Exercice 1

Une agence de voyage gère ses données grâce à une base de données formée des tables (ou relations) suivantes :

- `Vehicule(IdVehicule,Type,Compagnie)` : enregistre les véhicules disponibles - l'identifiant du véhicule, son type et sa compagnie.
- `Trajet(IdTrajet,VilleD,VilleA,IdVehicule)` : enregistre les trajets élémentaires possibles - l'identifiant du trajet, la ville de départ, la ville d'arrivée ainsi que le véhicule utilisé.
- `Ticket(IdTicket,IdTrajet,Place,Date,Heure,Prix)` : enregistre les tickets disponibles - l'identifiant du ticket, celui du trajet auquel ce ticket donne accès, le numéro de la place, la date, l'horaire et le prix.
- `Hotel(IdHotel,Classe,Ville)` : enregistre les hôtels connus - l'identifiant de l'hôtel, sa classe et sa ville.
- `Chambre(IdReservation,IdHotel,Date,Prix)` : enregistre les chambres d'hôtels qui sont disponibles - l'identifiant de réservation à utiliser, l'identifiant de l'hôtel où se trouve la chambre, la date et le prix.

1. donner les clefs étrangères de cette base de données

   On a 3 clefs étrangères : IdVehicule dans Trajetjet, IdTrajet dans Ticket et IdHotel dans chambre

2. écrire une requête SQL qui renvoie tous les trajets dont la ville de départ est Brest.

   select * from Trajet where VilleD='Brest'

3. écrire une requête qui renvoie le prix moyen de tous les tickets.

   select avg(Prix) from Ticket

4. écrire une requête qui renvoie le nombre de réservations de chambres d'hôtel à Bayonne le 07/07/2020.

   select count(IdReservation) from Chambre join Hotel on Chambre.IdHotel=Hotel.IdHotel where Hotel.Ville='Bayonne' and Cambre.Date=07/07/2020

   select count(IdReservation) from Chambre where Date=07/07/2020 and IdHotel in (select IdHotel from Hotel where Ville='Bayonne')

5. écrire une requête qui renvoie la liste formée des identifiants d'hôtel et de leur classe dans la ville d'arrivée de chaque trajet.

   select Trajet.IdTrajet,Hotel.idHotel,Hotel.Classe from Hotel join Trajet on Hotel.Ville=Trajet.VilleA group by IdTrajet

6. écrire une requête qui donne les différents identifiants de véhicules qui circulent le 07/07/2020.

   select Trajet.IdVehicule from Trajet join Ticket on Trajet.IdTrajet=Ticket.IdTrajet where Ticket.Date=07/07/2020

7. écrire une requête qui renvoie l'identifiant du véhicule le plus utilisé aujourd'hui. le plus = Max, utilisé =Nb de fois=count

   select IdVehicule,max(nb) from (select Trajet.IdVehicule, count(Trajet.IdTrajet) as nb from Trajet join Ticket on Trajet.IdTrajet=Ticket.IdTrajet where Ticket.Date=30/04/2020 group by Trajet.IdVehicule)

8. Que donne la requête SQL ci-dessous ?

```
SELECT * FROM Chambre WHERE Chambre.Prix = '100'
AND Chambre.IdHotel IN (SELECT Hotel.IdHotel
FROM Hotel JOIN Trajet ON Hotel.Ville = Trajet.VilleA JOIN Ticket
ON Trajet.IdTrajet = Ticket.IdTrajet WHERE Ticket.Prix = '50'
AND Trajet.VilleD = 'Brest')
```

   Donne les chambres de 100 euros qui sont accessible pour un trajet de 50 euros à partir de Brest

Exercice 2

Soit la BDD définie par le modèle logique suivant:

- ACTOR (id, name)
- MOVIE (id, title, year, score, votes, director)

  où director réfère à ACTOR(id)
- CASTING (movieid,actorid, ord)

  où movieid réfère à MOVIE(id) et actorid réfère à ACTOR(id)

1. Trouver les films dont le score est le plus élevé. Le plus = Max

   select * from MOVIE where score in (select max(score) from MOVIE)

2. Sortir la liste des films ayant comptabilisé le moins de votes mais dont le score est supérieur à 8.

   select * from MOVIE where score>8 and vote in (select min(vote) from MOVIE where score>8)

