République Tunisienne	**Section** : S.I.	**Niveau** : *4ème Année*
Ministère de l'Education	***DS n° 3***	**Matière** : S.T.I.
CREs : Kebili – Médenine – Tataouine & Gabès	**Durée** : *3h*	**Date** : *14-05-2024 à 8h*

Nom & Prénom	…….	***Total points***	……/40
Classe	…….	***Note***	……/20

*Le sujet comporte **14** pages numérotées de **1/14** à **14/14** qui sont toutes à remettre à la fin de l'épreuve.*

Partie A (8 points)

Exercice n°1 (03 points)

Pour chacune des questions suivantes, une seule proposition est correcte. Mettre une croix **(X)** dans la case correspondante à la proposition correcte.

Important : Pour chaque question, toute réponse comportant plus d'une croix est considérée erronée.

1. En Html, l'événement qui se déclenche lors de la saisie du contenu d'une entrée de type texte est:

- ☐ Oninput
- ☐ Onfocus
- ☐ Onchange

2. En Html, la balise qui permet de définir une légende à une image ou à l'élément figure est :

- ☐ <legend>
- ☐ <figcaption>
- ☐ <figure>

3. En Html, la balise qui permet de spécifier une liste d'options prédéfinies connecté à une entrée de type liste est :

- ☐ <Option>
- ☐ <Select>
- ☐ <Datalist>

4. En Html, la balise qui permet de définir le titre d'une œuvre à savoir : le nom d'un livre, d'une chanson, d'un film…

- ☐ <title>
- ☐ <cite>
- ☐ <label>

5. En CSS, la règle qui permet d'appliquer un flou gaussien de rayon égale à 8 pixels sur une image est:

- ☐ transform: blur(8px);
- ☐ blur: 8px;
- ☐ filter: blur(8px);

6. En Javascript, l'instruction qui permet de convertir le nombre 2024 de la base 8 vers la base 10 est :

- ☐ parseInt("2024", 10)
- ☐ parseInt("2024", 8)
- ☐ parseInt("2024", 8, 10)

7. En Javascript, l'expression qui détermine le résultat de la division euclidienne d'un entier a par un entier b non nul est :

- ☐ Math.trunc(a/b)
- ☐ Math.round(a/b)
- ☐ a//b

8. En CSS, la règle à utiliser pour appliquer la couleur rouge aux textes des deux paragraphes identifiés respectivement par p1 et p2 est :

- ☐ #p1 , #p2 {color :red ;}
- ☐ #p1 || #p2 {color :red ;}
- ☐ #p1 && #p2 {color :red ;}

9. Soit l'élément Html : **<input type="text" id="n" class="cl" name="nom">**, pour récupérer, en PHP, la valeur saisie dans cette entrée on utilisera l'instruction :

- ☐ $_POST["n"];
- ☐ $_POST["nom"];
- ☐ $_POST["cl"];

10. En CSS, la règle à utiliser pour masquer toutes les images est :

- ☐ img {visibility: hidden}
- ☐ img {opacity : 1}
- ☐ Img{display : none}

11. L'élément Html définissant un lien hypertexte qui permet d'ouvrir la page "**AjoutEleve.Html**" dans l'iframe nommé "**if1**" est:

- ☐ Ajout
- ☐ Ajout
- ☐ Ajout

12. En CSS, la règle à utiliser pour fixer sur l'écran du navigateur un élément Html dont l'identifiant est "**elem**" est :

- ☐ #elem {position: absolute;}
- ☐ #elem {position: fixed;}
- ☐ #elem {position: relative;}

Exercice n°2 (03.50 points)

1- Dans un contexte de développement **WEB** et de **Bases de données**, compléter le tableau suivant par le terme convenable à chaque description :

Description	Terme
Une clause **SQL** utilisée pour indiquer une valeur par défaut.	F
Une fonction **PHP** qui permet d'effacer les espaces inutiles qui peuvent exister au début et/ou à la fin d'une chaine.	i
Un évènement qui se déclenche lorsqu'un élément perd le focus.	o u
Un attribut **Html** qui permet d'afficher ou de masquer un élément Html.	h d
Une propriété **CSS** qui permet de définir l'espacement intérieur d'un élément Html.	g

Description	Terme
Une fonction PHP qui permet d'appeler un fichier **PHP**.						r	
Une méthode d'envoi des données d'un formulaire vers une page **PHP**			
Une fonction **SQL** qui retourne la date et l'heure courante.			

