

Interaction PHP / base de données

Accès à MySQL depuis PHP

- L'interface MySQL/PHP permet de récupérer, modifier, créer, toute information d'une base de données MySQL depuis un script PHP.
- L'accès à MySQL se fait via des fonctions PHP prédéfinies (module `mysqli`)
- PHP a un module différent pour l'accès à la plupart des SGBD
- Il existe également le module PHP `PDO`, qui uniformise l'accès au différents SGBD (légèrement plus complexe à utiliser)

PHP/mysqli : connexion à la base de données

Accès à MySQL depuis PHP au travers de l'extension `mysqli`

Connexion à une base de données

```
$connex = mysqli_connect($serveur, $login, $mdp, $base);
```

Tester si la connexion a eu lieu

```
if (! $connex) {  
    //afficher page d'erreur  
}
```

En phase de débogage il peut être utile de récupérer l'erreur renvoyé par MySQL

```
echo mysqli_connect_error($connex);
```

PHP/mysqli : exécution de requêtes

Exécution d'une requête

```
$req = "SELECT ...FROM ..." // ou $req = "INSERT ...INTO ..." etc.  
$resultat = mysqli_query($connex, $req);
```

Tester si la requête a été exécutée sans erreur

```
if (! $resultat) {  
    //afficher page d'erreur  
}
```

Récupérer la dernière erreur de MySQL :

```
echo mysqli_error($connex);
```

PHP/mysqli : traitement du résultat d'une requête

Si `$req` est une requête `SELECT`

```
$resultat = mysqli_query($connex, $req);
```

`$resultat` est de type `resource`,

on peut voir cela comme une collection de lignes, chaque ligne représentée par un tableau

Récupération d'une ligne du résultat sous forme de tableau associatif

```
$ligne = mysqli_fetch_assoc ($resultat)
```

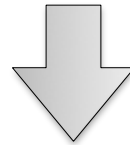
Récupération d'une ligne du résultat sous forme de tableau indicé

```
$ligne = mysqli_fetch_row ($resultat);
```

PHP/mysqli : traitement du résultat d'une requête

Une ligne du résultat en tant que tableau associatif :
entrées du tableau indexées par 'attribut'

```
+-----+-----+-----+  
| titre      | annee | realisateur |  
+-----+-----+-----+  
| Pulp Fiction | 1995 | Tarantino  |  
+-----+-----+-----+
```



\$ligne

"Pulp Fiction"	1995	"Tarantino"
----------------	------	-------------

"titre"

"annee"

"realisateur"

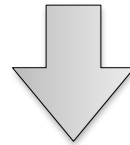
Ex : `$ligne['realisateur']` donne le nom du réalisateur

PHP/mysqli : traitement du résultat d'une requête

Une ligne du résultat en tant que tableau indicé :

entrées du tableau indexés par position :

```
+-----+-----+-----+
| titre      | annee | realisateur |
+-----+-----+-----+
| Pulp Fiction | 1995 | Tarantino  |
+-----+-----+-----+
```



`$ligne`

"Pulp Fiction"	1995	"Tarantino"
0	1	2

Ex : `$ligne[2]` donne le nom du réalisateur

PHP/mysqli : traitement du résultat d'une requête

`mysqli_fetch_assoc ($resultat)` et `mysqli_fetch_row ($resultat)` renvoient une ligne du résultat et font avancer un pointeur interne de ligne dans `$resultat`

les deux renvoient NULL s'il n'y a plus de ligne à lire

⇒ une suite d'appels à ces fonctions parcourt toutes les lignes du résultat :

```
$ligne = mysqli_fetch_assoc($resultat);  
while ($ligne) {  
    //traiter $ligne  
    $ligne = mysqli_fetch_assoc($resultat); //prochaine ligne  
}
```


PHP/mysqli : traitement du résultat d'une requête

`mysqli_fetch_assoc ($resultat)` et `mysqli_fetch_row ($resultat)` renvoient une ligne du résultat et font avancer un pointeur interne de ligne dans `$resultat`

les deux renvoient NULL s'il n'y a plus de ligne à lire

⇒ une suite d'appels à ces fonctions parcourt toutes les lignes du résultat :

...ou de façon équivalente (plus compacte)

```
while ($ligne = mysqli_fetch_assoc($resultat)) {  
    //traiter $ligne  
}
```

possible parce qu'une affectation (`$var1 = $var2`) est aussi une expression qui renvoie la valeur affectée

PHP/mysqli : traitement du résultat d'une requête

Nombre de lignes du résultat

```
mysqli_num_rows ($resultat)
```

Libération des ressources allouées pour stocker le résultat

```
mysqli_free_result ($resultat)
```

Il est conseillé de toujours utiliser cette fonction lorsque l'objet `$resultat` n'est plus utile

PHP/mysqli : fermeture de la connexion

Fermeture de la connexion

```
mysqli_close($connex);
```

À appeler lorsque l'interaction du script PHP avec la base de données est terminée (aucune autre requête à envoyer)

Exemple de page Web qui interagit avec une BD

- Page Web très simple qui réalise une tâche de base
 - ▶ récupérer la liste des films de la BD et l'afficher sous forme de tableau HTML
- Mais on structure le script PHP de façon très générale, qui peut être facilement adaptée à gérer une interaction plus complexe

05-films /films.php
/connex.php
/affichage.php
localhost/.../films.php

```
1 <?php
2 require_once "affichage.php";
3 require_once "connex.php";
4
5 $connex = connexion_bd();
6
7 $films = lire_films($connex);
8
9 page_films($films);
10
11 mysqli_free_result($films);
12
13 mysqli_close($connex);
14
15 ?>
16
17
```

```
1 <?php
2 require_once "affichage.php";
3
4
5 function connexion_bd() {
6     $serv = "127.0.0.1";
7     $username = "sirangelo";
8     $pwd = "sirangelo";
9     $bd = "sirangelo";
10    $connex = mysqli_connect($serv, $username,
... $pwd, $bd);
11    if (! $connex) {
12        page_erreur(ERR_CONNEX,
... mysqli_connect_error($connex)); exit;
13    }
14    return $connex;
15 }
16
17 function lire_films($connex) {
18     $req = "select * from Film";
19     $resultat = mysqli_query($connex, $req);
20     if (!$resultat) {
21         page_erreur(ERR_REQUETE,
... mysqli_error($connex));
22         mysqli_close($connex);
23         exit;
24     }
25     return $resultat;
26 }
27
28 ?>
```

```
1 <?php
2
3 //le codes d'erreurs, pour savoir quel type d'erreur afficher
4 define('ERR_CONNEXION', 0);
5 define ('ERR_REQUETE', 1);
6
7
8 //les fonctions de base pour l'affichage
9 function afficher_entete($titre) {
10 ?>
11     <html>
12     <head>
13     <title> <?=$titre?> </title>
14     <meta charset="utf-8" >
15     <link rel="stylesheet" type="text/css" href="films.css">
16     </head>
17     <body>
18 <?php
19 }
20
21 function afficher_pied_page() {
22     echo "</body></html>";
23 }
24
25 //affiche une ligne de la tables des films
26 //parametre : un tableau associatif representant un film
27 //noter la protection des chaines avant l'affichage
28 function afficher_ligne(&$ligne) {
29     echo "<tr>",
30         "<td>". htmlspecialchars($ligne["titre"])."</td>",
31         "<td>". htmlspecialchars($ligne["annee"])."</td>",
32         "<td>". htmlspecialchars($ligne["realisateur"])."</td>",
33         "</tr>";
34 }
35
36
37 //affiche la table des films
38 //parametre : la ressource contenant les films à afficher
39 function afficher_table ($films) { ?>
40     <table>
41         <tr> <th>Titre du film</th>
42         <th>Année de realisation</th>
43         <th>Réalisateur</th></tr>
44 <?php
```

```
45     while($ligne = mysqli_fetch_assoc($films))
46         afficher_ligne($ligne);
47     echo "</table>" ;
48 }
49
50
51
52
53 //les vues
54
55 //la page qui affiche les films disponibles
56 //parametre : la ressource contenant les films à afficher
57 function page_films ($films) {
58     afficher_entete("Films");
59     echo "<h2> Nos films </h2>";
60     afficher_table($films);
61     afficher_pied_page();
62 }
63
64 //affiche une page d'erreur
65 //paramteres : le code de l'erreur et un message d'erreur
66 function page_erreur($err_code, $error) {
67     afficher_entete("Erreur");
68     switch ($err_code) {
69         case ERR_CONNEXION:
70             echo "<h2>Desolé, connexion impossible</h2>";
71             break;
72         case ERR_REQUETE:
73             echo "<h2>Erreur dans l'execution de la
... requete</h2>";
74             break;
75     }
76     //pour le debogage :
77     echo "<p>".$error."</p>";
78     afficher_pied_page();
79 }
80 ?>
81
82
83
84
```


