Exercices Corrigés en SQL

ESSADDOUKI Mostafa (essaddouki@gmail.com), 26 Septembre 2016

Exercice 1

Soit la base de données d'un festival de musique : Dans une représentation peut participer un ou plusieurs musiciens. Un musicien ne peut participer qu'à une seule représentation.

— Representation (**Num_Rep** , titre_Rep , lieu)
— Musicien (**Num_mus** , nom , #**Num_Rep**)
— Programmer (Date , #**Num_Rep** , tarif)

Ecrire la commande SQL permettant de rechercher :

1. La liste des titres des représentations.

```sql
SELECT * FROM Representation
```

2. La liste des titres des représentations ayant lieu au « théâtre allissa ».

```sql
SELECT * FROM Representation WHERE lieu="theatre allissa"
```

3. La liste des noms des musiciens et des titres et les titres des représentations auxquelles ils participent.

```sql
SELECT M.nom, R. titre FROM Musicien M INNER JOIN Representation R
ON R.Num_rep=M.Num_rep
```

4. La liste des titres des représentations, les lieux et les tarifs du 25/07/2008.

```sql
SELECT R. titre, R.lieu,P.tarif FROM Programmer P INNER JOIN Representation R
ON P.Num_rep=R.Num_rep WHERE P.date="25-07-2008"
```

5. Le nombre des musiciens qui participent à la représentations n°20.

```sql
SELECT COUNT(*) FROM Musicien WHERE Num_rep=20
```

6. Les représentations et leurs dates dont le tarif ne dépasse pas 20DH.

```sql
SELECT R.Num_Rep, R.titre, P.Date FROM Representation R
INNER JOIN Programmer P ON R.Num_Rep=P.Num_Rep WHERE P.tarif<=20
```

Exercice 2

Soit la base de données suivante :

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

Exprimez en SQL les requêtes suivantes :

1. Donnez la liste des employés ayant une commission

```sql
SELECT * FROM Employes WHERE COMM NOT NULL
```

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

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

3. Donnez le salaire moyen des employés

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

4. 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"
```

5. Donnes les numéros de département et leur salaire maximum

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

6. Donnez les différentes professions et leur salaire moyen

```
SELECT PROF, MAX(SAL) FROM Employes GROUP BY PROF
```

7. Donnez le salaire moyen par profession le plus bas

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

8. 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)
```

Exercice 3

Soit le modèle relationnel suivant relatif à la gestion des notes annuelles d'une promotion d'étudiants :
— ETUDIANT(**NEtudiant**, Nom, Prénom)
— MATIERE(**CodeMat**, LibelléMat, CoeffMat)
— EVALUER(#**NEtudiant**, #**CodeMat**, Date, Note)
Remarque : les clés primaires sont soulignées et les clés étrangères sont marquées par #
Exprimez en SQL les requêtes suivantes :

1. Quel est le nombre total d'étudiants ? SELECT count(*) FROM ETUDIANT
2. Quelles sont, parmi l'ensemble des notes, la note la plus haute et la note la plus basse ? SELECT MIN(Note) as 'plus basse note', MAX(Note) as 'plus haute note' FROM EVALUER
3. Quelles sont les moyennes de chaque étudiant dans chacune des matières ? SELECT E.NEtudiant, E.Nom, E.Prénom, M.LibelléMat, M.CoeffMat, AVG(EV.Note) AS MoyEtuMat FROM EVALUER EV, MATIERE M, ETUDIANT E WHERE EV.CodeMat = M.CodeMat AND EV.NEtudiant = E.NEtudiant GROUP BY E.NEtudiant, M.LibelléMat
4. Quelles sont les moyennes par matière ? Avec la vue MGETU de la question 3 (MOYETUMAT)

```
SELECT LibelleMat, AVG(MoyEtuMat) FROM MOYETUMAT  GROUP BY LibelleMat
```

5. Quelle est la moyenne générale de chaque étudiant ? Avec la vue MGETU de la question 3 (MOYETUMAT)

```
SELECT NEtudiant, Nom, SUM(MoyEtuMat*CoeffMat)/SUM(CoeffMat) AS MgEtu
FROM MOYETUMAT  GROUP BY NEtudiant
```

6. Quelle est la moyenne générale de la promotion ? Avec la vue MGETU de la question 5 :

```
SELECT AVG(MgEtu)  FROM MGETU
```

