plus grosse charge de travail. Et c'est une activité difficilement sous-traitable, car elle nécessite une bonne connaissance fonctionnelle des applications.

La phase de bascule est généralement très simple. Le jour J, il faut pour l'essentiel, enchaîner migration des données puis mise en ligne du code applicatif modifié. En cas de souci lors de la migration des données, l'activité peut continuer avec l'ancienne base. Naturellement, un suivi particulier du fonctionnement de l'application et de la base de données doit être assurer durant les quelques jours qui suivent la bascule, afin de détecter et corriger s'il y a lieu les quelques scories résiduelles.

Si l'application est particulièrement critique, il peut être judicieux d'avoir envisagé et testé

une retour arrière, au cas où un problème très grave survenait après l'ouverture en mise à jour de la base de données.

« Migration continue » ?

Les phases de transformation et de test peuvent aussi être menées dans une démarche itérative. Dès qu'un embryon de migration de base est opérationnel, il est possible de créer un environnement de test puis de l'enrichir au fil du temps. A intervalle régulier (1 ou quelques jours), une nouvelle migration de base permettra aux testeurs de disposer d'une base : * contenant des données resynchronisées par rapport à la source, ce qui peut faciliter les comparaisons entre ancienne et nouvelle bases et * intégrant d'éventuels compléments de code procédural fraîchement adaptés ainsi que les corrections de problèmes détectés lors des tests précédents. Lors de la bascule, le processus de migration aura ainsi été parfaitement rodé.

ÉTAPE 5 - SUPPORT

Toute exploitation d'une base de données nécessite un contrat de support. Les SGBD open-source et PostgreSQL en particulier n'échappent pas à cette règle. Ce qui est différent avec ce dernier, c'est qu'il n'y a pas un éditeur unique imposant un modèle de support monolithique.

Comme souvent avec l'open-source, les utilisateurs ont l'embarras du choix. Dès lors, pour ce qui concerne le support PostgreSQL, il est crucial de définir des critères de sélection précis pour choisir un prestataire de support capable de répondre aux besoins et attentes.

Afin de choisir une offre de support, les questions suivantes sont déterminantes :

- Comment le prestataire de support est-il impliqué dans la communauté PostgreSQL ?
- Le prestataire de support a-t-il une équipe d'experts PostgreSQL francophones compétents ?
- Combien de personnes font partie de l'équipe support ?
- Le support est-il en intégralité traité par l'équipe du prestataire ?
- Quelles sont les versions de PostgreSQL supportées ?
- Quelles sont les logiciels satellites supportés ?
- Quelle est la durée de prise en charge de chaque version ?
- Quels sont les engagements de délais de prise en compte et de traitement des appels ?

- Le prestataire de support est-il en capacité d'apporter des services complémentaires de conseil, de formation,... ?
- Le modèle de tarification encourage-t-il à la proactivité ?
- Dispose-t-il d'une base de connaissances accessible ?

> **Pour aller plus loin :**
>
> http ://www.postgresql.org/about/sponsors/

PARTIE 3 - ENJEUX ÉCONOMIQUES

Open-source ne veut pas dire gratuit. PostgreSQL est dénué de coût de licence, ce qui rend bien sûr son coût de possession beaucoup plus faible que les autres SGBD propriétaires et notamment Oracle. Il n'en reste pas moins que toute technologie comporte des coûts associés, aussi bien en termes d'investissement (études, socle, migrations) que de fonctionnement (hébergement, administration, support, formation, conseil). Mesurer les coûts induits et pouvoir déterminer le retour sur investissement d'une migration est un enjeu essentiel.

ÉVALUER L'INVESTISSEMENT

L'investissement lié à une migration dépend des choix opérés au niveau des étapes 1, 3 et 4 présentées précédemment. La phase d'étude, la réalisation d'un socle technique et la migration elle-même représentent les trois principaux postes de dépenses pour garantir une transition maîtrisée.

Comme vu précédemment, chacune de ces 3 étapes peut-être internalisée ou confiée à un ou plusieurs prestataires.

Nous proposons ci-dessous un chiffrage pour la migration de 40 instances de type A réalisées intégralement par un prestataire externe :

Étude	5 à 10 jours	10 k€
Socle	50 à 100 jours	70 k€
Migrations données	40 à 60 jours	50 k€

Il faut y ajouter une charge de test, activité qui, compte tenu de la connaissance fonctionnelle qu'elle nécessite, ne peut pas être sous-traitée, ainsi que l'éventuelle adaptation des traitements.

