GMSI 15 **CORRECTION**
SGBD(r) 2016 **Quizz 2**
Time Limit: 30 Minutes **17/06/2016**

Ce quizz contient 3 pages et 10 questions, pour un total de 30 points.

Chambres				
id	Etage	Num	Taille	Prix
1	1	01	22.00	23.50
2	1	02	25.00	24.00
3	2	02	32.00	75.00
4	2	01	18.00	19.99

ListeChambres		
id	Nom	Taille
1	101	22.00
2	102	25.00
4	201	18.00
3	202	32.00

Clients			
id	idCh	Nom	Date
1	1	Paul	Lu
2	2	Marie	Lu
3	2	Marie	Ma
4	4	Eric	Lu
5	1	Paul	Me

1. (3 points) Quelle est la différence entre un SGBD et une base de donnée par fichier ? Dans quel cas privéligier l'un ou l'autre ?

Solution: SGBD: programme qui va ajouter une couche supplémentaire à la gestion de la BDD. Cela permet d'avoir une gestion simultanée multi-clients, sécurisé avec un langage dédié à sa gestion et à la consultation/ modification des données.
Fichiers:
+ Pas de compléxité dans les données, ni de récurrence. Ex: fichier de configuration
+ Pas de modification régulière des données, ni multi-ressouces/utilisateurs.

2. (4 points) Quelles sont les propriétés d'un SGBD (installée et configurée sur une machine A) ?
   - √ **Plusieurs personnes peuvent lire des données en même temps**
   - √ **Un utilisateur sur une machine B peut, avoir le droit d'extraire des données**
   - √ **Un utilisateur sur une machine B peut, ne pas avoir le droit d'écrire dans la BDD**
   - √ **Le SGBD va modifier la requête SQL qui lui est envoyée**

3. Donner pour la table **Chambres**:
   (a) (2 points) La valeur de $\Pi_{Chambres.Etage}$. Comment se nomme ce genre de valeur ?
   (b) (2 points) Les types des attributs Etage, Num et Prix
   (c) (1 point) Un 4-uplet (d'une des tables)

Solution: Un domaine.
$\Pi_{Chambres.Etage} = \{1, 2\}$

Solution:
Etage: INTEGER(> 1)
Num: VARCHAR(2)
Prix: DECIMAL(> 4, 2)
$* > x$: *au moins supérieur à x.*

Solution:
$< 1, 01, 22.00, 23.50 >$
$< 2, 2, Marie, Lu >$
$< 4, 2, 18.00, 19.99 >$

4. (2 points) Donner la reqûete SQL qui permet de constuire la table **ListeChambres**

Solution:

```
SELECT id, CONCAT(Etage, Num) as nom, taille FROM Chambres
```

5. (2 points) Donner la table résultant de la division des **Chambres** par "SELECT id, taille FROM **ListeChambres** WHERE taille BETWEEN 20 AND 27"

Solution:

Etage	Num	Prix
1	101	22.00
2	102	25.00

6. (3 points) Donner la reqûete SQL qui permet d'obtenir une vue contenant la liste des chambres et le nom du client qui l'occupe le Lundi. Ne pas utiliser la 'vue' déjà générée. Le résultat doit être trié par etage et numéro de chambre (ordre croissant). (+1 point bonus)

Solution:

```
SELECT ch.id, ch.etage, ch.num, cl.nom, cl.‘date‘
    FROM Chambres as ch
        LEFT JOIN Clients ON ch.id = cl.idCh
    WHERE cl.‘Date‘ = "Lu"
    ORDER BY ch.etage ASC, ch.num ASC
```

7. (2 points) Qu'est ce qu'une clé primaire ? Definition. En citer une, extraite des trois tables.