2- Soit la figure suivante qui représente le principe de fonctionnement d'un site web dynamique :

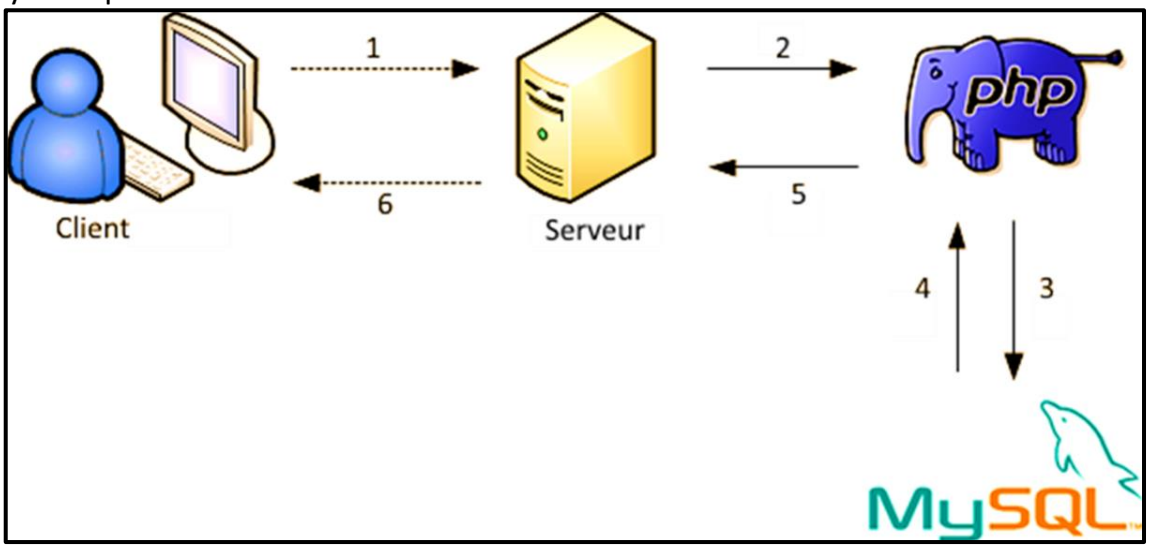


Figure 1: Principe de fonctionnement d'un site web dynamique

Compléter le tableau suivant en écrivant le numéro qui correspond à chacune des 6 étapes :

Description de l'étape	N° de l'étape
Envoi des données demandées	……
Réception d'une page web affichable	……
Demande d'exécution de code PHP	……
Demande d'une page web	……
Interrogation de la base de données	……
Envoi de la page web affichable	……

Exercice n°3 (01.50 points)

Soit la base de données simplifiée "**GestionLangues**", qui permet de gérer les langues parlées par les candidats d'un concours.

Candidat (**cin**, nom, prenom)
Langue (***idl***, libelle)
Parler (**cin#**, **idl#**)

Soit le formulaire "**Ajout.Html**" qui permet d'ajouter, à la base, les langues parlées par un candidat.

Figure 2 : le formulaire de la page "Ajout.Html"

Ci- après un extrait du code relatif aux composants du formulaire.

```
<fieldset><legend><h2>Ajout de langues : </h2></legend>
                <label for="cin" class="cl">CIN : </label>
        <input type="text" id="cin" name="cin"><br><br>
                <label class="cl">Langues parlées : </label>
        <input type="checkbox" name="Lang[]" id="ar" value="ar">
                <label for="ar">Arabe </label>
        <input type="checkbox" name="Lang[]" id="en" value="en">
                <label for="en">Anglais </label>
        <input type="checkbox" name="Lang[]" id="ch" value="ch">
                <label for="ch">Chinois </label>
        <input type="checkbox" name="Lang[]" id="esp" value="esp">
                <label for="esp">Espagnol</label><br><br>
        <input type ="reset" value="Annuler"> 
        <input type="submit" value="Ajouter">
</fieldset>
```

Le code Html relatif au fichier "Ajout.Html"

