Note Pays mont modifié job 4

SQL

1. Qu’est ce qu’une donnée ? Sous quelle forme peut-elle se présenter ?

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

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

House. Illustrez les différences à l’aide de schémas.

D. Donnez une définition et des exemples de systèmes de gestion de bases de

données avec des illustrations.

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.

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

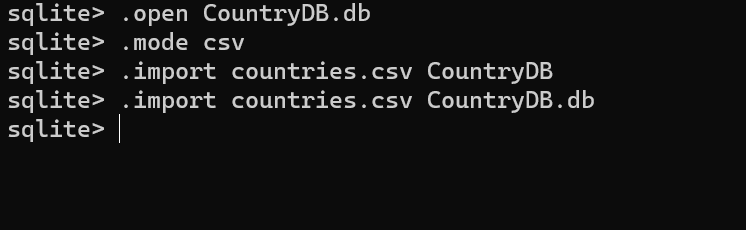
G. Quelles sont les propriétés ACID ?

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.

I. Définissez le langage SQL. Donnez les commandes les plus utilisées de ce

langage et les différentes jointures qu’il est possible de faire.



Job 1 :

**1.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) = 'Germany'

**2.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **IN** ('Sweden', 'Norway', 'Denmark');

**3.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** Area > '200000' **AND** Area < '300000'

**Job 2 :**

**1. Créez une requête permettant de trouver les noms de pays commençant par la**

**lettre B.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **LIKE** 'B%'

**2. Créez une requête permettant de trouver les noms de pays commençant par “Al”.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **LIKE** 'Al%'

**3. Créez une requête permettant de trouver les noms de pays finissant par la lettre**

**y.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **LIKE** '%y'

**4. Créez une requête permettant de trouver les noms de pays finissant par “land”.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **LIKE** '%land'

**5. Créez une requête permettant de trouver les noms de pays contenant la lettre w.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **LIKE** '%w%'

**6. Créez une requête permettant de trouver les noms de pays contenant “oo” ou**

**“ee”.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **LIKE** '%oo%' **OR** **TRIM**(Country) **LIKE** '%ee%'

**7. Créez une requête permettant de trouver les noms de pays contenant au moins**

**trois fois la lettre a.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **LENGTH**(**REPLACE**(Country, 'a', '')) <= **LENGTH**(Country) - 3;

**SELECT** \*

**FROM** Countryyolow

**WHERE** (Country **LIKE** '%a%a%a%') **AND** **NOT** (Country **LIKE** '%a%a%a%a%');

**8. Créez une requête permettant de trouver les noms de pays ayant la lettre r**

**comme seconde lettre.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** **UPPER**(Country) **LIKE** '\_R%';

**Job 3**

**1. Créez une requête permettant d’afficher toutes les colonnes de la table students.**

**SELECT** \*

**FROM** students

**2. Créez une requête permettant de filtrer la table et d’afficher les élèves âgés de**

**strictement plus de 20 ans.**

**SELECT** \*

**FROM** students

**WHERE** age > 20

**3. Créez une requête permettant de faire un classement des élèves selon leur note**

**dans un ordre croissant, puis dans un ordre décroissant.**

La question pose une grosse problématique car A+ et A- seront quel que soit le tri effectuer afficher avant ou après A, il faut donc soit utiliser une condition CASE dans le order by ou bien garder un tableau parallèle associant des valeurs numériques à chaque possibilitées de notes, puis effectuer une requête INNER Join and ORDER BY la colonne de tri numérique.

Croissant

**SELECT** \*

**FROM** students

**WHERE** Ranking **IN** ('A+', 'A', 'A-', 'B+', 'B', 'B-', 'C+', 'C', 'C-', 'D+', 'D', 'D-', 'F')

**ORDER** **BY**

**CASE**

**WHEN** Ranking = 'A+' **THEN** 1

**WHEN** Ranking = 'A' **THEN** 2

**WHEN** Ranking = 'A-' **THEN** 3

**WHEN** Ranking = 'B+' **THEN** 4

**WHEN** Ranking = 'B' **THEN** 5

**WHEN** Ranking = 'B-' **THEN** 6

**WHEN** Ranking = 'C+' **THEN** 7

**WHEN** Ranking = 'C' **THEN** 8

**WHEN** Ranking = 'C-' **THEN** 9

**WHEN** Ranking = 'D+' **THEN** 10

**WHEN** Ranking = 'D' **THEN** 11

**WHEN** Ranking = 'D-' **THEN** 12

**WHEN** Ranking = 'F' **THEN** 13

**END** **ASC**;

Décroissant

Il suffit d’inverser la méthode d’order by sur la dernière ligne avec :