7. Quels sont les étudiants qui ont une moyenne générale supérieure ou égale à la moyenne générale de la promotion ? Avec la vue MGETU de la question 5

```
SELECT NEtudiant, Nom, Prenom, MgEtu FROM MGETU
WHERE MgEtu >= (SELECT AVG(MgEtu) FROM MGETU)
```

Exercice 4

Soit la base de données intitulée "gestion_projet" permettant de gérer les projets relatifs au développement de logiciels. Elle est décrite par la représentation textuelle simplifiée suivante :

- Developpeur (**NumDev**, NomDev, AdrDev, EmailDev, TelDev)
- Projet (**NumProj**, TitreProj, DateDeb, DateFin)
- Logiciel (**CodLog**, NomLog, PrixLog, #**NumProj**)
- Realisation (#**NumProj**, #**NumDev**)

Ecrire les requêtes SQL permettant :

1. D'afficher les noms et les prix des logiciels appartenant au projet ayant comme titre « gestion de stock », triés dans l'ordre décroissant des prix .

```
SELECT L.NomLog, L.PrixLog FROM Logiciel L INNER JOIN Projet P
ON L.NumProj=P.NumProj WHERE P.TitreProj="gestion de stock"
ORDER BY L.PrixLog DESC
```

2. D'afficher le total des prix des logiciels du projet numéro 10. Lors de l'affichage, le titre de la colonne sera « cours total du projet ».

```
SELECT SUM(PrixLog) as "cout total du projet" FROM Logiciel WHERE NumPRoj=10
```

3. Afficher le nombre de développeurs qui ont participé au projet intitulé « gestion de stock »

```
SELECT count(*) FROM Developpeur D INNER JOIN Realisation R
ON D.NumDev=R.NumDev INNER JOIN Projet P ON P.NumProj=R.NumProj
```

4. Afficher les projets qui ont plus que 5 logiciels SELECT NumProj, TitreProj FROM PRojet P INNER JOIN Logiciel L ON P.NumProj=L.NumProj GROUP BY NumProj HAVING count(*)>5
5. Les numéros et noms des développeurs qui ont participés dans tout les projets.

```
SELECT NumDev, NomDev FROM Developpeur D INNER JOIN Realisation R ON
D.NumDev=R.NumDev GROUP BY NumDev HAVING
count(*)=(SELECT COUNT(*) FROM Projet)
```

6. Les numéros de projets dans lesquelles tous les développeurs y participent dans sa réalisation.

```
SELECT NumProj, TitreProj FROM Projet P INNER JOIN Realisation R
ON P.NumProj=R.NumProj GROUP BY NumProj HAVING
count(*)=(SELECT COUNT(*) FROM Developpeur)
```

Exercice 5

Ci-après, on donne la représentation textuelle simplifiée d'une base de données concernant un cycle de formation destiné à des étudiants. Il regroupe un ensemble de matières. On considère que chaque enseignant n'enseigne qu'une seule matière et qu'à la fin du cycle de formation, une note par matière, est attribuée à chaque étudiant. D'autre par, les étudiants peuvent ne pas suivre les mêmes matières.

- ETUDIANT(**CodeEt**, NomEt, DatnEt)
- MATIERE(**CodeMat**, NomMat, CoefMat)
- ENSEIGNANT(**CodeEns**, NomEns, GradeEns, #**CodeMat**)
- NOTE(#**CodeEt,** #**CodeMat**, note)

Ecrire les requêtes SQL permettant d'afficher :

1. Les informations relatives aux étudiants (Code, Nom et Date de naissance) selon l'ordre alphabétique croisant du nom

```
SELECT * FROM ETUDIANT ORDER BY NomEt ASC
```

2. Les noms et les grades des enseignants de la matière dont le nom est 'BD'.

```
SELECT E.NomEns, E.GradeEns FROM ENSEIGNANT E INNER JOIN MATIERE M
ON M.CodeMat=E.CodeMat WHERE M.NomMat="BD"
```

3. La liste distincte formée des noms et les coefficients des différentes matières qui sont enseignées par des enseignants de grade 'Grd3'.

```
SELECT DISTINCT(M.NomMat), M.CoefMat FROM ENSEIGNANT E
INNER JOIN MATIERE M ON M.CodeMat=E.CodeMat WHERE E.GradeEns="Grd3"
```