1) Le clic sur le bouton **Ajouter** fait appel à un script PHP "**Ajout.PHP**" qui permet d'ajouter la ou les langues parlées par un candidat à la base "**GestionLangues**".
L'appel du fichier du fichier "**Ajout.PHP**" se fait via l'élément Html :

☐	<form action="Ajout.PHP" onsubmit="return verif()">
☐	<form method="Ajout.PHP" action="POST" onsubmit="return verif()">
☐	<form method="Ajout.PHP" action="GET" onsubmit="return verif()">

2) Ci-après, les taches à exécuter par le fichier "**Ajout.PHP**"

```
< ?php
//Tache 1 : Récupération du N° Cin saisi.
//Tache 2 : Récupération des langues sélectionnées.
//Tache 3 : Affichage du nombre des langues sélectionnées.
//Tache 4 : Connexion au serveur et sélection de la base de données.
// Autres taches……
……
…..
//Tache 5 : Insertion des données récupérées dans la table Parler.
//Autres taches
……
?>
```

Des taches à exécuter par le fichier "Ajout.PHP"

a- Tache 1 :

L'instruction PHP qui permet la récupération du numéro de la carte d'identité nationale est :

☐	$cin=$_get["cin"]
☐	$cin=$_GET["cin"]
☐	$cin=$_GET("cin")

b- Tache 2 :

L'instruction PHP qui permet la récupération les langues sélectionnées est :

☐	$lang=$_GET["Lang"]
☐	If(isset($_GET["Lang"]) $Lang=$_GET["Lang”]
☐	$lang=isset($_GET["Lang"])

c- Tache 3 :

L'instruction PHP qui permet l'affichage du nombre des langues sélectionnées est :

☐	echo($Lang) ;
☐	echo (count($Lang))
☐	echo (Select count(*) from langue)

d- Tache 4 :

Sachant que les instructions relatives à la connexion au serveur et à la sélection de la base sont stockées dans le fichier intitulé "**Connexion.PHP**". L'instruction PHP qui permet d'importer le contenu de ce fichier est :

☐	Require("Connexion.PHP")
☐	Import (connexion.PHP)
☐	Require(Connexion.PHP)

e- Tache 5 :

L'instruction PHP qui permet l'insertion des données adéquates dans la table **Parler** est :

☐	for($i=0 ;$i<count($lang) ;$i++){ $x= $lang[$i]; mysql_query("insert into Parler values('$cin','$x')"); }
☐	mysql_query("insert into parle values('$cin','$lang')");
☐	for(i=0 ; i<count($lang) ; i =i+1){ $x= $lang[i]; mysql_query("insert into Parler values('$cin','$x')"); }

Partie B (17 points)

Exercice n°1 (5 points)

Soit la page "**Index.Html**" qui représente la page d'accueil d'un site web de paiement de factures.

```
<!doctype html>
<html lang="fr">
<head>
        <title>Gestion factures</title>
        <link rel=”stylesheet” type=”text/css” href=”styles.css”>
</head>
<body>
        <nav>
                <ul>
                        <li><a href="Abonner.html">Abonnement</a></li>
                        <li><a href="Editer.html">Etat de paiement</a></li>
                        <li><a href="Payer.html">Paiement d'une facture</a></li>
                </ul>
        </nav>
        <main>
                <iframe src="Editer.html" name="c">Hello</iframe>
        </main>
</body>
</html>
```

Code source de la page "lindex.Html"

1- On se propose d'ajouter un pied de page à cette page d'accueil.
Ecrire l'élément Html qui permet de définir ce pied de page et qui doit comporter **l'adresse URL** du site :**"https://www.Facturation.com.tn"**.

