

MISSION MARATHON



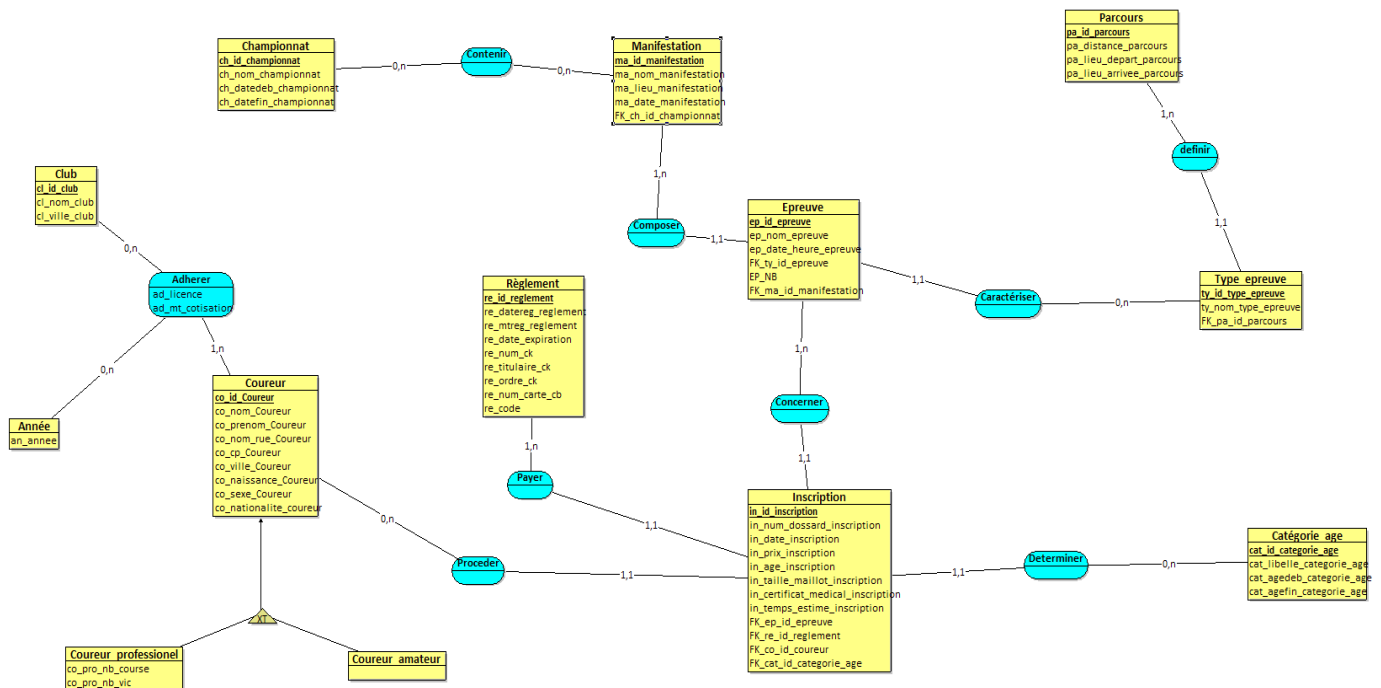
PARTIE 1

Contexte :

Le contexte Marathon s'inscrit dans le contexte M2L, La Maison des Ligues de Lorraine, qui a pour mission de fournir des espaces et des services aux différentes ligues sportives régionales et à d'autres structures hébergées. La M2L est une structure financée par le Conseil Régional de Lorraine dont l'administration est déléguée au Comité Régional Olympique et Sportif de Lorraine (CROSL).

Déroulé des Mission 2 à 4 :

1/ Voici le MCD de la mission marathon :



2/ Voici les contraintes du cahier des charges :

- Un coureur ne peut participer qu'à une seule épreuve dans le cadre d'une manifestation.
- Les enfants de moins de 10 ans ne peuvent pas s'inscrire à ce type d'épreuves.
- La catégorie d'âge d'un coureur est déterminée à partir de son âge le jour d'une épreuve.
- Chaque coureur ne peut participer que s'il est adhérent dans un club
- Il y a pour chaque épreuve, un classement suivant le sexe et la catégorie d'âge des coureurs. Ces derniers sont classés du plus rapide au plus lent.

3/ Voici le MLD de la mission marathon :

```
create database marathon;
```

```
use marathon;
```

```
create table adherer(fk_cl_id, fk_an_annee, fk_co_id, ad_licence int(10),  
ad_cotisation int(5));
```

```
create table championnat(ch_id int not null primary key, ch_datedeb date,  
ch_datefin date, ch_nom varchar(50));
```

```
create table annee(an_annee int not null primary key);
```

```
create table categorie_age(cat_id int not null primary key, cat_libelle varchar(50),  
cat_agedeb int(5), cat_agefin(5));
```

```
create table club(cl_id int not null primary key, cl_nom varchar(50), cl_ville  
varchar(50));
```

```
create table coureur(co_id int not null primary key, co_prenom varchar(50),  
co_nom varchar(50), co_nom_rue_coureur varchar(50), co_cp int(10), co_ville  
varchar(50), co_naissane date, co_sexe varchar(50), co_nationalite  
varchar(50));
```

```
create table epreuve(ep_id int not null primary key, ep_nom varchar(50),  
ep_dateheure datetime, fk_man_id, fk_typ_id);
```

```
create table manifestation(ma_id int not null primary key, ma_nom varchar(50),  
ma_lieu varchar(50), ma_date date, fk_ch_id);
```

```
create table type_epreuve(typ_id int not null primary key, typ_nom varchar(50),  
fk_par_id);
```

```
create table parcours(par_id int not null primary key, par_distance int(5),  
par_lieudeb varchar(50), par_lieufin varchar(50));
```

```
create table inscription(in_id int not null primary key, in_numdos int(10),  
in_date date, in_prix int(10), in_age int(5), in_maillot varchar(50), in_certifmed  
varchar(50), in_tpsestim time, in_tpsfin time, fk_ep_id, fk_reg_id, fk_co_id,  
fk_cat_id);
```

```
create table reglement(reg_id int not null primary key, reg_date date,  
reg_montant int(5), reg_numchk int(30), reg_numcb int(30),  
reg_date_expiration_cb date, reg_titulaire_chk varchar(50), reg_ordre_chk  
varchar(50), reg_code varchar(50));
```

Déroulé de la Mission 5 :

