

## ex1

### Réalisez les requêtes suivantes :

**a. Afficher toutes les informations sur les véhicules loués par le Client n°T122**

```
SELECT v.*  
  
FROM `vehicule` v  
  
INNER JOIN a_loue al ON v.id_vehicule = al.id_vehicule  
  
WHERE al.id_client = "T122"
```

**b. Afficher toutes les locations réalisées par le client n° T122**

```
SELECT v.Immatriculation  
  
FROM `vehicule` v  
  
INNER JOIN a_loue al ON v.id_vehicule = al.id_vehicule  
  
WHERE al.id_client = "T122"
```

**c. Afficher l'immatriculation, l'âge et l'état de tous les véhicules.**

```
SELECT Immatriculation, Age, Etat  
  
FROM `vehicule`
```

**d. Afficher les noms des clients et les adresses, des clients qui habitent à << Nice >>.**

```
SELECT `Nom`, `Adresse`  
  
FROM `client`  
  
WHERE `Ville` = "Nice"
```

**e. Affiche la liste des clients par ordre alphabétique croissant des noms**

```
SELECT *  
  
FROM client  
  
ORDER BY nom ASC
```

**f. Ajouter l'attribut kilométrage et Afficher la liste des voitures par ordre décroissant des**

compteurs (kilométrage)

```
SELECT *
```

```
FROM vehicule
```

```
ORDER BY `kilometrage` DESC*
```

**g. Afficher les informations sur les clients qui ont loué la voiture EW 25EW**

```
SELECT c . *
```

```
FROM `client` c
```

```
INNER JOIN a_loue al ON c.id_Client = al.id_client
```

```
INNER JOIN vehicule v ON al.id_vehicule = v.id_vehicule
```

```
WHERE v.immatriculation = "EW25EW"
```

**h. Afficher toutes les voitures noires :)**

```
SELECT *
```

```
FROM `vehicule`
```

```
WHERE `couleur` = 'noir'
```

**i. Afficher toutes les voitures ayant un kilométrage <10000 km**

```
SELECT *
```

```
FROM `vehicule`
```

```
WHERE `kilometrage` < 10000
```

**j. Afficher toutes les informations sur les locations réalisées avant 2018**

```
SELECT *
```

```
FROM a_loue
```

```
WHERE `Date_de_retrait` < '2018-01-01';
```

**k. Afficher la moyenne des kilométrages de tous les véhicules du parc.**

```
SELECT AVG( `kilometrage` )
```

```
FROM vehicule
```

**L. Afficher toutes les locations réalisées en 2018**

```
SELECT *  
  
FROM a_loue  
  
WHERE `Date_de_retrait`  
  
BETWEEN '2018-01-01'  
  
AND '2018-12-31';
```

**M. Afficher le nombre de voitures ayant un kilométrage <10 000 kilomètres**

```
SELECT COUNT( * )  
  
FROM `vehicule`  
  
WHERE `kilometrage` <10000
```

## Partie 2

• **Obtenir la liste des véhicules empruntés et rendu le même jour ainsi que l'agence de rattachement**

```
SELECT v.id_vehicule, a.nom, al.date_de_retrait, al.date_de_retour  
  
FROM vehicule v  
  
INNER JOIN a_loue al ON v.id_vehicule = al.id_vehicule  
  
INNER JOIN agence a ON a.id_agence = v.id_agence  
  
WHERE al.date_de_retrait = al.date_de_retour
```

• **Obtenir le nombre véhicules pour chaque marque**

```
SELECT m.nom, COUNT( * )  
  
FROM vehicule v  
  
INNER JOIN marque m ON v.id_marque = m.id_marque  
  
GROUP BY m.nom
```

• **Obtenir les noms des clients qui ont loué plus de 10 véhicules de marque « Renault »**

```
SELECT c.nom, COUNT( * )
```

```

FROM vehicule v

INNER JOIN marque m ON v.id_marque = m.id_marque

INNER JOIN a_loue al ON v.id_vehicule = al.id_vehicule

INNER JOIN client c ON al.id_client = c.id_client

WHERE m.nom = "Renault"

GROUP BY m.nom

HAVING COUNT( * ) >10