Nos films

Titre du film	Année de realisation	Réalisateur
Match Point		Allen
Alien	1979	Scott
Vertigo	1958	Hitchcock
Psychose	1960	Hitchcock
Kagemusha	1980	Kurosawa
Volte-face	1997	Woo
Pulp Fiction	1995	Tarantino
Titanic	1997	Cameron
Sacrifice	1986	Tarkovski

Affichage des données lues de la BD

- Les données sont en général stockées dans la BD sans protection par rapport aux caractères spéciaux HTML (i.e. sans `htmlspecialchars` préalable)
- Raisons : les données peuvent avoir une autre utilisation que l'affichage HTML
 - ▶ ⇒ les données lues de la bases de données devront être protégées avec `htmlspecialchars` avant l'affichage, tout comme les données d'un formulaire
 - ▶ Exemple : pour afficher une ligne du tableau des Films

```
function afficher_ligne(&$ligne) {  
    echo "<tr>",  
        "<td>". htmlspecialchars($ligne["titre"])."</td>",&br/>        "<td>".htmlspecialchars($ligne["annee"])."</td>",&br/>        "<td>". htmlspecialchars($ligne["realisateur"])."</td>",&br/>        "</tr>";  
}
```

Envoyer des requêtes avec paramètres : une page de recherche

- Dans la plus part des cas les requêtes envoyées à la BD contiennent des paramètres
 - ▶ valeurs récupérés de l'input de l'utilisateur (par exemple à travers un formulaire)
- Exemple :
 - ▶ avec un formulaire l'utilisateur spécifie le nom d'un réalisateur
 - ▶ le serveur récupère ce paramètre et construit une requête qui demande les films de ce réalisateur
 - ▶ la requête est envoyée à la BD, le résultat est récupéré et affiché

05-recherche /recherche.php

/connex.php

/affichage.php

localhost/.../recherche.php

```
1 <?php
2 require_once "affichage.php";
3 require_once "connex.php";
4
5
6
7 $real = $_POST['realisateur'];
8
9 //s'il n'y a aucune demande de recherche en cours on affiche la
... page de recherche
10 if (! isset($real)) {
11     page_recherche();
12     exit;
13 }
14
15 //sinon on traite la demande
16 $connex = connexion_bd();
17
18
19 //on passe le realisateur
20 //à la fonction qui s'occupe de lire de la BD
21 $films = lire_films($connex, $real);
22
23
24 // on traite le resultat : deux cas possibles
25
26 //1) resultat vide
27 if (mysqli_num_rows($films) == 0) page_erreur(ERR_VIDE, '');
28
29 //2) resultat à afficher
30 else page_films($films);
31
32 mysqli_free_result($films);
33
34 mysqli_close($connex);
35
36 ?>
37
38
```

```
1 <?php
2
3 //les modeles
4
5 function connexion_bd() {
6     $serv = "127.0.0.1";
7     $username = "sirangelo";
8     $pwd = "sirangelo";
9     $bd = "sirangelo";
10    $connex = mysqli_connect($serv, $username, $pwd, $bd);
11    if (! $connex) {
12        page_erreur(ERR_CONNEX, mysqli_connect_error($connex));
13    }
14    return $connex;
15 }
16
17 function lire_films($connex, $real) {
18 // $real = mysqli_real_escape_string($connex,$real);
19 $req = "select * from Film where realisateur = '$real.'";
20 $resultat = mysqli_query($connex, $req);
21 //cas de problèmes pendant l'exécution de la requête
22 if (!$resultat) {
23     page_erreur(ERR_REQUETE, mysqli_error($connex));
24     mysqli_close($connex);
25     exit;
26 }
27 return $resultat;
28 }
29
30 ?>
```

```
1 <?php
2
3 //le codes d'erreurs, pour savoir quel type d'erreur afficher
4 define('ERR_CONNEXION', 0);
5 define ('ERR_REQUETE', 1);
6 define ('ERR_VIDE', 2);
7
8 //les fonctions de base pour l'affichage
9 function afficher_entete($titre) {
10 ?>
11     <html>
12     <head>
13     <title> <?=$titre?> </title>
14     <meta charset="utf-8" >
15     <link rel="stylesheet" type="text/css" href="films.css">
16     </head>
17     <body>
18 <?php
19 }
20
21 function afficher_pied_page() {
22     echo "</body></html>";
23 }
24
25 //affiche une ligne de la tables des films
26 //parametre : un tableau associatif representant un film
27 function afficher_ligne(&$ligne) {
28     echo "<tr>",
29         "<td>". htmlspecialchars($ligne["titre"])."</td>",
30         "<td>". htmlspecialchars($ligne["annee"])."</td>",
31         "<td>". htmlspecialchars($ligne["realisateur"])."</td>",
32         "</tr>";
33 }
34
35
36 //affiche la table des films
37 //parametre : la ressource contenant les films à afficher
38 function afficher_table ($films) { ?>
39     <table>
40         <tr> <th>Titre du film</th> <th>Année de
... realisation</th> <th>Réalisateur</th></tr>
41 <?php
42     while($ligne = mysqli_fetch_assoc($films))
... afficher_ligne($ligne);
```

```
43     echo "</table>" ;
44 }
45
46
47
48
49 //les vues
50
51 //la page qui affiche les films disponibles
52 //parametre : la ressource contenant les films à afficher
53 function page_films ($films) {
54     afficher_entete("Films");
55     echo "<h2>Résultat de la recherche</h2>";
56     afficher_table($films);
57     afficher_pied_page();
58 }
59
60 //affiche une page d'erreur
61 //paramteres : le code de l'erreur et un message d'erreur
62 function page_erreur($err_code, $error) {
63     afficher_entete("Erreur");
64     switch ($err_code) {
65         case ERR_CONNEXION:
66             echo "<h2>Desolé, connexion impossible</h2>";
67             break;
68         case ERR_REQUETE:
69             echo "<h2>Erreur dans l'execution de la
... requete</h2>";
70             break;
71         case ERR_VIDE:
72             echo "<h2>Aucun résultat correspond à vos
... critères</h2>";
73             break;
74     }
75     //pour le debogage :
76     echo "<p>".$error."</p>";
77     afficher_pied_page();
78 }
79
80 function page_recherche() {
81     afficher_entete ("Chercher");
82     echo "<h2> Chercher les films d'un réalisateur </h2>"; ?>
83     <form action ="recherche.php" method ="post">
84         Renseigner le nom du réalisateur : <input type="text"
```

```
84... name ="realisateur">
85         <input type="submit" value="Chercher">
86     </form>
87 <?php
88     afficher_pied_page();
89 }
90 ?>
91
92
93
94
```