4. La liste des matières (Nom et Coefficient) qui sont suivies par l'étudiant de code 'Et321'.

```
SELECT M.NomMat, M.CoefMat FROM MATIERE M INNER JOIN NOTE N
ON M.CodeMat=N.CodeMat INNER JOIN ETUDIANT E ON E.CodeEt=N.CodeEt
WHERE E.CodeEt="Et321"
```

5. Le nombre d'enseignants de la matière dont le nom est 'Informatique' SELECT COUNT(*) FROM ENSEIGNANT E INNER JOIN MATIERE M ON M.CodeMat=E.CodeMat WHERE M.NomMat="Informatique"

Exercice 6

On considère la base de données BD_AIR_MAROC suivante :
— PILOTE (NUMPIL, NOMPIL, VILLE, SALAIRE)
— AVION (NUMAV, NOMAV, CAPACITE, VILLE)
— VOL (**NUMVOL,** #**NUMPIL,** #**NUMAV**, VILLE_DEP, VILLE_ARR, H_DEP, H_ARR)

1. Donnez la liste des avions dont la capacité est supérieure à 350 passagers.

```
SELECT * FROM AVION WHERE CAPACITE>350
```

2. Quels sont les numéros et noms des avions localisés à Marrakech ?

```
SELECT NUMAV, NOMAV FROM AVION WHERE VILLE='Marrakech'
```

3. Quels sont les numéros des pilotes en service et les villes de départ de leurs vols ?

```
SELECT NUMPIL, VILLE_DEP FROM VOL
```

4. Donnez toutes les informations sur les pilotes de la compagnie.

```
SELECT * FROM PILOTE
```

5. Quel est le nom des pilotes domiciliés à Meknès dont le salaire est supérieur à 20000 DH ?

```
SELECT NOMPIL FROM PILOTE WHERE VILLE='Meknes' AND SALAIRE>20000
```

6. Quels sont les avions (numéro et nom) localisés à Marrakech ou dont la capacié est inférieure à 350 passagers ?

```
SELECT NUMAV, NOMAV FROM AVION WHERE VILLE='Marrakech' AND CAPACITE<350
```

7. Quels sont les numéros des pilotes qui ne sont pas en service ?

```
SELECT NUMPIL FROM PILOTE
WHERE NUMPIL NOT IN (SELECT DISTINCT NUMPIL FROM VOL)
```

8. Donnez le numéro des vols effectués au départ de Marrakech par des pilotes de Meknès ?

```
SELECT DISTINCT V.NUMVOL FROM VOL AS V, PILOTE AS P
WHERE V.NUMPIL=P.NUMPIL AND V.VILLE_DEP='Marrakech' AND P.VILLE='Meknes'
```

Ou

```
SELECT DISTINCT NUMVOL FROM VOL WHERE V.VILLE_DEP='Marrakech' AND
NUMPIL NOT IN (SELECT NUMPIL FROM PILOTE WHERE VILLE='Meknes')
```

9. Quels sont les vols effectu"s par un avion qui n'est pas localisé à Marrakech ?

```
SELECT DISTINCT V.NUMVOL FROM VOL V, AVION A WHERE A.NUMAV=V.NUMAV
AND A.VILLE !='Marrakech'
```

10. Quelles sont les villes desservies à partir de la ville d'arrivée d'un vol au départ de Guelmim ?

```
SELECT DISTINCT VILLE_ARR FROM VOL WHERE VILLE_DEP='Guelmim'
AND VILLE_DEP!=VILLE_ARR
```

Exercice 7

Soit le schéma relationnel suivant :

— Departement (NomD, **N_Dep**, Directeur)
— Employe (**Matricule,** Nom, Prénom, DateNaissance, Adresse, Salaire, #**N_dep**, superieur)
— Projet (NomP, **N_pro**, Lieu, #**N_Dep**)
— Travaille (#**Matricule,** #**N_Proj**, Heures)

L'attribut supérieur dans la relation Employe contient le matricule du supérieur direct de l'employé. Chaque employé appartient à un département et travaille sur zéro, un ou plusieurs projets. Chaque projet est rattaché à un département qui peut être différent de celui des employés travaillant sur ce projet. Exprimer en SQL les requêtes suivantes :

1. Date de naissance et l'adresse de Taha Lamharchi.

```
SELECT DateNaissance, Adresse FROM Employe WHERE Nom='Lamharchi'
AND Prenom ='Taha'
```

