Hello DBMS

Hello veille ...

A. Qu'est ce qu'**une donnée** ? Sous **quelle forme** peut-elle se présenter ?

En data science, le terme "données" représente des informations enregistrées sur différents types de support et qui peuvent être enregistrées, stockées et traitées.
Ces informations sont classées en trois catégories :

- **structurés:** Les données structurées sont organisées de manière tabulaire avec des lignes et des colonnes, suivant un schéma prédéfini. Elles sont souvent stockées dans des bases de données relationnelles.
- **semi-structurés :** Les données semi-structurées ne suivent pas le format d'un modèle de données tabulaire ou de bases de données relationnelles car elles n'ont pas de schéma fixe. Elles peuvent inclure des balises, des paires clé-valeur, des documents JSON, XML..
- **non structurés:** Les données non structurées ne suivent aucune organisation prédéfinie. Elles peuvent inclure du texte, des images, des vidéos, des fichiers audio.

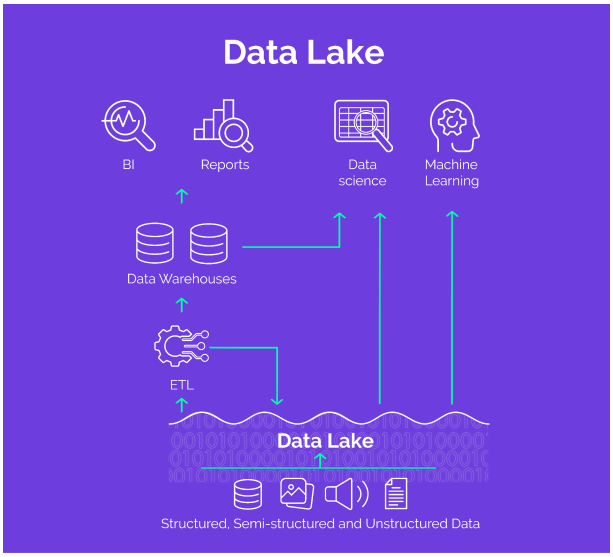
Voici une liste non exhaustive des différentes formes de données :

- Numériques : 42, 3.14.
- Textuelles : chaînes de caractères, documents
- Binaires : True / False
- Temporelles : dates / heures / température /
- Géospatiales : latitudes / longitudes
- Multimédias : images, vidéos
- Tableaux
- Graphes
- Codes informatiques
- Scientifique

B. Donnez et expliquez **les critères de mesure de qualité des données**.

- **Unicité :** l'utilisateur ne doit pas pouvoir confondre deux informations ni deux produits.
- **Exhaustivité** : les données ne doivent pas comporter de champs manquants.
- **Conformité aux conventions de l'entreprise** : le cadre d'utilisation des données doit être défini selon des règles de nommage et de formats de données afin d'éviter des valeurs inexactes et des formats inutilisables.
- **Précision** : chaque donnée doit être correcte et correspondre à la réalité.
- **Incorruptibilité** : la donnée ne doit pas être altérée après utilisation.
- **Cohérence** : les données doivent respecter une certaine logique entre elles et ne pas entrer en conflit en présentant par exemple des informations contradictoires.

C. Définissez et comparez les notions de **Data Lake, Data Warehouse et Lake House**.



Data Lake : c'est un référentiel de données volumineuses et diversifiées qui stocke les données sous leur forme brute, non transformée, sans nécessiter de structures prédéfinies. Les données stockées dans un Data Lake peuvent être structurées, semi-structurées ou non structurées, et elles peuvent provenir de différentes sources.

Avantages :

- Volume de stockage important
- Flexibilité : Les données stockées dans un Data Lake n'ont pas besoin d'être structurées.
- Faible coût
- Evolutif

Inconvénients:

- **La gestion de ces données peut être difficile**, en particulier lorsqu'il s'agit de garantir leur qualité et leur traçabilité.
- **Sécurité** : Comme les données sont stockées sous leur forme brute, le contrôle des accès peut être difficile.
- **Difficulté d'utilisation** : La nécessité de manipuler des données brutes peut rendre plus difficile l'intégration des données avec des outils de BI ou d'analyse.