1/ Voici un scripts de chargement (de la table championnat) :

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.2969 seconde(s).)

```
create table championnat(ch_id int not null primary key, ch_datedeb date, ch_datefin date, ch_nom varchar(50))
```

2/ Spécificité de la base :

- La table année doit contenir les années 2015 à l'année en cours.
- La date de naissance des coureurs doit être comprise entre 1925 et l'année en cours et doit contenir au moins un homme et une femme sur chaque date de naissance.
- Pour faciliter la cohérence des futures requêtes, le sexe sera de type caractère, une position de valeur M ou F
- Faire en sorte d'avoir une bonne mixité des coureurs
- La première lettre du nom des sportifs doit balayer l'alphabet
- Vous devez avoir au moins deux clubs sportifs sur la ville de Bordeaux
- Les années d'adhésion à un club doivent être comprises entre 2014 et l'année en cours
- Vous devez renseigner le temps annoncé et le temps effectué pour chaque inscription réalisée

3/ Structure de la base :

Table	Action	Lignes	Type	Interclassement	Taille	Perte
<input type="checkbox"/> adherer	★	95	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> annee	★	6	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> categorie_age	★	7	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> championnat	★	5	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> club	★	7	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> coureur	★	90	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> epreuve	★	18	InnoDB	utf8mb4_0900_ai_ci	48,0 kio	-
<input type="checkbox"/> inscription	★	90	InnoDB	utf8mb4_0900_ai_ci	80,0 kio	-
<input type="checkbox"/> manifestation	★	13	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
<input type="checkbox"/> parcours	★	3	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> reglement	★	8	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
<input type="checkbox"/> type_epreuve	★	6	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
12 tables	Somme	348	MyISAM	utf8mb4_0900_ai_ci	320,0 kio	0 o

Déroulé des Missions 6 à 9 :

Durant les missions 6 à 9 on étudie :

- requête avancée
- jointure
- agrégation
- requête imbriquée

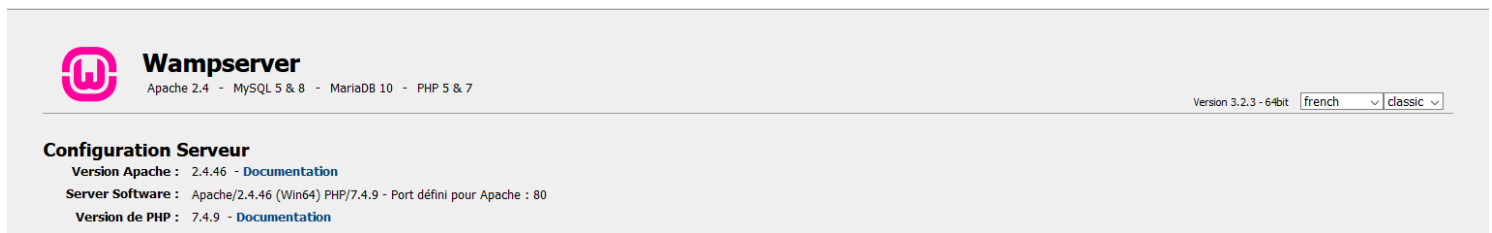
D'un point de vue professionnel le dérouler de ses missions m'ont appris à réfléchir méthodiquement. Car pour faire les requêtes SQL il faut savoir trier les données dans le bon ordre, et donc faire attention à ne pas se perdre, car les requêtes peuvent facilement prendre quelques lignes.

Déroulé de la Mission 10 (créer un formulaire) :

Utilitaires :

Pour se faire on vas devoir on vas devoir faire fonctionner des scripts PHP en local, on vas donc utiliser une plateforme de développement Web, **WampServer**.

1/ Voici la config de mon WampServer :



The screenshot shows the WampServer configuration interface. At the top left is the WampServer logo and the text 'Wampserver'. Below it, it lists the installed components: 'Apache 2.4 - MySQL 5 & 8 - MariaDB 10 - PHP 5 & 7'. On the top right, it shows 'Version 3.2.3 - 64-bit' and two dropdown menus for 'french' and 'classic'. Below this is the 'Configuration Serveur' section, which includes links for 'Version Apache : 2.4.46 - Documentation', 'Server Software : Apache/2.4.46 (Win64) PHP/7.4.9 - Port défini pour Apache : 80', and 'Version de PHP : 7.4.9 - Documentation'.

2/ Les avantages de WAMP pour une entreprise :

- on peut ajouter des addons
 - possibilité d'avoir plusieurs versions de PHP, MySQL, ...
- on peut modifier la config de PHP, MySQL, ...

3/ Voici la config de mon PHPmyAdmin:



The screenshot shows the PHPmyAdmin configuration page. It is divided into three main sections: 'Serveur de base de données', 'Serveur Web', and 'phpMyAdmin'. The 'Serveur de base de données' section lists: 'Serveur : MySQL (127.0.0.1 via TCP/IP)', 'Type de serveur : MySQL', 'Connexion au serveur : SSL n'est pas utilisé', 'Version du serveur : 8.0.21 - MySQL Community Server - GPL', 'Version du protocole : 10', 'Utilisateur : root@localhost', and 'Jeu de caractères du serveur : UTF-8 Unicode (utf8mb4)'. The 'Serveur Web' section lists: 'Apache/2.4.46 (Win64) PHP/7.4.9', 'Version du client de base de données : libmysql - mysqlnd 7.4.9', 'Extension PHP : mysqli, curl, mbstring', and 'Version de PHP : 7.4.9'. The 'phpMyAdmin' section lists: 'Version : 5.0.2, dernière version stable : 5.1.0', 'Documentation', 'Site officiel', 'Contribuer', 'Obtenir de l'aide', 'Liste des changements', and 'Licence'.

4/ Les avantages de PHPmyAdmin pour une entreprise :

L'avantage majeur de PHPmyAdmin est que l'on va pouvoir traiter, opérer les données rapidement (sans utilisé forcément des commandes).