2. Nom et adresse des employés qui travaillent au département de recherche.

```
SELECT E.Nom, E.Adresse FROM Employe as E, Departement as D
WHERE E.N_dep=D.N_dep AND NomD='recherche'
```

3. Nom et Prénom des employés dont le supérieur est Taha Lamharchi.

```
SELECT Nom, Prenom FROM Employe
WHERE superieur=(SELECT Matricule FROM Employe WHERE Nom='Lamharchi'
AND Prenom ='Taha')
```

4. Nom des employés qui travaillent plus de 10heures sur un projet à Guelmim

```
SELECT E.Nom FROM Employe as E, Travaille as T , Projet P
WHERE E.Matricule=T.Matricule
AND T.N_proj=P.N_proj AND T.heures>=10 AND P.Lieu='Guelmim'
```

5. Nom des projets sur lesquelles travaillent Taha Lamharchi et Dounia Mahmoud.

```
SELECT T.N_proj FROM Travaille as T,Employe as E  WHERE T.Matricule=E.Matricule
AND E.Nom='Lamharchi' AND E.Prenom='Taha'
INTERSECT
SELECT T.N_proj FROM Travaille as T,Employe as E  WHERE T.Matricule=E.Matricule
AND E.Nom='Mahmoud' AND E.Prenom='Dounia'
```

6. Nom et prénom des employés qui ne travaillent sur aucun projet.

```
SELECT Nom, Prenom FROM Employe
WHERE Matricule NOT IN ( SELECT Matricule FROM Travaille)
```

7. Numéro des projets qui ont au moins un participant de chaque département.

```
SELECT T.N_proj FROM Travaille as T,Projet as P, Employe as E
WHERE T.N_proj=P.N_proj ANDT.Matricule=E.Matricule
GROUP BY T.N_proj
HAVING count(DISTINCT E.N_dep)=(SELECT count(*) FROM Departement)
```

8. Nom des employés qui ne travaillent pas sur un projet à Guelmim.

```
SELECT Nom FROM Employe WHERE
Matricule NOT IN(SELECT T.Matricule FROM Travaille as T, Projet as P
WHERE T.N_proj=P.N_proj AND P.Lieu='Guelmim')
```

Exercice 8

Soit le schéma relationnel suivant qui représente la base de données d'une agence de voyage en ligne.
— CLIENT (**NumCli**, Nom, Prénom, e-mail, NumCB)
— VOYAGE (**CodeVoyage**, Destination, Durée, Prix)
— RESERVATION (#**NumCli,** #**CodeVoyage**, DateRes)
Formuler en SQL les requêtes suivantes :

1. Nom, prénom et e-mail des clients ayant une réservation en cours

```
SELECT Nom, Prenom, e-mail FROM CLIENT
WHERE NumCli IN (SELECT DISTINCT NumCli FROM RESERVATION)
```

2. Nom, prénom et e-mail des clients n'ayant aucune réservation en cours

```
SELECT Nom, Prenom, e-mail FROM CLIENT
WHERE NumCli NOT IN (SELECT DISTINCT NumCli FROM RESERVATION)
```

3. Destination et liste des clients ayant réservés pour un voyage de plus de 10 jours et coûtant moins de 1000 DH.

```
SELECT C.Nom, C.Prenom, V.Destination FROM CLIENT as C, VOYAGE as V,
RESERVATION as R
WHERE C.NumCli=R.NumCli and  V.CodeVoyage=R.CodeVoyage
AND Duree>=10 AND Prix<1000
```

4. Numéros de tous les clients ayant réservés sur tous les voyages proposés.

```
SELECT NumCli FROM RESERVATION GROUP BY NumCli
HAVING count(*)=(SELECT count(*) FROM VOYAGE)
```

Exercice 9

Soit la base de données « cinéma » dont le schéma relationnel est donné ci-dessous :
— VILLE (CodePostal, NomVille)
— CINEMA (**NumCine**, NomCine, Adresse, #**CodePostal**)
— SALLE (**NumSalle**, Capacité, #**NumCine**)
— FILM (**NumExploit**, Titre, Durée)
— PROJECTION (#**NumExploit,** #**NumSalle**, NumSemaine, Nbentrees)
Ecrivez les requêtes suivantes en algèbre relationnelle :

1. Titre des films dont la durée est supérieure ou égale à deux heures

```
SELECT NumExploit, Titre FROM FILM WHERE Duree>=2
```