```

- **Obtenir le nombre d'agences et d'employés par pa**

```

SELECT p.nom, Nb_employees, COUNT( a.id_agence ) AS "Nb agence"

FROM agence a

INNER JOIN pays p ON a.id_pays = p.id_pays

GROUP BY p.nom

```

## Exercice 2

**Ecrire les requêtes SQL permettant d'afficher :**

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

```

SELECT CodeEt, NomEt, DatnEt

FROM ETUDIANT

ORDER BY NomEt ASC ;

```

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

```

SELECT NomEns, GradeEns

FROM ENSEIGNANT

WHERE CodeMat = (

SELECT CodeMat

FROM MATIERE

```

```
WHERE 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 NomMat, CoefMat
```

```
FROM MATIERE
```

```
WHERE CodeMat
```

```
IN (
```

```
SELECT CodeMat
```

```
FROM ENSEIGNANT
```

```
WHERE GradeEns = 'Grd3'
```

```
);
```

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

```
SELECT NomMat, CoefMat
```

```
FROM MATIERE
```

```
WHERE CodeMat
```

```
IN (
```

```
SELECT CodeMat
```

```
FROM NOTE
```

```
WHERE CodeEt = 'Et321'
```

```
);
```

**5. Le nombre d'enseignants de la matière dont le nom est 'Informatique'**

```
SELECT COUNT( * )
```

```
FROM ENSEIGNANT
```

```
WHERE CodeMat = (
```

```
SELECT CodeMat
```

FROM MATIERE

WHERE NomMat = 'Informatique' );

## Exercice 3 :

Exprimez en SQL les requêtes suivantes :

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

```
SELECT C.NumeroCoureur, C.NomCoureur, PAYS.NomPays
```

```
FROM COUREUR C
```

```
INNER JOIN EQUIPE E ON C.CodeEquipe = E.CodeEquipe
```

```
INNER JOIN PAYS ON C.CodePays = PAYS.CodePays
```

```
WHERE E.NomEquipe = 'Festina'
```

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

```
SELECT SUM( NbKm ) AS "Nombre kilometre total"
```

```
FROM etape
```

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

```
SELECT SUM( e.NbKm ) AS "Nombre de kilomètres total pour le type Haute Montagne"
```

```
FROM etape e
```

```
INNER JOIN type_etape t ON e.CodeType = t.CodeType
```

```
WHERE t.LibelleType = 'Haute Montagne'
```

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

```
SELECT c.NomCoureur
```

```
FROM coureur c
```

```
WHERE c.NumeroCoureur NOT
```

```
IN (
```

```
SELECT a.NumeroCoureur
```

```
FROM ATTRIBUER_BONIFICATION a
```

)

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

```
SELECT c.NomCoureur
```

```
FROM coureur c
```

```
WHERE NOT
```

```
EXISTS (
```

```
SELECT e.NumeroEtap
```

```
FROM etape e
```

```
WHERE e.NumeroEtap NOT
```

```
IN (
```

```
SELECT p.NumeroEtap
```

```
FROM participer p
```

```
WHERE p.NumeroCoureur = c.NumeroCoureur
```

```
)
```

```
)
```

**7. 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 ?**

```
SELECT c.NomCoureur, c.CodeEquipe, c.CodePays, SUM( p.TempsRealise + ab.NbSecondes ) AS  
TotalTemps
```

```
FROM coureur c
```

```
INNER JOIN participer p ON c.NumeroCoureur = p.NumeroCoureur
```

```
INNER JOIN ATTRIBUER_BONIFICATION ab ON p.NumeroCoureur = ab.NumeroCoureur
```

```
AND p.NumeroEtap = ab.NumeroEtap
```

```
INNER JOIN etape e ON e.NumeroEtap = p.NumeroEtap
```

```
WHERE e.NumeroEtap <=13
```