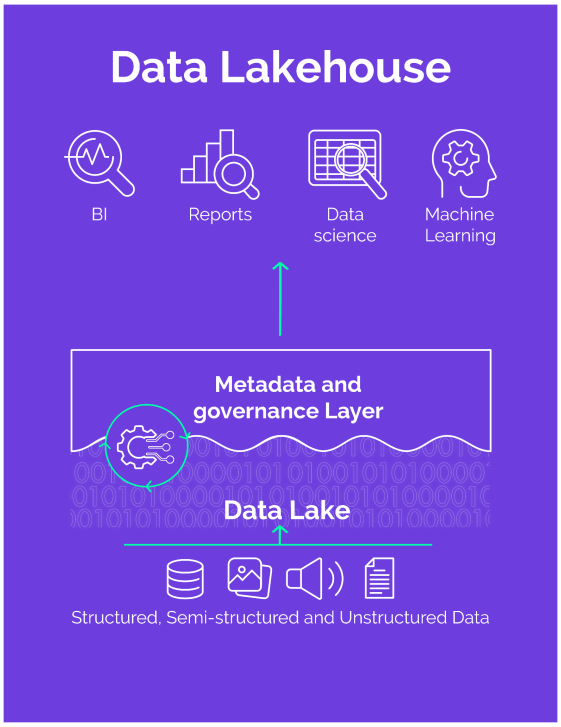
Data Warehouse : C'est un système centralisé qui est optimisé pour le traitement de données structurées. Les données sont extraites, transformées et chargées (ETL) à partir de différentes sources de données afin de créer un référentiel unique de données fiables et cohérentes.

Avantages :

- Centralisation des données
- **Performance** : Les données stockées sont optimisées pour les requêtes analytiques, ce qui permet d'effectuer des analyses et générer des rapports plus rapidement.
- **Fiabilité** : Les données stockées sont plus fiables et de meilleure qualité que celles stockées dans d'autres types de systèmes, en raison des processus de nettoyage et de validation des données.

Inconvénients :

- Coût élevé
- Pas de streaming : Un Data Warehouse n'est pas conçu pour stocker des données en temps réel.
- Flexibilité limitée : Un Data Warehouse ne stocke que des données structurées.



Data Lakehouse : Un Data Lakehouse est une architecture qui combine les avantages du Data Warehouse et du Data Lake. Dans un Data Lakehouse, les données sont stockées sous leur forme brute, mais également organisées en tables pour permettre des requêtes SQL standards. Cela permet aux analystes de travailler sur des données à la fois brutes et agrégées, tout en utilisant des outils SQL standards pour les interroger.

Avantages :

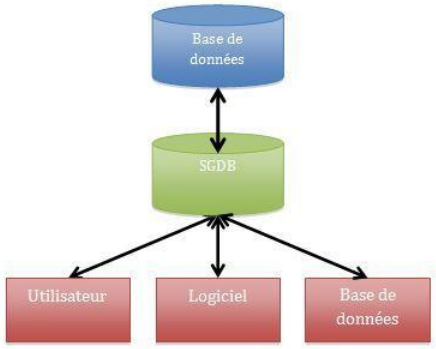
- Flexibilité
- Performances élevées
- Gouvernance simplifiée

Inconvénients :

- Complexe à mettre en place.
- Coût : Les coûts de mise en place et de maintenance sont plus élevés que pour un Data Lake ou Data Warehouse.

D. Donnez une définition et des exemples de **systèmes de gestion de bases de données** avec des illustrations.

Un SGBD est un logiciel système servant à stocker, gérer, trier et utiliser des données dans une base de données. Il garantit la confidentialité, la pérennité et la qualité des informations.



Exemples de Systèmes de Gestion de Bases de Données :

E. Qu'est ce qu'**une base de données relationnelle** ? Qu'est ce qu'**une base de données non relationnelle** ? Donnez **la différence entre les deux** avec des exemples d'applications.

Les bases de données relationnelles sont conçues pour stocker des données dans des tables, qui sont ensuite reliées les unes aux autres par des clés primaires et étrangères. Cela permet aux utilisateurs de faire des recherches complexes en utilisant des requêtes SQL (Structured Query Language) pour extraire des informations à partir de plusieurs tables à la fois.