2. Nom des villes abritant un cinéma nommé « RIF »

```
SELECT NomVille FROM VILLE
WHERE CodePostal IN (SELECT CodePostal FROM CINEMA WHERE NomCine='RIF')
```

3. Nom des cinémas situés à Meknès ou contenant au moins une salle de plus 100 places

```
SELECT NomCine FROM CINEMA WHERE CodePostal=(SELECT CodePostal
FROM VILLE WHERE NomVille='Meknes')
OR NumCine IN (SELECT NumCine FROM SALLE WHERE Capacite>=100)
```

4. Nom, adresse et ville des cinémas dans lesquels on joue le film « Hypnose » la semaine 18

```
SELECT C.NomCine, C.Adresse, V.NomVille FROM CINEMA as C, VILLE as V
WHER C.CodePostal=V.CodePostal
AND C.NumCine IN (SELECT S.NumCine FROM SALLE as S, FILM as F, PROJECTION as P
WHERE P.NumExploit=F.NumExploit AND P.NumSalle=S.NumSalle
AND F.Titre='Hypnose' AND P.NumSemaine=18)
```

5. Numéro d'exploitation des films projetés dans toutes les salles

```
SELECT NumExploit FROM PROJECTION GROUP BY NumExploit
HAVING count(*)=(SELECT count(*) FROM SALLE)
```

6. Titre des films qui n'ont pas été projetés

```
SELECT Titre FROM FILM WHERE NumExploit NOT IN (SELECT NumExploit FROM PROJECTION)
```

Exercice 10

Soit le modèle relationnel suivant relatif à la gestion simplifiée des étapes du Tour de France 97, dont une des étapes de type "contre la montre individuel" se déroula à Saint-Etienne :

- EQUIPE(CodeEquipe, NomEquipe, DirecteurSportif)
- COUREUR(NuméroCoureur, NomCoureur, #**CodeEquipe,** #**CodePays**)
- PAYS(CodePays, NomPays)
- TYPE_ETAPE(CodeType, LibelleType)
- ETAPE(NuméroEtap, DateEtape, VilleDép, VilleArr, NbKm, #CodeType
- PARTICIPER(#**NuméroCoureur,** #**NuméroEtape**, TempsRealisé
- ATTRIBUER_BONIFICATION(#**NuméroEtape,** #**NuméroCoureur,** km, Rang, NbSecondes)

Exprimez en SQL les requetes suivantes :

1. Quelle est la composition de l'équipe Festina (Numéro, nom et pays des coureurs) ?

```
SELECT NumeroCoureur, NomCoureur, NomPays  FROM EQUIPE A, COUREUR B, PAYS C
WHERE A.CodeEquipe=B.CodeEquipe And B.CodePays=C.CodePays
And NomEquipe="FESTINA"
```

2. Quel est le nombre de kilomètres total du Tour de France 97 ?

```
SELECT SUM(Nbkm) FROM ETAPE
```

3. Quel est le nombre de kilomètres total des étapes de type "Haute Montagne" ?

```
SELECT SUM(Nbkm) FROM ETAPE A, TYPE_ETAPE B
WHERE A.CodeType=B.CodeType And LibelleType="HAUTE MONTAGNE"
```

4. Quels sont les noms des coureurs qui n'ont pas obtenu de bonifications ?

```
SELECT NomCoureur FROM COUREUR
WHERE NumeroCoureur NOT IN (SELECT NumeroCoureur FROM ATTRIBUER_BONIFICATION)
```

5. Quels sont les noms des coureurs qui ont participé à toutes les étapes ?

```
SELECT NomCoureur FROM PARTICIPER A, COUREUR B
WHERE A.NumeroCoureur=B.NumeroCoureur
GROUP BY NumeroCoureur, NomCoureur
HAVING COUNT(*)=(SELECT COUNT(*) FROM ETAPE)
```

6. Quel est le classement général des coureurs (nom, code équipe, code pays et temps des coureurs) à l'issue des 13 premières étapes sachant que les bonifications ont été intégrées dans les temps réalisés à chaque étape ?

```sql
SELECT NomCoureur, CodeEquipe, CodePays, SUM(TempsRealise) AS Total
FROM PARTICIPER A, COUREUR B
WHERE A.NumeroCoureur=B.NumeroCoureur and NumeroEtape<=13
GROUP BY A.NumeroCoureur, NomCoureur, CodeEquipe, CodePays
ORDER BY Total
```