Solution:
Une clé primaire est une contrainte d'unicité qui permet d'identifier de manière unique un enregistrement dans une table.
Une clé primaire peut être composée d'un ou de plusieurs champs de la table. Deux lignes distinctes de la table ne peuvent pas avoir les mêmes valeurs pour les champs définis au niveau de la clé primaire. Il est possible de définir pour une même table plusieurs contraintes d'unicité, mais au plus une seule clé primaire. La clé primaire ne peut pas être NULL.
Exemple: Chambres.id, ListeChambres.id, Clients.id

8. (2 points) Qu'est ce qu'une clé étrangère ? Definition. En citer une, extraite des trois tables.

Solution: Une clé étrangère, dans une base de données relationnelle, est une contrainte qui garantit l'intégrité référentielle entre deux tables.

Une clé étrangère identifie une colonne ou un ensemble de colonnes d'une table comme référençant une colonne ou un ensemble de colonnes d'une autre table (la table référencée). Les colonnes de la table référencée doivent faire partie d'une contrainte de clé primaire ou d'une contrainte d'unicité. La contrainte de clé étrangère garantit que les valeurs de chaque ligne de la table référençant existent dans la table référencée : ainsi une ligne de la table référençant ne peut pas contenir un ensemble de valeurs qui n'existe pas dans la table référencée.

Une contrainte de clé étrangère permet ainsi d'établir des liens entre plusieurs tables : il s'agit d'un des principes fondamentaux des bases de données relationnelles.
Exemple: Clients.idCh

9. (3 points) Donner la reqûete SQL qui permet de créer la table **Clients**

Solution:

```
CREATE TABLE IF NOT EXISTS ‘Clients‘ (
  ‘id‘ int(11) NOT NULL AUTO_INCREMENT,
  ‘idCh‘ int(11),
  ‘Nom‘ varchar(255) NOT NULL,
```

```
  `Date` date NOT NULL,
  PRIMARY KEY (id),
  FOREIGN KEY (idCh) REFERENCES Chambres(id) -- for MySQL
);


-- Possibilité d'ajouter la contrainte avec (à la place du précédent FOREIGN ...):
ALTER TABLE Clients
  ADD CONSTRAINT `clients_chambres_ibfk_1`
    FOREIGN KEY (idCh) REFERENCES Chambres (id)
        ON DELETE NO ACTION
        ON UPDATE NO ACTION;
```

10. (4 points) Quelles écritures sont vraies ? Si c'est possible, corriger. (+2 points bonus si vous décrivez les comportements de chaque ligne)

    ☐ `SELECT * FROM Chambres WHERE type = 'Suite Royale'`

    √ `SELECT * FROM Chambres LIMIT 1, 3`

    ☐ `UPDATE Clients SET nom = Eric WHERE id = 3`

    ☐ `UPDATE Chambres SET prix = prix+1 AND taille = 22.5f WHERE id = 2`

    ☐ `DROP TABLE Clients.Nom`

    ☐ `SELECT nom FROM Clients /* c */ WHERE date = "Lu"`

    √ `SELECT c.taille FROM Chambres c ORDER BY c.taille`

    √ `ALTER TABLE Chambres MODIFY( Num CONSTRAINT TailleNum CHECK(DATALENGTH(Num) = 3))`

Solution:

1. Le champs 'type' n'existe pas
2. OK
3. "Eric" est un litéral (texte)

   ```
   UPDATE Clients SET nom = 'Eric' WHERE id = 3
   ```

4. On sépare les valeurs du update par des virgules

   ```
   UPDATE Chambres SET prix = prix+1, taille = 22.5f WHERE id = 2
   ```

5. Clients.Nom n'est pas une table ! Soit on supprime la table, soit une colonne

   ```
   DROP TABLE Clients -- Pour supprimer la table

   ALTER TABLE Clients DROP nom -- Pour supprimer la colonne
   ```

6. date est un mot réservé (fonction et type date). Il faut le transformé en litéral texte.

   ```
   SELECT nom FROM Clients /* c */ WHERE `date` = "Lu"
   ```

7. OK
8. OK