Formulaire :

PARTIE INSCRIPTION

1/ Pour commencer, on va créer le formulaire. Pour ce faire, on va utiliser la balise HTML **<form>** et toutes ses propriétés.

```
<html>
<body>
  <section id="section_form">
    <form method="post" action="recapinscription.php">
      <label>Nom :</label>
      <br><input type="text" name="nom" required>
      <label>Prenom :</label>
      <br><input type="text" name="prenom" required>
      <label>Date de naissance :</label>
      <br><input type="date" name="date_naissance" required>
      <label>Genre :</label>
      <br><input name="genre" type="radio" value="M" required/><label for="M"> Homme</label>
      <br><input name="genre" type="radio" value="F" required/><label for="F"> Femme</label>
      <label>Adresse :</label>
      <br><input type="text" name="adresse" required>
      <label>Code Postal :</label>
      <br><input type="number" name="code_postal" required/>
      <label>Ville :</label>
      <br><input type="text" name="ville" required>
      <label>Taille de Maillots :</label>
      <br>
      <select name="taille_maillots" required>
        <option value="">Taille de maillot</option>
        <option value="S">S</option>
        <option value="M">M</option>
        <option value="L">L</option>
        <option value="XL">XL</option>
        <option value="XXL">XXL</option>
      </select>
      <label>Certificat médical :</label>
      <br><input name="certificat_medical" value="oui" type="radio" required/><label for="oui"> Oui</label>
      <br><input name="certificat_medical" value="non" type="radio" required/><label for="non"> Non</label>
      <label>Temps estime :</label>
      <br><input type="time" name="temps_estime" required/>
      <label>Nationalité :</label>
      <br><input type="text" name="nationalite" required>
```

```

<label>Categorie d'age :</label>
<br>
<select name="categorie_age" required>
  <option value="">Categorie d'age :</option>
  <option value="1">poussin</option>
  <option value="2">junior</option>
  <option value="3">ado</option>
  <option value="4">jeune</option>
  <option value="5">adulte</option>
  <option value="6">confirmé</option>
  <option value="7">senior</option>
</select>
<label>Epreuve :</label>
<br>
<select name="epreuve" required>
  <option value="">Epreuve :</option>
  <option value="1">Arkétop</option>
  <option value="2">Aventure</option>
  <option value="3">Suprême sensation</option>
  <option value="4">Coule douce</option>
  <option value="5">Admirer le paysage</option>
  <option value="6">Verdoyants parcours</option>
  <option value="7">Pour les mordues</option>
  <option value="8">Vive demain</option>
  <option value="9">La perle de la montagne</option>
  <option value="10">Tous en plaine</option>
  <option value="11">Parcours aventure</option>
  <option value="12">Le chemin des petits</option>
  <option value="13">Warriors</option>
  <option value="14">Baby course</option>
  <option value="15">Upgrade</option>
  <option value="16">Toujours plus loin</option>
  <option value="17">Les 5 bornes</option>
</select>
<br>
<input type="submit" name="btn" value="Inscription"/>
</form>
</section>
</body>
</html>

```

On retrouve beaucoup l'attribut **required**, qui permet de rendre obligatoire le remplissage d'un champ (ici tout les champs sont obligatoire).

L'attribut **action** permet d'envoyer les données traité dans le **<form>**, vers la page souhaité (ici : test 1.php).

On retrouve aussi beaucoup l'attribut **type**, qui permet de déterminer quel type de donnée y seras renseigné.

Pour finir, l'attribut **name**, permet de référencer l'élément inscrits pour pouvoir l'utiliser pour l'insérer dans une base de donnée.

2/ On va maintenant faire la connexion à la base de donnée, avec le langage PHP.

```
//connexion bd
date_default_timezone_set('Europe/Paris');
$server='127.0.0.1:3309';
$username='root';
$password='';
$dbname = "marathon1";

$mysqli = mysqli_connect($server,$username,$password,$dbname);
$mysqli->set_charset("utf-8");
//message d'erreur si la connexion fonctionne pas
$connect = mysqli_connect($server, $username, $password, $dbname);
if(!$connect){
    /*Erreur : impossible de se connecter à sql*/
    exit;
} else {
    echo "tu es co";
}
```

On peut remplacer à la ligne 3, l'IP et le port (IP::PORT) par localhost (si votre base de donnée est en local).

La ligne 8 définit le jeu de caractères du client MySQL.

De la ligne 10 à 16 on as un **if** qui permet de savoir si oui ou non on est connecté à la base de donnée.

3/ Maintenant on va s'assurer qu'une fois que l'utilisateur à rentré toutes les données dans le form, après appuie sur le bouton « Inscription » les données sont insérés dans la base de donnée. A l'aide de PHP.

```
//traitement de la requête
if(isset($_POST['nom'])){ //détecte si le nom est remplie

    // variables
    $nom = $_POST['nom'];
    $prenom = $_POST['prenom'];
    $date = $_POST['date_naissance'];
    $sexe = $_POST['genre'];
    $adresse = $_POST['adresse'];
    $code_postal = $_POST['code_postal'];
    $ville = $_POST['ville'];
    $taille = $_POST['taille_maillots'];
    $certif = $_POST['certificat_medical'];
    $temps = $_POST['temps_estime'];
    $nationalite = $_POST['nationalite'];
    $date = NULL;
    $montant = NULL;
    $numchk = NULL;
    $numcb = NULL;
    $date_expiration_cb = NULL;
    $titulaire_chk = NULL;
    $ordre_chk = NULL;
    $code = NULL;
    $numero_dossard = NULL;
    $date_inscription = NULL;
    $prix = NULL;
    $age = NULL;
    $tpsfin = NULL;
```

À la ligne 2 on retrouve la fonction **isset()**, qui permet de déterminé si une variable est déclarer. Ici si la valeur 'nom' est déclaré, on peut ainsi exécuter les valeurs du dessous.

De la ligne 5 à 28 on attribut une variable au **name** définit dans le **<form>**. Pour ensuite, à la ligne 14 insérer ses variables dans la base de donnée.






