</main> <……>
<……> **"https://www.Facturation.com.tn" </**……>
</……> **</body>**

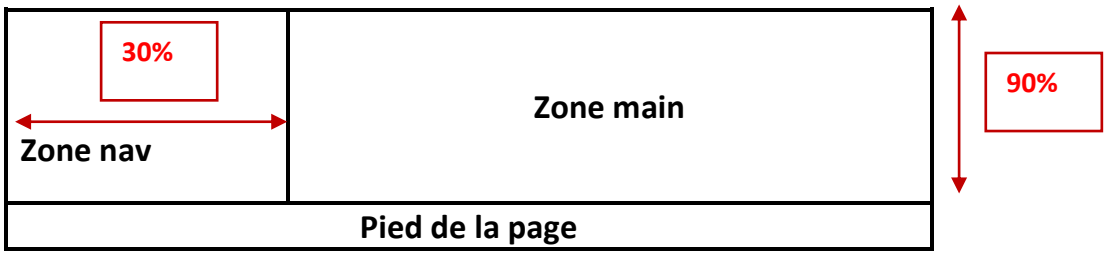
2- On se propose d'améliorer cette page d'accueil en ajoutant, dans le conteneur principal et après la fenêtre en ligne (<**iframe**>), une section dont l'identifiant est "**S1**" et contenant la séquence vidéo "**MyVideo.mp4**".

<main> <iframe src="Editer.Html" name="c"> </iframe>
<……, ……=……>
<video controls> <source src="MyVideo.mp4"> </video>
</……>
</main>

3- On se propose de modifier l'affichage de la page "**Abonner.Html**"pour qu'il soit dans l'élément **iframe**, indiquer l'attribut et sa valeur qu'il doit les ajouter dans l'élément relatif au lien hypertexte.

<a href="Abonner.Html" ……=
**……..>Abonnement
**

4- On se propose de positionner la zone "**nav**" à gauche, la zone "**main**" à droite, le **pied de la page** en bas et centrer l'adresse **URL** du site, comme l'illustre l'aperçu ci-après :

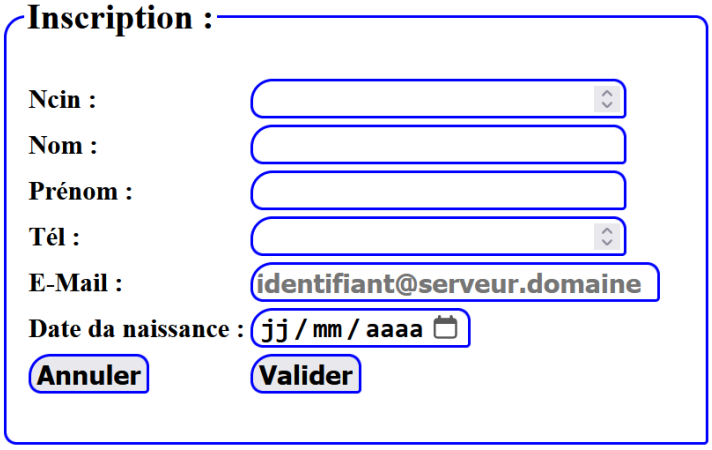


Compléter les règles CSS pour avoir la disposition CSS demandée

```
nav{……:30%; ……:90%; position:……; left:0;
top:0}
main{width:……; height:……; left:……; top:0;
position:……}
footer{width:100%; height:……; left:0; ……:0; position: absolute}
address {text-align: ……}
```

Exercice n°2 (07.50 points)

Le formulaire ci-après permet l'inscription d'un apprenant à un centre de formation à distance.



Formulaire d'inscription

Le code Html relatif à ce formulaire est :

```
<form action="inscription.php">
<fieldset><legend><h2>Inscription : </h2></legend>
        <label for="ncin">Ncin :</label> <input type="number" id="ncin" name="ncin">
        <label for="nom">Nom :</label> <input type="text" id="nom" name="nom">
        <label for="pr">Prénom :</label> <input type="text" id="pr" name="pr">
        <label for="tel">Tél :</label> <input type="number" id="tel" name="tel">
        <label for="mail">E-Mail :</label>
                <input type="email" id="mail" name="email" size="22"
                        placeholder="identifiant@serveur.domaine">
        <label for="dn">Date da naissance :</label> <input type="date" id="dn" name="dn"><br>
        <input type="reset" value="Annuler">
        <input type="submit" value="Valider">
        <p id="mess"></p>
</fieldset>
</form>
```

Code source de la page Inscription.Html

Le clic sur le bouton **Valider** fait appel à une fonction javascript intitulée **verifInscription()** et implémentée dans le fichier **"D:\Site\Controls.js"** qui permet de vérifier les données du formulaire avant de les envoyer vers le fichier "**Inscription.PHP**".

