

# Hello DBMS+

An SQL query walks into a bar, walks up to two tables and says: "CAN I JOIN YOU?"

## Hello data...

---

Les données sont des éléments servant de base à un **raisonnement** ou de **point de départ à une recherche**. A partir de **données** dites **transformées**, il est possible d'**extraire de l'information primordiale** pour les entreprises.

Une donnée peut être perdue ou erronée, **c'est ici qu'apparaît l'intérêt d'un système de gestion de bases de données approprié**. Par ailleurs, les données doivent être stockées sur une base de données **pour améliorer la performance de l'entreprise**. Il est **important** que celles-ci **soient non corrompues, propres, centralisées et accessibles**.

Selon des estimations recensées par Statista, le volume de données numériques créées ou répliquées à l'échelle mondiale a été **multiplié par 30**



**au cours de la dernière décennie**, passant de 2 zettaoctets en 2010 à 64 zettaoctets l'année dernière.

Mais cette quantité apparaît peu élevée en comparaison avec ce qui est attendu dans



les prochaines années. D'après ces prévisions, le volume de données générées dans le monde devrait dépasser 180 zettaoctets à l'horizon 2025, soit une croissance annuelle moyenne de près de 40% sur cinq ans.

Afin de se donner une idée de l'échelle : **un zettaoctet équivaut à un milliard de téraoctets**, soit mille milliards de gigaoctets. Cela signifie qu'il faudrait se procurer 640 millions des **plus gros disques SSD** actuellement commercialisés (100 To de stockage) pour être en mesure de sauvegarder les 64 zettaoctets de données générées en 2020.

**La démocratisation croissante des objets connectés et le développement de la 5G constituent les principaux moteurs de ce "Big Bang" de la donnée.**

## Hello veille ...

---

Friand d'apprendre de nouvelles choses, vous souhaitez approfondir vos **connaissances sur la donnée**, matière première des métiers liés à l'intelligence artificielle.

Vous documentez donc et réalisez une veille complète sur les éléments suivants :

- A. 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**. Illustriez 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.

J. En plus des recherches et exercices demandés, vous devez accompagner chaque réponse ou livrable d'une explication claire et imagée destinée à un **public non initié**.

- Vous devez utiliser des métaphores simples, des exemples du quotidien et des schémas visuels pour expliquer chaque notion (ex. comparer une base de données à une bibliothèque, une clé primaire à un numéro de carte d'identité, etc.).
- Les termes techniques doivent être définis dans un lexique simplifié en fin de document, afin de rendre le contenu accessible à toute personne ne disposant pas de connaissances préalables en informatique ou en bases de données.



- Cette approche vise à rendre la donnée, les systèmes de gestion et le SQL compréhensibles même pour un lecteur qui n'a jamais manipulé ce type de concepts

## Hello SQL...

---

### Job 1

Soit la table de données **world** contenant des **informations sur différents pays du monde**, avec des détails tels que la région, la population, la superficie en miles carrés, la densité de la population, le taux de mortalité infantile, le PIB par habitant ainsi que d'autres informations pertinentes. Elle peut être récupérée [ici](#).



Considérons la requête suivante permettant d'afficher la population de **“France”** depuis la table **world** :

```
● ● ●  
SELECT population FROM world  
WHERE name = 'France'
```

1. Modifiez la requête ci-dessus afin d'**afficher la population de “Germany”**.
2. Modifiez la requête ci-dessus afin d'afficher le nom et la population des pays **“Sweden”**, **“Norway”** et **“Denmark”**.



3. Créez une requête permettant d'**afficher les pays dont la superficie est supérieure à 200 000 mais inférieure à 300 000.**

## Job 2

Considérons la table **world** du job précédent :

1. Créez une requête permettant de trouver les noms de pays **commençant par la lettre B.**
2. Créez une requête permettant de trouver les noms de pays **commençant par "Al".**
3. Créez une requête permettant de trouver les noms de pays **finissant par la lettre y.**
4. Créez une requête permettant de trouver les noms de pays **finissant par "land".**
5. Créez une requête permettant de trouver les noms de pays **contenant la lettre "w".**
6. Créez une requête permettant de trouver les noms de pays **contenant "oo" ou "ee".**
7. Créez une requête permettant de trouver les noms de pays **contenant au moins trois fois la lettre a.**
8. Créez une requête permettant de trouver les noms de pays **ayant la lettre r comme seconde lettre.**