Elles sont adaptées à des applications nécessitant une gestion structurée des données, couvrant ainsi une variété de secteurs tels que le commerce, les ressources humaines, la finance et la gestion de projets.

Les bases de données non relationnelles, également appelées NoSQL, sont une alternative aux bases de données relationnelles. Contrairement aux bases de données relationnelles, les bases de données NoSQL ne stockent pas les données dans des tables reliées les unes aux autres. Au lieu de cela, les données sont stockées dans des documents, des graphes ou des paires clé-valeur, ce qui permet une grande flexibilité dans le stockage et la gestion des données.

Elles sont idéales pour gérer des données variées des réseaux sociaux, l'analyse de données massives, l'IoT, les jeux en ligne et le streaming de données.

	SQL	NoSQL
Type	Relationnelle	Non-Relationnelle
Données	Données structurées	Données non-structurées
Schéma	Statique	Dynamique
Scalabilité	Verticale	Horizontale
Langage	Structured Query Language	Un-structured Query language
Flexibilité	Rigide	Flexible
Support	Support éditeurs	Support communauté Open-Source

F. Définissez les notions de **clé étrangère** et **clé primaire**.

Une **clé primaire** est une colonne ou un groupe de colonnes qui identifie de manière unique chaque ligne de la table du système de gestion de base de données relationnelle (SGBD). Il ne peut pas s'agir d'un doublon, ce qui signifie que la même valeur ne doit pas apparaître plus d'une fois dans le tableau.

Une table ne peut pas avoir plus d'une clé primaire. La clé primaire peut être définie au niveau de la colonne ou de la table.

Une **clé étrangère** est une colonne qui crée une relation entre deux tables. Le but de la clé étrangère est de maintenir l'intégrité des données et de permettre la navigation entre deux instances différentes d'une entité. Il agit comme une référence croisée entre deux tables car il fait référence à la clé primaire d'une autre table. Chaque relation dans la base de données doit être prise en charge par une clé étrangère.

G. Quelles sont **les propriétés ACID** ?

Quatre propriétés essentielles définissent les transactions des bases de données relationnelles : atomicité, cohérence, isolement et durabilité, généralement appelées ACID.

Atomicité définit tous les éléments constituant une transaction de base de données complète.
Cohérence définit les règles permettant de maintenir les points de données dans un état correct après une transaction.
Isolement conserve l'effet d'une transaction invisible aux autres jusqu'à ce que son engagement soit effectif, afin d'éviter toute confusion.
Durabilité garantit que les modifications de données deviennent permanentes une fois que l'engagement de la transaction est effectif.

H. Définissez les **méthodes Merise et UML**. Quelles sont leur utilité dans le monde de l'informatique ? **Donnez des cas précis d'utilisation avec des schémas**.

Merise est une méthode qui aide à planifier et organiser la création d'un système informatique, en mettant souvent l'accent sur la modélisation des données. C'est comme un plan qui détaille la manière dont les données sont organisées dans une base de données.

Elle possède un certain nombre de modèles (ou schémas) qui sont répartis sur trois niveaux :

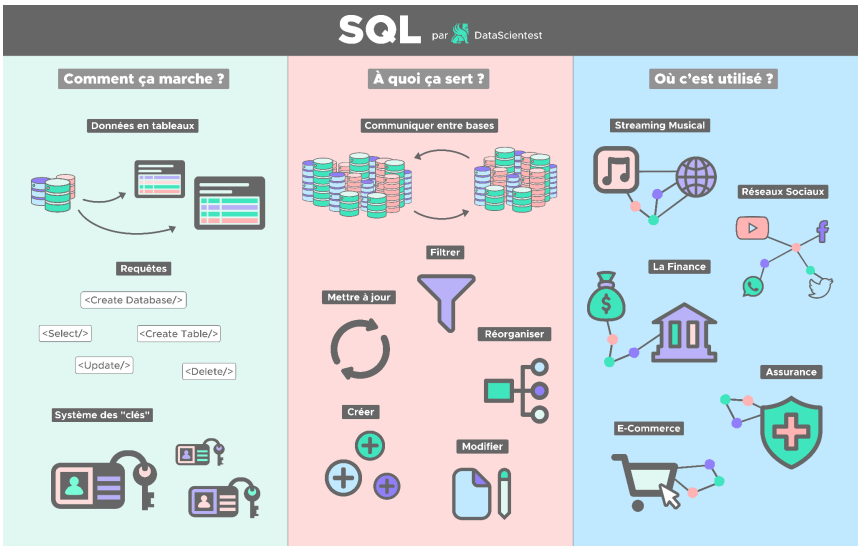
- le niveau conceptuel ;
- le niveau logique ou organisationnel ;
- le niveau physique.