1. Compléter les deux éléments **script** et **form** par les paramètres adéquats.

```
<script src = ……..></script>
<form action="inscription.PHP" ……= ”return ……..”
method=”POST”>
```

2. Compléter la règle de mise en forme à appliquer à toutes les entrées (**input**) pour les encadrer par une bordure solide, de largeur 2 pixels, de couleur bleu et en appliquant des arrondissements de **15 pixels** au coin gauche supérieur, de **8 pixels** aux coin gauche inférieur et droite supérieure et de **3 pixels** au coin droite inférieur.

```
Input {border: ……. ;   border-……:
…….. ;}
#mess {color: red ; font-size: 20px; font-weight: bolder}
```

3. Soit la fonction intitulée **estFormeePar(ch, binf, bsup)** qui retourne **Vrai** lorsque tous les caractères de la chaine **ch** sont compris entre les deux caractères **binf** et **bsup** avec bornes incluses ou retourne **Faux** dans le cas contraire. Ci-après l'algorithme de cette fonction, compléter les pointillées pour avoir son implémentation en Javascript.

Déclaration de la fonction en Algorithmique

```
Fonction estFormeePar (ch :Chaine ; binf, bsup : Caractère) : Booléen
Début
        j←-1
        Répéter
                j←j+1
        Jusqu'à (ch[j]<binf) ou (ch[j]>bsup) ou (j=long(ch)-1)

        Retourner (ch[j] >=binf) et (ch[j]<=bsup)
Fin
```

Implémentation de la fonction en Javascript

```
function estFormeePar(ch, binf, bsup)
{ var j = -1
do { ……. }
while ( ch.charAt(j)>=binf and ch.charAt(j)<=bsup and j<ch.length-1)
return (…… and ……..……)
}
```

4. On s'intéresse au contenu de la fonction **verifInscription**()
   - **a-** Compléter la pointillée par la méthode convenable pour stocker dans la variable **n** l'équivalent majuscule de la valeur saisie dans la zone **nom**.

```
var n =document.getElementById("nom").value. ……..
```

   - **b-** Le nom doit avoir une longueur comprise entre **3** et **25** et il doit être alphabétique. En utilisant la fonction **estFormeePar(ch, binf, bsup)** compéter la pointillée par l'appel de la fonction afin d'afficher un message dans le cas d'erreur.

```
ok = true
if (n.length < 3 ||n.length > 25 || estFormeePar(……., ”A” , “Z” ) == false)
```

c- Dans le cas où la valeur saisie du nom est non valide, le massage "**Le *nom est non valide***" sera le contenu du paragraphe ayant comme identifiant "**mess**" et la variable booléen **ok** aura la valeur **faux**. Compléter les pointillées par la méthode adéquate.

```
document.getElementById("mess"). ……="Le nom est non valide"
ok=false
```

d- Le numéro du téléphone saisi doit avoir une longueur égale à 8, formé que par des chiffres et dont le premier chiffres est "2" ou "3" ou "4" ou "5" ou "7" ou "9". Compléter les pointillées par les informations adéquates.

```
var t =document.getElementById("tel").value
if (t. …… !=8 || estFormeePar(t, ……., …….) == false ||"0168".indexOf(t.charAt(…) )>=0)
      ok=false
```

e- L'administrateur du site a développé une fonction intitulée **verifMail(m)** qui retourne **Vrai** lorsque l'adresse mail **m** est valide ou **Faux** dans le cas contraire. Une adresse mail est dite valide lorsqu'elle respecte le format : **Identifiant@Serveur.domaine**, sachant que :

- l'**Identifiant** doit avoir une longueur minimale de 5 caractères,
- Le **serveur** est formé que par des lettres et de longueur minimale 5 lettres
- Le domaine est formé que par des lettres et de longueur minimale 2 lettres

Compléter les pointillés pour que la fonction **verifMail()** retourne **Vrai** lorsque l'adresse mail respecte les conditions citées précédemment ou la valeur **Faux** dans le cas contraire.

```
function verifMail(m)
{ //conversion en minuscule de la chaine m
m=m.to……
//Extraction de l'identifiant
 var id=m.substr(0, .....)
//Extraction du nom du serveur
 var srv=m.substr(m.indexOf("@")+1, m.lastIndexOf(".")-m.indexOf("@")-1)
//Extraction du nom de domaine
var dom=m.substr(……..)
 //Discussion pour calculer le résultat à retourner
if (id.length<….. ||srv. ……..<5 ||estFormeePar(srv, ……, …….)== ……
 || dom.length<2 ||estFormeePar(dom, ……., ……..,)==false)
      test=……..;  else test=true;  return test }
```

f- La valeur de la zone entrée dans le champ date de naissance doit:

- Être définie (non vide)
- Avoir une année de naissance qui dépasse l'an 2000.