Chercher les films d'un réalisateur

Renseigner le nom du réalisateur :

Résultat de la recherche

Titre du film	Année de realisation	Réalisateur
Match Point		Allen

Envoyer des requêtes avec paramètres : authentication

- Avec une structure similaire : authentication des utilisateurs sur le site
- La base de données contiendra une table utilisateurs

```
CREATE TABLE Users (  
    login varchar(30) primary key,  
    mdp varchar(6)  
    -- etc. par exemple email, ...  
);
```

- L'utilisateur renseigne login et mdp dans un formulaire
- le script qui gère le login récupère ces valeurs dans les variables `$login`, `$mdp` et prépare la requête :

```
$req = "SELECT * FROM Users WHERE login = '$login' AND mdp = '$mdp'";
```

- Si le résultat est vide \Rightarrow réaffichage du formulaire
- Sinon \Rightarrow page "Bienvenue" (ainsi que affectation des variables de sessions etc.)

```
mysql> source 05-login/user.sql      05-login/      localhost/.../05-login/login.php
```

```
1 create table Users (  
2     login varchar(30) primary key,  
3     mdp varchar(255),  
4     email varchar(80)  
5 );  
6  
7 insert into Users values  
8 ('cristina', 's3cr3t', 'cristina@emal.com'),  
9 ('jean', '3iff3l', 'jean@emal.com');
```

```
1 <?php
2 require_once "affichage.php";
3 require_once "valid.php";
4
5
6 $login = $_POST['login'];
7 $mdp = $_POST['mdp'];
8
9 if (isset($login)) {
10     //alors il y a un login en cours
11     if (verifier_login($login, $mdp)) {
12         page_welcome($login); exit;
13     }
14 }
15 //ici soit il n'y a pas de login en cours soit
16 // les identifiants ne son pas corrects
17 //dans les deux cas on affiche le formulaire de login
18 //(possiblement pre-rempli)
19 page_login($login, $mdp);
20
21
22 ?>
23
24
```

```
1 <?php
2
3 require_once ('connex.php');
4
5 //le modele pour la verification
6
7 function verifier_login($login, $mdp) {
8     $connex = connexion_bd();
9     //on passe login et mdp
10    //à la fonction qui s'occupe de lire de la BD
11    $user = lire_login($connex, $login, $mdp);
12    //on traite le resultat : deux cas possibles
13    //1) resultat vide : login et/ou mdp non valides
14    //2) resultat non vide : utilisateur existant dans la bd et
... bon mdp
15    $success = mysqli_num_rows($user) != 0;
16    mysqli_free_result($user);
17    mysqli_close($connex);
18    return $success;
19 }
20
21 //ici d'autres possible fonction de verification
22 //(format du mot de passe, du login etc)
23 ?>
```

```
1 <?php
2
3 require_once ("affichage.php");
4
5 //les modeles pour la connexion à la bd
6
7 function connexion_bd() {
8     $serv = "127.0.0.1";
9     $username = "sirangelo";
10    $pwd = "sirangelo";
11    $bd = "sirangelo";
12    $connex = mysqli_connect($serv, $username, $pwd, $bd);
13    if (! $connex) {
14        page_erreur(ERR_CONNEX, mysqli_connect_error($connex));
15    }
16    return $connex;
17 }
18
19
20 function lire_login($connex, $login, $mdp) {
21     //$login = mysqli_real_escape_string($connex,$login);
22     //$mdp = mysqli_real_escape_string($connex,$mdp);
23     $req = "select * from Users where login = '$login' AND mdp =
... '$mdp'";
24     $resultat = mysqli_query($connex, $req);
25     //cas de problèmes pendant l'execution de la requête
26     if (!$resultat) {
27         page_erreur(ERR_REQUETE, mysqli_error($connex));
28         mysqli_close($connex);
29         exit;
30     }
31     return $resultat;
32 }
33
34
35 ?>
```

```
1 <?php
2
3 //le codes d'erreurs, pour savoir quel type d'erreur afficher
4 define('ERR_CONNEXION', 0);
5 define ('ERR_REQUETE', 1);
6
7 //les fonctions de base pour l'affichage
8 function afficher_entete($titre) {
9 ?>
10     <html>
11     <head>
12     <title> <?=$titre?> </title>
13     <meta charset="utf-8" >
14     <link rel="stylesheet" type="text/css" href="mon.css">
15     </head>
16     <body>
17 <?php
18 }
19
20 function afficher_pied_page() {
21     echo "</body></html>";
22 }
23
24 //les vues
25
26 //la page de Bienvenue
27 function page_welcome ($login) {
28     afficher_entete("Welcome");
29     echo "<h2>Bienvenue $login </h2>";
30     afficher_pied_page();
31 }
32
33
34
35 function page_login($login, $mdp) {
36     afficher_entete ("Login");
37     if(isset($login)) {
38         echo '<h2 class = "error"> Login et mdp non reconnus!
... </h2>';
39         $login = htmlspecialchars($login);
40         $mdp = htmlspecialchars($mdp);
41     }
42     else echo "<h2> Login </h2>"; ?>
43     <form action ="login.php" method ="post">
```



```
44         Login : <input type="text" name ="login" value ="<?=
... $login ?>">
45         Mot de passe : <input type="password" name ="mdp" value
... ="<?= $mdp?>">
46         <input type="submit" value="OK">
47     </form>
48 <?php
49     afficher_pied_page();
50 }
51
52
53 //affiche une page d'erreur
54 //paramteres : le code de l'erreur et un message d'erreur
55 function page_erreur($err_code, $error) {
56     afficher_entete("Erreur");
57     switch ($err_code) {
58         case ERR_CONNEXION:
59             echo "<h2>Desolé, connexion impossible</h2>";
60             break;
61         case ERR_REQUETE:
62             echo "<h2>Erreur dans l'execution de la
... requete</h2>";
63             break;
64     }
65     //pour le debogage :
66     echo "<p>".$error."</p>";
67     afficher_pied_page();
68 }
69
70 ?>
71
72
73
74
```

Login

Login : Mot de passe :

26/03/2020

Welcome

Bienvenue cristina

Login

Login : Mot de passe :

Login et mdp non reconnus!

