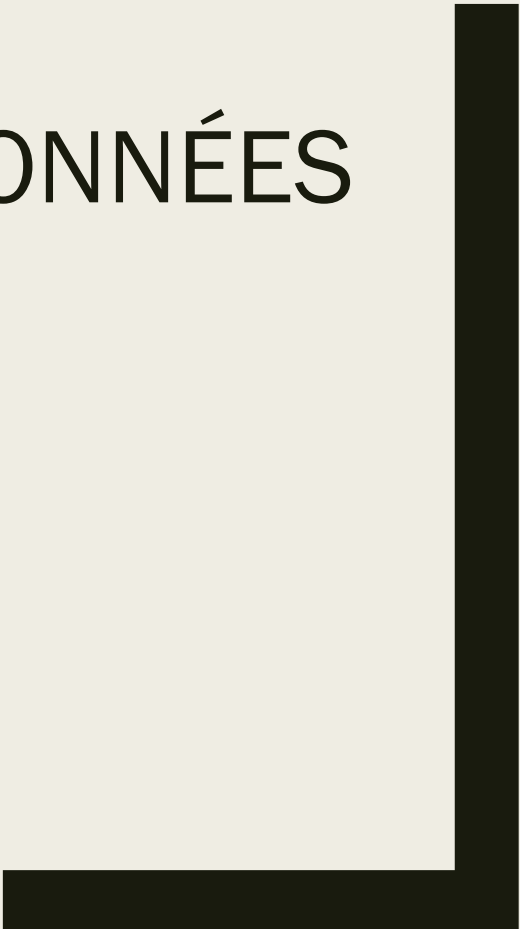




WEB ET BASES DE DONNÉES

Sarah BEN OTHMAN
Sara.ben-othman@ec-lille.fr



Plan

- Une base de données, pourquoi?
- Mysql
- Interconnexion avec PHP : exemples
- Connexions multiples (odbc)

**Une base de données :
pourquoi?**

Rôle d'une base de données

- Manipulation de grandes quantités d'information
 - *représentation des données de l'application*
 - *interrogation des données (requêtes, langage)*
 - *optimisation des requêtes*
- Sécurité (pannes)
 - *Gestion de droits d'accès*
- Accès multiples en simultané

Modèle relationnel

- Modèle le plus courant
- données = table
- Le modèle relationnel représente la base de données comme un ensemble de tables, sans préjuger de la façon dont les informations sont stockées dans la machine. Les tables constituent donc la structure logique du modèle relationnel.

Nom	Prenom	Email	CB
Gross	David	dgram@cnam.fr	33203343
John	Jimmy	truc@bidule.com	45443321
Onyme	Anne	onyme@chez.com	44543343

- Typage des données
- requête : SQL (standard galactique)

MySQL : un sgbd à la mode

Qu'est-ce que MySQL ?

- C'est un SGBD relationnel
 - *Gratuit (www.mysql.com), sous la GNU GPL*
 - *Disponible sous Windows et tous les Unix*
 - *Propose le minimum vital*
 - (Presque) tout le langage SQL
 - Stockage, indexation, optimisation
 - *Qualités reconnues (outre la gratuité):*
 - Simplicité
 - Efficacité
 - Robustesse