Compléter les pointillés par les informations adéquates.

```
//Récupération de la valeur de la date de naissance.
var dn =document.getElementById("dn").value
// Création d'une variable datnaiss comportant la date de naissance déjà récupérée.
   Var datnaiss=new …… (dn)
//création d'une variable annNaiss comporatant l'année de naissance.
   Var annNaiss= datnaiss. ……..()
//discussion sur la valeur de la date de naissance saisie et l'année de naissance calculée
   if (dn. ……..==0 || annNaiss<=……)
```

Exercice n°3 (04.50 points)

On s'intéresse au fichier "**inscription.PHP**" qui consiste à récupérer les données en provenance du formulaire de la page "**inscription.Html**" (*voir Partie BI Exercice 2*) puis les insérer dans la table "**candidats**" de la base "**formation**". Sachant que la représentation textuelle de la table candidat est la suivante : **candidats** (**ncin**, nomprenom, tel, adrmail, age)

On suppose que la récupération des données est déjà faite comme la présente l'illustration suivante :

```
<?php
//récupération des données du formulaire
$ncin=$_POST["ncin"];
$nom=$_POST["nom"];
$pr=$_POST["pr"];
$tel=$_POST["tel"];
$email=$_POST["email"];
$dn=$_POST["dn"];
……
……
```

La récupération des données d'un candidat en PHP

1- Pour chacune des questions suivantes, compléter les pointillées par les informations adéquates :

a- Concaténer le nom et le prénom en les séparant par le caractère espace "␣"

```
$np=…….  "␣"  …….. ;
```

b- Calculer le nombre de secondes équivalent à la date actuelle

```
$nsdc=strtotime(……);
```

c- Calculer le nombre de secondes équivalent à la date de naissance

```
$nsdn=strtotime(……..);
```

d- Calculer l'âge en convertissant la différence des deux nombres de secondes calculés en un nombre d'années puis retrancher l'année 1970.

```
$age=date("……..",$nsdc-$nsdn)-1970;
```

e- Se connecter au serveur et sélectionner la base via l'appel du fichier "**cnxion.PHP**".

```
……("cnxion.PHP") ;
```

f- Chercher les candidats dont le champ **ncin**= **$ncin** et stocker le résultat dans **$r1**.

```
$r1=……(" select * from candidats
where'$ncin'=ncin");
```

g- Afficher le message "*candidat déjà inscrit*" lorsqu'il existe un candidat dans la base ayant une valeur égale à **$ncin**.

```
if (mysql_num_rows(……)==……) echo('Candidat déjà inscrit');
```

h- Insérer le candidat, dont les données sont reçues à partir du formulaire, dans la table appropriée **et** afficher le message de confirmation "*inscription faite avec succès*" **ou** afficher le message "*Echec d'exécution de la requête*" dans le cas contraire.

```
mysql_query("insert ……. candidat …….('$ncin','$np','$tel','$email','$age')")
…… die("inscription faite avec succès") or die ("Echec d'exécution de la
requête");
```

2- L'administrateur de la base de données à commis erreur lors de la conception, en effet il existe un problème d'intégrité de données dans la table candidats car le champ **age** est un

champ qui doit varier au cours du temps alors que son contenu et statique (fixe) ; par conséquent les données perdent leurs fiabilités et crédibilités. Pour ceci il a décidé de remplacer la colonne **age** par la colonne **datenaissance**.

Ecrire les requêtes nécessaires pour effectuer ces modifications à la structure de la table candidats.

N.B. : Le contenu du champ **age** n'est pas utile par conséquent il n'y aura aucun problème causé par la perte de ce contenu.