On définit certaines variable NULL car les données y seront renseigné dans la page reglement

4/ Pour pouvoir insérer des données dans la table inscription, on a besoin de renseigner 4 clés étrangères (sans ça l'insertion ne peut fonctionner). On va donc pour commencer aller chercher la clé étrangère :

« catégorie_age »

```
// récupérer l'id de categorie age
$fk_cat_id = 0;
$cat_age = '';
if (isset($_POST['categorie_age'])) {
    $choix1 = $_POST['categorie_age'];
    if ($choix1==1) {
        $fk_cat_id = 1;
        $cat_age = 'poussin';
    }
    elseif ($choix1==2) {
        $fk_cat_id = 2;
        $cat_age = 'junior';
    }
    elseif ($choix1==3) {
        $fk_cat_id = 3;
        $cat_age = 'ado';
    }
    elseif ($choix1==4) {
        $fk_cat_id = 4;
        $cat_age = 'jeune';
    }
    elseif ($choix1==5) {
        $fk_cat_id = 5;
        $cat_age = 'adulte';
    }
    elseif ($choix1==6) {
        $fk_cat_id = 6;
        $cat_age = 'confirmé';
    }
    elseif ($choix1==7) {
        $fk_cat_id = 7;
        $cat_age = 'senior';
    }
}
```

Options

					cat_id	cat_libelle
<input type="checkbox"/>		Éditer		Copier		Supprimer
					1	poussin
<input type="checkbox"/>		Éditer		Copier		Supprimer
					2	junior
<input type="checkbox"/>		Éditer		Copier		Supprimer
					3	ado
<input type="checkbox"/>		Éditer		Copier		Supprimer
					4	jeune
<input type="checkbox"/>		Éditer		Copier		Supprimer
					5	adulte
<input type="checkbox"/>		Éditer		Copier		Supprimer
					6	confirmé
<input type="checkbox"/>		Éditer		Copier		Supprimer
					7	seniors

On a un **if** qui permet d'attribuer le bonne id à la catégorie d'âge sélectionné dans le formulaire.

« epreuve »

```
// récupérer l'id d'epreuve
$fk_ep_id = 0;
if (isset($_POST['epreuve'])) {
    $choix2 = $_POST['epreuve'];
    if ($choix2==1) {
        $fk_ep_id = 1;
        $epreuve = 'Arkétop';
    }
    if ($choix2==2) {
        $fk_ep_id = 2;
        $epreuve = 'Aventure';
    }
    if ($choix2==3) {
        $fk_ep_id = 3;
        $epreuve = 'Suprême sensation';
    }
    if ($choix2==4) {
        $fk_ep_id = 4;
        $epreuve = 'Coule douce';
    }
    if ($choix2==5) {
        $fk_ep_id = 5;
        $epreuve = 'Admirer le paysage';
    }
    if ($choix2==6) {
        $fk_ep_id = 6;
        $epreuve = 'Verdoyants parcours';
    }
    if ($choix2==7) {
        $fk_ep_id = 7;
        $epreuve = 'Pour les mordues';
    }
    if ($choix2==8) {
        $fk_ep_id = 8;
        $epreuve = 'Vive demain';
    }
    if ($choix2==9) {
        $fk_ep_id = 9;
        $epreuve = 'La perle de la montagne';
    }
    if ($choix2==10) {
        $fk_ep_id = 10;
        $epreuve = 'Tous en plaine';
    }
}
```

```
if ($choix2==11) {
    $fk_ep_id = 11;
    $epreuve = 'Parcours aventure';
}
if ($choix2==12) {
    $fk_ep_id = 12;
    $epreuve = 'Le chemin des petits';
}
if ($choix2==13) {
    $fk_ep_id = 13;
    $epreuve = 'Warriors';
}
if ($choix2==14) {
    $fk_ep_id = 14;
    $epreuve = 'Baby course';
}
if ($choix2==15) {
    $fk_ep_id = 15;
    $epreuve = 'Upgrade';
}
if ($choix2==16) {
    $fk_ep_id = 16;
    $epreuve = 'Toujours plus loin';
}
if ($choix2==17) {
    $fk_ep_id = 17;
    $epreuve = 'Les 5 bornes';
}
```

	ep_id	ep_nom
 Supprimer	1	Arkétop
 Supprimer	2	Aventure
 Supprimer	3	Suprême sensation
 Supprimer	4	Coule douce
 Supprimer	5	Admirer le paysage
 Supprimer	6	Verdoyants le paysage
 Supprimer	7	Extrême
 Supprimer	8	Pour les mordues
 Supprimer	9	Vive demain
 Supprimer	10	La perle de la montagne
 Supprimer	11	Tous en plaine
 Supprimer	12	Parcours aventure
 Supprimer	13	Le chemin des petits
 Supprimer	14	Warriors
 Supprimer	15	Baby course
 Supprimer	16	Upgrade
 Supprimer	17	Toujours plus loin
 Supprimer	18	Les 5 bornes

On a un **if** qui permet d'attribuer le bonne id à l'épreuve sélectionné dans le formulaire.

« coureur »

```
// requête préparé COUREUR
$stmt1 = $connect -> prepare("INSERT INTO coureur (co_nom, co_prenom, co_nom_rue_coureur, co_cp, co_ville, co_naissance, co_sexe, nationalite) VALUES (?, ?, ?, ?, ?, ?, ?, ?)");
$stmt1 -> bind_param("sssisiss", $nom, $prenom, $adresse, $code_postal, $ville, $date, $sexe, $nationalite);
$stmt1 -> execute();
$fkco_id = $connect -> insert_id;
$stmt1 -> close();
```

Ici pour la clé étrangère coureur, on va directement insérer le profil du coureur à l'aide d'une requête préparé, et stocker son id dans une variable, grâce à la ligne 5.

« reglement »

```
// requête préparé REGLEMENT
$stmt2 = $connect -> prepare("INSERT INTO reglement (reg_date, reg_montant, reg_numchk, reg_numcb, reg_date_expiration_cb, reg_titulaire_chk, reg_ordre_chk, reg_code) VALUES (?, ?, ?, ?, ?, ?, ?, ?)");
$stmt2 -> bind_param("idiiiiss", $date, $montant, $numchk, $numcb, $date_expiration_cb, $titulaire_chk, $ordre_chk, $code);
$stmt2 -> execute();
$fkreg_id = $connect -> insert_id;
$stmt2 -> close();
```

Ici aussi, pour clé étrangère règlement, on va insérer des valeurs dites null à l'aide d'une requête préparé, et stocker l'id dans une variable, grâce à la ligne 5.

5/ Maintenant que on as nos 4 clé étrangères on peut faire notre insertion dans la table inscription :

```
// requête préparé INSCRIPTION
$stmt3 = $connect -> prepare("INSERT INTO inscription (in_numdos, in_date, in_prix, in_age, in_maillot, in_certifmed, in_tpsestim, in_tpsfin, FK_ep_id, FK_reg_id, FK_co_id, FK_cat_id) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)");
$stmt3 -> bind_param("iidiisssiiii", $numero_dossard, $date_inscription, $prix, $age, $taille, $certif, $temps, $tpsfin, $fk_ep_id, $fkreg_id, $fkco_id, $fk_cat_id);
$stmt3 -> execute();
$stmt3 -> close();
```

On retrouve bien nos 4 clés étrangères à la fin de la 3èmes lignes.