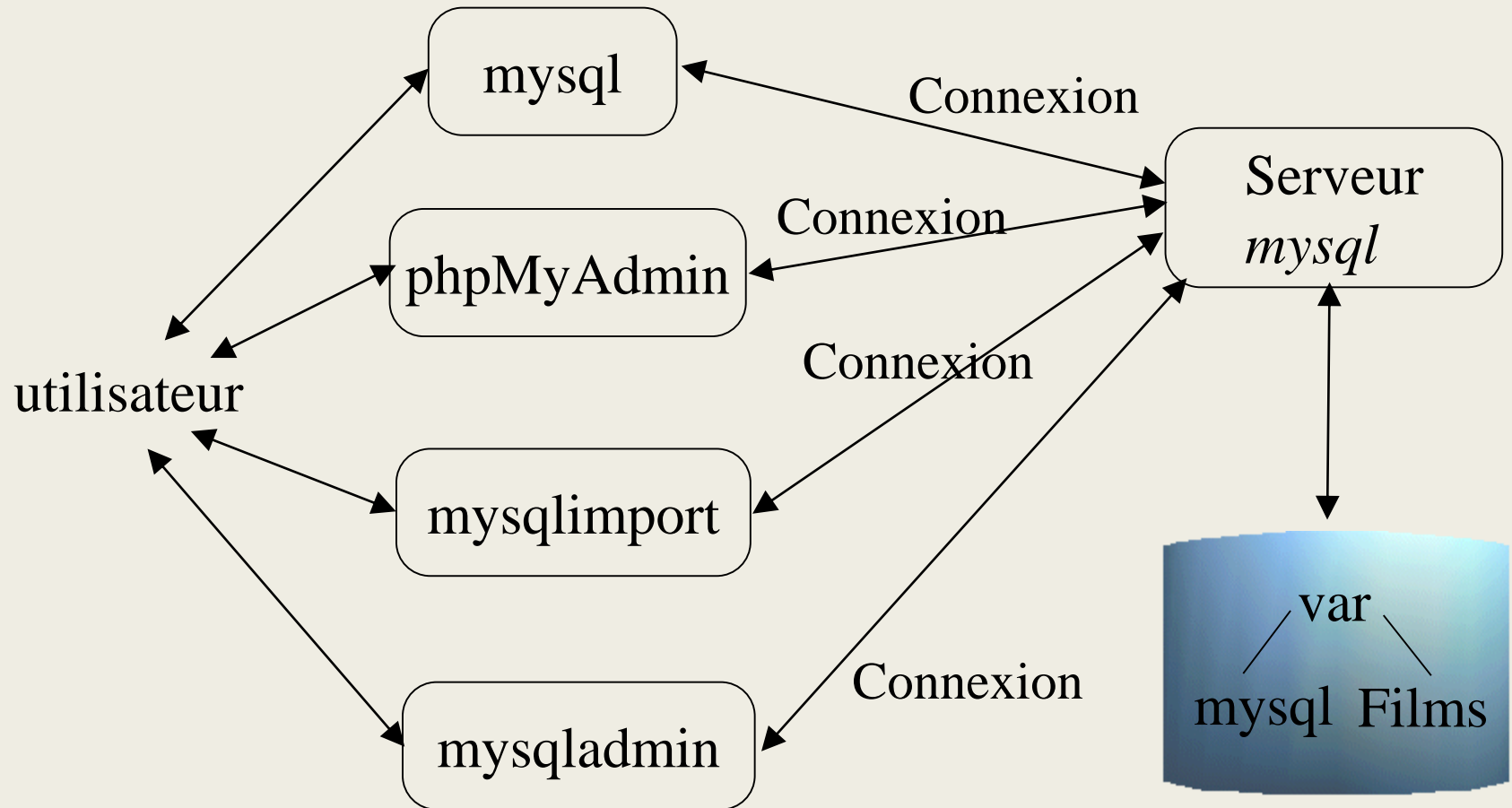
Quelques limites de MySQL

- Pas de transactions !
 - *Inadapté pour les applications transactionnelles (finances, réservations)*
 - *Ni commit, ni rollback, une reprise sur panne rudimentaire*
- Pas de requêtes imbriquées
 - *Pas très grave !?*
- Pas d'environnement de développement
 - *Pas de générateur d'écran ou de rapport, pas d'outil pour le DBA, etc. Juste une API en C !*

Statut de MySQL à l'heure actuelle

- Très prisé pour les sites web
 - *Très bonne intégration avec Apache et PHP*
 - *Tout est gratuit !*
- En fort développement
 - *Intégration des transactions ?*
- Beaucoup de « contributions » extérieures
 - *API en C++, Perl, PHP, Java (JDBC)*
 - *Des clients graphiques, des utilitaires*
- En résumé, un moteur SQL, simple et efficace