7. Quel est le classement par équipe à l'issue des 13 premières étapes (nom et temps des équipes) ?

```sql
SELECT NomEquipe, SUM(TempsRealise) AS Total
FROM PARTICIPER A, COUREUR B, EQUIPE C
WHERE A.NumeroCoureur=B.NumeroCoureur And B.CodeEquipe=C.CodeEquipe
And NumeroEtape<=13
GROUP BY B.CodeEquipe, NomEquipe ORDER BY Total
```

Exercice 11 : extrait HEC 2014

à partir du système d'information de l'entreprise. le service des ressources humaines peut extraire et analyser les informations relatives à tous les personnels. celui-ci lui permet en particulier d'exercer un suivi dans le domaine de la formation. un extrait de ce domaine est présenté sours forme d'un schéma relation :

Extrait du modèle relationnel du système d'information du service des ressources humaines

ORGFORM (Numorgform, Nomorgform, Adresseorgform, CPorgform, Villeorgform, Nomorgform, Telorgform, Melorgform)
DATEFORM (Datedebutform)
FORMATION (Codeform, Intituleform, Dureeform, Niveauform, Nbreplaceform, Coutform)
SALARIE (Matriculesal, Nomsal, Prenomsal, Qualificationsal, #Codecatégorie)
CATEGORIE (Codecatégorie, Libellécatégorie)
SERVICE (Codeserv, Nomserv, Anciennetéserv)
PASSERBILAN (#Matriculesal, #Matriculesal, Datebilan)
AFFECTER (#Matriculesal, #Codeserv)
PROPOSER (#Codeform, #Matriculesal, Dateproposition)
SUIVRE (#Codeform, #Matriculesal, Datedebut)
REALISER (#Numorgform, #Codeform, #Datedébuform, NbInscrits)

Construire les requêtes en langage SQL permettant de répondre aux questions suivantes :

1. quel est le nombre de formations suivies par catégories de salariés ayant débuté au cours de la période du 01/06/2011 au 31/12/2011 ?

```sql
SELECT Libellecategorie, count(distinct Codeform) FROM SUIVRE,
SALARIE, CATEGORIE WHERE SUIVRE.Matriculesal=SALARIE.Matriculesal
AND SALARIE.codecategorie=CATEGORIE.codecategorie
AND Datedebut BETWEEN "01/06/2011" AND "31/12/2011"
GROUP BY Libellecategorie
```

2. quelles sont les catégories pour lesquelles le nombre d'heures de formation est supérieur à la moyenne du nombre d'heures des formations suivies par l'ensemble des personnels ?

```sql
SELECT Libellecategorie FROM SUIVRE, SALARIE, CATEGORIE, FORMATION
WHERE SUIVRE.Matriculesal=SALARIE.Matriculesal AND
SALARIE.codecategorie=CATEGORIE.codecategorie AND
FORMATION.Codeform=SUIVRE.Codeform
```

```
GROUP BY Libellecategorie
HAVING SUM(Dureeform) > (SELECT AVG(Dureeform) SUIVRE, FORMATION
WHERE SUIVRE.Codeform=FORMATION.Codeform)
```

3. le responsable des ressources humaines souhaite intégrer dans la base de données une nouvelle formation liée au sertissage des boîtes de conserve.
   les nouvelles données à insérer sont les suivantes : "FORM587, sertissage niveau 1, 25j, perfectionnement, 12, 525 "
   Ecrire la requête permettant de mettre à jour la base.

```
INSERT INTO FORMATION VALUES("FORM587","sertissage niveau 1",600,
"perfectionnement",12,525)
```

Exercice 12

La société X utilise le logiciel de gestion de base de données Access pour gérer ses clients et ses représentants. Voici la liste des tables crées dans Access :

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 MÈROGIS	44
5	LEMOINE	5 rue Auboïs	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%

Ecrire les requêtes suivantes

1. 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
```

2. 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"
```

3. 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%"
```

4. 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"
```

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

```
SELECT * FROM departement
```

6. Afficher la liste des chefs de secteur

```
SELECT DISTINCT CHEF_SECTEUR FROM departement
```

Exercice 13

Le responsable du SAV d'une entreprise d'électroménager a mis en place une petite base de données afin de gérer les interventions de ces techniciens. Le modèle relationnel à la source de cette base de données est le suivant :