EVALUER LE TCO

Déterminer le coût d'exploitation de PostgreSQL est une tâche difficile. Comme dit précédemment les logiciels libres comportent des coûts cachés et difficiles à évaluer. On peut néanmoins identifier 3 grandes catégories de dépenses liées au fonctionnement de PostgreSQL :

- L'administration,
- La formation,

- Le support,
- Le conseil.

Les retours de nos clients nous indiquent que d'une manière générale, l'**administration** des bases de données PostgreSQL est plutôt moins coûteuse en charge d'administration que les mêmes bases Oracle. Même si les outils d'administration Oracle sont très performants, PostgreSQL s'avère plus simple à gérer. Ainsi, la bascule progressive du parc de bases d'Oracle vers PostgreSQL ne nécessite pas de renforcer les équipes en place.

La **formation** est considérée comme un coût récurrent. Même si le coût de formation initial peut-être un palier important, il n'est reste pas moins que les équipes techniques doivent-être formées régulièrement pour s'adapter aux nouvelles versions de PostgreSQL, maintenir leur savoir-faire et développer de nouvelles compétences.

Pour 40 instances et une équipe de 3 DBAs :

Conseil	5 à 10 jours / an
Formation	3 à 8 jours / an
Support	5 à 12 jours / an + 45 tickets ouverts

ÉTUDE DE CAS

Le scénario suivant représente un plan de migration classique pour une entreprise ayant un parc de 300 bases Oracle, dont 200 bases de Type A. Parmi ces bases, la société prévoit d'en migrer 150 sur 5 ans (de 2019 à 2024).

Le plan prévoie également que tous les nouveaux projets utiliseront désormais PostgreSQL, ce qui représente 20 instances supplémentaires sur 5 ans.

Coté investissement, elle réalise une étude complète (étape 1) en 2019 pour concevoir son plan de migration, suivi de la rédaction d'un "socle" (étape 3) en 2019-2020 afin d'intégrer PostgreSQL dans son parc logiciel.

Ces deux étapes représentent un coût de démarrage important. Toutefois lorsque ces actions sont réalisées, l'investissement se stabilise et se concentre sur l'effort de migration en lui-même (étape 4).

Comme vu précédemment, le coût de possession PostgreSQL (TCO) est composé essentiellement par l'assistance (conseil et support) et la formation, qui sont des besoins récurrents et indexés sur le nombre d'instances déployées.

Dès lors le TCO est assez simple à évaluer et il bénéficie d'un effet d'échelle évident avec

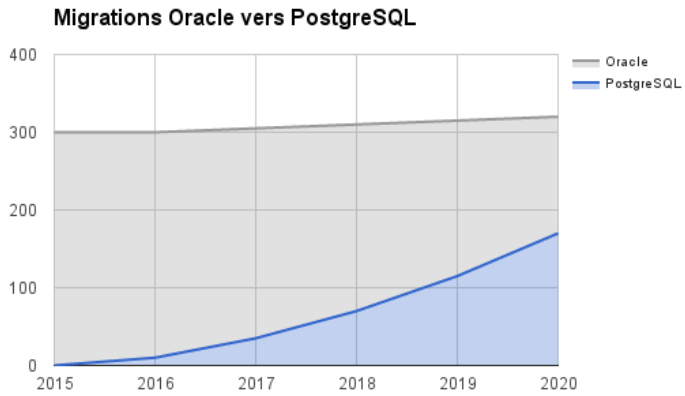

Fig. 5 : Plan de migration

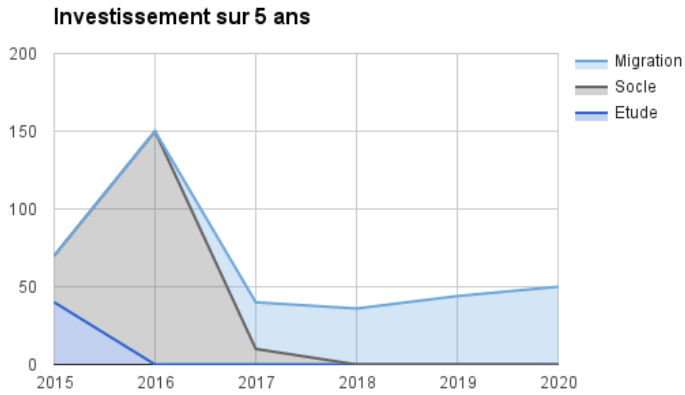

Fig. 6 : Plan d'investissement

la dégressivité relative des coûts du support.

