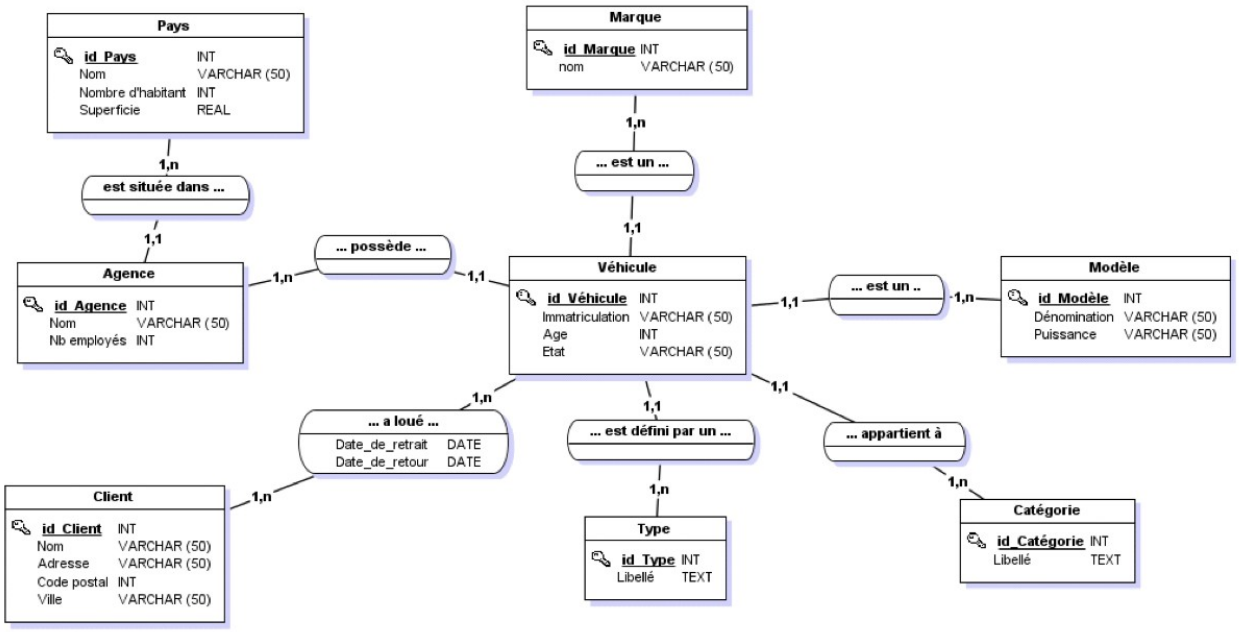
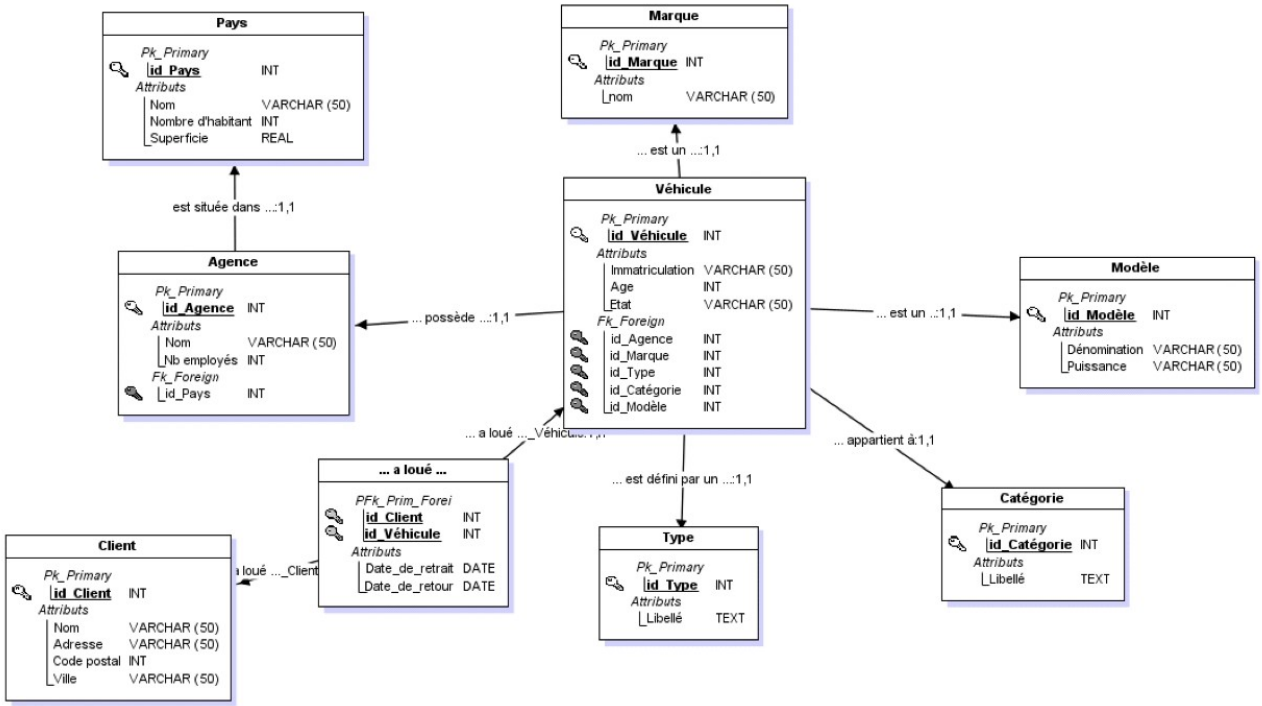
Raïhana Bouchelaghem

MISSION 3

MCD



MLD



Traduire le MLD Graphique en un représentation textuelle simplifiée d'une base de Données :

· Véhicule (**id_Vehicule**, immatriculation, age, etat, #id_Agence, #id_Marque, #id_Type, #id_categorie, #id_modele)

· Marque (**id_marque**, nom)

· Type (**id_type**, libelle)

· Catégorie (**id_categorie**, libelle)

· Modèle (**id_modele**, denomination, puissance)

· Agence (**id_Agence**, nom, nb_employes, #id_pays)

· Pays (**id_pays**, nom, nombre_habitant, superficie)

· a_loue (**id_client**, **id_vehicule**, date_de_retrait, date_de_retour)

· Client (**id_client**, nom, adresse, code_postal, ville)

Réalisez les requêtes suivantes :

• 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"

• 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"

• Afficher l'immatriculation, l'âge et l'état de tous les véhicules.
SELECT Immatriculation, Age, Etat**FROM** `vehicule`

• Afficher les noms des clients et les adresses, des clients qui habitent à << Nice >>.
SELECT `Nom` , `Adresse`**FROM** `client`**WHERE** `Ville` = "Nice"

• Affiche la liste des clients par ordre alphabétique croissant des noms
SELECT ***FROM** client**ORDER BY** nom **ASC**

• Ajouter l'attribut kilométrage et Afficher la liste des voitures par ordre décroissant des compteurs (kilométrage)
SELECT ***FROM vehicule**ORDER BY** `kilometrage` **DESC***

• 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"

• Afficher toutes les voitures noires :)
SELECT ***FROM** `vehicule`**WHERE** `couleur` = 'noir'

• Afficher toutes les voitures ayant un kilométrage <10000 km
SELECT ***FROM** `vehicule`**WHERE** `kilometrage` < 10000

• Afficher toutes les informations sur les locations réalisées avant 2018
SELECT ***FROM** a_loue**WHERE** `Date_de_retrait` < '2018-01-01';

• 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 pays.

SELECT p.nom, Nb_employes, 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 :

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

SELECT CodeEt, NomEt, DatnEt**FROM** ETUDIANT**ORDER BY** NomEt **ASC** ;

• **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** MATIER E**WHERE** NomMat = 'BD');

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

```
SELECT DISTINCT NomMat, CoefMatFROM MATIEREWHERE CodeMatIN (
SELECT CodeMatFROM ENSEIGNANTWHERE GradeEns = 'Grd3'
);
```

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

```
SELECT NomMat, CoefMatFROM MATIEREWHERE CodeMatIN (
SELECT CodeMatFROM NOTEWHERE CodeEt = 'Et321'
);
```

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

```
SELECT COUNT( * )FROM ENSEIGNANTWHERE CodeMat = (SELECT CodeMatFROM MATIERE
WHERE NomMat = 'Informatique' );
```

Exercice 3 :

Exprimez en SQL les requêtes suivantes :

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

```
SELECT C.NumeroCoureur, C.NomCoureur, PAYS.NomPaysFROM COUREUR CINNER JOIN EQUIPE E
ON C.CodeEquipe = E.CodeEquipeINNER JOIN PAYS ON C.CodePays = PAYS.CodePaysWHERE E.Nom
Equipe = 'Festina'
```

le nombre de kilomètres total du Tour de France 97 ?

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

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

les noms des coureurs qui n'ont pas obtenu de bonifications

```
SELECT c.NomCoureurFROM coureur cWHERE c.NumeroCoureur NOTIN (
```

```
SELECT a.NumeroCoureurFROM ATTRIBUER_BONIFICATION a
)
```

les noms des coureurs qui ont participé à toutes les étapes ?

```
SELECT c.NomCoureurFROM coureur cWHERE NOTEXISTS (
SELECT e.NumeroEtapFROM etape eWHERE e.NumeroEtap NOTIN (SELECT p.NumeroEtap
FROM participer pWHERE p.NumeroCoureur = c.NumeroCoureur)
)
```

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 TotalTe
mpsFROM coureur cINNER JOIN participer p ON c.NumeroCoureur = p.NumeroCoureurINNER
JOIN ATTRIBUER_BONIFICATION ab ON p.NumeroCoureur = ab.NumeroCoureurAND p.NumeroEtap = ab
.NumeroEtapINNER JOIN etape e ON e.NumeroEtap = p.NumeroEtapWHERE e.NumeroEtap <=13
GROUP BY c.NumeroCoureurORDER BY TotalTemps
```

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

```
SELECT NomEquipe, SUM( TempsRealise + NbSecondes ) AS TempsTotalFROM equipe eINNER
JOIN coureur c ON c.CodeEquipe = e.CodeEquipeINNER
JOIN participer p ON p.NumeroCoureur = c.NumeroCoureurLEFT JOIN ATTRIBUER_BONIFICATION ab O
N ab.NumeroCoureur = c.NumeroCoureurGROUP BY NomEquipeORDER BY TempsTotal
```

Exercice 4 :

LES COMMANDES:

```
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);
```

// Le premier qui m'envoie une tête de pierre en message privée discord, je lui donne ce qu'il veut =) c kdo

```
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));
```

la liste des clients de Marrakech.

```
SELECT Nomcli, Prenomcli, adressecliFROM ClientWHERE adressecli = 'Marrakech'
```

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

```
SELECT Numprod, designation, prixFROM ProduitORDER BY prix DESC
```

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

```
SELECT Nomvendeur, adresse_vendFROM VendeurWHERE Nomvendeur LIKE 'M%'
```

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

```
SELECT c.*FROM Commande cINNER JOIN Vendeur v ON c.Idvendeur = v.IdvendeurWHERE Nomvendeur = 'Mohammed'AND date_comBETWEEN '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
```

<u>Exercice 5 :</u>

// Le premier qui m'envoie un drapeau albanais en message privé discord, je lui donne ce qu'il veut =) c kdo (non-compatible si vous avez déjà envoyé une tête de pierre)

les commandes SQL permettant de rechercher :

La liste de tous les étudiants.

```
SELECT nom, prenomFROM etudiant
```

Nom et coefficient des matières.

```
SELECT nom_matiere, coefficientFROM matiere
```

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

```
SELECT numero_carte_etudiantFROM ETUDIANTWHERE numero_carte_etudiantIN (
SELECT numero_carte_etudiantFROM NOTEGROUP BY numero_carte_etudiantHAVING AVG( note_exam
en )BETWEEN 7AND 12
)
```

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

```
SELECT nom, prenom, numero_carte_etudiantFROM ETUDIANTWHERE nom LIKE 'ben%'
```

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

```
SELECT COUNT( * ) AS "nombre des étudiants qui ont comme matière 12518"FROM (
SELECT DISTINCT numero_carte_etudiantFROM NOTEWHERE code_matiere = '12518'
) AS students
```

La somme des coefficients des matières.

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

Les noms des étudiants qui une note_examen >10.

```
SELECT nom, prenomFROM etudiant eINNER
JOIN note n ON e.numero_carte_etudiant = n.numero_carte_etudiantWHERE note_examen >10
```

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