## Job 3

Soit la table **students** définie comme suit :

student_id	first_name	last_name	age	grade
1	Alice	Johnson	22	A+
2	Bob	Smith	20	B
3	Charlie	Williams	21	C
4	David	Brown	23	B+
5	Eva	Davis	19	A
6	Frank	Jones	22	C+

1. Créez une requête permettant d'afficher **toutes les colonnes de la table students.**
2. Créez une requête permettant de filtrer la table et d'**afficher les élèves âgés de strictement plus de 20 ans.**
3. Créez une requête permettant de faire **un classement des élèves selon leur note dans un ordre croissant, puis dans un ordre décroissant.**

## Job 4

Soit la table **nobel** définie comme suit :

yr	subject	winner
1960	Chemistry	Willard F. Libby
1960	Literature	Saint-John Perse
1960	Medicine	Sir Frank Macfarlane Burnet
1960	Medicine	Peter Madawar
...		



1. Créez une requête permettant d'**afficher les prix nobels de 1986**.
2. Créez une requête permettant d'**afficher les prix nobels de littérature de 1967**.
3. Créez une requête permettant d'**afficher l'année et le sujet du prix nobel d'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**.
5. Créez une requête permettant d'**afficher les détails des lauréats du prix de Mathématiques**. Combien y en a-t-il ?

## Job 5

Considérons la table **world** précédente :

1. Créez une requête permettant d'**afficher les pays dont la population est supérieure à celle de "Russia"**.
2. Créez une requête permettant d'**afficher les pays d'Europe dont le PIB par habitant est supérieur à celui d' "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**.
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. Exemple :

name	pourcentage
Albania	3%
Andorra	0%
Austria	11%
...	

5. Créez une requête permettant de **trouver le plus grand pays de chaque continent, en indiquant son continent, son nom et sa superficie.**
6. Créez une requête permettant de **trouver les continents où tous les pays ont une population inférieure ou égale à 25 000 000.**

## Job 6

Considérons une nouvelle fois la table **world** précédente :

1. Créez une requête permettant d'**afficher la population totale du monde.**
2. Créez une requête permettant d'**afficher la population totale de chacun des continents.**
3. Créez une requête permettant d'**afficher le PIB total du continent de chacun des continents.**
4. Créez une requête permettant d'**afficher le PIB total du continent africain.**



5. Créez une requête permettant d'**afficher le nombre de pays ayant une superficie supérieure ou égale à 1 000 000m<sup>2</sup>**.
6. Créez une requête permettant d'**afficher la population totale des pays suivants : Estonia, Latvia, Lithuania.**
7. Créez une requête permettant d'**afficher le nombre de pays de chaque continent.**
8. Créez une requête permettant d'**afficher les continents ayant une population totale d'au moins 100 millions d'individus.**

## Job 7

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

**Game**

<b>id</b>	<b>mdate</b>	<b>stadium</b>	<b>team1</b>	<b>team2</b>
1001	8 June 2012	National Stadium, Warsaw	POL	GRE
1002	8 June 2012	Stadion Miejski (Wroclaw)	RUS	CZE
1003	12 June 2012	Stadion Miejski (Wroclaw)	GRE	CZE
1004	12 June 2012	National Stadium, Warsaw	POL	RUS
...				