3. Trouver les réalisateurs qui comptabilisent les meilleurs scores moyens pour leurs films. Max(AVG)

   select name, max(moyscore) from (select ACTOR.name, AVG(MOVIE.score) as moyscore from MOVIE JOIN ACTOR on MOVIE.director=ACTOR.id group by ACTOR.id) un seul réalisateur

   select ACTOR.name, AVG(MOVIE.score) as moyscore from MOVIE JOIN ACTOR on MOVIE.director=ACTOR.id group by ACTOR.id where moyscore in (select max(moyscore) from (select AVG(MOVIE.score) as moyscore from MOVIE JOIN ACTOR on MOVIE.director=ACTOR.id group by ACTOR.id))

4. Pareil pour les acteurs.

   select name, max(moyscore) from (select ACTOR.name, AVG(MOVIE.score) as moyscore from MOVIE JOIN CASTING on MOVIE.id =CASTING.movieid JOIN ACTOR on CASTING.actorid=ACTOR.id group by ACTOR.id) Un seul acteur

   select ACTOR.name, AVG(MOVIE.score) as moyscore from MOVIE JOIN CASTING on MOVIE.id =CASTING.movieid JOIN ACTOR on CASTING.actorid=ACTOR.id group by ACTOR.id where moyscore in (select max(moyscore) from (select AVG(MOVIE.score) as moyscore from MOVIE JOIN CASTING on MOVIE.id =CASTING.movieid JOIN ACTOR on CASTING.actorid=ACTOR.id group by ACTOR.id))

5. Sortir le casting du film "Star Wars"

   select ACTOR.name from MOVIE JOIN CASTING on MOVIE.id =CASTING.movieid join ACTOR on CASTING.actorid=ACTOR.id where MOVIE.title='Star Wars'

6. Sortir la liste des films dont le réalisateur est aussi acteur.

   select MOVIE.* from MOVIE JOIN CASTING on MOVIE.id =CASTING.movieid where MOVIE.director=CASTING.actorid

Exercice 3

On a une base de donnée avec une seule classe

`enregistrement_etat_civil(prenom,nombre,sexe,annee)`

Cette base de données contient les prénoms des bébés nés à Paris depuis 1955. `nombre` donne le nombre de fois où un prénom est donné par sexe et par année.

1. Pour chacune de requêtes SQL suivantes, donner la traduction "en français".

   a. `SELECT DISTINCT prenom FROM enregistrement_etat_civil`

liste des prénoms donnés

b. `SELECT SUM(nombre) FROM enregistrement_etat_civil`

nombres de naissances enregistrées

c. `SELECT COUNT(DISTINCT prenom) FROM enregistrement_etat_civil`

Nombre de prénoms différents enregistrés

d. `SELECT annee,SUM(nombre) FROM enregistrement_etat_civil GROUP BY annee`

nombres de naissances enregistrées par année

e. `SELECT prenom,annee,SUM(nombre) as s FROM enregistrement_etat_civil GROUP BY prenom,annee HAVING s>=300`

Les prénoms qui ont été donnée plus de 300 fois par année

f. `SELECT prenom,SUM(nombre) as somme FROM enregistrement_etat_civil GROUP BY prenom HAVING somme>=2000`

Les prénoms qui ont été donnée plus de 2000 fois depuis 1955

2. Donner les commandes SQL qui

a. donne le nombre de naissances de filles qui ont été enregistrées.

`SELECT SUM(nombre) FROM enregistrement_etat_civil` where sexe='F'

b. donne le nombre de fois votre prénom a-t-il été donné à Paris pendant les années concernées.

`SELECT SUM(nombre) FROM enregistrement_etat_civil` where prenom='Gwénolé'

c. donne, pour chaque année, le nombre de prénoms différents qui ont été enregistrés.

`SELECT annee,COUNT(DISTINCT prenom) FROM enregistrement_etat_civil` group by annee

d. donne les prénoms qui ont été donnés exactement 100 fois.

`SELECT prenom,SUM(nombre) as somme FROM enregistrement_etat_civil GROUP BY prenom having sommme=100`

e. donne le prénom qui a été le plus donné sur l'ensemble de la période.

select prenom, max(somme) from (SELECT prenom,SUM(nombre) as somme FROM enregistrement_etat_civil GROUP BY prenom)

f. donne le prénom féminin qui a été le plus donné sur l'ensemble de la période ?

select prenom, max(somme) from (SELECT prenom,SUM(nombre) as somme FROM enregistrement_etat_civil GROUP BY prenom where sexe='F')

Exercice 4

La base de données d'une bibliothèque est constituée du modèle logique suivante:

- Auteur(CODE_AUTEUR, NOM_AUTEUR, PRENOM_AUTEUR)

  contenant la liste des auteurs.

