1) Adresses IP(v4) (2+ 1 + 2 = 5pts)

1.1) Soit une station st1 ayant l'adresse 172.27.115.24 /16
- Donner le masque de sous-réseau, l'adresse réseau et l'adresse de diffusion du réseau de st1.
- Donner le nombre d'hôtes maximal de ce réseau.

1.2) Soient deux stations st1 et st2 ayant respectivement les adresses suivantes:
@st1: 192.30.129.12 @st2: 192.30.129.24

1.2.1) On configure les stations avec le masque 255.255.255.0
- Déterminer les adresses réseau et diffusion du réseau.

1.2.2) Les stations sont maintenant configurées avec le masque 255.255.255.240.
- Déterminer les adresses réseau et diffusion du réseau de chaque station.
- Donner le nombre d'hôtes maximal de chaque sous-réseau.

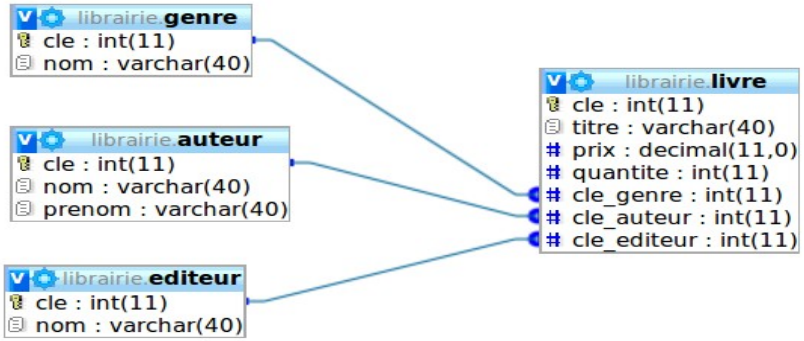
2) Requêtes SQL (5 pts)

2.1) Sélectionner le titre des livres de l'auteur "VIAN".

2.2) Sélectionner le prix de " L'arrache coeur " de "VIAN".

2.3) Sélectionner le nom du genre du livre "L'écume des jours" de l'auteur " VIAN " .

2.4) Déterminer les noms des différents éditeurs de "Les morts ont tous la même peau " de " VIAN ".



3) Javascript (5 pts) Voir annexe 1

3.1) Réaliser une fonction *Parite(objetMessage)*
Cette fonction retourne le nombre 1 si la valeur de l'objetMessage correspond à un nombre pair. Retourne le nombre 0 pour un nombre impair.

```
function parite(objetMessage)
{?}
```

3.2) Donner la modification à apporter à la fonction afin qu'elle affiche un message d'alerte si la valeur de l'objetMessage n'existe pas. Ex :" svp remplir le champ".

3.3) Modifier la fonction afin qu'elle affiche un message si la valeur ne correspond pas à un nombre. Ex" svp tapez un nombre".

3.4) Réalise la fonction *Parite2(objetMessage)* qui n'affiche plus rien mais retourne un nombre dépendant de la valeur de *objetMessage* :
0 si la valeur est impaire
1 si la valeur est paire
-1 si la valeur n'existe pas
-2 si la valeur ne correspond pas à un nombre

4) Trame Ethernet (5 pts) Voir annexe 2

A partir de la trame Ethernet ci-jointe, donner:
4.1) les adresses mac: destination et source en hexadécimal
4.2) les adresses IP destination et source en hexa puis en 4 nbres décimaux
4.3) les ports source et destination en hexa puis en décimal
4.4) l'état des bits SYN, ACK et FIN
4.5) Déduire à quoi correspond cette trame.

n°octet *TRAME ETHERNET*

```
0000  00 07 c0 ef c0 12 e1 1e aa 11 ac dc 08 00 45 00
0010  00 30 18 17 40 00 80 06 0e 5f a0 08 10 05 0a 0b
0020  40 19 04 06 00 50 63 9d d4 35 00 00 00 00 70 02
0030  40 00 31 f9 00 00 02 04 05 b4 01 01 04 02
```

Annexe 1

Exemple de code javascript dont on pourrait s'inspirer : Attention ce n'est pas la réponse !!

```
if (objetMessage.value){
   n=objetMessage.value ;
   if (isNaN(n) )      alert("ce n'est pas un nombre");
    else   {
        reste=n%2;
        if (reste==1)  alert(n+" est un nombre impair");
           else alert(n+" est nombre pair");
     }
}
```

Annexe 2

En-tête Ethernet :

6 octets	Adresse Physique (MAC) Destination
6 octets	Adresse Physique Emetteur
2 octets	Type protocole réseau

En-tête Ip (type de protocole 0X0800) :

1er mot de 32 bits :

4 bits	Version
4bits	Longueur en-tête en mots de 32 bits
8bits -3 bits -1bit -1bit -1bit - 2 bits	Type de service priorité (0 normal) délai (0 normal, 1 lent) Vitesse de transmission (0 normal, 1rapide) Sécurité (0 normal, 1 haute)
16 bits	Longueur totale du datagramme

2ème mot de 32 bits :

16 bits	Identification du datagramme
3 bits	Flags (gauche à droite) bit 0 toujours à 0 bit 1 1: fragmenté, 0 : non fragmenté bit 2 Dernier fragment (1 : Oui, 0 : Non)
13 bits	Offset du fragment

3ème mot de 32 bits :

8 bits	Temps de vie: TimeToLive
8 bits	Type de protocole
16 bits	Checksum en-tête

4ème mot de 32 bits :

32 bits	Adresse IP Emetteur

5ème mot de 32 bits :

32 bits	Adresse IP Destinataire

6ème mot de 32 bits (**éventuel** voir longueur dans 1er mot):

x bits	Options
y bits	Bourrage

En-tête TCP :Placé après un en-tête IP (type de protocole :0x06)

16 bits	Port source
16 bits	Port destination
32bits	Offset depuis début de la transmission
32 bits	Accusé de réception
4 bits	La taille de l'en-tête TCP en nombre de mots de 32 bits.
6 bits	Réservé (bits à 0)
6 bits	Bits de contrôle voir note ci-dessous
	Infos diverses, checksum, bourrage

Note:*Bits de contrôle (de gauche à droite):*

URG: Pointeur de données urgentes significatif
ACK: Accusé de réception significatif
PSH: Fonction Push
RST: Réinitialisation de la connexion
SYN: Synchronisation des numéros de séquence
FIN: Fin de transmission

Encapsulation des couches: