R3.07 SQL dans le langage PL/SQL (0/2)
TP 1 révisions requêtes

I. Cadre de travail

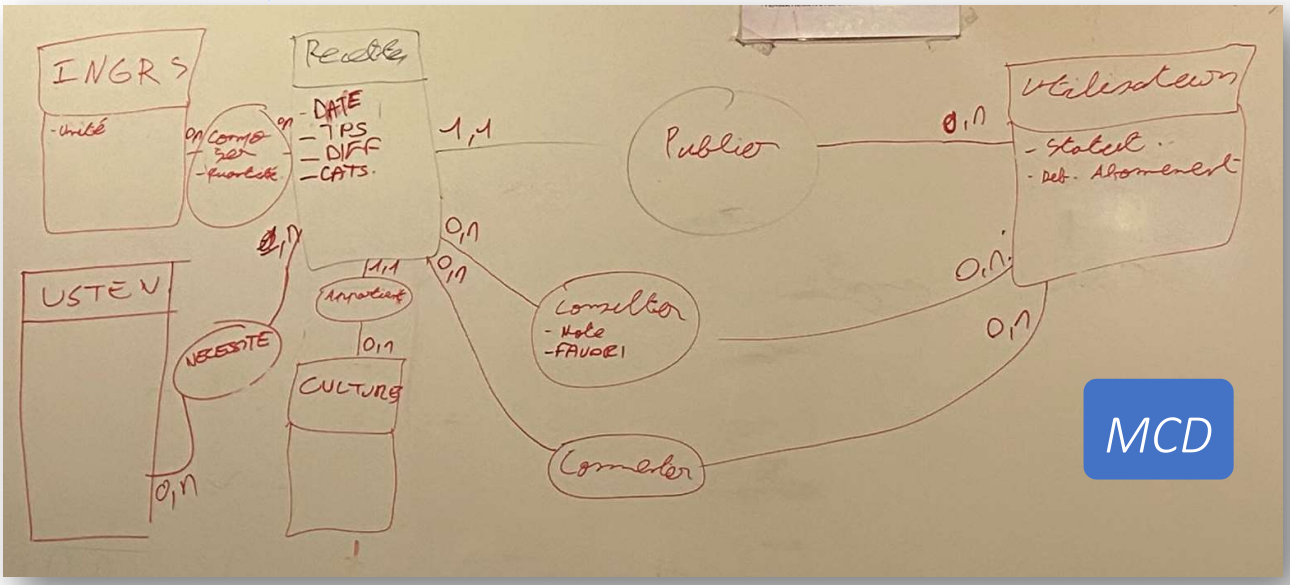
Vous travaillerez sur http://paris.uca.local pour faciliter le partage.

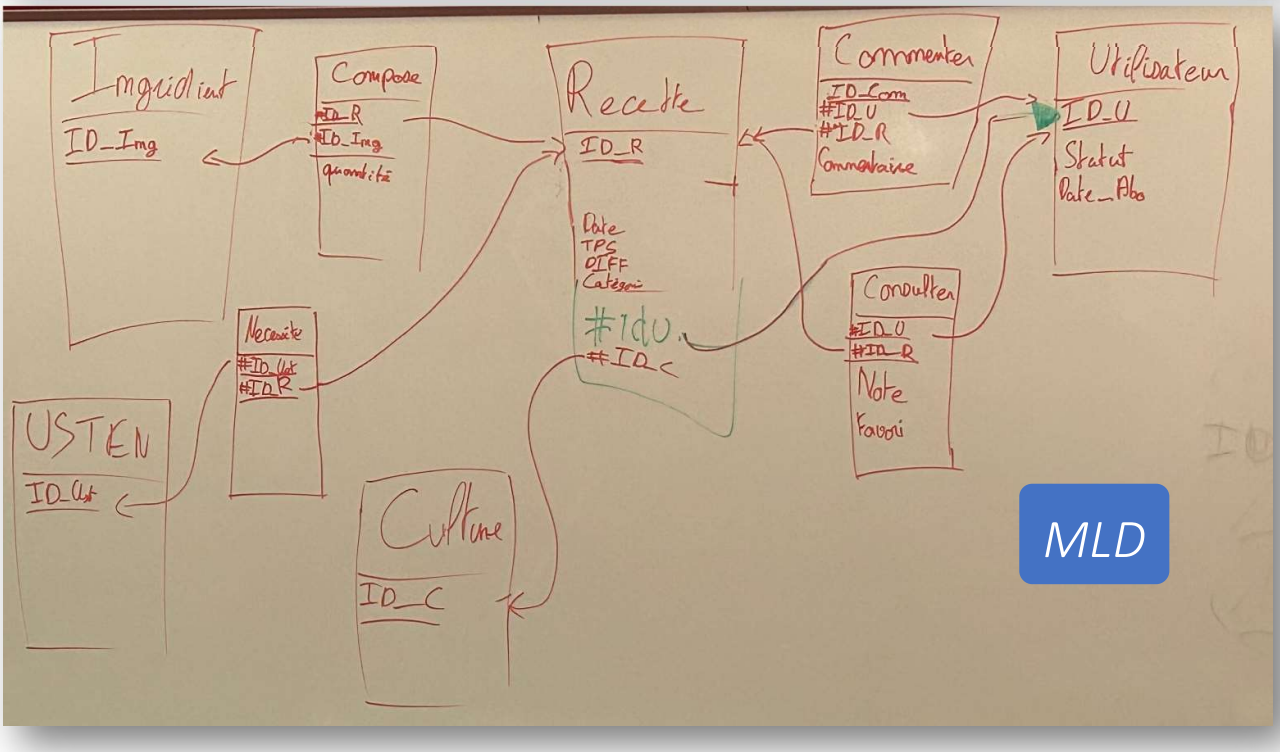
L'étude de cas concerne un site internet qui propose à ses abonnés de poster ou consulter des recettes de cuisine. La conception est déjà réalisée : les MCD, MLD et scripts de création de la base sont fournis. Il faudra probablement la compléter par quelques exemples pour tester certains exercices.

La base de données complétée sera utilisée pendant plusieurs séances. Conservez votre travail ainsi que les requêtes/ réponses aux questions.

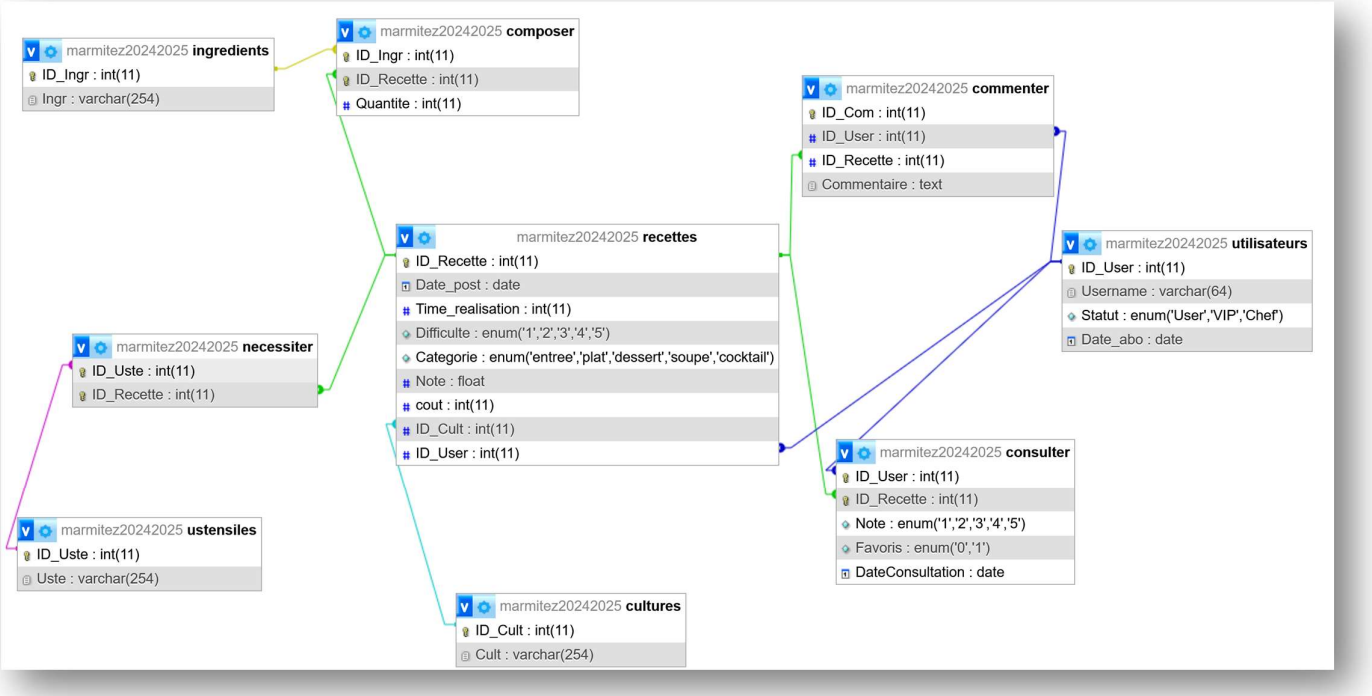
II. Rappel : Règles de gestion et contraintes d'intégrités

A. Conception de la base de données : Marmitez !





MPD



Vous tiendrez compte de ce MPD, plus détaillé et complet que les MLD/MCD et correspondant aux scripts de création de base de données fourni.