Il est important de se souvenir du code du règlement. C'est grâce à lui que nous allons pouvoir régler.

6/ Pour terminer la partie formulaire, on va faire notre page qui récapitule le dossier d'inscription. À l'aide d'HTML et de PHP.

```
<head>
  <meta charset="utf-8"/>
  <title>Inscription Marathon</title>
  <link rel='stylesheet' href="style.css"/>
</head>

<html>
  <body>
    <section>
      <div>
        Voici votre dossier d'inscription :
        <br>
        <br>
        Nom : <?php echo $nom = $_POST['nom'];?>
        <br>
        Prenom : <?php echo $prenom = $_POST['prenom'];?>
        <br>
        Date : <?php echo $date = $_POST['date_naissance'];?>
        <br>
        Sexe : <?php echo $sexe = $_POST['genre'];?>
        <br>
        Adresse : <?php echo $adresse = $_POST['adresse'];?>
        <br>
        Code Postal : <?php echo $code = $_POST['code_postal'];?>
        <br>
        Ville : <?php echo $ville = $_POST['ville'];?>
        <br>
        Taille : <?php echo $taille = $_POST['taille_maillots'];?>
        <br>
        Certificat Médical : <?php echo $certif = $_POST['certificat_medical'];?>
        <br>
        Temps estimé : <?php echo $temps = $_POST['temps_estime'];?>
        <br>
        Nationalité : <?php echo $nationalite = $_POST['nationalite'];?>
      </div>
    </section>
  </body>
</html>
```

De la ligne 13 à 33, on affiche toute les données qu'on avait rentré dans notre **<form>**.

7/ On vas faire la mise en forme de notre site d'inscription (basique), avec le CSS.

```
body {
  background-color: #212f3c;
  display: flex;
  justify-content: center;
}

section {
  width: 50%;
  display: flex;
  justify-content: center;
  background: #b2babb;
}

form {
  font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
  display: flex;
  flex-direction: column;
  font-weight: bold;
}
```

8/ Résultat

Nom :
Bourdel

Prenom :
Théo

Date de naissance :
20 / 12 / 2001

Genre :

☒ Homme

☐ Femme

Adresse :
11 rue de Paris

Code Postal :
75000

Ville :
Paris

Taille de Maillots :
S

Certificat médical :

☒ Oui

☐ Non

Temps estime :
02 : 50

Nationalité :
francais

Inscription

Voici votre dossier d'inscription :

Nom : Bourdel
Prenom : Théo
Date : 2001-12-20
Sexe : M
Adresse : 11 rue de Paris
Code Postal : 75000
Ville : Paris
Taille : S
Certificat Médical : oui
Temps estimé : 02:50
Nationalité : francais

PARTIE REGLEMENT

1/ Il y a 3 types de règlements : chèque / cb / liquide
On va donc faire 3 formulaires :

```
<head>
  <meta charset="utf-8"/>
  <title>Projet Marathon (chèque)</title>
  <link rel='stylesheet' href="style.css"/>
</head>

<html>
  <body>
    <section>
      <form method="post" action="recapcheque.php">
        <label>Date :</label>
        <br>
        <input type="date" name="date" required>
        <br>
        <label>Numero de chèque :</label>
        <br>
        <input type="number" name="num" required>
        <br>
        <label>Titulaire du chèque :</label>
        <br>
        <input type="text" name="titulaire" required>
        <br>
        <label>Ordre :</label>
        <br>
        <input type="text" name="ordre" required>
        <br>
        <label>Code du reglement</label>
        <br>
        <input type="number" name="code_reglement" required>
        <br>
        <label>Prix :</label>
        <br>
        <p>55.00€</p>
        <br>
        <input type="submit" name="btn" value="RECAP"/>
      </form>
    </section>
  </body>
</html>
```

```
<head>
  <meta charset="utf-8"/>
  <title>Projet Marathon (carte)</title>
  <link rel='stylesheet' href="style.css"/>
</head>

<html>
  <body>
    <section>
      <form method="post" action="recapcarte.php">
        <label>Date :</label>
        <br>
        <input type="date" name="date" required>
        <br>
        <label>Numero de carte :</label>
        <br>
        <input type="text" name="num" required>
        <br>
        <label>Date expiration de la carte :</label>
        <br>
        <input type="date" name="date_expiration" required>
        <br>
        <label>Code (à 3 chiffres) :</label>
        <br>
        <input type="text" name="code" required>
        <br>
        <label>Code du reglement</label>
        <br>
        <input type="number" name="code_reglement" required>
        <br>
        <label>Prix :</label>
        <br>
        <p>55.00€</p>
        <br>
        <input type="submit" name="btn" value="RECAP"/>
      </form>
    </section>
  </body>
</html>
```



```
<head>
  <meta charset="utf-8"/>
  <title>Projet Marathon (carte)</title>
  <link rel='stylesheet' href="style.css"/>
</head>

<html>
  <body>
    <section>
      <form method="post" action="recapcarte.php">
        <label>Date :</label>
        <br>
        <input type="date" name="date" required>
        <br>
        <label>Numero de carte :</label>
        <br>
        <input type="text" name="num" required>
        <br>
        <label>Date expiration de la carte :</label>
        <br>
        <input type="date" name="date_expiration" required>
        <br>
        <label>Code (à 3 chiffres) :</label>
        <br>
        <input type="text" name="code" required>
        <br>
        <label>Code du reglement</label>
        <br>
        <input type="number" name="code_reglement" required>
        <br>
        <label>Prix :</label>
        <br>
        <p>55.00€</p>
        <br>
        <input type="submit" name="btn" value="RECAP"/>
      </form>
    </section>
  </body>
</html>
```