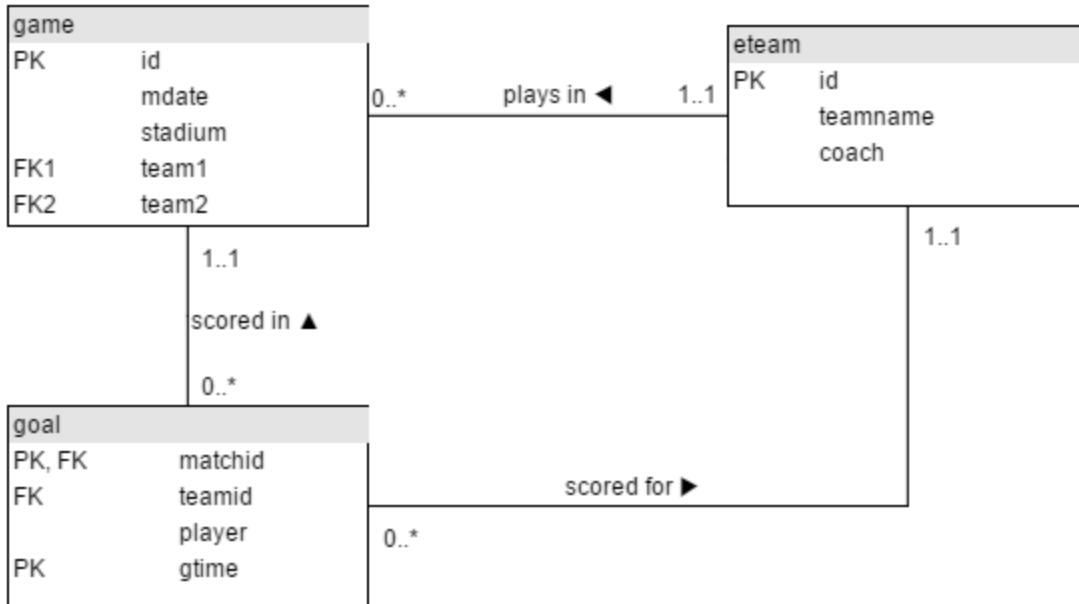
**Goal**



matchid	teamid	player	gtime
1001	POL	Robert Lewandowski	17
1001	GRE	Dimitris Salpingidis	51
1002	RUS	Alan Dzagoev	15
1002	RUS	Roman Pavlyuchenko	82
...			

### Eteam

id	teamname	coach
POL	Poland	Franciszek Smuda
RUS	Russia	Dick Advocaat
CZE	Czech Republic	Michal Bilek
GRE	Greece	Fernando Santos
...		



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.**
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 * FROM goal  
WHERE player LIKE '%Bender'
```

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



```
SELECT *  
FROM game JOIN goal ON (id=matchid)
```

5. Créez une requête permettant d'**afficher team1, team2 et player pour chaque but marqué par un joueur appelé Mario**.
6. Créez une requête permettant de **joindre la table goal et la table eteam sur les clés id - teamid**.
7. Créez une requête permettant d'**afficher player, teamid, coach, gtime pour tous les buts marqués dans les 10 premières minutes des matchs**.
8. La requête suivante permet de joindre la table **game** et la table **eteam** sur les clés **team1 - 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 *
FROM game JOIN eteam ON (team1=eteam.id)
```

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"**.

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"**.

## Job 8

Soient les tables **Employees** et **Departments** constituants la base de données **SomeCompany** définies comme suit :

1. **Employees:** employee\_id (INT, PK), first\_name (VARCHAR), last\_name (VARCHAR), birthdate (DATE), position (VARCHAR), department\_id (INT, FK).



2. **Departments:** department\_id (INT, PK), department\_name (VARCHAR), department\_head (INT, FK), location (VARCHAR).
3. **Projects:** project\_id (INT, PK), project\_name (VARCHAR), start\_date(DATE), end\_date(DATE), department\_id (INT, FK).

**Employees**

employee_id	first_name	last_name	birthdate	position	department_id
1	John	Doe	1990-05-15	Software Engineer	1
2	Jane	Smith	1985-08-20	Project Manager	2
3	Mike	Johnson	1992-03-10	Data Analyst	1
4	Emily	Brown	1988-12-03	UX Designer	1
5	Alex	Williams	1995-06-28	Software Developer	1
6	Sarah	Miller	1987-09-18	HR Specialist	3
7	Ethan	Clark	1991-02-14	Database Administrator	1
8	Olivia	Garcia	1984-07-22	Marketing Manager	2
9	Emilia	Clark	1986-01-12	HR Manager	3
10	Daniel	Taylor	1993-11-05	Systems Analyst	1
11	William	Lee	1994-08-15	Software Engineer	1



12	Sophia	Baker	1990-06-2 5	IT Manager	2
----	--------	-------	----------------	------------	---

### Departments

department_id	department_name	department_head	location
1	IT	11	Headquarters
2	Project Management	2	Branch Office West
3	Human Resources	6	Branch Office East

1. Créez la base de données **SomeCompany** à l'aide d'une requête, ajoutez une **condition sur l'existence** de SomeCompany.
2. Créez la **table Employees**.
3. Créez la **table Departments**.
4. Insérez 6 à 9 nouveaux employés dans la table **Employees**.
5. Récupérez le nom et le poste de tous les employés.
6. Mettez à jour le poste d'un employé dans la table **Employees**.
7. Supprimez un employé de la table **Employees**.
8. Affichez le nom, le département et le bureau de chaque employé.
9. Affichez, à l'aide d'un filtre, les membres de l'équipe IT, puis le management, puis les ressources humaines.