UML(Unified Modeling Language) :

C'est un langage de modélisation visuelle qui permet de créer des schémas visuels pour expliquer comment un logiciel, y compris les bases de données, doit fonctionner.

La différence majeure entre les deux est que Merise est une méthode de projet informatique complète et UML est une notation/langage graphique.

I. Définissez **le langage SQL.** Donnez les commandes les plus utilisées de ce langage et les différentes jointures qu'il est possible de faire.



SQL ou « Structured Query Language » est un langage de programmation permettant de manipuler les données et les systèmes de bases de données relationnelles. Ce langage permet principalement de communiquer avec les bases de données afin de gérer les données qu'elles contiennent.

Il permet notamment de stocker, de manipuler et de retrouver ces données. Il est aussi possible d'effectuer des requêtes, de mettre à jour les données, de les réorganiser, ou encore de créer et de modifier le schéma et la structure d'un système de base de données et de contrôler l'accès à ses données.

Commandes les plus utilisées:

CREATE DATABASE : créer une base de données

CREATE TABLE : créer une table

SELECT : sélectionner les données

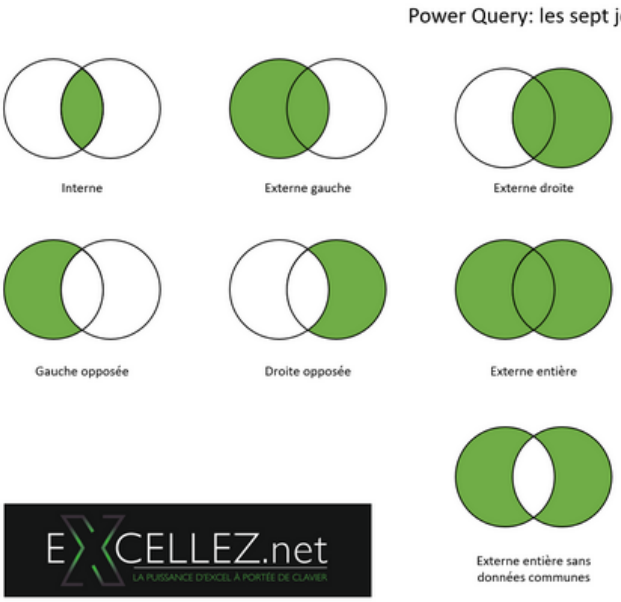
INSERT INTO : ajouter des données dans une table

WHERE : filtre les données

ORDER BY : classer les données

GROUP BY : regrouper les données

Les jointures:



Jointure	Explications
Interne	Associe uniquement les lignes des deux tables dont la clause d'union est vérifiée
Externe gauche	Reprend toutes les lignes de gauche et y associe les lignes de droite qui vérifient la clause d'union
Externe droite	Reprend toutes les lignes de droite et y associe les lignes de gauche qui vérifient la clause d'union
Gauche opposée	Reprend les lignes de gauche qui ne trouvent pas de correspondance à droite sur la clause d'union
Droite opposée	Reprend les lignes de droite qui ne trouvent pas de correspondance à gauche sur la clause d'union
Externe entière	Reprend toute les lignes des deux tables en associant celles qui vérifient la clause d'union
Externe entière sans données communes (N'existe pas nativement mais peut être construite)	Reprend uniquement les lignes des deux tables qui ne vérifient pas la clause d'union

INNER JOIN : jointure interne pour retourner les enregistrements quand la condition est vraie dans les 2 tables.

CROSS JOIN : jointure croisée permettant de joindre chaque lignes d'une table avec chaque lignes d'une seconde table.

LEFT JOIN (ou LEFT OUTER JOIN) : jointure externe pour retourner tous les enregistrements de la table de gauche (LEFT = gauche) même si la condition n'est pas vérifiée dans l'autre table.