B. Critique de cette conception

Que dire des contraintes des champs `difficultés` et `catégories` ?
Quelle était la solution plus souple ? Pourquoi cette solution est plus lourde ?

Que dire de la contrainte sur le `coût` ?
Pour trouver dans PHPMyAdmin la contrainte de type Check, exécuter : `SHOW CREATE TABLE recettes;`
Quelle étaient les 2 alternatives ?

On ne peut pas garder un historique des consultations d'un recette pour un utilisateur donné !
Pourquoi ?
Que faudrait-il apporter comme modifications pour pouvoir le faire ?
Dans la solution actuelle, quel est le sens du champ `DateConsultation` ?

Un utilisateur ne devrait pas pouvoir commenter une recette dont il est l'auteur ; Est-ce toutefois possible ?
Si non, pourquoi? si oui , comment modifier pour ajouter cette contrainte ?

Un utilisateur devrait pouvoir apporter plusieurs commentaires à une même recette; Est-ce possible ?
Si oui comment ? si non , comment modifier pour ajouter cette contrainte ?

Un utilisateur ne devrait pouvoir apporter un commentaire qu'aux recettes qu'il a déjà consultées ; Est-ce vérifié ?
Si oui comment ? si non , comment modifier pour ajouter cette contrainte ?

III. Requêtes d'interrogation

A. Requêtes simples.

Quelle est la recette la plus consultée ? *Utiliser* `GroupBy` *et* `Limit`

La durée d'un abonnement premium('VIP') est d'1 an.
Quels sont les utilisateurs à relancer, i.e. ceux dont l'abonnement va expirer d'ici 1 mois. *Utiliser* `DateDiff`

Pour chaque ustensile, afficher le nombre de recettes où il est utilisé. *Utiliser* `GroupBy` *et* `Count`

Quel est le nombre moyen d'ustensiles utilisés par les recettes ? *Calculer le nombre d'ustensiles utilisés / par le nombre de recette différentes.*

Quel est le nombre moyen d'ustensiles utilisés par les recettes des chefs ? *Même chose mais juste pour les recettes de Chef*

Quelle est la catégorie de recettes la plus consultée. `Count, GroupBy, OrderBy et Limit`

Quel est le pourcentage d'utilisateurs premium('VIP') , soit le nombre de premium / par le nombre total d'utilisateurs, tous confondus. *petite astuce :* `sum(utilisateurs.Statut="VIP")`

B. Requêtes avancées.

Donner le nom et le créateur des 10 recettes les plus consultées. *Une requête principale retourne le nom et créateurs en utilisant une sous-requête qui retourne les 10 recettes les plus consultées)*

Pour une recette (au choix), donner les ingrédients nécessaires pour 2,4 et 10 personnes. La quantité par défaut est fixée pour une personne.

Proposer la requête donnant les 10 recettes les plus consultées le dernier mois.

C. Vues

Ajouter la vue CoutMoyenRecette qui donne le coût moyen des recettes

```
DROP  VIEW  IF EXISTS CoutMoyenRecettes ;
CREATE VIEW CoutMoyenRecettes AS
      SELECT AVG(R.cout) AS CMrecettes FROM    Recettes R  ;
SELECT * FROM coutMoyenRecettes;
```

Créer une vue RecettesLesMoinsCheres qui donne les recettes inférieures au coût moyen des recettes.*(utiliser la vue précédente)*

D. Requêtes avancées & vues

Créer une requête qui donne la catégorie principale des recettes les moins chères consultées.
Pas si difficile une fois que les vues ci-dessus ont été créées :)

On veut fidéliser l'utilisateur en l'assistant ! On souhaite donner pour un utilisateur donné (e.g. 6) sa prochaine recette idéale, celle qui ressemble le plus à ses favoris au regard de l'information `catégorie` … ***prochaine*** donc qu'il n'a pas encore consultée …. De la catégorie la plus présente parmi ses recettes favorites .