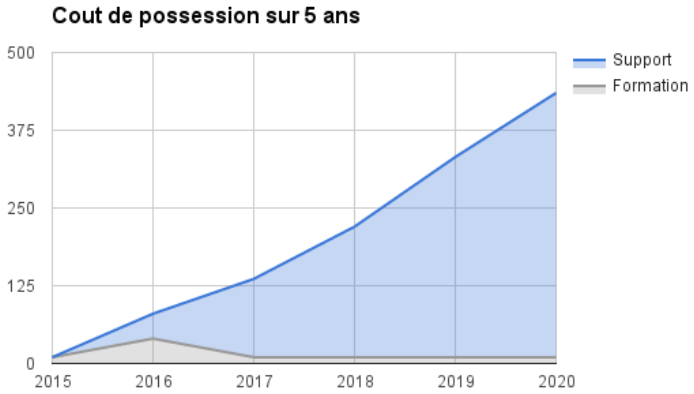

Fig. 7 : Evolution du TCO

PARTIE 4 - POSTGRESQL : POURQUOI MÉTÉO-FRANCE A PARIÉ SUR L'OPEN-SOURCE ?

Bien avant la diffusion de la directive Ayrault promulguant l'utilisation du logiciel libre dans l'administration française, Météo-France prônait en interne l'utilisation de logiciels communautaires (logiciels bureautiques, systèmes d'exploitation, etc). Ainsi, dès les prémices de systèmes de gestion de bases de données relationnel open-source et notamment PostgreSQL, l'ambition était claire : d'une part se libérer de la dépendance à un fournisseur technologique due aux verrous propriétaires et, d'autre part, conserver une maîtrise totale des coûts liés aux bases de données en limitant les coûts de licence.

Pour valider a posteriori le choix stratégique en faveur d'une technologie, ainsi que l'investissement financier et humain qu'il a requis, l'analyse porte autant sur la pérennité du service rendu que sur la constance du niveau de qualité au fil des années en dépit même de l'inévitable accroissement des besoins. Le choix effectué il y a plus de 15 ans par Météo-France d'un système de gestion de bases de données relationnel open-source en remplacement progressif de systèmes propriétaires s'avère judicieux.

Depuis plusieurs années l'Établissement se doit d'atteindre des objectifs qui lui sont fixés par l'État. Ces objectifs concernent en grande partie le domaine de la prévision numérique. Dans ce contexte, une prévision au meilleur de l'état de l'art est une nécessité. Les choix logiciels et matériels effectués ces dernières années par Météo France ont été déterminants pour atteindre ce niveau de qualité. PostgreSQL en fait partie !

LES DONNÉES GÉRÉES SUR SGBDR SONT CRUCIALES POUR LA PRODUCTION DE MÉTÉO-FRANCE.

Une grande partie des activités de Météo-France nécessitent l'acquisition massive et temps réel de données issues d'une variété de systèmes et de techniques tels que les stations automatiques en surface, les satellites, les radars météorologiques, les radiosondages, les capteurs embarqués, etc. Toutes ces données sont stockées dans des bases de données relationnelles.

Parmi ces activités de production centrale figure la prévision numérique du temps. Pour ce domaine, les données sont exploitées par des modèles de prévision numérique hébergés sur un supercalculateur. En effet, pour pouvoir affiner les prévisions sur un domaine géographique, pour pouvoir étendre leur domaine temporel ou encore multiplier les scénarios, il est impératif de disposer de moyens de calculs conséquents.

Depuis plusieurs décennies, Météo-France se dote à cet effet de supercalculateurs dont la tâche principale est de réaliser des prévisions sur l'évolution de l'atmosphère à plus ou

moins long terme. Ces supercalculateurs sont également utilisés dans la recherche sur les phénomènes atmosphériques ou encore pour réaliser les projections utiles aux travaux du GIEC (Groupe d'experts intergouvernemental) sur l'impact du changement climatique. Plusieurs aspects essentiels sont à prendre en compte pour atteindre les objectifs de réalisation des prévisions numériques : la puissance de calcul mais aussi la quantité, la qualité des données disponibles et l'efficacité des accès aux données nécessaires pour pouvoir initialiser les modèles.

Par ailleurs, au-delà du domaine de la prévision numérique, les bases de données relationnelles s'avèrent indispensables sinon critiques pour la plupart des secteurs d'activités de Météo-France. Par exemple, les bases de données sous PostgreSQL hébergent la quasi-totalité des systèmes dédiés aux extranets mis à disposition des clients ou partenaires institutionnels. Pour ces derniers et dans le contexte de vigilance météorologique, il faut pouvoir quantifier et informer de risques locaux utiles aux partenaires de la sécurité civile d'une manière fiable.

Les bases de données sont donc un élément crucial pour le fonctionnement de la chaîne de prévision numérique et plus largement pour la chaîne de production de l'Établissement.