1- ……
……
2- ……
……

Partie C (15 points)

Exercice n°1 (05.50 points)

Soit la base de données **"MaisonEdition"** qui sert à gérer les abonnements aux revues électroniques publiées par une maison d'édition. Cette base est décrite par la représentation textuelle suivante :

ABONNE (**Cin**, NomPr, Tel, Adresse, Email)
REVUE (**Code**, Titre, PrixMens)
ABONNEMENT (**Cin #, Code #,** DateAbon, Duree)

La description des différents champs est présentée dans le tableau suivant :

Champ	Description	Type et taille
Cin	N° de la Cin d'un abonné.	Chaine de 8 caractères.
NomPr	Nom et prénom d'un abonné.	Chaine de 30 caractères.
Tel	N° du téléphone d'un abonné	Chaine de 8 caractères.
Adresse	Adresse d'un abonné.	Chaine de 50 caractères.
Email	Adresse mail d'un abonné.	Chaine de 40 caractères.
Code	Code d'une revue.	Entier incrémenté automatiquement.
Titre	Titre d'une revue.	Chaine de 30 caractères.
PrixMens	Prix mensuel à une revue.	Entier > 5.
DateAbon	Date d'abonnement d'une revue	Date.
Duree	Durée d'abonnement d'une	Entier > 0.

1) Lors de la saisie des abonnements, l'administrateur de la base a remarqué qu'il a commis une erreur lors de la définition de la clé primaire de la table **ABONNEMENT**.
Ecrire les deux requêtes **SQL** qui permettent de modifier la clé primaire de la table **ABONNEMENT** pour qu'elle soit formée par les trois champs : **Cin**, **Code** et **DateAbon**.
……
……

2) Vue l'existence des deux colonnes **Email** et **Tel**, l'administrateur se propose d'éliminer le colonne **Adresse**.
Ecrire une requête **SQL** qui permet de supprimer le champ **Adresse** de la table **ABONNE**.
……
……

3) L'administrateur se propose de définir une contrainte pour garantir la non-redondance des numéros téléphoniques.
Ecrire une requête **SQL** qui permet de définir la contrainte d'unicité sur la colonne **Tel**.

……
……

4) Puisqu'il existe une grande variété de revues, l'administrateur se propose d'améliorer la base en ajoutant la table **TYPE**, ayant la représentation textuelle suivante : **Type**(**CodeType**, Libelle).
Sachant que la colonne **CodeType** est une lettre majuscule et le champ **Libelle** est une chaine non nulle de taille maximale égale à 50.
Ecrire une requête **SQL** qui permet de créer la table **TYPE**.

……
……
……
……
……
……

5) Puisque chaque revue doit avoir un type, l'administrateur doit ajouter un champ à la table **REVUE** qui renseigne le type.
Ecrire une requête **SQL** qui permet d'ajouter le champ **CodeType** à la table **REVUE**.

……
……

6) Pour achever cette modification dans la structure de la base, le champ **CodeType** de la table **REVUE** doit être une clé étrangère reliée avec la clé primaire **CodeType** de la table **TYPE**.
Ecrire une requête **SQL** qui permet de définir une contrainte d'intégrité référentielle entre les deux tables **TYPE** et **REVUE**.

……
……
……
……

Exercice n°2 (07.50 points)

La représentation textuelle finale, après amélioration, de la base de données **"Maison_Edition"** est : **ABONNE** (**Cin**, NomPr, Tel, Email)
REVUE (**Code**, Titre, PrixMens, CodeType#)
TYPE(**CodeType**, Libelle)
ABONNEMENT (**Cin #, Code #, DateAbon**, Duree)

Ecrire la (les) requête(s) **SQL** pour chacune des taches suivantes :.

1- Augmenter de **10%** le prix d'abonnement mensuel des revues scientifiques (**CodeType="S"**)

……
……

2- Afficher les abonnés **(Cin, NomPr**) ayant une ligne téléphonique de l'opérateur **Ooredoo** (dont le 1er chiffre est **"2"**)

……
……

3- Afficher les revues ordonnées en sens décroissant de leurs prix mensuels.

……
……
……
……

4- Afficher pour chaque type de revue son nombre d'abonnements.

……
……
……
……
……

5- Afficher le montant total des abonnements pour toutes les abonnements trimestrielles (**Durée=3 mois**).

……
……
……
……
……

6- Afficher, pour chaque année et pour chaque type de revue, le nombre d'abonnements.