2/ Maintenant, grâce aux notions acquises avec la partie inscription, on a juste à UPDATE la colonne reglement du client pour y insérer ses données de règlements.

```
<?php

//connexion bd
date_default_timezone_set('Europe/Paris');
$server='127.0.0.1:3309';
$username='root';
$password='';
$dbname = "marathon1";

// connexion mysqli
$connect = mysqli_connect($server, $username, $password, $dbname);

// vérifier si on est bien connecté
if(!$connect){
    die("Erreur : impossible de se connecter à sql".mysqli_connect_error());
}

//traitement de la requête
if(isset($_POST['date'])){ //détecte si la date est remplie

    // variables
    $date = $_POST['date'];
    $code_reglement = $_POST['code_reglement'];
    $prix = 55.00;

    // update reglement
    $stmt4 = "UPDATE reglement SET reg_date = $date, reg_montant = $prix WHERE reg_id = $code_reglement";

    // vérifier si l'update est bien passé
    if ($connect->query($stmt4) === TRUE) {
        //echo "Record updated successfully";
    } else {
        echo "Error updating record: " . $connect->error;
    }

    $connect->close();
}
?>
```

```

<?php

//connexion bd
date_default_timezone_set('Europe/Paris');
$server='127.0.0.1:3309';
$username='root';
$password='';
$dbname = "marathon1";

// connexion mysqli
$connect = mysqli_connect($server, $username, $password, $dbname);

// vérifier si on est bien connecté
if(!$connect){
    die("Erreur : impossible de se connecter à sql".mysqli_connect_error());
}

//traitement de la requête
if(isset($_POST['date'])){ //détecte si la date est remplie

    // variables
    $date = $_POST['date'];
    $num = $_POST['num'];
    $date_expiration = $_POST['date_expiration'];
    $code = $_POST['code'];
    $code_reglement = $_POST['code_reglement'];
    $prix = 55.00;

    // update reglement
    $stmt6 = "UPDATE reglement SET reg_date = $date, reg_montant = $prix, reg_numcb = $num, reg_date_expiration_cb = $date_expiration, reg_code = $code WHERE reg_id = $code_reglement";

    // vérifier si l'update est bien passé
    // '$connect->query($stmt6)' est indispensable
    if ($connect->query($stmt6) === TRUE) {
        //echo "Record updated successfully";
    } else {
        echo "Error updating record: " . $connect->error;
    }
}

$connect->close();
}
?>

```

```

<?php

//connexion bd
date_default_timezone_set('Europe/Paris');
$server='127.0.0.1:3309';
$username='root';
$password='';
$dbname = "marathon1";

// connexion mysqli
$connect = mysqli_connect($server, $username, $password, $dbname);

// vérifier si on est bien connecté
if(!$connect){
    die("Erreur : impossible de se connecter à sql".mysqli_connect_error());
}

//traitement de la requête
if(isset($_POST['date'])){ //détecte si la date est remplie

    // variables
    $date = $_POST['date'];
    $num = $_POST['num'];
    $titulaire = $_POST['titulaire'];
    $ordre = $_POST['ordre'];
    $code_reglement = $_POST['code_reglement'];
    $prix = 55.00;

    // update reglement
    $stmt5 = "UPDATE reglement SET reg_date = $date, reg_montant = $prix, reg_numchk = $num, reg_titulaire_chk = '$titulaire', reg_ordre_chk = '$ordre' WHERE reg_id = $code_reglement";

    // vérifier si l'update est bien passé est bien passé
    if ($connect->query($stmt5) === TRUE) {
        //echo "Record updated successfully";
    } else {
        echo "Error updating record: " . $connect->error;
    }
}

$connect->close();
}

```

PARTIE 2

Déroulé de la Mission 12 :

1/ a/ Chaque user possède une connexion propre à lui-même. Grâce à cette commande : *CREATE USER 'login'@'hote' [IDENTIFIED BY 'mdp']*

login → nom d'utilisateur

hote → le port (localhost ou un IP)

identified → créer un mot de passe

Ici dans notre Mission on a 4 utilisateurs à créer :

- utilisateur gestionnaire de la base marathon : il est représenté comme « le directeur » de la base marathon, c'est l'utilisateur qui possède tous les droits
- utilisateur accueil inscription de la base marathon : cet utilisateur gère toutes les informations relatives à la table inscription, et peut réaliser des interrogations sur les tables : coureur, épreuve, les types d'épreuves et catégorie d'âge
- utilisateur évènementiel : il gère les championnats, les manifestations ainsi que toutes les informations relatives aux épreuves. Ce dernier peut s'informer sur les inscriptions
- utilisateur relation client : il possède des relations clubs et compétiteurs (coureurs). Ce dernier peut interroger toutes les informations relatives à la base marathon

1/ b/ Concernant les droits, il existe de multiples commandes pour les gérer.

GRANT → permet de donner des droits (SELECT, UPDATE, INSERT, ...) à un utilisateur sur une base de données

REVOKE → permet de révoquer des autorisations

USAGE → il permet de modifier les caractéristiques d'un compte avec la commande GRANT, sans modifier les privilèges du compte

L'avantage d'attribuer les bons droits aux bons utilisateurs, c'est que ça améliore la sécurité de la base de données, mais aussi l'organisation.

2/ Les différents utilisateurs

L'avantage de mettre en place différents utilisateurs est d'assurer une bonne sécurité / une bonne organisation de la base de donnée.

3/ Tests d'affectation des droits

a/ l'utilisateur gestionnaire

Commande	Résultat attendu	Résultat obtenu
Create table test	Création d'une table	OUI

Screen :

```
Invite de commandes - mysql.exe -u gestion -p
Database changed
mysql> create table test
-> (
-> age int(10)
-> )
-> ;
Query OK, 0 rows affected (0.03 sec)

mysql> show tables;
+-----+
| Tables_in_marathon |
+-----+
| adherer             |
| categorie_age       |
| championnat        |
| classement          |
| club                |
| coureur             |
| date                |
| epreuve             |
| inscrire            |
| manifestation       |
| paiement            |
| parcours            |
| test                |
| type_epreuve        |
+-----+
14 rows in set (0.01 sec)
```

b/ l'utilisateur accueil inscription

Commande	Résultat attendu	Résultat obtenu
Select * from epreuve	Pouvoir surveiller la table épreuve	OUI

Screen :