RIGHT JOIN (ou RIGHT OUTER JOIN) : jointure externe pour retourner tous les enregistrements de la table de droite (RIGHT = droite) même si la condition n'est pas vérifiée dans l'autre table.

FULL JOIN (ou FULL OUTER JOIN) : jointure externe pour retourner les résultats quand la condition est vraie dans au moins une des 2 tables.

SELF JOIN : permet d'effectuer une jointure d'une table avec elle-même comme si c'était une autre table.

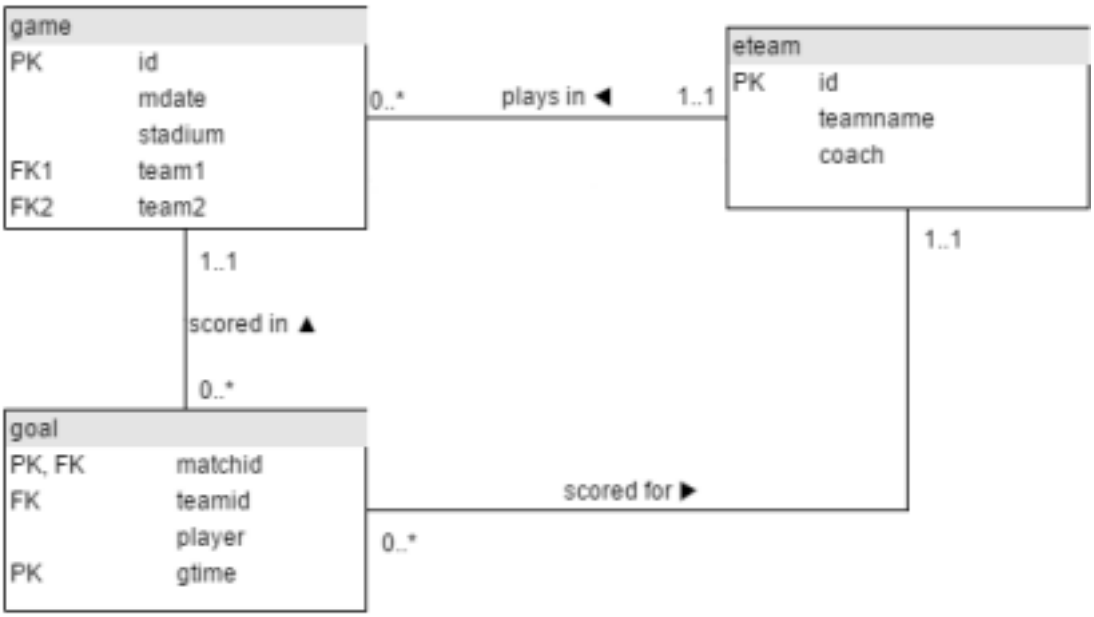
NATURAL JOIN : jointure naturelle entre 2 tables s'il y a au moins une colonne qui porte le même nom entre les 2 tables SQL

Hello SQL...

Job 7

Soit la base de données **UEFA EURO 2012** constituée des tables suivantes : **Game**

Modèle relationnel de la base de donnée :



1. Observez le schéma relationnel de la base de données **UEFA EURO 2012** ci-dessus. **Analysez les cardinalités**.

1..1 (Un à un) : une entité est associée à exactement un enregistrement dans une autre entité, et vice versa. Chaque enregistrement a une correspondance unique.
0 ..*(0 à plusieurs) :Indique qu'une entité peut être associée à zéro ou plusieurs enregistrements dans une autre entité. Il n'y a pas de limite supérieure au nombre d'associations possibles.

Table "game" et "goal" :
“1..1” : Indique qu’un match peut avoir 0 ou plusieurs buts
“0..*” : Indique qu’il peut y avoir 0 ou plusieurs buts dans un match

Table “goal” et “eteam”:

“0..*” : Indique que 0 ou plusieurs buts peuvent être marqué par une équipe

“1..1” : Indique qu’une équipe peut marquer 0 ou plusieurs buts.

Table “eteam” et “game”:

“1..1”: Indique qu’une équipe peut faire 0 ou plusieurs matchs

“0..*” : Indique qu’il peut y avoir 0 ou plusieurs matchs par équipe.