……
……
……
……
……
……
……

7- Afficher les clients qui ont souscrit à plus de deux abonnements.

……
……
……
……
……
……
……

8- L'administrateur ajoute une nouvelle revue à la base. Sachant que :

- le titre de cette nouvelle revue est "**Revue médicale du sud"**, son prix d'abonnement mensuel est de **25 dinars** et la référence de son type est "**M**".
- Le type "**M**" existe déjà dans le champ **RefType** de la table **TYPE**.

Ajouter cette nouvelle revue à la table appropriée.

……
……
……

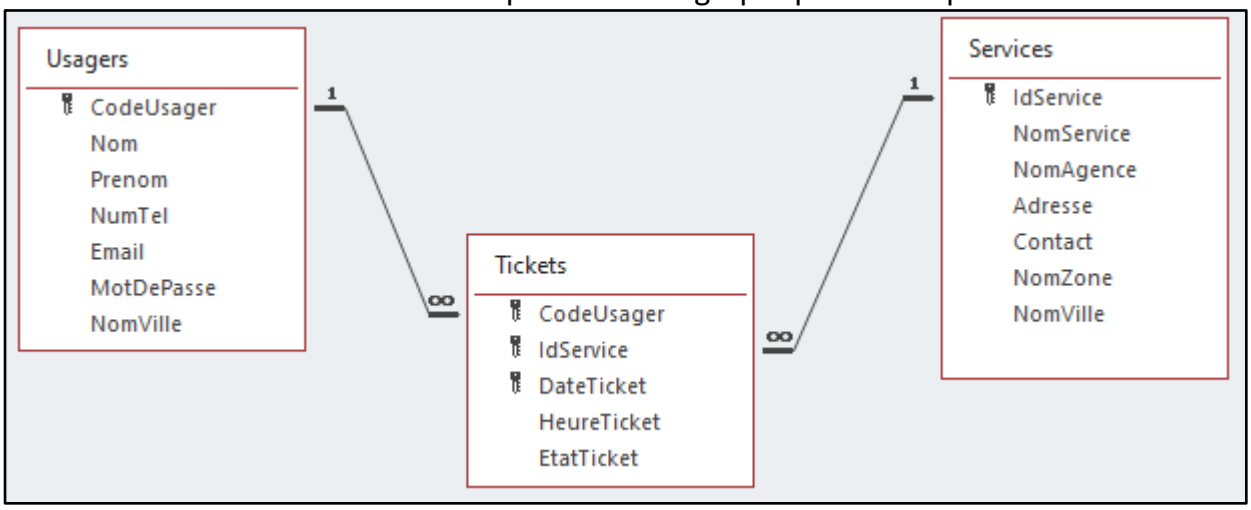
9- Lister toutes les revues (**code, titre, prix mensuel et libellé**) qui ne sont pas demandées par les abonnés (**ayant un nombre d'abonnement null**).

……
……
……
……
……
……
……

Exercice n°3 (02 points)

Pour assurer le confort et la sécurité des usagers ainsi que du personnel, l'administration d'une entreprise publique désire réduire le temps d'attente dans ses différentes agences qui existent dans plusieurs zones d'une même ville. Pour ceci l'administration propose un service de prise de rendez-vous en ligne via des tickets afin de permettre à ses usagers de s'épargner de longues heures d'attente et de réduire au maximum les foules.

L'équipe informatique responsable à la conception de la base de données a proposé la base de données "**GestionTickets**" dont la représentation graphique est ci-après:



Représentation graphique de la base de données "GestionTickets"

On propose les quatre règles de gestion suivantes :

- **R1** : Un usager peut avoir plusieurs rendez-vous chaque jour.
- **R2** : Un usager peut avoir un seul rendez-vous pour un service spécifique.
- **R3** : Un ticket est valide pour un seul service dans les différentes agences d'une ville.
- **R4** : Chaque client peut accéder seulement aux agences de sa ville originaire.

En se référant à la représentation graphique précédente et pour chaque règle de gestion cocher la case de la colonne **Respectée** si elle est respectée ou la case à cocher de la colonne **Non respectée** dans le cas contraire.

Règle	Respectée	Non respectée
R1	☐	☐
R2	☐	☐
R3	☐	☐
R4	☐	☐