BENCHMARK : SYSTÈMES DE GESTION DE BASES DE DONNÉES (SGBD)

Pourquoi choisir un SGBD SQL plutôt que NoSQL ?

Avant de comparer MySQL avec d'autres SGBD SQL, il est important de justifier pourquoi SQL est nécessaire pour le WMS de NTL et pourquoi NoSQL n'est pas adapté.

1. Le WMS repose sur des relations fortes entre données

Exemples :

- un client → plusieurs commandes
- une commande → plusieurs mouvements
- un article → plusieurs localisations
- un site → plusieurs stocks

Ces relations nécessitent **intégrité référentielle, contraintes, clés étrangères** impossibles ou très limitées en NoSQL.

2. Les opérations sont transactionnelles (ACID)

Les mouvements de stock doivent être exactement corrects.
Pas de données perdues, pas de données partiellement écrites.

NoSQL ne garantit pas ACID complet dans la majorité des cas.

3. Le WMS fait beaucoup de lectures + écritures structurées

SQL = optimisé pour :

- requêtes complexes
- jointures
- filtres

- index
- cohérence stricte

NoSQL = optimisé pour des données non structurées ou massivement distribuées, ce qui n'est pas le cas ici.

4. Historique : le WMS utilise déjà MySQL

Le système actuel est déjà en SQL/MySQL, donc NoSQL :

- imposerait une ré-écriture complète
- casserait les modèles existants
- augmenterait les risques
- compliquerait les opérations

Conclusion

Un SGBD SQL est obligatoire pour la cohérence, les transactions, les relations fortes, et la continuité technique.
NoSQL n'est pas adapté aux besoins métiers de NordTransit Logistics.

1. Benchmark SGBD : MySQL vs PostgreSQL vs SQL Server

Critère	MySQL	PostgreSQL	Microsoft SQL Server
Licence	Open source (GPL) + offres commerciales	Open source (PostgreSQL License)	Propriétaire (licences payantes)
Coût total	Très faible	Très faible	Élevé (licences + CAL + édition)
OS supportés	Linux, Windows, macOS	Linux, Windows, macOS	Windows + Linux (partiel)
Popularité web / SaaS	Très forte (LAMP)	Forte, surtout data/analytics	Forte en entreprise Microsoft
Performances OLTP	Excellentes (InnoDB)	Excellentes	Excellentes

Réplication / HA	Réplication native + Group Replication	Réplication avancée (streaming logical)	AlwaysOn, mirroring, clustering
Complexité de mise en œuvre	Faible à moyenne	Moyenne à élevée	Moyenne
Outils & écosystème	MySQL Workbench + large écosystème	PGAdmin + outils avancés	SSMS + outils Microsoft
Communauté / support	Très grande + support éditeurs	Communauté très active	Support Microsoft
Intégration existante	WMS déjà sur MySQL	Nécessite migration	Migration + licences
Apprentissage équipe	Simple	Plus technique	Dépendant Microsoft

2. Analyse :

PostgreSQL n'est pas le meilleur choix

PostgreSQL est un excellent SGBD, mais pas optimal dans le contexte NTL :

Plus complexe que MySQL

PostgreSQL propose des fonctionnalités avancées (types custom, JSONB, procédures complexes…), mais cela augmente la complexité de maintenance pour une équipe de PME.

Moins aligné avec l'existant

Le WMS actuel fonctionne déjà sur MySQL
→ PostgreSQL nécessiterait :

- une migration complète du schéma
- des adaptations SQL
- une montée en compétence importante

Réplication logique plus délicate pour un RPO court

Même si PostgreSQL est excellent en HA, la configuration est :

- plus technique
- plus sensible
- plus complexe pour une équipe réduite

Conclusion :
Troppo complexe pour NTL, migration trop lourde, pas nécessaire pour un WMS transactionnel classique.

SQL Server n'est pas le meilleur choix

SQL Server est puissant, mais totalement inadapté ici :

Coût très élevé

SQL Server = licences + CAL + options selon édition.
Pour une PME comme NTL, cela représente **des milliers d'euros par an**.

Verrouillage sur l'écosystème Microsoft

Il nécessite :

- Windows Server (coût)
- plus d'administration système
- un environnement Microsoft cohérent

NTL a déjà un environnement hétérogène (Linux + Windows)
→ SQL Server n'apporte aucun avantage.

Migration encore plus lourde que PostgreSQL

Le WMS actuel tourne sous MySQL.
Passer à SQL Server = rewriting complet.

Surdimensionné pour le besoin

SQL Server excelle dans des scénarios BI / reporting / gros entrepôts de données.
Le WMS, lui, nécessite :

- OLTP rapide
- intégrité
- réplication
- simplicité

➔Conclusion :

Trop cher, trop complexe, trop surdimensionné.

4. Conclusion finale : Pourquoi MySQL ?

MySQL représente le **meilleur compromis** pour NFL IT et NTL :

- Déjà utilisé par le WMS actuel
- Simple à administrer
- Très performant en OLTP (InnoDB)
- Open source → coûts bas
- HA simple (réplication native + group replication)
- Idéal pour une PME avec une petite équipe IT
- Large communauté et support

MySQL = Fiabilité + Simplicité + Coût faible + Performance

Parfait pour un WMS critique nécessitant un RTO=1h et RPO=15min.