# evaluation1_l_noah

| | |
|---|---|
| Responsable | noah lhote |
| Tags | Infrastructure SQL |

Lhote Noah.

1. ***L'algèbre relationnelle est une fermeture. Rappeler brièvement ce que cela signifie.***
   ***Comment cette opération se traduit-elle concrètement en SQL (quelles implications) ?***

   En algèbre relationnelle, la fermeture signifie que les opérations génèrent stoujours des résultats sous forme de relations.

   En SQL, cela se traduit par le fait que chaque requête produit une table relationnelle en résultat, assurant ainsi la conformité aux principes de l'algèbre relationnelle. les résultats des requêtes SQL sont toujours des ensembles de données relationnels qui respectent les règles de l'algèbre relationnelle.

2. ***Évalue la véracité de chaque proposition suivante en indiquant VRAI ou FAUX***

   1:Faux,
   2:Faux,
   3:Faux,
   4:Vrais,
   5:Vrais,
   6:Vrais
   7:Vrais,
   8:Vrais,

9:Vrais,
10:Vrais,
11:Vrais,
12:Faux,
13:Faux,
14:Faux,
15:Faux.

3. ***Quelle instruction SQL permet d'afficher toutes les contraintes présentes sur une table ?***

```sql
SHOW CREATE TABLE nom_de_la_table;
```

4. ***Qu'est ce que la normalisation des données ?***
   ***Que peut-on utiliser pour être guidé dans le processus de normalisation de son schéma de base de données ?***

   la normalisation vise à résoudre des problèmes pour garantir que la base de données est bien structurée, facile à maintenir, et que les opérations de gestion des données se déroulent de manière fiable et efficace.

   et pour être guidé dans le processus de normalisation de son schma de bdd on peut s'appuyer sur plusieurs technique tel que:
   utilisation des clés (primaire/etrangère), utiliser des outils de modélisation, analyser les besoins métier, Identifiez les dépendances fonctionnelles entre les attributs pour comprendre comment les données sont liées les unes aux autres

5. ***Quel risque encourt on à utiliser une base de données non normalisée ?***

   utiliser une base de données non normalisée peut entraîner des problèmes d'intégrité des données, de performances, de maintenance et de cohérence. La normalisation vise à minimiser ces risques en structurant les données de manière optimale et en éliminant les redondances inutiles.

6. ***A quoi servent les opérations de jointure dans une base de données relationnelle ?***

   Les opérations de jointure dans une base de données relationnelle servent à combiner des données provenant de différentes tables en fonction de relations prédéfinies.

7. ***Quelle est la différence entre une jointure interne et une jointure externe ? Comment ces opérations sont-elles définies en MySQL (instructions) ?***

   - **Jointure Interne (INNER JOIN) :** Retourne uniquement les lignes qui ont des correspondances dans les deux tables. Les lignes où il n'y a pas de correspondance dans l'une des tables ne sont pas incluses dans le résultat.

   - **jointure Externe (OUTER JOIN) :** Retourne toutes les lignes de l'une des tables, ainsi que les lignes correspondantes de l'autre table. Si aucune correspondance n'est trouvée dans la table secondaire, les colonnes de cette table contiendront des valeurs NULL dans le résultat.

```
SELECT colonnes
FROM table1
LEFT JOIN table2 ON table1.colonne = table2.colonne;
```

```
Jointure Interne (INNER JOIN) :
SELECT colonnes
FROM table1
INNER JOIN table2 ON table1.colonne = table2.colonne;
```

8. ***Donner un usage, une utilité des Vues au niveau externe.***

   ca offre une simplification de l'accès aux données, une sécurité renforcée, et la possibilité de personnaliser les résultats sans exiger une connaissance approfondie de la structure sous-jacente de la base de données.

9. ***À quoi sert l'instruction GROUP BY ?***
   ***À quoi sert l'instruction HAVING ?***
   ***Donner un exemple en MySQL d'une opération de regroupement et d'usage de HAVING sur une table.***

   **GROUP BY :**
   L'instruction GROUP BY en SQL est utilisée pour regrouper les lignes qui ont les mêmes valeurs dans des colonnes spécifiées. Elle est souvent utilisée en combinaison avec des fonctions d'agrégation telles que COUNT, SUM, AVG, etc.

   **HAVING :**
   L'instruction HAVING en SQL est utilisée pour filtrer les résultats des opérations de regroupement basées sur des conditions spécifiées. Elle s'applique après l'instruction GROUP BY et permet de filtrer les résultats agrégés.

```sql
SELECT ClientID, SUM(Montant) as MontantTotal
FROM Commandes
GROUP BY ClientID
HAVING MontantTotal > 1000;
```

10. ***Qu'est-ce qu'une sous-requête ?***
    ***Quelle alternative existe-t-il en MySQL à l'usage des sous-requêtes ?***
    ***Quel est l'avantage de cette alternative ?***

- Une sous-requête est une requête SQL incorporée à l'intérieur d'une autre requête. Elle est utilisée pour effectuer une opération de requête à l'intérieur d'une requête principale.
- Une alternative à l'utilisation de sous-requêtes est l'utilisation de jointures, où vous combinez les données de plusieurs tables dans une seule requête plutôt que d'utiliser une requête imbriquée.
- L'avantage des jointures par rapport aux sous-requêtes est souvent une meilleure performance, car les jointures permettent au moteur de base de données d'optimiser l'exécution de la requête de manière plus efficace.

11. **Pourquoi est-il important de bien préciser le SGBDR utilisé lors de la recherche de documentation :**

    car chaque SGBDR peut avoir des syntaxes SQL spécifiques, des fonctionnalités différentes et des comportements uniques.

12. ***Définir (sans nécessairement justifier) le type de données le mieux approp***
    a. Un nom de jour de la semaine : Chaîne de caractères (VARCHAR).
    b. Un nom de mois de l'année : Chaîne de caractères (VARCHAR).
    c. Un numéro de semaine : Entier (INTEGER).
    d. Un prénom : Chaîne de caractères (VARCHAR).
    e. Un numéro de téléphone (sans l'indicateur pays) : Chaîne de caractères (VARCHAR) ou Numérique (INTEGER).
    f. Le contenu d'un article (de journal, de blog) : Texte (TEXT).
    g. Un trigramme (un mot composé de trois lettres) : Chaîne de caractères fixe de longueur 3 (CHAR(3)).

13. ***Expliquer pourquoi la syntaxe de cette instruction CREATE TABLE est incorrecte***
    a. **Type de données INTEGER(16) :** La spécification de la longueur (16) n'est pas valide pour le type INTEGER. INTEGER est généralement sans

spécification de longueur. Vous devriez simplement utiliser "id INTEGER" sans indiquer une taille.

b. **DEFAULT CURRENT DATETIME :** L'utilisation correcte serait "DEFAULT CURRENT_TIMESTAMP" pour définir la valeur par défaut à la date et à l'heure actuelles.

c. **CONSTRAINT UNIQUE date, code :** La syntaxe pour définir plusieurs colonnes comme clés uniques devrait être "CONSTRAINT nom_de_contrainte UNIQUE (date, code)" au lieu de "CONSTRAINT UNIQUE date, code".

```
CREATE TABLE MaTable (
    id INTEGER PRIMARY KEY,
    nom CHAR(32) NOT NULL,
    date DATETIME DEFAULT CURRENT_TIMESTAMP,
    libelle VARCHAR(128),
    code VARCHAR(4) CHECK(code BETWEEN 'AAAA' AND 'ZZZZ'),
    ordre INT,
    CONSTRAINT nom_de_contrainte UNIQUE (date, code)
);
```