Login : Mot de passe :

Sécurité de l'interaction avec la BD

- Pour que l'interaction avec la BD n'ait pas de failles de sécurité il faut protéger les données reçues par l'utilisateur (par exemple par un formulaire) avant de les inclure dans une requête
- Raison : un utilisateur malveillant pourrait inclure du code SQL dangereux dans ce paramètres
- Qu'est-ce qui se passe si l'utilisateur renseigne dans le champ login :
(avec un espace à la fin)



The image shows a web form titled "Login". Below the title, there are two input fields. The first field, labeled "Login :", contains the text " or 1 --". The second field, labeled "Mot de passe :", is empty. To the right of the password field is a button labeled "OK".

- Rappel:
 - ▶ 1 est une condition booléenne toujours vrai en SQL
 - ▶ -- introduit un commentaire SQL

Injection de code SQL

Qu'est-ce qui s'est passé ?

- Rappel : on a construit la requête à envoyer comme :

```
SELECT * FROM Users WHERE login = '$login' AND mdp = '$mdp'
```

- Avec l'input donné par l'utilisateur :

- ▶ `$login` a valeur ' **OR 1 --**

- ▶ `$mdp` a valeur vide

- Donc la requête envoyée a valeur :

```
SELECT * FROM Users WHERE login = ' ' or 1 -- ' AND mdp = ''
```

- Ou bien, avec un peu de coloration syntaxique :

```
SELECT * FROM Users WHERE login = ' ' or 1 -- ' AND mdp = ''
```

Cette requête sélectionne tous les utilisateurs dans la table Users : résultat non vide

Accès accordé!!!

Injection de code SQL : solution

- Chaque module PHP associé à un SGBD offre une fonction qui échappe les caractères spéciaux SQL dans une chaîne de caractères
- pour mysql :

`mysqli_real_escape_string($connex,$str)`

- Utiliser cette fonction sur toutes les chaines de caractères php destinées à être incluse dans une requête SQL !

```
$login = mysqli_real_escape_string($connex,$login);
```

```
$mdp = mysqli_real_escape_string($connex,$mdp);
```

(Remarque : `stripslashes($str)` fait l'opération inverse si nécessaire)

[05-login/connex.php](#)

[localhost/.../05-login/login.php](#)


```
1 <?php
2
3 require_once ("affichage.php");
4
5 //les modeles pour la connexion à la bd
6
7 function connexion_bd() {
8     $serv = "127.0.0.1";
9     $username = "sirangelo";
10    $pwd = "sirangelo";
11    $bd = "sirangelo";
12    $connex = mysqli_connect($serv, $username, $pwd, $bd);
13    if (! $connex) {
14        page_erreur(ERR_CONNEX, mysqli_connect_error($connex));
15    }
16    return $connex;
17 }
18
19
20 function lire_login($connex, $login, $mdp) {
21     $login = mysqli_real_escape_string($connex,$login);
22     $mdp = mysqli_real_escape_string($connex,$mdp);
23     $req = "select * from Users where login = '$login' AND mdp =
24     '$mdp'";
25     $resultat = mysqli_query($connex, $req);
26     //cas de problèmes pendant l'execution de la requête
27     if (!$resultat) {
28         page_erreur(ERR_REQUETE, mysqli_error($connex));
29         mysqli_close($connex);
30         exit;
31     }
32     return $resultat;
33 }
34
35 ?>
```

Un autre exemple : une page d'enregistrement

- Effectuer l'enregistrement d'un nouvel utilisateur : plusieurs étapes
 - ▶ le formulaire d'enregistrement est validé comme vu dans le cours sur le traitement des formulaires
 - ▶ Une fois les données validées, elles sont insérées dans la BD par une requête INSERT INTO

