Réservations d'hôtel

5.1) Soit la Base de données 'hôtel' qui contient 3 Tables " Chambre", "Client" et "Réservation" qui sont définis avec le schéma relationnel suivant:

Chambre (**Num_Chambre**, Prix, Nbr_Lit, Nbr_Pers, Confort, Equ)

Client (**Num_Client**, Nom, Prenom, Adresse)

Réservation (**Num_Client#, Num_Chambre#**, Date_Arr, Date_Dep)

A. B. et C. **voir script solution 'script correction laboratoire 05'**

Locations des films

5.2)

Soit la base de données 'Films' qui contient 3 tables 'Clients', 'Films' et 'Locations' qui sont définis comme suit :

Table Clients (codecli, prenomcli, nomcli, ruecli, cpcli, villecli)
Clé primaire : codecli

Table Films (codefilm, nomfilm)
Clé primaire : codefilm

Table Locations (codecli, codefilm, datedebut, duree)
Clé primaire : codecli, codefilm
Clé étrangère : codefilm de la table Films, codecli de la table Clients

A. Saisissez la requête qui permettra d'insérer le film n°12 "The Raid"

```
insert into Films values(12,'The Raid');
```

B. Saisissez la requête qui permettra d'insérer le film n°13 "Le loup de Wall Street".

```
insert into Films values(13,'Le loup de Wall Street');
```

C. Saisissez la requête qui permettra d'insérer le client n°124 qui s'appelle "Jean" "Talu" (les autres informations sur ce client ne sont pas connues).

```
insert into Clients values(124,'Jean','Talu',null,null,null);
-- ou bien
insert into Clients(codecli,prenomcli,nomcli) values(124,'Jean','Talu',null,null,null);
```

D. Quel est l'effet de cette requête ? :

```
DELETE
FROM Locations;
```

Dans MySQL, cette requête ne donnera pas le même résultat à obtenir. Pour avoir le même résultat il faut ajouter la clause 'Where codecli and codefilm'

E. Dans quelle table la requête a-t-elle supprimé une ligne ?

```
DELETE
FROM Films WHERE codefilm=11;
```

Films.

F. Supprimez toutes les lignes de la table clients.

```
delete from Locations where codecli and codefilm;
delete from Clients where codecli;
```

G. Supprimez de la table FILMS les films 1, 2, et 3.

```
delete from Films where codefilm in (1,2,3);
-- ou bien
delete from Films where codefilm = 1 or codefilm = 2 or codefilm = 3;
```

Départements/Employés

5.3) Soit la base de données suivante :

- Départements :(DNO, DNOM, DIR, VILLE)
- Employés: (ENO, ENOM, PROF, DATEEMB, SAL, COMM, #DNO)

A. Donnez la liste des employés ayant une commission

```
SELECT * FROM Employes WHERE COMM IS NOT NULL;
```

B. Donnez les noms, emplois et salaires des employés par emploi croissant, et pour chaque emploi, par salaire décroissant

```
SELECT ENOM,PROF, SAL FROM Employes ORDER BY PROF ASC, SAL DESC;
```

C. Donnez le salaire moyen des employés

```
SELECT AVG(SAL) FROM Employes;
```

D. Donnez le salaire moyen du département Production

```
SELECT AVG(E.SAL) FROM Employes E INNER JOIN Departement D
ON E.DNO=D.DNO WHERE D.DNOM="production";
```

E. Donnez les numéros de département et leur salaire maximum

```
SELECT DNO, MAX(SAL) FROM Employes GROUP BY DNO;
```

F. Donnez les différentes professions et leur salaire moyen

```
SELECT PROF, AVG(SAL) FROM Employes GROUP BY PROF;
```

G. Donnez le salaire moyen par profession le plus bas

```
SELECT PROF, AVG(SAL) as moy FROM Employes
GROUP BY PROF
ORDER BY moy ASC
LIMIT 1 ;
```

H. Donnez-le ou les emplois ayant le salaire moyen le plus bas, ainsi que ce salaire moyen

```
SELECT PROF FROM Employes GROUP BY PROF
HAVING AVG(SAL)=(SELECT AVG(SAL) as moy FROM Employes
GROUP BY PROF ORDER BY moy ASC LIMIT 1);
```

Astuce : on ajoute le mot clé LIMIT qui permet de préciser combien de lignes à retourner à partir des résultats du select.
Syntaxe : SELECT … LIMIT *nombre_enregistrements;*
Exemple: SELECT * FROM Employé GROUP BY PROF ORDER BY AVG(SAL) ASC LIMIT 1;