**END DESC**

**Job 4**

**1. Créez une requête permettant d’afficher les prix nobels de 1986.**

**SELECT** \*

**FROM** nobel

WHERE yr = 1986

**2. Créez une requête permettant d’afficher les prix nobels de littérature de 1967.**

**SELECT** \*

**FROM** nobel

**WHERE** yr = 1967 **AND** subject = 'Literature'

**3. Créez une requête permettant d’afficher l’année et le sujet du prix nobel d’Albert**

**Einstein.**

**SELECT** yr, subject

**FROM** nobel

**WHERE** winner = 'Albert Einstein'

**4. Créez une requête permettant d’afficher les détails (année, sujet, lauréat) des**

**lauréats du prix de Littérature de 1980 à 1989 inclus.**

**SELECT** \*

**FROM** nobel

**WHERE** yr >= '1980' **AND** yr <= '1989'

**5. Créez une requête permettant d’afficher les détails des lauréats du prix de**

**Mathématiques. Combien y en a-t-il ?**

N\_Winners devrait afficher Le nombre de lauréats du prix de Mathématiques que j’ai traduit par Mathematics vu que la table est en anglais.

J’ai choisi COUNT(winner) au lieu de COUNT(\*) pour m’assurer de ne pas récupérer les colonnes avec des valeurs NULL.

**SELECT** \*, **COUNT**(winner) **AS** N\_Winners

**FROM** nobel

**WHERE** subject = 'Mathematics'

**Job 5**

**1. Créez une requête permettant d’afficher les pays dont la population est**

**supérieure à celle de "Russia".**

**SELECT** \*

**FROM** Countryyolow

**WHERE** Population > (**SELECT** Population **FROM** Countryyolow **WHERE** **TRIM**(Country) = 'Russia');

**2. Créez une requête permettant d’afficher les pays d'Europe dont le PIB par habitant est supérieur à celui d’ "Italy".**

Il me semble que le PIB et le GDP ne sont pas exactement la même chose mais :

**SELECT** \*

**FROM** Countryyolow

**WHERE** GDP > (**SELECT** GDP **FROM** Countryyolow **WHERE** **TRIM**(Country) = 'Italy');

**3. Créez une requête permettant d’afficher les pays dont la population est supérieure à celle du Royaume-Uni mais inférieure à celle de l'Allemagne.**

**SELECT** \*

**FROM** Countryyolow

**WHERE** Population > (**SELECT** Population **FROM** Countryyolow **WHERE** **TRIM**(Country) = 'United Kingdom') **AND** Population < (**SELECT** Population **FROM** Countryyolow **WHERE** **TRIM**(Country) = 'Germany') ;

**4. L'Allemagne (80 millions d'habitants) est le pays le plus peuplé d'Europe.**

**L'Autriche (8,5 millions d'habitants) compte 11% de la population allemande.**

**Créez une requête permettant d’afficher le nom et la population de chaque pays d'Europe, en pourcentage de la population de l'Allemagne.**

**SELECT**

Country,

Population,

**ROUND**((Population \* 100 / 80000000.0), 1) || '%' **AS** pourcentage

**FROM** Countryyolow

**WHERE** **TRIM**(Region) **LIKE** '%EUROPE%';

**5. Créez une requête permettant de trouver le plus grand pays de chaque continent, en indiquant son continent, son nom et sa superficie.**

**SELECT** Country, Region, **Max**(Population)

**FROM** Countryyolow

**GROUP** **BY** Region

**6. Créez une requête permettant de trouver les continents où tous les pays ont une population inférieure ou égale à 25 000 000.**

**SELECT** Region

**FROM** Countryyolow

**GROUP** **BY** **TRIM**(Region)

**HAVING** **MAX**(Population) <= 25000000

**Job 6**

**1. Créez une requête permettant d’afficher la population totale du monde.**

**SELECT** **SUM**(Population)

**FROM** Countryyolow

**2. Créez une requête permettant d’afficher la population totale de chacun des continents.**

**SELECT** Region, **SUM**(Population) **as** Total\_Population

**FROM** Countryyolow

**GROUP** **BY** Region

**3. Créez une requête permettant d’afficher le PIB total du continent de chacun des continents.**

**SELECT** Region, **SUM**(GDP) **as** Total\_PIB

**FROM** Countryyolow

**GROUP** **BY** Region

**4. Créez une requête permettant d’afficher le PIB total du continent africain.**

**SELECT** Region, **SUM**(GDP) **as** Total\_GDP

**FROM** Countryyolow

**WHERE** **TRIM**(Region) **LIKE** '%AFRICA%'

**5. Créez une requête permettant d’afficher le nombre de pays ayant une superficie supérieure ou égale à 1 000 000m2.**

**SELECT** **COUNT**(Country) **as** Country\_1M²\_Count

**FROM** Countryyolow

**WHERE** AREA < 1000000

**6. Créez une requête permettant d’afficher la population totale des pays suivants : Estonia, Latvia, Lithuania.**

**SELECT** **SUM**(Population)

**FROM** Countryyolow

**WHERE** **TRIM**(Country) **IN** ('Estonia', 'Latvia', 'Denmark');

**7. Créez une requête permettant d’afficher le nombre de pays de chaque continent.**

**SELECT** Region, **COUNT**(Country)

**FROM** Countryyolow

**GROUP** **BY** Region

**8. Créez une requête permettant d’afficher les continents ayant une population totale d’au moins 100 millions d’individus.**

**SELECT** Region, Population

**FROM** Countryyolow

**GROUP** **BY** **TRIM**(Region)

**HAVING** **MAX**(Population) >= 100000000

**Job 7**

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

Eteam = typo ?