05-enreg /05-enreg.php

/valid.php

/affichage.php

/connex.php

localhost/.../05-enreg.php

```
1
2 <?php
3
4 require_once ("affichage.php");
5 require_once("valid.php");
6 require_once("connex.php");
7
8 $erreurs_requis = array();
9 $erreurs_format = array();
10 $donnees = array();
11
12 if ($_SERVER ["REQUEST_METHOD"]=="POST") {
13     $donnees = $_POST;
14     $ok1 = verifierRequis($donnees, $erreurs_requis);
15     $ok2 = verifierFormat($donnees, $erreurs_format);
16     if ($ok1 && $ok2) {
17         enregistrer ($donnees);
18         page_succes($donnees);
19         //ici en general il serait preferable de rediriger
20         //comme egalement quand on affiche les pages d'erreur
21         exit;
22     }
23 }
24 page_enreg($donnees,$erreurs_requis, $erreurs_format);
25
26 ?>
```

```
1 <?php
2 require_once ("affichage.php");
3
4
5 //les modeles pour la connexion à la bd
6
7 function connexion_bd() {
8     $serv = "127.0.0.1";
9     $username = "sirangelo";
10    $pwd = "sirangelo";
11    $bd = "sirangelo";
12    $connex = mysqli_connect($serv, $username, $pwd, $bd);
13    if (! $connex) {
14        page_erreur(ERR_CONNEX, mysqli_connect_error($connex));
15    }
16    return $connex;
17 }
18
19
20 function enregistrer(&$donnees) {
21     $connex = connexion_bd();
22     //on stocke les données brutes,
23     //on protege uniquement par rapport aux injection sql,
24     //htmlspecialchars uniquement avant l'affichage
25     $login =
26     ... mysqli_real_escape_string($connex,$donnees['login']);
27     $email =
28     ... mysqli_real_escape_string($connex,$donnees['email']);
29     $mdp =
30     ... mysqli_real_escape_string($connex,$donnees['passwd']);
31     //le mot de passe n'est pas stocké en clair
32     // mais haché pour plus de securite
33     $mdp = password_hash($mdp, PASSWORD_DEFAULT);
34     //utiliserr password_verify(<#string password#>, <#string
35     ... hash#>)
36     //en phase de login pour verifier si un mdp correspond à
37     ... celui haché
38
39     //on verifie si un utilisateur avec le meme login existe
40     ... deja
41     $req = "select * from Users where login ='$login'";
42     $resultat = mysqli_query($connex, $req);
```

```
38     if ($resultat && mysqli_num_rows($resultat) == 0) {
39         //enregistrement dans la bd
40         mysqli_free_result($resultat);
41         $req = "insert into Users values ('$login', '$mdp',
... '$email')";
42         $resultat = mysqli_query($connex, $req);
43         if ($resultat) {
44             mysqli_close($connex); return;
45         }
46     }
47     //on est ici seulement en cas d'erreur dans les requetes ou
... login deja existant
48     if (!$resultat) {
49         page_erreur(ERR_REQUETE, mysqli_error($connex));
50     }
51     elseif (mysqli_num_rows($resultat) > 0) {
52         page_erreur(ERR_LOGIN, '');
53     }
54 }
55 mysqli_close($connex);
56 exit;
57
58 }
59
60
61 ?>
```

```
1 <?php
2 //les champs requis
3 $REQUIS["login"] = true;
4 $REQUIS["passwd"] = true;
5 $REQUIS["rpasswd"] = true;
6 $REQUIS["email"] =true;
7
8 function verifierRequis (&$donnees, &$erreur) {
9     global $REQUIS;
10    $ok = true;
11    foreach ($REQUIS as $champ => $valeur) {
12        if (empty(trim($donnees[$champ]))) {
13            $erreur[$champ] = true;
14            $ok = false;
15        }
16    }
17    return $ok;
18 }
19
20 function pwd_valide ($pwd) {
21     return preg_match('/[0-9]/', $pwd);
22     //ou quelque chose de plus compliqué
23 }
24
25 function email_valide ($email) {
26     return filter_var($email, FILTER_VALIDATE_EMAIL);
27 }
28
29 function verifierFormat(&$donnees, &$erreur) {
30     $ok = true;
31     if (!pwd_valide($donnees["passwd"])) {
32         $erreur["passwd"] = true;
33         $ok = false;
34     }
35     if (!email_valide($donnees["email"])) {
36         $erreur["email"] = true;
37         $ok = false;
38     }
39     if ($donnees["passwd"] != $donnees["rpasswd"]) {
40         $erreur["rpasswd"] = true;
41         $ok = false;
42     }
43     return $ok;
44 }
```

```
1 <?php
2 //le codes d'erreurs, pour savoir quel type d'erreur afficher
3 define('ERR_CONNEXION', 0);
4 define ('ERR_REQUETE', 1);
5 define ('ERR_LOGIN', 1);
6
7 //les fonctions de base pour l'affichage
8
9 function afficher_entete($titre) { ?>
10     <!DOCTYPE html>
11     <html>
12         <head>
13             <title> <?=$titre?> </title>
14             <meta charset="utf-8">
15             <link rel="stylesheet" type="text/css"
... href="mon.css">
16         </head>
17         <body>
18             <!-- ici par exemple le menu commun à toutes les pages
... -->
19 <?php
20 }
21
22 function afficher_pied_page() { ?>
23     <!--ici par exemple un footer commun à toutes les pages
... (adresse, contacte,...) -->
24     </body> </html>
25 <?php
26 }
27
28
29
30
31
32 //fonction qui affiche le formulaire,
33 //possiblement pre-rempli avec les données soumise
34 // et complété avec les erreurs eventuels pour chaque champ
35 function afficher_formulaire(&$donnees, &$erreurs) { ?>
36
37     <form action="05-enreg.php" method="post">
38     Login : <input type="text" name="login" size="30"
39             value="<?php echo
... htmlspecialchars($donnees['login'])?>">
40     <span class="error"> <?php echo $erreurs['login']; ?>
```

```
40... </span>
41    <br><br>
42    Email : <input type="text" name="email" size="30"
43              value="<?php echo
... htmlspecialchars($donnees['email'])?>">
44    <span class="error"> <?php echo $erreurs['email']; ?>
... </span>
45    <br><br>
46    Mot de passe : <input type="password" name="passwd"
... size="16"
47              value="<?php echo
... htmlspecialchars($donnees['passwd'])?>">
48    <span class="error"> <?php echo $erreurs['passwd']; ?>
... </span>
49    <br><br>
50    Retaper le mot de passe : <input type="password"
... name="rpasswd" size="16">
51    <span class="error"> <?php echo $erreurs['rpasswd']; ?>
... </span>
52    <br><br>
53    <input type="submit" value="Envoyer" name="go">
54    <input type="reset" value="Effacer">
55    </form>
56
57 <?php
58 }
59
60
61 //fonction qui calcule l'erreur à afficher à coté de chaque
... champ
62 //du formulaire, en fonction des erreurs de champ requi et/ou
... de format
63 function erreurs (&$erreurs_requis, &$erreurs_format) {
64     $erreurs = array();
65     if ($erreurs_format["passwd"])
66         $erreurs["passwd"] = " * le mot de passe doit contenir
... au moins un chiffre ! * ";
67     if ($erreurs_format["rpasswd"])
68         $erreurs["rpasswd"] = "* les mots de passe sont
... différents ! * ";
69     if ($erreurs_format["email"])
70         $erreurs["email"] = "* le format de l'email n'est pas
... valide * ";
71     //plus en general on pourrait avoir un code erreur et
```



```
71... différents erreurs de format
72    //possibles pour le meme champ
73
74    foreach ($erreurs_requis as $champ => $val) {
75        $erreurs[$champ] = " * ce champ est requis ! * ";
76        //l'erreur requis est prioritaire (ecrase l'erreur
... de format)
77    }
78    return $erreurs;
79 }
80
81
82 //la vue pour la page d'enregistrement
83 function page_enreg(&$donnees, &$erreurs_requis,
... &$erreurs_format) {
84     afficher_entete("Enregistrement");
85     echo '<h2> Enregistrement </h2>';
86     $erreurs = erreurs($erreurs_requis, $erreurs_format);
87     afficher_formulaire($donnees, $erreurs);
88     afficher_pied_page();
89 }
90
91 //la vue pour la page de enregistrement effectué
92 function page_succes (&$donnees) {
93     afficher_entete("Félicitations");
94     echo "<h1>Félicitations ".
... htmlspecialchars($donnees['login']).", enregistrement effectué
... !</h1>";
95     //ici par exemple un résumé des informations
... d'enregistrement
96     afficher_pied_page();
97 }
98
99
100 //affiche une page d'erreur
101 //paramteres : le code de l'erreur et un message d'erreur
102 function page_erreur($err_code, $error) {
103     afficher_entete("Erreur");
104     switch ($err_code) {
105         case ERR_CONNEXION:
106             echo "<h2>Desolé, connexion impossible</h2>";
107             break;
108         case ERR_REQUETE:
109             echo "<h2>Erreur dans l'execution de la
```

```
109... requete</h2>";
110         break;
111     case ERR_LOGIN:
112         echo "<h2>Login deja existant</h2>";
113         break;
114     }
115     //pour le deboggage :
116     echo "<p>".$error."</p>";
117     afficher_pied_page();
118 }
119
120 ?>
121
```

Une page d'enregistrement : remarques

- Les données brutes sont insérées dans la BD (htmlspecialchars appelé uniquement avant l'affichage)

```
Login : <input type="text" name="login" size="30"  
        value="<?php echo htmlspecialchars($donnees['login'])?>">
```

...

- Comme toute variable destinée à faire partie d'une requête SQL, les données sont protégées par mysqli_real_escape_string avant l'insertion

```
$login = mysqli_real_escape_string($connex,$donnees['login']);
```

...