Invite de commandes - mysql.exe -u acinsc -p

```
mysql> select * from epreuve;
```

ep_id	ep_ville	ep_entass	ep_heuredeb	ep_heurefin
1	Bordeaux	PROZIS, Netflix et Uber	10:30:00	11:50:00
2	Bergerac	Michel et Augustin, myProtein et Amazon	08:11:00	20:05:00
3	Mont-de-Marsan	Le Parisien, La Banque Postale et Apple	08:20:00	12:42:00
4	Arjuzanx	Hancock, Peugeot et Orange	14:00:00	19:30:00
5	Toulouse	SFR, Audi, Haribo et Samsung	10:34:20	17:50:30
6	Mont-de-Marsan	Le Parisien, La Banque Postale et Apple	10:00:00	15:25:30
7	Montauban	Paper Mate, Decathlon et Nivea	08:00:00	08:45:59
8	20180720	91500	00:00:01	00:00:04
9	20180720	80000	00:00:03	00:00:03
10	20180720	100000	00:00:05	00:00:07
11	20180720	80000	00:00:06	00:00:01
12	20180720	80000	00:00:07	00:00:07
13	20180920	151500	00:00:01	00:00:06
14	20180920	131500	00:00:02	00:00:06
15	20180920	121500	00:00:03	00:00:07
16	20180920	140000	00:00:05	00:00:07
17	20180920	150000	00:00:06	00:00:01
18	20180920	110000	00:00:07	00:00:01

18 rows in set (0.01 sec)

c/ l'utilisateur évènementiel

Commande	Résultat attendu	Résultat obtenu
Drop table inscrire	La table inscrire ne doit pas être supprimé	OUI

Screen :

Invite de commandes - mysql.exe -u event -p

```
Microsoft Windows [version 10.0.17763.973]
(c) 2018 Microsoft Corporation. Tous droits réservés.

C:\Users\bound>cd
C:\Users\bound

C:\Users\bound>cd\

C:\>cd C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin

C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin>mysql.exe -u event -p
Enter password: ***
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 32
Server version: 5.1.73-community MySQL Community Server (GPL)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use marathon
Database changed
mysql> drop table inscrire;
ERROR 1142 (42000): DROP command denied to user 'event'@'localhost' for table 'inscrire'
```

c/ l'utilisateur relation client

Commande	Résultat attendu	Résultat obtenu
Drop table club	l'utilisateur ne doit pas pouvoir faire autre chose que lire les données de toutes les tables	OUI

Screen :

```
Invite de commandes - mysql.exe -u relat -p
Microsoft Windows [version 10.0.17763.973]
(c) 2018 Microsoft Corporation. Tous droits réservés.

C:\Users\bound>cd\

C:\>cd C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin

C:\Program Files (x86)\MySQL\MySQL Server 5.1\bin>mysql.exe -u relat -p
Enter password: ***
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 30
Server version: 5.1.73-community MySQL Community Server (GPL)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use marathon
Database changed
mysql> drop table club;
ERROR 1142 (42000): DROP command denied to user 'relat'@'localhost' for table 'club'
```

Déroulé des Missions 13 à 16 :

1/a/ Les fonctions

```
1 DELIMITER $$
2
3 DROP FUNCTION IF EXISTS age_epreuve $$
4
5 CREATE FUNCTION age_epreuve (p_co_id int, p_ep_id int)
6 RETURNS int(11)
7 BEGIN
8 RETURN
9 (
10 SELECT
11     ROUND (datediff(ep.ep_date, co.co_naissance) / 365, 0)
12 FROM
13     inscrire ins
14 LEFT JOIN coureur co on (ins.in_coureur_fk = co.co_id)
15 LEFT JOIN epreuve ep on (ins.in_epreuve_fk = ep.ep_id)
16 WHERE
17     ins.in_coureur_fk = p_co_id
18     AND ins.in_epreuve_fk = p_ep_id
19 );
20 END $$
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,4315 seconde(s).)

DROP FUNCTION IF EXISTS age_epreuve

[Éditer en ligne] [Éditer] [Créer le

⚠ Note: #1305 FUNCTION marathon1.age_epreuve does not exist

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,1068 seconde(s).)

```
CREATE FUNCTION age_epreuve (p_co_id int, p_ep_id int) RETURNS int(11) BEGIN RETURN ( SELECT ROUND (datediff(ep.ep_date, co.co_naissance) / 365, 0) FROM inscrire ins LEFT JOIN coureur co on
(ins.in_coureur_fk = co.co_id) LEFT JOIN epreuve ep on (ins.in_epreuve_fk = ep.ep_id) WHERE ins.in_coureur_fk = p_co_id AND ins.in_epreuve_fk = p_ep_id ); END
```

[Éditer en ligne] [Éditer] [Créer le

b/ Les vues

```
mysql> CREATE VIEW V_Coureur_officiel AS
-> SELECT coureur.co_id_coureur, co_nom_coureur, co_prenom_coureur, co_naissance_coureur, co_sexe_coureur, in_temps_effectue_inscription
-> FROM coureur,inscription
-> WHERE coureur.co_id_coureur = inscription.FK_co_id_coureur
-> AND in_temps_effectue_inscription IS NOT NULL;
Query OK, 0 rows affected (0.04 sec)

mysql> select * from V_Coureur_officiel;
Empty set (0.00 sec)
```


c/ Les vues matérialisé

```
mysql> CREATE TABLE VUEM_resultat_coureur AS SELECT concat (cat_libelle_categorie_age) AS abc, count(*) AS nb_participans FROM coureur, inscription, categorie_age, epreuve, type_epreuve WHERE coureur.co_id_coureur = inscription.FK_co_id_coureur AND epreuve.FK_ty_id_type_epreuve = type_epreuve.ty_id_type_epreuve AND inscription.FK_cat_id_categorie_age = categorie_age.cat_id_categorie_age GROUP BY abc;
ERROR 1054 (42S22): Unknown column 'inscription.FK_cat_id_categorie_age' in 'where clause'
```

d/ Les triggers

```
1 DELIMITER |
2
3 CREATE TRIGGER inscription_after_insert after INSERT ON inscription
4 FOR EACH ROW
5 BEGIN
6 UPDATE epreuve SET ep_nb = (SELECT COUNT(*) FROM inscription WHERE FK_ep_id = new.FK_ep_id) WHERE ep_id = new.FK_ep_id;
7
8 END |
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,2740 seconde(s).)

```
CREATE TRIGGER inscription_after_insert after INSERT ON inscription FOR EACH ROW BEGIN UPDATE epreuve SET ep_nb = (SELECT COUNT(*) FROM inscription WHERE FK_ep_id = new.FK_ep_id) WHERE ep_id = new.FK_ep_id; END
```

[\[Éditer en ligne\]](#) [\[Éditer \]](#) [\[Créer le code \]](#)

```
1 DELIMITER |
2
3 CREATE TRIGGER inscription_after_delete AFTER DELETE ON inscription
4 FOR EACH ROW
5 BEGIN
6 UPDATE epreuve SET ep_nb = (SELECT COUNT(*) FROM inscription WHERE FK_ep_id = old.FK_ep_id) WHERE ep_id = old.FK_ep_id;
7
8 END |
```

✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,1160 seconde(s).)

```
CREATE TRIGGER inscription_after_delete AFTER DELETE ON inscription FOR EACH ROW BEGIN UPDATE epreuve SET ep_nb = (SELECT COUNT(*) FROM inscription WHERE FK_ep_id = old.FK_ep_id) WHERE ep_id = old.FK_ep_id; END
```

[\[Éditer en ligne\]](#) [\[Éditer \]](#) [\[Créer le code \]](#)

```
mysql> use marathon;
Database changed
mysql>
mysql> CREATE TRIGGER after_insert_inscription AFTER INSERT
-> ON inscription FOR EACH ROW
-> CALL MAJ_VM_coureur_resultat();
Query OK, 0 rows affected (0.16 sec)

mysql>
mysql> CREATE TRIGGER after_delete_inscription AFTER DELETE
-> ON coureur FOR EACH ROW
-> CALL MAJ_VM_coureur_resultat();
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> CREATE TRIGGER after_update_inscription AFTER UPDATE
-> ON inscription FOR EACH ROW
-> CALL MAJ_VM_coureur_resultat();
Query OK, 0 rows affected (0.03 sec)

mysql> show triggers;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| Trigger                               | Event | Table | Statement                               | Timing | Created | sql_mode                               | Definer           | character_set_client | collation_connection | Database Collation |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| after_delete_inscription              | DELETE | coureur | CALL MAJ_VM_coureur_resultat()         | AFTER  | NULL    | STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION | root@localhost    | latin1               | latin1_swedish_ci   | latin1_swedish_ci |
| after_insert_inscription              | INSERT | inscription | CALL MAJ_VM_coureur_resultat()         | AFTER  | NULL    | STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION | root@localhost    | latin1               | latin1_swedish_ci   | latin1_swedish_ci |
| after_update_inscription              | UPDATE | inscription | CALL MAJ_VM_coureur_resultat()         | AFTER  | NULL    | STRICT_TRANS_TABLES,NO_AUTO_CREATE_USER,NO_ENGINE_SUBSTITUTION | root@localhost    | latin1               | latin1_swedish_ci   | latin1_swedish_ci |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.04 sec)
```

e/ Les procédures stockées

```
1 DELIMITER $$
2
3 DROP PROCEDURE IF EXISTS inscription $$
4
5 CREATE PROCEDURE inscription (in co_id int, in ep_id int, in in_taillemaillet char(3))
6 sortie:
7 BEGIN
8 DECLARE coureur_age_epreuve int;
9 START TRANSACTION;
10 INSERT INTO inscrire (in_numinscription, in_epreuve, in_certificatmedical, in_dateinscription, in_taillemaillet) VALUES (co_id, ep_id, 1, now(), in_taillemaillet);
11 SET coureur_age_epreuve = age_epreuve(co_id, ep_id);
12 IF (coureur_age_epreuve < 10) THEN
13     ROLLBACK;
14     LEAVE sortie;
15 END IF;
16 CALL dossard(ep_id);
17 COMMIT;
18
19 END $$
```

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.1999 seconde(s.))

DROP PROCEDURE IF EXISTS inscription

[Éditer en ligne] [Éditer] [Créer le code source PHP]

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0.0043 seconde(s.))

CREATE PROCEDURE inscription (in co_id int, in ep_id int, in in_taillemaillet char(3)) sortie: BEGIN DECLARE coureur_age_epreuve int; START TRANSACTION; INSERT INTO inscrire (in_numinscription, in_epreuve, in_certificatmedical, in_dateinscription, in_taillemaillet) VALUES (co_id, ep_id, 1, now(), in_taillemaillet); SET coureur_age_epreuve = age_epreuve(co_id, ep_id); IF (coureur_age_epreuve < 10) THEN ROLLBACK; LEAVE sortie; END IF; CALL dossard(ep_id); COMMIT; END

```
mysql> DELIMITER $$
mysql> CREATE PROCEDURE MAJ_VM_coureur_resultat()
  -> BEGIN
  -> DELETE FROM VM_coureur_resultat;
  -> INSERT INTO VM_coureur_resultat
  -> SELECT concat(ty_nom_type_epreuve," ", cat_libelle_categorie_age) AS abc, count(*) as nb_participants
  -> FROM coureur, inscription, categorie_age, epreuve, type_epreuve
  -> WHERE coureur.co_id_coureur= inscription.FK_co_id_coureur
  -> AND inscription.FK_ep_id_epreuve = epreuve.ep_id_epreuve
  -> AND epreuve.FK_ty_id_type_epreuve=type_epreuve.ty_id_type_epreuve
  -> AND inscription.FK_cat_id_categorie_age=categorie_age.cat_id_categorie_age
  -> GROUP BY abc;
  -> END $$
Query OK, 0 rows affected (0.02 sec)
```

2/ D'un point de vue professionnel, les missions 13 à 16, ont renforcé mes connaissances en SQL.

PARTIE 3

Déroulé des Mission 18 à 25 :

1/ Le nombre et la taille des bases de données sont illimités mais vous devez optimiser vos requêtes, vos tables (avec des indexes) afin de ne pas surcharger le serveur sur lequel vos bases se trouvent, surtout lorsque la taille de la base augmente considérablement.

Pour cela on va utiliser principalement 2 commandes : EXPLAIN / INDEX

2/ a/ La commande EXPLAIN permet d'auditer une requête SQL et ainsi connaître le nombre de lignes parcourues pour exécuter la requête.

b/ Si ce nombre est très élevé, l'utilisation de la commande INDEX peut s'avérer utile afin d'optimiser la requête.

3/ La sécurité des bases de données couvre un ensemble de contrôles de sécurité conçu pour protéger un système de gestion des bases de données (SGBD). Il est ici nécessaire de protéger la base de donnée marathon car elle contient des informations privées concernant nos utilisateurs.

4/ Il est nécessaire de mettre la base marathon en mode client serveur, pour que les clients puissent y accéder sur internet.

5/ Le contenu de ces missions, m'ont appris qu'une base de donnée peut-être optimisée, pour augmenter sa performance. J'ai appris aussi qu'une base de donnée doit-être sécurisée, pour préserver les données des clients.