Il est possible que nous soyons plusieurs à avoir lu le tableau dans le mauvais sens pour les cardinalités, ce qui rendrait la formulation originale plus cohérente.

Game – goal et eteam – goal paraissent selon moi être à l’envers, dans la mesure où un match devrait être associé à 0 ou plus de goals (1 : 0..N) et chaque goal ne peut être associé qu’à un seul match où il a été marqué (N..1).

C’est également vrai pour eteam – goal, où une équipe peut être associée à plusieurs goals (1 :N) et un goal ne peut être associé qu’à une seule équipe qui l’a marqué (0 :1).

Pour ce qui est de la relation game – eteam elle n’est pas cohérente, plusieurs games devraient pouvoir être associés à plusieurs équipes dans une relation (N : N) tandis qu’une eteam devrait pouvoir être associée à 0 ou plus games (0..N)

**2. La requête ci-dessous permet d’afficher le but marqué par un joueur dont le nom de famille est "Bender". L'astérisque (\*) indique qu'il faut énumérer toutes les colonnes du tableau - une façon d’appeler toutes les colonnes de la table goal (matchid, teamid, player, gtime). Modifiez cette requête afin d’afficher le numéro de match et le nom du joueur pour tous les buts marqués par l'Allemagne. Afin d’identifier les joueurs allemands, vérifiez que : teamid = 'GER'.**

**SELECT** matchid, player

**FROM** goal

**WHERE** **TRIM**(teamid) = 'GER'

**3. Créez une requête permettant d’afficher les colonnes id, stadium, team1, team2 pour le match dont l’id est 1012.**

**SELECT** id, stadium, team1, team2

**FROM** game

**WHERE** id = '1012'

**4. La requête suivante permet de joindre la table game et la table goal sur la colonne id-matchid. Modifiez cette requête afin d’afficher player, teamid, stadium et mdate de chaque but allemand.**

Je n’utilise pas d’ALIAS car les tables game et goal portent facilement à confusion si raccourci.

Go est également attribué en sql donc .go .ga n’est pas une option.

**SELECT** goal.player goal.teamid, game.stadium, game.mdate

**FROM** game

**JOIN** goal **ON** game.id = goal.matchid

**WHERE** goal.teamid = 'GER'

**5. Créez une requête permettant d’afficher team1, team2 et player pour chaque but marqué par un joueur appelé Mario.**

**SELECT** game.team1, game.team2, goal.player

**FROM** game

**JOIN** goal **ON** game.id = goal.matchid

**WHERE** goal.player **LIKE** '%Mario%'

**6. Créez une requête permettant de joindre la table goal et la table eteam sur les clés id - teamid.**

**SELECT** \*

**FROM** goal

**JOIN** eteam **ON** goal.teamid = eteam.id

**7. Créez une requête permettant d’afficher player, teamid, coach, gtime pour tous les buts marqués dans les 10 premières minutes des matchs.**

**SELECT** g.player, g.teamid, e.coach, g.gtime

**FROM** goal g

**JOIN** eteam e **ON** g.teamid = e.id

**WHERE** g.gtime <= 10

**8. La requête suivante permet de joindre la table game et la table eteam sur les clés team1 – et eteam.id. Créez une requête permettant d’afficher les dates des matches ainsi que le nom de l'équipe dont "Fernando Santos" était le coach de l’équipe team1.**

**SELECT** g.mdate, e.teamname

**FROM** game g

**JOIN** eteam e **ON** g.team1 = e.id

**WHERE** e.coach **LIKE** '%Fernando Santos%';

**9. Créez une requête permettant d’afficher la liste des joueurs pour chaque but marqué lors d'un match dont le stade était le “National Stadium, Warsaw”.**

**SELECT** g.player

**FROM** game ga

**JOIN** goal g **ON** ga.id = g.matchid

**WHERE** ga.stadium **LIKE** '%National Stadium, Warsaw%';

**10.Créez une requête permettant d’afficher le nombre total de buts marqués pour chaque équipe de la table goal.**

**11.Créez une requête permettant d’afficher les stades et le nombre de buts marqués dans chacun des stades de la jointure de game-goal.**

**12.Pour chaque match où l’équipe de France a marqué, créez une requête permettant d’afficher l'id du match, la date du match et le nombre de buts marqués par "FRA".**