Une page d'enregistrement : hachage du mdp

- Le mot de passe n'est pas stocké dans la BD en clair, on en stocke une version hachée

```
$mdp = password_hash($mdp, PASSWORD_DEFAULT);
```

- La fonction `password_hash` “transforme” le mdp pour qu’il devienne indéchiffrable (i.e. à priori très difficile de dériver le mdp d’origine de son chiffrement)
- Elle utilise un sel aléatoire et un algorithme de chiffrement (le deuxième paramètre)
- Ces deux informations, entre autres, sont incluses dans le chiffrement
- En phase de login utiliser la fonction

```
password_verify($mdp, $mdp_haché);
```

pour vérifier que le `$mdp` fourni par l'utilisateur coïncide avec le `$mdp_haché` lu dans la BD

Enregistrement

Login :

Email :

Mot de passe :

Retaper le mot de passe :

```
mysql> select * from users;
```

login	mdp	email
<crisrina>	\$2y\$10\$jS/0IwGrdKEj01hBKjZY80datT58pG573hvXYCgDlSju9dNhfNCv0	crisrna@irif.fr
crisrina	s3cr3t	crisrina@emal.com
jean	3iff3l	jean@emal.com

```
3 rows in set (0.00 sec)
```

Interrogation des données en SQL

Requêtes SQL : forme générique

- Forme générique d'une requête :

SELECT...

FROM ...

WHERE ...

- Au delà des requêtes de base : plusieurs options pour les clauses **SELECT** **FROM** et **WHERE**

Clause WHERE : opérateurs

égalité =

inégalité <>

inférieur <

inférieur ou égal <=

supérieur >

supérieur ou égal >=

intervalle BETWEEN ... AND ...

reconnaissance de motifs LIKE '...'

On peut utiliser les opérateurs booléens NOT, AND et OR

Clause WHERE : BETWEEN

SELECT titre, annee

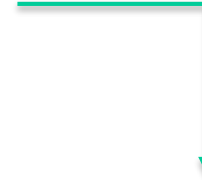
FROM Films

WHERE titre BETWEEN 'Psychose' AND 'Titanic';



Films

<i>titre</i>	<i>annee</i>	<i>realisateu</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7



+	-----	+	-----	+
	titre		annee	
+	-----	+	-----	+
	Pulp Fiction		1995	
	Psychose		1960	
	Titanic		1997	
	Sacrifice		1986	
+	-----	+	-----	+

Clause WHERE : opérateurs booléens

SELECT titre, annee

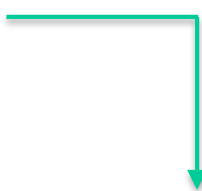
FROM Films

WHERE titre BETWEEN 'Psychose' AND 'Titanic'

OR annee < 1965

Films

<i>titre</i>	<i>annee</i>	<i>realisateu</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7



titre	annee
Vertigo	1958
Pulp Fiction	1995
Psychose	1960
Titanic	1997
Sacrifice	1986

Clause WHERE : LIKE

- A LIKE 'motif'
vrai si la valeur de A est conforme au motif
- 'motif': une chaîne de caractères qui peut inclure les caractères spéciaux "_" et "%"
- "_" : un seul caractère (n'importe lequel)
- "%" : un nombre quelconque de caractères (y compris zéro caractères)
- le motif est insensible à la casse (ne distingue pas minuscules et majuscules)

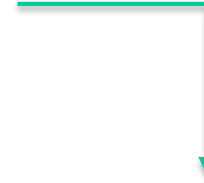
Clause WHERE : LIKE

```
SELECT titre, annee  
FROM Film  
WHERE titre LIKE '%ti%';
```



Films

<i>titre</i>	<i>annee</i>	<i>realisateu</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7



```
+-----+-----+  
| titre          | annee |  
+-----+-----+  
| Pulp Fiction   | 1995  |  
| Vertigo        | 1958  |  
| Titanic        | 1997  |  
+-----+-----+
```

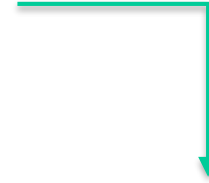
Clause WHERE : LIKE

```
SELECT titre, annee  
FROM Film  
WHERE titre LIKE '_a%';
```



Films

<i>titre</i>	<i>annee</i>	<i>realisateu</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7



titre	annee
Kagemusha	1995
Sacrifice	1958

Clause SELECT : DISTINCT

SELECT DISTINCT annee

FROM Film



Films

<i>titre</i>	<i>annee</i>	<i>realisateu</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7



+-----+
annee
+-----+
1979
1958
1960
1980
1997
1995
1986
+-----+

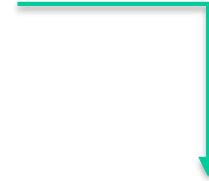
Clause SELECT : renommage

```
SELECT titre AS titre_film  
FROM Films ;
```



Films

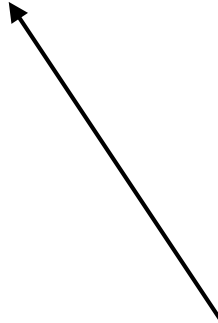
<i>titre</i>	<i>annee</i>	<i>realisateur</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7



```
+-----+  
| titre_film |  
+-----+  
| Alien      |  
| Vertigo    |  
| Psychose   |  
| Kagemusha  |  
| Volte-face |  
| Pulp Fiction |  
| Titanic    |  
| Sacrifice  |  
+-----+
```


Clause SELECT : attributs

```
SELECT Films.titre  
FROM Films
```



Optionnel (sauf en
cas d'ambiguïté)

Clause FROM : accéder à plusieurs tables

- Clause FROM : plusieurs tables dans la forme générale
- But : recomposer de l'information distribuée dans plusieurs tables
- Exemple : obtenir titre et réalisateur de tous les films à partir de ce schéma :

Films

<i>titre</i>	<i>annee</i>	<i>id_realisate</i>
Alien	1979	1
Sacrifice	1986	6

Artistes

<i>id</i>	<i>nom</i>	<i>prenom</i>	<i>naissan</i>
1	Scott	Ridley	1943
2	Hitchcock	Alfred	1899
3	Kurosawa	Akira	1910
4	Woo	John	1946
5	Cameron	James	1954
6	Tarkovski	Andrei	1932

Clause FROM : accéder à plusieurs tables

- la clause FROM avec deux tables renvoie leur “produit” :
 - ▶ chaque ligne de la première table concaténée avec chaque ligne de la deuxième
 - ▶ appelé **produit cartésien**

```
SELECT * FROM Films, Artiste;
```

titre	année	lid_réalisateur	lid	nom	prénom	naissance
Alien	1979	1	1	Scott	Ridley	1943
Alien	1979	1	2	Hitchcock	Alfred	1899
Alien	1979	1	3	Kurosawa	Akira	1910
Alien	1979	1	4	Woo	John	1946
Alien	1979	1	5	Tarkovski	Andrei	1932
Alien	1979	1	6	Cameron	James	1954
Sacrifice	1986	6	1	Scott	Ridley	1943
Sacrifice	1986	6	2	Hitchcock	Alfred	1899
Sacrifice	1986	6	3	Woo	John	1946
Sacrifice	1986	6	4	Kurosawa	Akira	1910
Sacrifice	1986	6	5	Cameron	James	1954
Sacrifice	1986	6	6	Tarkovski	Andrei	1932