OPEN-SOURCE : UN CHOIX FINANCIER ET STRATÉGIQUE PERTINENT SUR LE LONG TERME

Dès le début des années 2000,l'Établissement a fait le pari des standards ouverts en optant pour des logiciels open-source de classe entreprise supportés par du matériel lui aussi standard (CentOS, serveurs x86, ...). Ce choix repose largement sur des considérations économiques imposant de maîtriser les coûts sur le long terme sans compromis sur la qualité du service rendu.

Côté bases de données, un virage a été pris en 2013 avec le lancement du projet MoBiDiC de migration de l'ensemble des bases de données de production centrales sous PostgreSQL, dernières bases de l'Établissement encore hébergées sous SGBDR propriétaire. Depuis avril 2018 et la fin de ce projet, l'ensemble de la production de Météo-France s'appuie sur le SGBDR PostgreSQL.

Douze ans après la mise en production des premiers serveurs PostgreSQL, on peut dire que non seulement l'objectif a été atteint, mais aussi que le choix reste pertinent. A vrai dire, il l'est plus que jamais puisque la densité des observations s'accroît pour "nourrir" les supercalculateurs et améliorer les prévisions.

Les volumes de données ont augmenté ces dernières années et la tendance va se confirmer (le volume de la base contenant les données de prévision numérique pèse actuellement 11To – ingestion d'environ 1,5To/jour).

Dans ce contexte, le choix de PostgreSQL met Météo-France à l'abri des effets de seuil propres aux fournisseurs de solutions propriétaires : les coûts qui seront engendrés par ces nouvelles volumétries, tant vis-à-vis de l'acquisition de matériel (serveurs, stockage), que des services de support sont maintenant connus. Météo-France conserve le contrôle de ces différents aspects et n'est pas à la merci des changements de licence, de tarification, des fins de support.

OPEN-SOURCE : LA PERFORMANCE DE POSTGRESQL

Sur le plan logiciel, sachant que Météo-France privilégie à ce jour une organisation des données structurées, s'applique le challenge assez classique "volume, variété et vitesse". Il faut pouvoir absorber et insérer rapidement dans les bases de données à des fins d'exploitation parfois temps réel.

Par ailleurs, en météorologie les données historiques ont autant de valeur que les données récentes. Météo-France est soumis à une obligation d'augmenter régulièrement le volume de stockage nécessaire à leur conservation. L'utilisation de ces données devant rester relativement efficace pour ce cadre d'utilisation.

LE RESPECT DES NORMES ET DES STANDARDS : UN PAS EN AVANT VERS UNE COLLABORATION PLUS EFFICACE

Si ces différents bénéfices étaient attendus dès le départ, d'autres avantages ont découlé de ce choix open-source. Le partage de connaissances fait partie des missions que s'est fixé Météo-France en accord avec son contrat d'objectifs et la directive INSPIRE.

Bien que le respect des standards ouverts ne soit pas l'apanage des solutions open-source, ces dernières encouragent une approche vertueuse en matière de qualité des données. Par exemple, l'approche consistant à intégrer des données géo-référencées dans le cadre d'une utilisation de PostgreSQL associé à PostGIS (qui ajoute le support d'objets géographiques à la base de données PostgreSQL) s'avère très pertinente dans le cadre du respect de la Directive INSPIRE et du standard ISO 19115 :2003.

Par ailleurs, la gratuité des logiciels favorise dans une certaine mesure la réutilisation des données de Météo-France, dont certaines sont accessibles gratuitement sur le site de l'Établissement conformément à sa démarche Open Data.

Cette réutilisation peut servir non seulement la collaboration internationale pour une meilleure compréhension des mécanismes globaux complexes qui régissent le climat, mais aussi dans des contextes de recherches universitaires, académiques ou dans une volonté de curation des données.

Ainsi, l'ouverture tend à s'inscrire durablement dans l'ADN de Météo-France. Elle ne concerne pas seulement le code source des logiciels utilisés, mais s'inscrit dans une démarche plus large. L'ouverture y est aussi synonyme d'un décloisonnement visant à stimuler le partage de données, le partage de connaissances pour de meilleures prévisions et une compréhension plus fine des phénomènes météorologiques.

> David PEYRIERES est actuellement référent SGBDR pour Météo France et DBA à la Direction des Systèmes d'Information de Météo-France. Depuis 20 ans il a occupé des postes de DBA dans différentes directions de l'Établissement, principalement sur plateformes Oracle et PostgreSQL. Il a été le Chef du Projet MoBiDiC qui s'est achevé en 2018 et qui consistait à la Migration des Bases de Données Centrales sur PostgreSQL.