```
SELECT nom_matiere, coefficientFROM matiere mINNER JOIN note n ON n.code_matiere = n.code_matier
eINNER JOIN etudiant e ON n.numero_carte_etudiant = e.numero_carte_etudiantWHERE e.numero_carte_
etudiant = '01234568'
```

Traduire le MLD Graphique en un représentation textuelle simplifiée d'une base de Données :

· Véhicule (id_Vehicule, immatriculation, age, etat, #id_Agence, #id_Marque, #id_Type, #id_categorie, #id_modele)

· Marque (id_marque, nom)

· Type (id_type, libelle)

· Catégorie (id_categorie, libelle)

· Modèle (id_modele, denomination, puissance)

· Agence (id_Agence, nom, nb_employes, #id_pays)

· Pays (id_pays, nom, nombre_habitant, superficie)

· a_loue (id_client, id_vehicule, date_de_retrait, date_de_retour)

· Client (id_client, nom, adresse, code_postal, ville)

Réalisez les requêtes suivantes :

• Afficher toutes les informations sur les véhicules loués par le Client n°T122
SELECT v.*FROM `vehicule` vINNER JOIN a_loue al ON v.id_vehicule = al.id_vehiculeWHERE al.id_client = "T122"

• Afficher toutes les locations réalisées par le client n° T122

SELECT v.ImmatriculationFROM `vehicule` vINNER JOIN a_loue al ON v.id_vehicule = al.id_vehicule WHERE al.id_client = "T122"

• Afficher l'immatriculation, l'âge et l'état de tous les véhicules.
SELECT Immatriculation, Age, EtatFROM `vehicule`

• Afficher les noms des clients et les adresses, des clients qui habitent à << Nice >>.
SELECT `Nom` , `Adresse`FROM `client`WHERE `Ville` = "Nice"

• Affiche la liste des clients par ordre alphabétique croissant des noms
SELECT *FROM clientORDER BY nom ASC

• Ajouter l'attribut kilométrage et Afficher la liste des voitures par ordre décroissant des compteurs (kilométrage)
SELECT *FROM vehiculeORDER BY `kilometrage` DESC*

• Afficher les informations sur les clients qui ont loué la voiture EW 25EW

SELECT c . *FROM `client` cINNER JOIN a_loue al ON c.id_Client = al.id_clientINNER JOIN vehicule v ON al.id_vehicule = v.id_vehiculeWHERE v.immatriculation = "EW25EW"

• Afficher toutes les voitures noires :)
SELECT *FROM `vehicule`WHERE `couleur` = 'noir'

• Afficher toutes les voitures ayant un kilométrage <10000 km
SELECT *FROM `vehicule`WHERE `kilometrage` < 10000

• Afficher toutes les informations sur les locations réalisées avant 2018
SELECT *FROM a_loueWHERE `Date_de_retrait` < '2018-01-01';

• 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_loueWHERE `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_retourFROM vehicule vINNER JOIN a_loue al ON v.id_vehicule = al.id_vehiculeINNER JOIN agence a ON a.id_agence = v.id_agenceWHERE al.date_de_retrait = al.date_de_retour
```

• Obtenir le nombre véhicules pour chaque marque

```
SELECT m.nom, COUNT( * )FROM vehicule vINNER 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 vINNER JOIN marque m ON v.id_marque = m.id_marque INNER JOIN a_loue al ON v.id_vehicule = al.id_vehiculeINNER JOIN client c ON al.id_client = c.id_client WHERE m.nom = "Renault"GROUP BY m.nomHAVING COUNT( * ) >10
```

• Obtenir le nombre d'agences et d'employés par pays.

```
SELECT p.nom, Nb_employes, COUNT( a.id_agence ) AS "Nb agence"FROM agence a
INNER JOIN pays p ON a.id_pays = p.id_paysGROUP BY p.nom
```

Exercice 2 :

Ecrire les requêtes SQL permettant d'afficher :

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

```
SELECT CodeEt, NomEt, DatnEtFROM ETUDIANTORDER BY NomEt ASC ;
```

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

```
SELECT NomEns, GradeEnsFROM ENSEIGNANTWHERE CodeMat = (SELECT CodeMatFROM MATIERE WHERE NomMat = 'BD' );
```

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

```
SELECT DISTINCT NomMat, CoefMatFROM MATIEREWHERE CodeMatIN (
SELECT CodeMatFROM ENSEIGNANTWHERE GradeEns = 'Grd3'
);
```

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

```
SELECT NomMat, CoefMatFROM MATIEREWHERE CodeMatIN (
SELECT CodeMatFROM NOTEWHERE CodeEt = 'Et321'
);
```

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

```
SELECT COUNT( * )FROM ENSEIGNANTWHERE CodeMat = (SELECT CodeMatFROM MATIEREWHERE NomMat = 'Informatique' );
```

Exercice 3 :

Exprimez en SQL les requêtes suivantes :

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

```
SELECT C.NumeroCoureur, C.NomCoureur, PAYS.NomPaysFROM COUREUR CINNER JOIN EQUIPE E ON C.CodeEquipe = E.CodeEquipeINNER JOIN PAYS ON C.CodePays = PAYS.CodePaysWHERE E.Nom Equipe = 'Festina'
```

le nombre de kilomètres total du Tour de France 97 ?