Clause FROM : accéder à plusieurs tables

- La plupart du temps le produit cartésien contient des lignes “inutiles”
 - ▶ e.g. : pas significatif de concatener “Alien” avec “Hitchcock”
- On peut utiliser la condition WHERE pour sélectionner uniquement les lignes du produit qui sont “reliées”

```
SELECT * FROM Films, Artistes  
WHERE id_réalisateur = id ;
```

Titre	année	id_réalisateur	id	nom	prénom	naissance
Alien	1979	1	1	Scott	Ridley	1943
Sacrifice	1986	6	6	Tarkovski	Andrei	1932

Clause FROM : accéder à plusieurs tables

- La plupart du temps le produit cartésien contient des lignes “inutiles”
 - ▶ e.g. : pas significatif de concatener “Alien” avec “Hitchcock”
- On peut utiliser la condition WHERE pour sélectionner uniquement les lignes du produit qui sont “reliées”
- Ensuite la clause SELECT pour retenir uniquement les colonnes qui nous intéressent

```
SELECT titre, nom FROM Films, Artistes  
WHERE réalisateur = id ;
```

titre	nom
Alien	Scott
Sacrifice	Tarkovski

Clause FROM : accéder à plusieurs tables

- L'opération de produit cartésien (`FROM Table1, Tables2`) suivie d'une condition de sélection (`WHERE condition`) est appelée **JOINTURE** (JOIN)
- Syntaxe alternative pour la même requête

```
SELECT titre, nom  
FROM Films JOIN Artistes ON (réalisateur = id );
```

Clause FROM : renommage

```
SELECT F.titre  
FROM Films AS F
```

AS optionnel

Renommage des tables nécessaire si la même table est présente plusieurs fois dans la partie FROM, pour pouvoir distinguer les attributs

```
SELECT F1.titre  
FROM Film F1, Film F2  
WHERE F2.annee > F1.annee;
```

(les films qui ne sont pas le plus récents)

Utilisation de fonctions prédéfinies dans les requêtes

On peut utiliser des fonctions dans les requêtes

Exemple : dans les clauses SELECT, WHERE ou dans une expression pour affecter des valeurs à des champs

Pour la plupart ce sont des ajouts de MySQL à la norme SQL

Exemples :

ABS(num) : valeur absolue

CONCAT(str, [str2, ...]) : concaténation des chaînes

NOW() : la date et heure courante

...

Utilisation de fonctions prédéfinies dans les requêtes

```
SELECT CONCAT ('réalisateur : ', nom)
FROM Artistes ;
```



```
+-----+
| concat('realisateur : ', nom)|
+-----+
| réalisateur : Scott          |
| réalisateur : Hitchcock     |
| réalisateur : Kurosawa      |
+-----+
```

```
SELECT YEAR(NOW()) - naissance AS age
FROM Artistes ;
```



```
+-----+
| age  |
+-----+
| 73   |
| 117  |
| 106  |
+-----+
```



Artistes

<i>id</i>	<i>nom</i>	<i>prenom</i>	<i>naissan</i>
1	Scott	Ridley	1943
2	Hitchcock	Alfred	1899
3	Kurosawa	Akira	1910

Quelques fonctions MySQL

CEILING(num)

Renvoie l'entier immédiatement supérieur ou égal à num

FLOOR(num)

Renvoie l'entier immédiatement inférieur ou égal à num

CURDATE()

Renvoie la date courante AAAAMMJJ ou AAAA-MM-JJ

CURTIME

Renvoie l'heure courante HHMMSS ou HH:MM:SS

DATE_FORMAT(date, format)

Formate date selon format

Quelques fonctions MySQL

IF(test, val1, val2)

Renvoie val1 si test est vrai, val2 sinon

INSTR(str, substr)

Position de substr dans str

LENGTH(str)

Renvoie la longueur de str

STRCMP(str1, str2)

0 si égalité, -1 si str1 < str2, +1 sinon

Trier les résultats des requêtes : ORDER BY

```
mysql> SELECT titre, annee
```

```
-> FROM Films
```

```
-> WHERE titre BETWEEN 'Psychose' AND 'Titanic'
```

```
-> ORDER BY titre;
```

titre	annee
Pulp Fiction	1995
Psychose	1960
Sacrifice	1986
Titanic	1997

Trier les résultats des requêtes : ORDER BY

Note : le tri n'est pas lié à BETWEEN

On peut faire un tri sur plus d'une colonne

On peut trier dans l'ordre croissant (ASC) ou décroissant (DESC)

Exemple :

Liste les films par année et, dans une année, par ordre alphabétique inverse

```
SELECT annee, titre  
FROM Films  
ORDER BY annee ASC, titre DESC;
```

Résultat

```
SELECT annee, titre FROM Films  
ORDER BY annee ASC, titre DESC;
```

annee	titre
1958	Vertigo
1960	Psychose
1979	Alien
1980	Kagemusha
1986	Sacrifice
1995	Pulp Fiction
1997	Volte-face
1997	Titanic

Une condition de WHERE plus complexe : IN

Rechercher des attributs appartenant à un ensemble :

```
SELECT titre FROM Films
```

```
WHERE nom IN ('Hitchcock','Scott', 'Kurosawa');
```

Plus simple qu'une suite de OR

```
SELECT titre FROM film-simple
```

```
WHERE nom = 'Hitchcock' OR nom = 'Scott' OR nom = 'Kurosawa');
```

Une condition de WHERE plus complexe : IN

Plus intéressant : les valeur de l'ensemble dans lequel rechercher peuvent être le résultat d'une (sous-) requête

```
SELECT id_realisateur FROM Films  
WHERE titre IN (SELECT titre FROM Notation WHERE note > 5);
```

La condition IN peut être combinée avec d'autres conditions à l'aide des opérateurs booléens AND, OR, NOT

```
SELECT nom FROM Films, Artiste  
WHERE id = id_realisateur  
AND titre NOT IN (SELECT titre FROM Notation);
```

D'autre conditions complexes introduisent des sous-requêtes,...cf. cours BD L3

Un mot sur la modélisation des données

Modéliser plusieurs concepts

Rappel : la table Films

Films

<i>titre</i>	<i>annee</i>	<i>realisateur</i>
Alien	1979	Scott
Vertigo	1958	Hitchcock
Psychose	1960	Hitchcock
Kagemusha	1980	Kurosawa
Volte-face	1997	Woo
Pulp Fiction	1995	Tarantino
Titanic	1997	Cameron
Sacrifice	1986	Tarkovski

Et si on voulait représenter plusieurs informations sur les réalisateurs (nom, prénom, date de naissance, ...) ?

Modéliser plusieurs concepts

Pourquoi pas tout mettre dans la meme table?

