Les vues SQL

Une vue SQL est un objet de base de données qui représente le résultat d'une requête prédéfinie.
En d'autres termes, une vue est une sorte de « **table virtuelle** » dont la structure et les données sont issues du résultat d'une sélection effectuée sur une ou plusieurs tables existantes dans la base de données.

L'objectif principal des vues est de simplifier et structurer l'accès aux données en créant des objets correspondants à des sous-ensembles précis des données de la base.

Cela permet, par exemple, de préparer des données complexes pour faciliter leur utilisation ou de sécuriser l'accès à certaines informations sensibles.

L'application ou l'utilisateur peut utiliser ou manipuler une vue SQL en réalisant une requête sur la vue SQL.

Exécution dynamique : À chaque fois qu'une vue est utilisée, la requête associée est exécutée, fournissant des résultats à jour.

Pas de stockage physique : Les données ne sont pas stockées physiquement dans la base de données. La vue ne contient que la requête.

https://learn.microsoft.com/fr-fr/sql/relational-databases/views/views?view=sql-server-ver16

Pourquoi les vues SQL sont appréciées ?

- **Sécurité** : Une vue permet de restreindre l'accès aux données et de masquer la structure sous-jacente. On peut montrer différentes vues à différents utilisateurs et n'exposer que les données minimales pour chaque usage.
- **Simplicité et abstraction de la complexité** : Plutôt que de travailler directement avec les données réelles et devoir faire des requêtes complexes liant de nombreuses tables, travailler à partir des vues simplifie la compréhension des données mais aussi le travail à partir des données et réduit le risque d'erreurs.

- **Efficacité, maintenabilité** et **cohérence des données** : La création de vues peut être réalisée par des spécialistes et les opérations courantes déléguées à des personnes moins expertes. Ainsi, chacun pourra bénéficier d'une base de travail simple, propre et économe à partir du travail préparatoire réalisé sur les vues.
  Ça n'évitera pas la mauvaise interprétation mais au moins, les données utiles seront toutes de même qualité.
  Et lorsqu'il faudra apporter une modification, il suffira de le faire une fois sur la vue.

- **Centralisation des règles métier** : Proche du point précédent, si tout le monde utilise les vues, tout le monde se base sur les mêmes règles.

- **Encapsulation et réutilisation** : Les vues peuvent être utilisées de plusieurs façons. Inutile de recoder une vue pour l'utiliser dans un nouveau contexte, il suffit de la reprendre et de la réutiliser.

- **Mise en cache des vues** : Certains SGBD permettent de stocker en cache les vues, ce qui permet un accès beaucoup plus rapide et moins exigeant en ressources pour le serveur.

Les contraintes des vues SQL

- **Performances** : Forcément, si on ajoute une couche d'abstraction, ce sera au détriment des performances pures. Mais tout dépend de s'il est important de préserver les ressources du serveur de base de données ou bien s'il faut simplifier la vie aux utilisateurs.

- **Cohérence des données** : Que se passe-t-il si les données sous-jacentes sont actualisées alors que la vue est utilisée ? Quand et comment la vue doit-elle se mettre à jour ?

- **Mise à jour de données** : Il est possible de réaliser des opérations d'ajout/modification/suppression directement en attaquant une vue SQL mais cela impose un travail supplémentaire lors de la conception de la vue.

- **Nombre de vues nécessaires** : S'il y a beaucoup de besoins variés, il faut créer beaucoup de vues.

- **Fonctionnalités de tri et de filtres** : Comme la vue se base sur une table virtuelle, toutes les fonctionnalités classiques associées aux tables ne sont pas forcément utilisables. C'est le cas pour les fonctionnalités de filtres et de tris sur les colonnes de la vue SQL.

- **Surcouche limitante et manque de flexibilité** : En déportant de la logique métier dans les vues, on s'impose des limites pour l'accès aux données. C'est d'autant plus embêtant si le code qui s'exécute se base sur les vues plutôt que sur les données brutes. En cas de migration, le risque est bien réel de devoir réécrire les vues.

Créer des vues :

https://learn.microsoft.com/fr-fr/sql/relational-databases/views/create-views?view=sql-server-ver16

Exécuter les vues :

SELECT * FROM nomVue

Ressources utilisées :

https://www.pairform.fr/les-vues-sql.html

https://blog.alphorm.com/comprendre-vues-sql

https://www.base-de-donnees.com/avantages-inconvenients-vues-sql/