10. Affichez **les départements de SomeCompany** dans l'ordre alphabétique, avec les managers respectifs de chaque département.

11. Ajoutez **un nouveau département** à la table **Department** (Marketing peut-être?), **ajoutez ou mettez à jour les employés** de ce nouveau département.

12. Créez **une nouvelle table Project** : project\_id (INT, PK), project\_name (VARCHAR), start\_date (DATE), end\_date (DATE), departement\_id (INT, FK). **Ajoutez des observations à cette nouvelle table, analysez la productivité des départements en IT et du nouveau département** créé précédemment.

## Job 9

Considérons une dernière fois la table **world**. Il existe plusieurs colonnes de cette table que nous n'avons pas pu analyser. **Étudiez au moins 6 autres variables de world**, à l'aide de différentes fonctions et commandes SQL, afin d'**obtenir des insights pertinents (Literacy, Net migration, Birthrate, Deathrate, Infant mortality, Arable, Crops, ...)**.

## Big job : Calculateur d'Empreinte Carbone

L'empreinte carbone (ou le contenu carbone) d'une **activité humaine** est une mesure des **émissions de gaz à effet de serre** d'origine anthropique, c'est-à-dire qui peuvent lui être imputées.

Soucieux de l'environnement et de notre chère planète, vous avez eu l'idée d'un **outil pour**





**comprendre cette empreinte** afin de la diminuer et **minimiser son impact**.

Vous vous lancez dans le développement d'un **calculateur d'empreinte carbone**, visant à aider à l'évaluation et à la compensation de l'empreinte carbone, en particulier dans le contexte de la production d'énergie électrique.

1. Vous récupérez les données d'intérêt [ici](#). Vous avez à votre disposition un dataset composé de deux tables, **Country** et **World**. Elles **recensent le pourcentage d'utilisation de différentes sources d'énergie** (charbon, gaz, pétrole, nucléaire, ...) **en 2015 pour la production d'électricité** par pays dans la table *Country*, puis par région du monde dans la table *World*.
2. Créez la base de données **CarbonFootprint**, puis les tables **Country** et **World**.
3. Utilisez des requêtes SQL afin d'**analyser les données recueillies** et tirez un maximum d'informations sur les émissions en carbone. **Qu'est ce que vous observez ?** Notez ces observations pour la suite.
4. Créez une application Flask où vous présenterez le **contexte de ce mini projet et les observations faites précédemment**. Pensez à afficher un aperçu de votre jeu de données **CarbonFootprint**.
5. Le tableau suivant montre **les émissions de CO2 de différentes sources de production d'électricité** d'après une étude réalisée par le Groupe d'experts intergouvernemental sur l'évolution du climat datée de 2014. Par exemple, pour l'électricité produite à partir du charbon, les émissions de CO2 par kilowattheure varient de 740 grammes (au



minimum) à 910 grammes (au maximum), avec une médiane de 820 grammes. Calculez le **pourcentage de contribution des différentes sources du tableau aux émissions totales de CO<sub>2</sub>** lors de la production d'électricité **pour tous les pays de Country**.

**Indice : Contribution du charbon aux émissions totales de CO<sub>2</sub> d'un pays = Pourcentage d'utilisation du charbon du pays x Emission de gCO<sub>2</sub> par kWh du charbon.**

Source	Min de gCO <sub>2</sub> /kWh	Médiane de gCO <sub>2</sub> /kWh	Max de gCO <sub>2</sub> /kWh
Charbon	740	820	910
Gaze naturel	410	490	650
Pétrole	620	740	890
Hydro	1	24	2200
Renouvelable (Solaire)	26	41	60
Nucléaire	3.7	12	110

6. Modifiez votre application Flask afin de pouvoir **filtrer vos données selon un pays (ou une région du monde) sélectionnable depuis une selection box**.
7. Créez un tableau montrant, pour chaque pays sélectionné depuis la **selection box** créée précédemment, **le pourcentage d'utilisation de différentes ressources, l'émission médiane en gCO<sub>2</sub>kWh de ces ressources et la contribution spécifique de ses ressources aux émissions totales de CO<sub>2</sub>** lors de la production d'électricité.