Films

<i>titre</i>	<i>annee</i>	<i>realisateur</i>	<i>prenom</i>	<i>naissance</i>
Alien	1979	Scott	Ridley	1943
Vertigo	1958	Hitchcock	Alfred	1899
Psychose	1960	Hitchcock	Alfred	1899
Kagemusha	1980	Kurosawa	Akira	1910
Volte-face	1997	Woo	John	1946
Pulp Fiction	1995	Tarantino	Quentin	
Titanic	1997	Cameron	James	1954
Sacrifice	1986	Tarkovski	Andrei	1932

Problèmes avec la table simple

Redondance

Les informations sur les réalisateurs sont répétées pour chaque film qu'ils ont réalisé.

Anomalies d'insertion

Possibilité d'insérer des données incohérentes

exemple : le même réalisateur avec deux dates de naissance différentes

Anomalies de mise à jour

Si on a besoin de rectifier une erreur sur l'année de naissance, il faut penser à le faire pour tous les films. Sinon, la table contient des informations incohérentes...

Anomalies de suppression

La suppression d'un film de la table, entraîne la suppression des informations associées sur le réalisateur.

Si le réalisateur n'était présent que pour un seul film, la suppression de ce film entraîne la disparition de toutes les informations relatives au réalisateur.

Solution

Utiliser plusieurs tables pour représenter les films et les réalisateurs indépendamment les uns des autres

insertions, mises-à-jour et destructions indépendantes.

- Identifier les films (et les réalisateurs) pour s'assurer qu'aucun doublon ne figure dans nos tables
 - ▶ Films: 2 films ne peuvent avoir le même titre (supposons-le)
 - ▶ Réalisateurs : 2 réalisateurs peuvent avoir le même nom; on les distingue grâce à un identificateur (*id*)
- Lier les films et les réalisateurs sans introduire de redondance d'information

Solution

Ajout d'un attribut dans la table film : “*realisateur*”

Il n'y a plus de redondance dans la base de données :

Films

<i>titre</i>	<i>annee</i>	<i>realisateu</i>
Alien	1979	1
Vertigo	1958	2
Psychose	1960	2
Kagemusha	1980	3
Volte-face	1997	4
Pulp Fiction	1995	5
Titanic	1997	6
Sacrifice	1986	7

Realisateurs

<i>id</i>	<i>nom</i>	<i>prenom</i>	<i>naissanc</i>
1	Scott	Ridley	1943
2	Hitchcock	Alfred	1899
3	Kurosawa	Akira	1910
4	Woo	John	1946
5	Tarantino	Quentin	
6	Cameron	James	1954
7	Tarkovski	Andrei	1932

Solution

Insertion

Les informations concernant un même réalisateur sont présentes une seule fois dans la base : pas possible de stocker des informations incohérentes (e.g. deux dates de naissance différentes)

ou bien il s'agit d'un réalisateur différent !

Mise à jour

Plus de redondance, donc une mise à jour ne risque pas d'introduire d'incohérence

Suppression

La suppression d'un film n'affecte pas le réalisateur

Modélisation

Remarque :

la modélisation ne concerne que le schema de la base de données
pas les données (lignes)

Films

titre	annee	realisateur
--------------	-------	-------------

Realisateurs

id	nom	prenom	naissance
-----------	-----	--------	-----------

Une schema de base de données plus complexe

- On veut représenter:
 - ▶ Des films,
 - ▶ Les réalisateurs et les acteurs qui jouent,
 - ▶ Les pays où ces films ont été réalisés,
 - ▶ Des utilisateurs du site des films
- Permettre aux utilisateurs de noter les films

Solution : ébauche

- Les informations concernant les acteurs et les réalisateurs seront vraisemblablement les mêmes (nom, prénom, année de naissance)
 - ▶ on les représente avec une unique table Artistes

Artistes			
id	nom	prenom	naissance

- Avec les films on représente l'id de l'artiste qui en est le réalisateur

Films		
titre	annee	id_realisateur

- Comment représenter les acteurs qui jouent dans un film sans redondance?
 - ▶ Pourquoi la solution adoptée pour le réalisateur (ajouter son id dans la table Films) n'est pas valable?

Solution : ébauche

- Un film a plusieurs acteurs, un attribut id_acteur peut représenter une seule valeur!
- Solution : une nouvelle table qui fait le “lien” entre films et acteurs
 - ▶ Seuls les identifiants dans cette table, pour éviter la redondance

Cast

titre_film	id_acteur
------------	-----------

Films

titre	annee	id_realisateur
--------------	-------	----------------

Artistes

id	nom	prenom	naissance
-----------	-----	--------	-----------

- Et si on voulait représenter également les rôles des acteurs dans les film ?

Solution : ébauche

- Le rôle n'est pas associé au film ou à l'acteur, mais à la participation de l'acteur dans un film.
 - ▶ attribut de la table Cast

Cast

titre_film	id_acteur	rôle
------------	-----------	------

Films

titre	annee	id_realisateur
-------	-------	----------------

Artistes

id	nom	prenom	naissan
----	-----	--------	---------

Solution : ébauche

- Représentation des pays :

Pays		
code	nom	langue

- Et des utilisateurs : chaque utilisateur a un pseudo, nom, prénom, mdp, mais également un pays
 - ▶ comment représenter le pays?

Solution : ébauche

- Représentation des pays :

Pays		
code	nom	langue

- Et des utilisateurs : chaque utilisateur a un pseudo, nom, prénom, mdp, mais également un pays
 - ▶ comment représenter le pays?

Utilisateurs					
pseudo	email	nom	prenom	mdp	code_pays

- Comment représenter les notes que les utilisateurs donnent aux films?

Solution : ébauche

- Un utilisateur peut noter plusieurs films et un film peut être noté par plusieurs utilisateurs
 - ▶ \Rightarrow la note ne peut pas être un attribut du film, ni de l'utilisateur
- Solution : une nouvelle table qui fait le “lien” entre films et utilisateurs
 - ▶ Seuls les identifiants dans cette table, pour éviter la redondance
 - ▶ la note est un attribut additionnel de cette table

Notation

titre_film	pseudo	note
------------	--------	------

Solution : schema complet

Films

titre	annee	id_realisateur
-------	-------	----------------

Artistes

id	nom	prenom	naissance
----	-----	--------	-----------

Cast

titre_film	id_acteur
------------	-----------

Pays

code	nom	langue
------	-----	--------

Utilisateurs

pseudo	email	nom	prenom	mdp	code_pays
--------	-------	-----	--------	-----	-----------

Notation

titre_film	pseudo	note
------------	--------	------

Ce schéma sera ensuite implementé dans le SGBD avec un suite de commandes CREATE TABLE, après avoir choisi le type de chaque attribut

Modèles E/A

Pour simplifier le processus de modélisation en général on ne cherche pas à trouver les bonnes tables directement (comme dans l'exemple précédent)

On s'appuie sur des **modèles** dits “**Entités / Associations**” (ou E/A)

- modèles E/A (1976) à la base de méthodes de conception comme OMT (UML)
- plus haut-niveau que le modèle relationnel
- notions d'entité pour représenter les données d'intérêt et d'association pour représenter comment elles sont reliées

Il existe ensuite des règles qui nous permettent de traduire un schéma E/R en un schéma relationnel, et définir donc les tables de la base

cf. cours BD L3