— Client (Codecl, nomcl, prenomcl, adresse, cp, ville)
— Produit (Référence, désignation, prix)
— Techniciens (Codetec, nomtec, prenomtec, tauxhoraire)
— Intervention (Numéro, date, raison, #codecl, #référence, #codetec)

Le responsable vous demande d'écrire en langage SQL les requêtes suivantes :

1. La lite des produits (référence et désignation) classées du moins cher au plus cher. select Reference, designation from produit order by prix ASC
2. Le nombre d'intervention du technicien n°2381. select count (*) from Intervention where codetec =2381
3. La liste des clients ayant demandé une intervention pour des produits d'un prix supérieur à 300 dhs.

```
select nomcl from Client clt, Produit prod, Intervention int where clt.codecl=int.codecl
and prod.Reference=int.Reference and prod.prix>300
```

4. Les interventions effectuées par le technicien : 'Mestiri Mohamed' entre le 1er et le 31 août 2009.

```
select Numero , date , raison from Intervention int , Techniciens tec
where int.codetec=tec.codetec and tec.nomtec="Mestiri" and tec.prenomtec="Mohamed"
and int.date between "2009-08-01" and "2009-08-31"
select Numero , date , raison from Intervention int , Techniciens tec
where int.codetec=tec.codetec and tec.nomtec="Mestiri" and tec.prenomtec="Mohamed"
and MONTH(int.date)=8 and YEAR(int.date)=2009
```

5. Par ailleurs il vous informe que le produit référencé 548G a vu son prix augmenter (nouveau prix = 320 dhs).

```
update Produit set prix=320 where Reference="548G"
```

6. Vous apprenez également par le directeur des ressources humaines qu'un nouveau technicien a été recruté : son code est le 3294, il s'appelle 'El Abed Ridha' et est rémunéré à un taux horaire de 15 dhs.

```
insert into Technicien values(3294,"EL Abed","Ridha",15)
```

Exercice 14

La représentation textuelle suivante est une description simplifiée d'une base de données de gestion de facturation d'une entreprise commerciale.

- **Client** (**Numcli**, Nomcli, Prenomcli, adressecli, mailcli)
- **Produit** (**Numprod**, désignation, prix , qte_stock)
- **Vendeur** (**Idvendeur**, Nomvendeur, adresse_vend)
- **Commande** (**Numcom**, #Numcli, #Idvendeur, #Numprod, date_com, qte_com)

On suppose que Numcli, Numprod, Idvendeur et Numcom sont de type numérique.
Le nom, le prénom et l'adresse des clients ainsi que les vendeurs sont des informations obligatoires, le mail peut ne pas être indiqué.
La valeur par défaut de la quantité en stock des produits (qte_stock) est égale à 0
Exprimer en SQL les requêtes suivantes :

1. Créer les tables : Client, Produit, Vendeur et Commande. create table Produit(Numprod int primary key , designation varchar(30), prix float , qte_stock int default 0) create table commande(Numcom int primary key , Numcli int , idvendeur int , Numprod int date_com date qte_com int FOREIGN KEY(Numcli) REFERENCES Client(Numcli), FOREIGN KEY(idvendeur) REFERENCES Vendeur(idvendeur), FOREIGN KEY(Numprod) REFERENCES Produit(Numprod))
2. la liste des clients de marrakech.

```
select * from Client where adressecli like "%marrakech%"
```

3. la liste des produits (Numprod, désignation, prix) classés de plus cher au moins cher.

```
select Numprod , designation , prix from Produit order by prix ASC
```

4. noms et adresses des vendeurs dont le nom commence par la lettre 'M'.

```
select Nomvendeur, adresse_vend from Vendeur where Nomvendeur like "M%"
```

5. la liste des commandes effectuées par le vendeur "Mohammed" entre le 1er et 30 janvier 2012.

```
select Numcom,Numcli,Idvendeur,Numprod,date_com,qte_com from Commande cmd, Vendeur vend
where cmd.Idvendeur=vend.Idvendeur and vend.Nomvendeur="mohammed"
and cmd.date_com between "2012-01-01" and "2012-01-30"
```

6. le nombre des commandes contenant le produit n° 365.

```
select count (*) from Commande where Numprod =365
```

Exercice 15

Soit la base de données suivante :
Ecrire les commandes SQL permettant de rechercher :