Architecture de MySQL

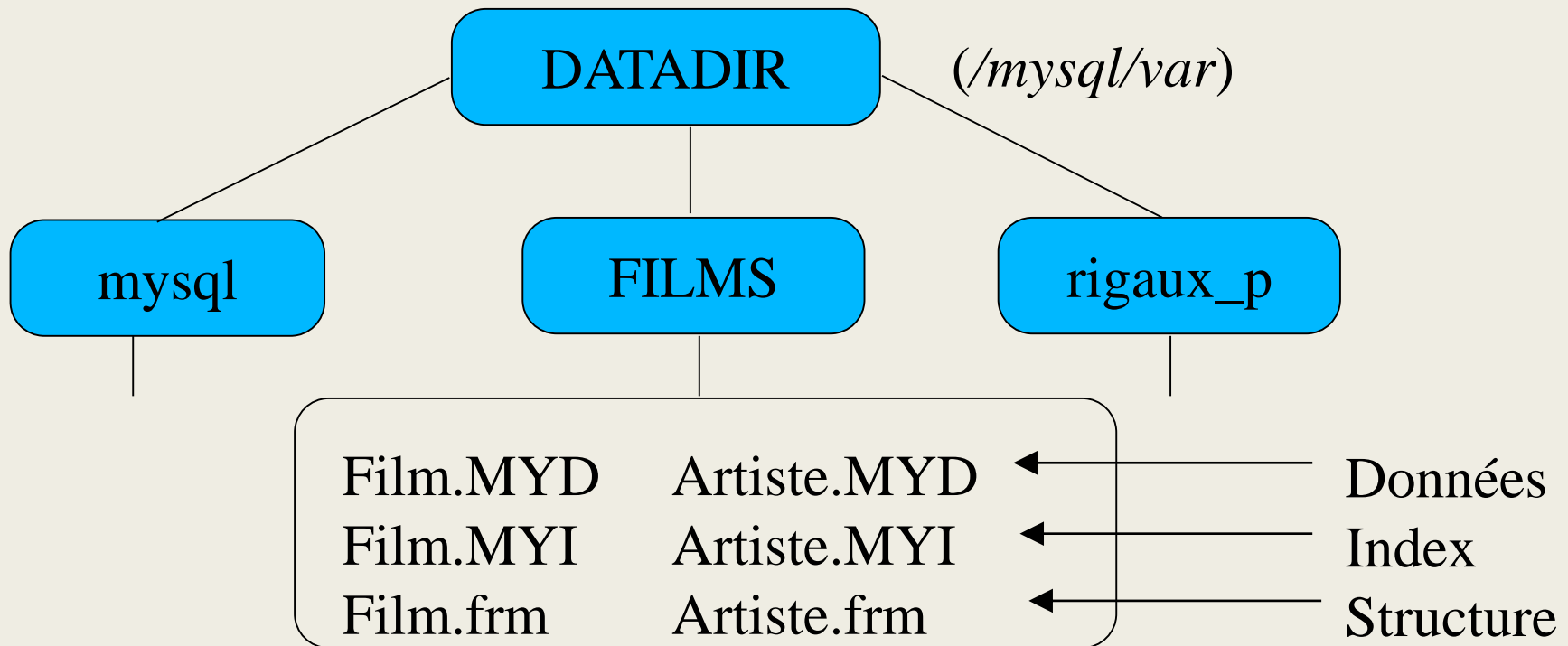


Le serveur *mysql*

- Il est en écoute sur un port réseau
- Il gère une ou plusieurs bases de données
 - *La base mysql est le «dictionnaire de données»*
 - *Les autres bases sont des bases utilisateur*
- Il dialogue avec ses clients en *multi-threading* (processus léger)

Les bases de données

- Une base = un répertoire !
- Une table = 3 fichiers !