GROUP BY c.NumeroCoureur

ORDER BY TotalTemps

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

SELECT NomEquipe, SUM( TempsRealise ) AS TempsTotal

FROM equipe e

INNER JOIN coureur c ON c.CodeEquipe = e.CodeEquipe

INNER JOIN participer p ON p.NumeroCoureur = c.NumeroCoureur

LEFT JOIN ATTRIBUER\_BONIFICATION ab ON ab.NumeroCoureur = c.NumeroCoureur

GROUP BY NomEquipe

ORDER BY TempsTotal

**Exercice 4 :**

CREATE TABLE Client (

Numcli INT PRIMARY KEY,

Nomcli VARCHAR(255) NOT NULL,

Prenomcli VARCHAR(255) NOT NULL,

adressecli VARCHAR(255) NOT NULL,

mailcli VARCHAR(255)

);

CREATE TABLE Produit (

Numprod INT PRIMARY KEY,

designation VARCHAR(255) NOT NULL,

prix DECIMAL(10, 2) NOT NULL,

qte\_stock INT DEFAULT 0

);

CREATE TABLE Vendeur



```

Idvendeur INT PRIMARY KEY,
Nomvendeur VARCHAR(255) NOT NULL,
adresse_vend VARCHAR(255) NOT NULL
);

CREATE TABLE Commande (
Numcom INT PRIMARY KEY,
Numcli INT NOT NULL,
Idvendeur INT NOT NULL,
Numprod INT NOT NULL,
date_com DATE NOT NULL,
qte_com INT NOT NULL,
FOREIGN KEY (Numcli) REFERENCES Client(Numcli),
FOREIGN KEY (Idvendeur) REFERENCES Vendeur(Idvendeur),
FOREIGN KEY (Numprod) REFERENCES Produit(Numprod)
);

```

**1. la liste des clients de Marrakech.**

```

SELECT Nomcli, Prenomcli, adressecli
FROM Client
WHERE adressecli = 'Marrakech'

```

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

```

SELECT Numprod, designation, prix
FROM Produit
ORDER BY prix DESC

```

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

```

SELECT Nomvendeur, adresse_vend

```

FROM Vendeur

WHERE Nomvendeur LIKE 'M%'

**4. la liste des commandes effectuées par le vendeur "Mohammed" entre le 1er et 30 janvier 2020.**

SELECT c.\*

FROM Commande c

INNER JOIN Vendeur v ON c.Idvendeur = v.Idvendeur

WHERE Nomvendeur = 'Mohammed'

AND date\_com

BETWEEN '2020-01-01'

AND '2020-01-30'

**5. le nombre des commandes contenant le produit n° 365.**

SELECT COUNT( \* ) AS "Nombre de commande pour le produit  
n°365"

FROM Commande

WHERE Numprod =365

## Exercice 5

### 1

SELECT SUM( coefficient ) AS "total coefficient"

FROM MATIERE

### 2

SELECT nom\_matiere, coefficient

FROM matiere

### 3

SELECT numero\_carte\_etudiant

FROM ETUDIANT

WHERE numero\_carte\_etudiant

IN (

SELECT numero\_carte\_etudiant

FROM NOTE

GROUP BY numero\_carte\_etudiant

HAVING AVG( note\_examen )

BETWEEN 7

AND 12

)

### 4

SELECT nom, prenom, numero\_carte\_etudiant

FROM ETUDIANT

WHERE nom LIKE 'ben%'

### 5

SELECT COUNT( \* ) AS "nombre des étudiants qui ont  
comme matière 12518"

FROM (

SELECT DISTINCT numero\_carte\_etudiant

FROM NOTE

WHERE code\_matiere = '12518'

) AS students

**6**

```
SELECT SUM( coefficient ) AS "total coefficient"  
  
FROM MATIERE
```

**7**

```
SELECT e.nom, e.prenom  
  
FROM etudiant e  
  
INNER JOIN note n ON e.numero_carte_etudiant = n.numero_carte_etudiant  
  
WHERE note_examen >10
```

**8**

```
SELECT m.nom_matiere, m.coefficient  
  
FROM matiere m  
  
INNER JOIN note n ON n.code_matiere = m.code_matiere  
  
INNER JOIN etudiant e ON n.numero_carte_etudiant = e.  
numero_carte_etudiant  
  
WHERE e.numero_carte_etudiant = '01234568'
```