1. La liste de tous les étudiants.

```
select * from Etudiant
```

2. Nom et coefficient des matières.

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

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

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

```sql
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
```

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

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

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

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

6. La somme des coefficients des matières.

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

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

```sql
select distinct Nom from Note , Etudiant
where Note.numero_carte_etudiant=Etudiant.numero_carte_etudiant and note_examen >10
```

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

```sql
select nom_matiere , coefficient from Note , Matiere
where Note.numero_carte_etudiant="01234568"
```

Exercice 16

Afin d'assurer la qualité des produits attendues par les Clients, l'entreprise cherche à optimiser la gestion des pannes pouvant survenir dans les infrastructures de production nécessaires à la fabrication du Ciment.
voici un extrait de la base de données :

— TECHNICIEN (idTech, nom, prénom, spécialité)
— STATION (idstat, nom, Position, coordLat, coordLong,phase)
— MACHINE (idmach, état, dateMiseEnService, dateDernièreRévision, #idStat)
— TYPEINCIDENT (id, description, tempsRéparationPrévu)

— INCIDENT (idInd, remarques, dateHeure, dateHeureCloture,#idmach,#idType)
— INTERVENTION (idInterv, dateHeureDébut, dateHeureFin, #idInd, #idTech)

1. Rédiger la requête SQL permettant d'obtenir la liste par ordre alphabétique des noms et prénoms des techniciens ayant réalisé une intervention sur la Machine identifiée par Ber001.

```
select nom , prenom from TECHNICIEN tec , INCIDENT inc , INTERVENTION int
where tec.idTech=int.idTech and int.idInd=inc.idInd and idmach="Ber001"
order by nom ASC , prenom ASC
```

2. Rédiger la requête SQL permettant d'obtenir la liste des phases ayant connue un incident de "sur-chauffage" pour le mois Mai 2013.

```
select distinct phase from STATION st, MACHINE mch, INCIDENT inc, TYPEINCDENT type
where inc.idmach=mch.idmach and st.idstat=mch.idstat and type.id=inc.idType
and type.description="sur-chauffage" and MONTH(dateHeure)=5 and YEAR(dateHeure)=2013
```

3. Rédiger la requête SQL permettant d'obtenir le nombre d'incidents non clôturés.

```
select count(*) from INCIDENT where dateHeureCloture is NULL
```

4. Rédiger la requête SQL permettant d'obtenir la liste des noms des stations ayant eu plus de dix incidents.

```
select nom from STATION st, MACHINE mch, INCIDENT inc where inc.idmach=mch.idmach
and st.idstat=mch.idstat
GROUP by nom having count (*) >10
```

Exercice 17

voici un extrait de la base de données gestion des ventes :
— **Produit** (**Ref**, Designation, PrixUnitaire, Dimension, #code_Machine)
— **Vente** (**Ncom**, Ref, Qte , DateLiv)
— **Commande** (**Ncom**, DateCmd, #CodeClt,#Code_Salarie)
— **Produit_concurrent** (**Ref**, Designation, PrixUnitaire, PrixUnitaire,Dimension, #code_Machine,Nom_Concurrent)

1. Donner la requête qui permet d'obtenir le chiffre d'affaire mensuel de l'année en cours.

```
select sum(Qte*PrixUnitaire), MONTH(DateCmd) from Produit, Vente, Commande
where Produit.Ref=Vente.Ref and Vente.Ncom=Commande.Ncom and
YEAR(DateCmd)=YEAR(NOW()) group by MONTH(DateCmd)
```

2. Donner la requête qui calcule le taux de vente de chaque produit.

```
select Ref, sum(Qte)/(select sum(Qte) from Vente) from Vente group by Ref
```

3. Donner la requête qui affiche le produit le plus vendu du mois en cour.

```
select Ref, Designation, tot from (select Ref, Designation, sum(Qte) as tot from Produit,
Vente where Produit.Ref=Vente.Ref group by Ref) order by tot DESC LIMIT 1
```

4. La table produit concurrent est composée des informations sur les produits vedettes des concurrents ;
Donner la requête qui permet d'ajouter tous les produits du concurrent GleenAlu à la table Produits.

```
insert into Produit (Ref,Designation,PrixUnitaire,Dimension,code_machine)
(SELECT Ref, Designation, PrixUnitaire, Dimension,code_machine FROM Produit_concurrent
where Nom_Concurent="GleenAlu")
```