- Livre(NO_LIVRE, ISBN, TITRE, EDITEUR, DATE_EDITION)

  contenant la liste des livres.

- Ecrit(CODE_AUTEUR, NO_LIVRE)

  associant à chaque auteur les livres qu'il a écrits.

- Exemplaire(NO_LIVRE, NO_EXEMPL, ETAT, DATE_ACQ)

  contenant la liste des exemplaires des livres de la bibliothèque.

- Emprunteur(NO_INSCR, NOM_EMPR, PRENOM_EMPR, ADR_EMPR)

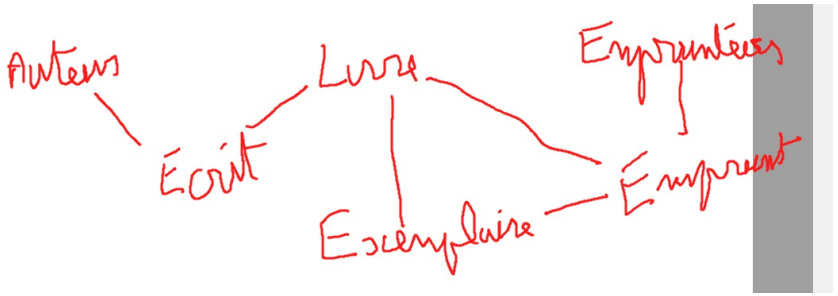
  contenant la liste des emprunteurs.

- Emprunt(NO_LIVRE, NO_EXEMPL, NO_INSCR, DATE_EMPR)

  contenant la liste des emprunts en cours.

1. En regardant ce modèle, donner l'ensemble des clefs primaires et des clefs étrangères avec leur référence.

2. Donner son modèle conceptuel.



3. Exprimer les requêtes suivantes en SQL.

   a. Rechercher le titre des livres écrits par Yves Fonctionrecurs.

      select Livre.TITRE from Livre join Ecrit on Livre.NO_LIVRE=Ecrit.NO_Livre join Auteur on Auteur.CODE_AUTEUR=Ecrit.CODE_AUTEUR where Auteur.PRENOM= 'Yves' and Auteur.Nom='Fonctionrecurs'

   b. Rechercher les numéro, nom et prénom des emprunteurs qui ont en prêt un exemplaire du livre numéro 10.

      select Emprunteur.NOM_EMPR,Emprunteur.PRENOM_EMPR from Emprunteur join Emprunt on Emprunt.NO_INSCR=Emprunteur.NO_INSCR where Emprunt.NO_LIVRE=10

   c. Rechercher le nombre d'emprunts en cours.

      select count(*) from Emprunt

   d. Pour chaque livre, rechercher son numéro, son titre et le nombre de ses exemplaires en cours de prêt.

      select Livre.NO_LIVRE,Livre.TITRE,count(NO_EXEMPL) from Livre join Emprunt on Livre.NO_LIVRE=Emprunt.NO_Livre group by Livre.NO_LIVRE

   e. Rechercher le numéro et le titre des livres de la bibliothèque édités depuis le 1 septembre 2014.

      select NO_LIVRE,TITRE from Livre where DATE_EDITION>01/09/2014

   f. Rechercher le nom, le prénom et le code des auteurs ayant écrit le plus de livres de la bibliothèque.

select * from Auteur where CODE_AUTEUR in (select CODE_AUTEUR,max(nb) from (select CODE_AUTEUR,count(NO_LIVRE) from Ecrit group by CODE_AUTEUR))

g. Rechercher le numéro et le titre des livres disponibles (c-à-d dont au moins un exemplaire n'est pas emprunté).

   select NO_LIVRE,TITRE from Livre where NO_LIVRE not in (select NO_LIVRE from Emprunt)

h. Rechercher le numéro et le titre des livres dont tous les exemplaires sont empruntés.

   select NO_LIVRE,TITRE from Livre where NO_LIVRE not in (select NO_LIVRE from Exemplaire where NO_EXEMPL not in (select NO_EXEMPL from Emprunt))

i. Rechercher le numéro et le titre des livres disponibles (c-à-d dont au moins un exemplaire n'est pas emprunté) édités après le 1 septembre 2010 dont le titre contient le mot "motivation".

   select NO_LIVRE,TITRE from Livre where NO_LIVRE not in (select NO_LIVRE from Exemplaire where NO_EXEMPL not in (select NO_EXEMPL from Emprunt)) and DATE_EDITION>01/09/2010 and TITRE like '%motivation%'