Gestion des clients/représentants

5.4) La société X utilise un logiciel pour gérer ses clients et ses représentants. Voici la BDD utilisée dans le logiciel:

la table représentant

NUM_REP	NOM_REP	AD_REP	CP_REP	VIL_REP	AGE_REP
1	DELMOTTE	18 rue Aristide Briand	75012	PARIS	26
2	HINAUD	25 rue Martel	94120	FONTENAY SOUS BOIS	31
3	LAPIERRE	89 rue Gaston berger	95100	ARGENTEUIL	52
4	LATOUR	7 rue du Four	91700	FLEURY MÉROGIS	44
5	LEMOINE	5 rue Aubois	91700	FLEURY MEROGIS	28
6	LEMOINE	12 route des Fiacres	93140	BONDY	34

la table couvrir

NUM_REP	COD_DEP
1	75
1	94
2	93
2	94
3	91
3	75
4	95
5	93
5	91
6	92
6	95

la table département

COD_DEP	NOM_DEP	CHEF SECTEUR
75	Paris	PONS
91	Essonne	BERTRAND
92	Hauts de Seine	FISCHER
93	Seine Saint Denis	FISCHER
94	Val de Marne	BERTRAND
95	Val d'Oise	BERTAND

la table client

CODE_CLT	NOM_CLT	NUM_REP	NUM_CAT
1	BOCCARD	1	1
2	RALDI	2	1
3	PIERROL	2	3
4	ENGELI	2	3
5	ATR	4	2
6	PARTOLI	4	3
.....			

la table catégorie tarifaire

NUM_CAT	NOM_CAT	REMISE
1	ENTREPRISES	10%
2	COLLECTIVITES	5%
3	PARTICULIERS	0%

Écrire les requêtes suivantes:

A. Afficher la liste des clients appartenant à la catégorie tarifaire n°1, classée par ordre alphabétique

```
SELECT CODE_CLT , NOM_CLT FROM client WHERE NUM_CAT =1
ORDER BY NOM_CLT ASC;
```

B. Afficher la liste des clients (code, nom de client) rattachés au représentant HINAUD

```
SELECT CODE_CLT , NOM_CLT FROM client , representant
WHERE client.NUM_REP=representant.NUM_REP AND NOM_REP="HINAUD";
```

C. Afficher la liste des clients bénéficiant d'une remise de 10%

```
SELECT CODE_CLT, NOM_CLT FROM client,categorie_tarifaire
WHERE client.NUM_CAT=categorie_tarifaire.NUM_CAT AND REMISE="10%";
```

D. Afficher la liste des représentants (Numéro et nom) dépendant du chef de secteur PONS

```
SELECT NUM_REP ,NOM_REP FROM representant ,couvrir ,departement
WHERE reprsentant.NUM_REP=couvrir.NUM_REP AND couvrir.CODE_DEP=departement.CODE_DEP
AND CHEF_SECTEUR="PONS";
```

E. Afficher la liste des départements (code, nom, chef de secteur)

```
SELECT * FROM departement;
```

F. Afficher la liste des chefs de secteur

```
SELECT DISTINCT CHEF_SECTEUR FROM departement ;
```

BDD Étudiants

5.5) Soit la base de données suivante :

Etudiant

numéro_carte_etudiant	Nom	Prénom	Date_naissance	Section
01234567	Ben Salah	Ahmed	12/08/1988	Informatique
01234568	Ben Mahmoud	Sami	02/09/1990	Math
01234569	Marzougui	Rami	23/01/1988	Informatique

Matière

code_matière	nom_matière	coefficient
12508	Base de données	1.5
12518	Algorithme	3

Note

numéro_carte_etudiant	code_matière	note_examen
01234567	12508	15.5
01234567	12518	5.5
01234568	12518	10.5
01234569	12518	8.75

Écrire les commandes SQL permettant de rechercher :

A. La liste de tous les étudiants.

```
select * from Etudiant;
```

B. Nom et coefficient des matières.

```
select nom_matiere , coefficient from Matiere ;
```

C. Les numéros des cartes d'identité des étudiants dont la moyenne entre 7 et 12.

```
SELECT numero_carte_etudiant FROM Note, Matiere mat
WHERE Note.code_matiere=Mat.code_matiere
GROUP BY numero_carte_etudiant
HAVING (sum(note_examen*coefficient)/sum(coefficient)) between 7 and 12 ;
```

D. La liste des étudiants dont le nom commence par 'ben'.

```
select * from Etudiant where Nom like "Ben%";
```

E. Le nombre des étudiants qui ont comme matière '12518'.

```
select * from Note where code_matiere=12518 ;
```

F. La somme des coefficients des matières.

```
select sum(coefficient) from Matiere ;
```

G. Les noms des étudiants qui une note_examen >10.

```
SELECT distinct Nom FROM Note , Etudiant
WHERE Note.numero_carte_etudiant=Etudiant.numero_carte_etudiant AND note_examen >10 ;
```

H. Afficher les noms et les coefficients des matières étudier par l'étudiant "01234568".

```
SELECT nom_matiere , coefficient FROM Note , Matiere
WHERE Note.numero_carte_etudiant="01234568";
```