Par exemple, si l'on sélectionne l'Albanie depuis la selection box, le tableau doit afficher les éléments suivants :

Source de production	% d'utilisation	Médiane de gCO2/kWh	Contribution en émission gCO2/kWh
Charbon	0	820	0% x 820 = 0
Gaze Naturel	0	490	0% x 490 = 0
Pétrole	0	740	0% x 740 = 0
Hydro	100	24	100% x 24 = 24
Renouvelable	0	41	0 % x 41 = 0
Nucléaire	0	12	0 % x 12 = 0

8. Calculez et affichez l'**émission totale des différentes sources** d'un pays sélectionné : **émissions totales = émission de charbon + émission de gaze + ... + émission de nucléaire.** Par exemple : **0 + 0 + 0 + 24 +... + 0 = 24 gCO2/kWh.**

9. Calculez et affichez l'**émission totale annuelle pour un pays** (toujours depuis une selection box). On définit la formule permettant de calculer cette valeur comme suit :

**Émissions annuelles totales de CO2 = Émissions totales en kgCO2/kWh x nombre d'heures dans une année x consommation électrique, où la consommation électrique en kw doit être spécifiée par l'utilisateur.**

Afin d'y voir pour clair, faisons ce calcul pour l'Albanie, on a :



1. Émissions totales de l'Albanie en kgCO<sub>2</sub>/kWh : **0,024 (kgCO<sub>2</sub>/kwh)**.
2. Nombre d'heures dans une année : **24 x 365.**
3. Puissance électrique consommée de manière continue par l'Albanie (donnée choisie par l'utilisateur) : **1 (kw).**

On calcule donc à l'aide de **la formule précédente**, les émissions en CO<sub>2</sub> de l'Albanie durant une année pour la production d'électricité consommée de 1 (kw) par heure, par jour : **0,024 x 24 x 365 x 1 = 210,24 (kg of CO<sub>2</sub>).**

10. Pour finir, en sachant qu'**un arbre absorbe environ 25 kg de CO<sub>2</sub> par an**, affichez le **nombre nécessaire d'arbres à planter afin d'absorber le CO<sub>2</sub> engendré par un pays** (sélectionné depuis la selection box, et oui) durant la production d'électricité.





## Compétences visées

---

- Systèmes de gestion de base de données
- SQL
- Flask
- Analyse exploratoire de données
- Concevoir un processus de collecte de données sécurisé et conforme rgpd
- Extraire des données via scripts personnalisés (python, sql)
- Créer un système automatisé de collecte de données (etl, python, sql)

## Rendu

---

L'évaluation de ce projet se fera sur deux aspects :

1. Une présentation explicative de votre travail sous forme de diapositives.
2. Un repository github public nommé **hello-dbms**, contenant les éléments suivants :
  - a. **La veille scientifique** réalisée dans votre fichier **README** (en plus du contexte du projet, comme à votre habitude).
  - b. Les **scripts SQL** des différents jobs de la section Hello SQL. Faites en sorte de nommer chaque script selon le job (job1.sql, job2.sql,...).



- c. Le **script Python de l'application Flask du big job** ainsi que les **scripts HTML et CSS** (l'application doit être un minimum plaisante à regarder).

## Base de connaissances

---

- [Volume of data/information created, captured, copied, and consumed worldwide from 2010 to 2020, with forecasts from 2021 to 2025](#)
- [Qu'est-ce qu'un système de gestion de base de données](#)
- [MySQL – The Basics // Learn SQL in 23 Easy Steps](#)
- [SQL – Tout savoir sur le langage de programmation des bases de données](#)
- [SQL.sh – Apprendre le SQL](#)
- [Practice SQL](#)
- [SQL Cheatsheet](#)
- [NoSQL : Tout comprendre sur les bases de données non relationnelles](#)
- [7 Database Paradigms](#)