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

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

les noms des coureurs qui n'ont pas obtenu de bonifications

```
SELECT c.NomCoureurFROM coureur cWHERE c.NumeroCoureur NOTIN (
SELECT a.NumeroCoureurFROM ATTRIBUER_BONIFICATION a
)
```

les noms des coureurs qui ont participé à toutes les étapes ?

```
SELECT c.NomCoureurFROM coureur cWHERE NOTEXISTS (
SELECT e.NumeroEtapFROM etape eWHERE e.NumeroEtap NOTIN (SELECT p.NumeroEtap
FROM participer pWHERE p.NumeroCoureur = c.NumeroCoureur)
)
```

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 TotalTe
mpsFROM coureur cINNER JOIN participer p ON c.NumeroCoureur = p.NumeroCoureurINNER
JOIN ATTRIBUER_BONIFICATION ab ON p.NumeroCoureur = ab.NumeroCoureurAND p.NumeroEtap = ab
.NumeroEtapINNER JOIN etape e ON e.NumeroEtap = p.NumeroEtapWHERE e.NumeroEtap <=13
GROUP BY c.NumeroCoureurORDER BY TotalTemps
```

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

```
SELECT NomEquipe, SUM( TempsRealise + NbSecondes ) AS TempsTotalFROM equipe eINNER
JOIN coureur c ON c.CodeEquipe = e.CodeEquipeINNER
JOIN participer p ON p.NumeroCoureur = c.NumeroCoureurLEFT JOIN ATTRIBUER_BONIFICATION ab O
N ab.NumeroCoureur = c.NumeroCoureurGROUP BY NomEquipeORDER BY TempsTotal
```

Exercice 4 :

LES COMMANDES:

```
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);
```

// Le premier qui m'envoie une tête de pierre en message privée discord, je lui donne ce qu'il veut =) c kdo

```
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));

la liste des clients de Marrakech.
SELECT Nomcli, Prenomcli, adresseclIFROM ClientWHERE adressecli = 'Marrakech'

la liste des produits (Numprod, désignation, prix) classés de plus cher au moins cher.
SELECT Numprod, designation, prixFROM ProduitORDER BY prix DESC

noms et adresses des vendeurs dont le nom commence par la lettre ‘M’.

SELECT Nomvendeur, adresse_vendFROM VendeurWHERE Nomvendeur LIKE 'M%'

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

SELECT c.*FROM Commande cINNER JOIN Vendeur v ON c.Idvendeur = v.IdvendeurWHERE Nomvendeu r = 'Mohammed'AND date_comBETWEEN '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 :

// Le premier qui m’envoie un drapeau albanais en message privé discord, je lui donne ce qu’il veut =) c kdo (non-compatible si vous avez déjà envoyé une tête de pierre)

les commandes SQL permettant de rechercher :

La liste de tous les étudiants.
SELECT nom, prenomFROM etudiant

Nom et coefficient des matières.

```
SELECT nom_matiere, coefficientFROM matiere
```

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

```
SELECT numero_carte_etudiantFROM ETUDIANTWHERE numero_carte_etudiantIN (
SELECT numero_carte_etudiantFROM NOTEGROUP BY numero_carte_etudiantHAVING AVG( note_exam
en )BETWEEN 7AND 12
)
```

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

```
SELECT nom, prenom, numero_carte_etudiantFROM ETUDIANTWHERE nom LIKE 'ben%'
```

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

```
SELECT COUNT( * ) AS "nombre des étudiants qui ont comme matière 12518"FROM (
SELECT DISTINCT numero_carte_etudiantFROM NOTEWHERE code_matiere = '12518'
) AS students
```

La somme des coefficients des matières.

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

Les noms des étudiants qui une note_examen >10.

```
SELECT nom, prenomFROM etudiant eINNER
JOIN note n ON e.numero_carte_etudiant = n.numero_carte_etudiantWHERE note_examen >10
```

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

```
SELECT nom_matiere, coefficientFROM matiere mINNER JOIN note n ON n.code_matiere = n.code_matier
eINNER JOIN etudiant e ON n.numero_carte_etudiant = e.numero_carte_etudiantWHERE e.numero_carte_
etudiant = '01234568'
```