Avancer par étape :

- ⇨ Commencer par faire une vue* qui retourne la catégorie la plus citée dans les recettes favorites de l'utilisateur.
- ⇨ Proposer ensuite une requête qui retourne les recettes qu'un utilisateur n'a jamais consulté
- ⇨ Proposer une requête globale

* certaines limitations de Mysql nous imposent de passer par une vue (pas de limit dans une sous requête)

E. Lever une contrainte

Dans des cas particuliers, on souhaite lever une contrainte. Il faut la supprimer ! Pour la réactiver, il faudra la recréer.
Syntaxe et test pour lever une contrainte. Nous allons lever temporairement la contrainte de vérification du coût qui était : `CONSTRAINT VerifierCoutRecette CHECK (cout  BETWEEN 1 and 10);`

⇨ `ALTER TABLE Recettes DROP CONSTRAINT VerifierCoutRecette;`

Il vous est maintenant possible de définir librement le coût d'une recette (e.g. un coût de 11)
`UPDATE Recettes R SET cout=11 WHERE R.ID_Recette=1 ;`

Pour réactiver la contrainte, il faut la recréer :
`ALTER TABLE Recettes ADD CONSTRAINT VerifierCoutRecette CHECK (cout BETWEEN 1 and 10);`

Qu'observez-vous ? corrigez.

Nous allons maintenant définir une contrainte supplémentaire sur le coût : définir une valeur par défaut à 0
`ALTER TABLE Recettes ADD CONSTRAINT CoutRecetteParDefaut DEFAULT 0;`

Essayer d'ajouter une recette sans en préciser le coût.
Essayer d'ajouter une recette en précisant un coût de 11.

Remarques :

- Pour lever (ou réactiver) toutes les contraintes de clé étrangère : `SET FOREIGN_KEY_CHECKS= 0 (ou 1) ;`
- On différence les *contraintes de table* qui vont comparer entre eux différentes champs de la table et des *contraintes de champ* dont la portée des test est restreinte à un seul champ.

F. Optimisation et index

1. A quoi ça sert ?

A accélérer les requêtes !
Comment ça fonctionne ? Sans index chaque ligne d'une requête est testée pour savoir si elle doit être retenue (condition émise sur un champ de la ligne). Si un index est défini sur ce champ, les données sont prétriées et les valeurs autorisées ou recherchées sont plus rapidement retrouvées.

2. Quand les utiliser les index ?

Sur des tables de grande taille
Sur des champs qui font très fréquemment l'objet de sélection (clause WHER) ou de tri dans les requêtes

3. Syntaxe

```
CREATE TABLE test (
    id         INT NOT NULL,
    last_name  CHAR(30) NOT NULL,
    first_name CHAR(30) NOT NULL,
    PRIMARY KEY (id),
    INDEX name (last_name,first_name)
);
```

```
CREATE INDEX idx_lastname
ON Persons (LastName);

CREATE INDEX idx_pname
ON Persons (LastName, FirstName);

ALTER TABLE table_name
DROP INDEX index_name;
```

4. Des index créés automatiquement

Lors d'une création de contrainte de `Primary Key` ou `Unique`, un index est automatiquement créé !

5. Quel index ajouter judicieusement dans notre base Marmitez! ?

⇨ Quel champ devrait faire l'objet de beaucoup de test dans de nombreuses requêtes. ?

IV. Je vivrais bien en Théorie ….

… parce qu' "*en Théorie tout va bien* »!

Mais nous évoluons dans un monde imparfait ... avec des utilisateurs maladroits et plus rarement d'autres malveillants. Le rôle de l'informaticien est aussi de sécuriser au maximum ses programmes et ses données avant même d'optimiser.
Si vous avez en charge le stockage des données (administrateur de base de données) , et en particulier leur intégrité, il peut être délicat de s'appuyer sur la phase applicative (les traitement, l'interface de saisie ou modification des données) . Il est même possible que vous n'ayez pas à développer vous-même cette partie. Il faut donc sécuriser les données sans a priori trop optimiste sur la suite applicative qui va les manipuler.

Essayer de saisir les requêtes suivantes pour enregistrer des faits qui ne devraient jamais arriver ... en théorie
Si elles passent, elles vont préparer nos prochains TP (fonctions, procédures , triggers et events) qui consolideront encore d'avantage l'intégrité des données.

Essayer d'ajouter un commentaire sur une recette par un utilisateur qui n'a pas consulté cette recette
Essayer la consultation d'une recette d'un chef par un utilisateur qui n'a pas l'accès premium (‘VIP’)
Essayer de modifier le statut en premium sans mettre à jour la date
Essayer de changer le statut d'un chef en utilisateur standard.
Essayer de fixer un statut prémium sans date d'abonnement ou avec une date postérieur à la date du jour.