SQL / Python, couple gagnant du géotraitement ?

AXES Conseil

Pôle Systèmes d'information de Ségic

axes.fr

axesconseil@axes.fr

Dès son apparition (1999-2001), le SQL Spatial s'était progressivement imposé dans la réalisation de géotraitements dans les bases de données SIG. Mais depuis les années 2010, le langage Python se développe progressivement dans le monde des SIG avec l'apparition de librairies dédiées (Shapely, GeoPandas, etc.) et son intégration dans des logiciels SIG (ArcGIS, QGIS, etc.). La « simplicité » de codage de Python a fortement favorisé cette diffusion. Pour la première fois, Python est devenu en 2025 le langage n° 1 dans l'index TIOBE, qui fait référence dans la communauté des développeurs ; SQL se positionnant à la 10ième position.

Cependant, si le langage Python dispose d'avantages indéniables, il présente également certaines faiblesses.

Python, le dynamisme…au suivi exigeant

L'environnement Python fait preuve d'un dynamisme très (trop ?) marqué : la plateforme Python a une fréquence de mise à jour variant de 18 à 24 mois ; le rythme des mises à jour des librairies peut être plus rapide, parfois moins de 3 mois !

Développer dans un environnement Python, c'est donc accepter et assumer une forte évolutivité dans l'environnement du code. Cela augmente les actions de surveillance et de maintenance du code, pouvant interroger sur le gain de temps effectif entre le codage et la maintenance.

Python offre donc une grande accessibilité et une large plage fonctionnelle, mais au prix d'évolutions de versions très rapides nécessitant une charge de vérification importante.

SQL (et le PL/SQL), la stabilité…un peu austère

Elaboré en 1970, le langage SQL est un cinquantenaire bien portant, faisant toujours l'objet d'évolutions régulières (2016 et 2023 pour les 2 dernières). Le langage SQL se caractérise donc par une grande stabilité : des scripts SQL peuvent être encore opérationnels 5 ou 10 ans après leur primo création. Le temps alors investi est donc bien rentabilisé. Les actions de maintenance sont limitées

Cependant, le codage en SQL (et PL/SQL) a probablement moins d'atouts (vis à vis de Python) pour attirer les primo-codeurs notamment par son formalisme plus exigeant comme par exemple la nécessité de coder en détail des actions plus concises sous Python (ex : gestion des tableaux).

La complémentarité : SQL structure la donnée, Python la valorise.

	Points forts	Points faibles
Python	>Accessibilité >Flexibilité >Fonctions avancées « prêtes à l'emploi » (statistiques, analyse, dataviz, etc.) >Framework	>Evolution très rapide >Lenteur de traitement sur les grands volumes >Travail en mémoire vive
SQL (plSQL)	>Stabilité >Performance sur les grands volumes >Traitement en base > Sécurité	>Formalisme du codage exigeant >Absence de fonctions de dataviz >Environnement de codage moins avancé

Les langages Python et SQL présentent une réelle une complémentarité. Aborder un problème de géotraitement en prenant en compte les deux mondes permet de tirer le meilleur des deux parties :

- SQL pour réaliser les traitements structurants sur la donnée,
- Python pour explorer et visualiser ces données pré-traitées sous SQL.

L'enjeu est alors de bien identifier la répartition des actions entre le SQL et le Python en fonction du contexte :

- Pérennité du développement,
- Compétences techniques en présence,
- Temps disponible pour la réalisation.