Le client *mysql*

- Accéder et utiliser Mysql en ligne de commande peut se révéler pratique pour diverses opération de traitement en masse, ou de sauvegarde / restauration notamment.
- Permet de se connecter sous un nom d'utilisateur:
mysql -u root -p Films
-> se connecte sous *root* (MySQL) à la base *Films*, avec un mot de passe
- Permet d'entrer toutes les commandes (en fonction des droits d'accès biensûr)

Utilisateur : Les droits

- Un utilisateur peut avoir plusieurs types de droits:
 - *sur le serveur (administration)*
 - *sur les bases et tables (création, destruction)*
 - *sur les données (lecture, écriture)*
 - *sur d'autres utilisateurs (transmission de droits)*
- Le compte *root* est le DBA: il a tous les droits sur tout

Création d'utilisateurs : la commande GRANT

- Pour créer un utilisateur avec tous les droits sur la base Films:

*GRANT ALL ON Film.**

TO admin@localhost

IDENTIFIED BY 'mdpAdmin'

- Pour créer un utilisateur avec des droits en lecture sur la table Artiste:

GRANT select ON Film.Artiste

TO visiteur@localhost

IDENTIFIED BY 'mdpVisiteur'

Création de bases et de tables

- On crée des bases sous *root* :

```
% mysql -u root -p
```

```
mysql> CREATE DATABASE Films;
```

- Tout utilisateur ayant des droits peut créer des tables dans une base:

```
% mysql -u adminFilms -p Films
```

```
mysql> CREATE TABLE Film (...);
```

- Les commandes SQL *CREATE*, *ALTER*, *DROP* sont reconnues

Types de données MySQL

- MySQL reconnaît les principaux, types SQL2, SQL3 et quelques autres:
 - *INTEGER* : les entiers
 - *FLOAT* : numériques flottants
 - *DECIMAL (M, D)* : numériques exacts
 - *CHAR* : chaînes de longueur fixe
 - *VARCHAR* : chaînes de longueur variable
 - *TEXT* : textes longs
 - *DATE* : dates
 - *TIME* : moments

Exemple de création de table

```
CREATE TABLE Film
```

```
(id INT NOT NULL AUTO_INCREMENT
```

```
titre VARCHAR (50) NOT NULL,
```

```
annee INTEGER NOT NULL,
```

```
idMES INTEGER,
```

```
resume TEXT,
```

```
codePays VARCHAR (4),
```

```
PRIMARY KEY (titre),
```

```
FOREIGN KEY (idMES) REFERENCES Artiste,
```

```
FOREIGN KEY (codePays) REFERENCES Pays);
```

Insertion dans une table

```
mysql> CREATE TABLE MaTable
```

```
-> (MaDonnee INTEGER NOT NULL,
```

```
-> PRIMARY KEY (MaDonnee));
```

```
Query OK, 0 rows affected (0.02 sec)
```

```
mysql> INSERT INTO MaTable (MaDonnee) VALUES (1);
```

```
Query OK, 1 row affected (0.05 sec)
```

- Il vaut mieux placer toutes les commandes dans un fichier de script,
MonF.sql

Exemple d'un fichier de script

```
# Création d'une table 'FilmSimple'
```

```
USE Films;
```

```
CREATE TABLE FilmSimple  
  (titre    VARCHAR (30),  
   annee    INTEGER,  
   nomMES   VARCHAR (30),  
   prenomMES VARCHAR (30),  
   anneeNaiss INTEGER  
  )  
;
```

MySQL en résumé

- Tout à fait conforme à la norme SQL ANSI
 - *CREATE TABLE, DROP TABLE, ALTER TABLE*
 - *SELECT, INSERT, DELETE, UPDATE*
- Avec des extensions (très utiles)
 - *Types de données (TEXT, BLOB)*
 - *Contraintes (AUTO_INCREMENT)*
 - *Beaucoup de fonctions*

Programmation MySQL/PHP

Rappel : langage PHP

- Du code inséré dans une page HTML
- Situé entre balises `<?php` et `?>`
- Code exécuté par le serveur
- Résultat : une nouvelle page HTML

Rappel : langage PHP

<HTML>

<BODY>

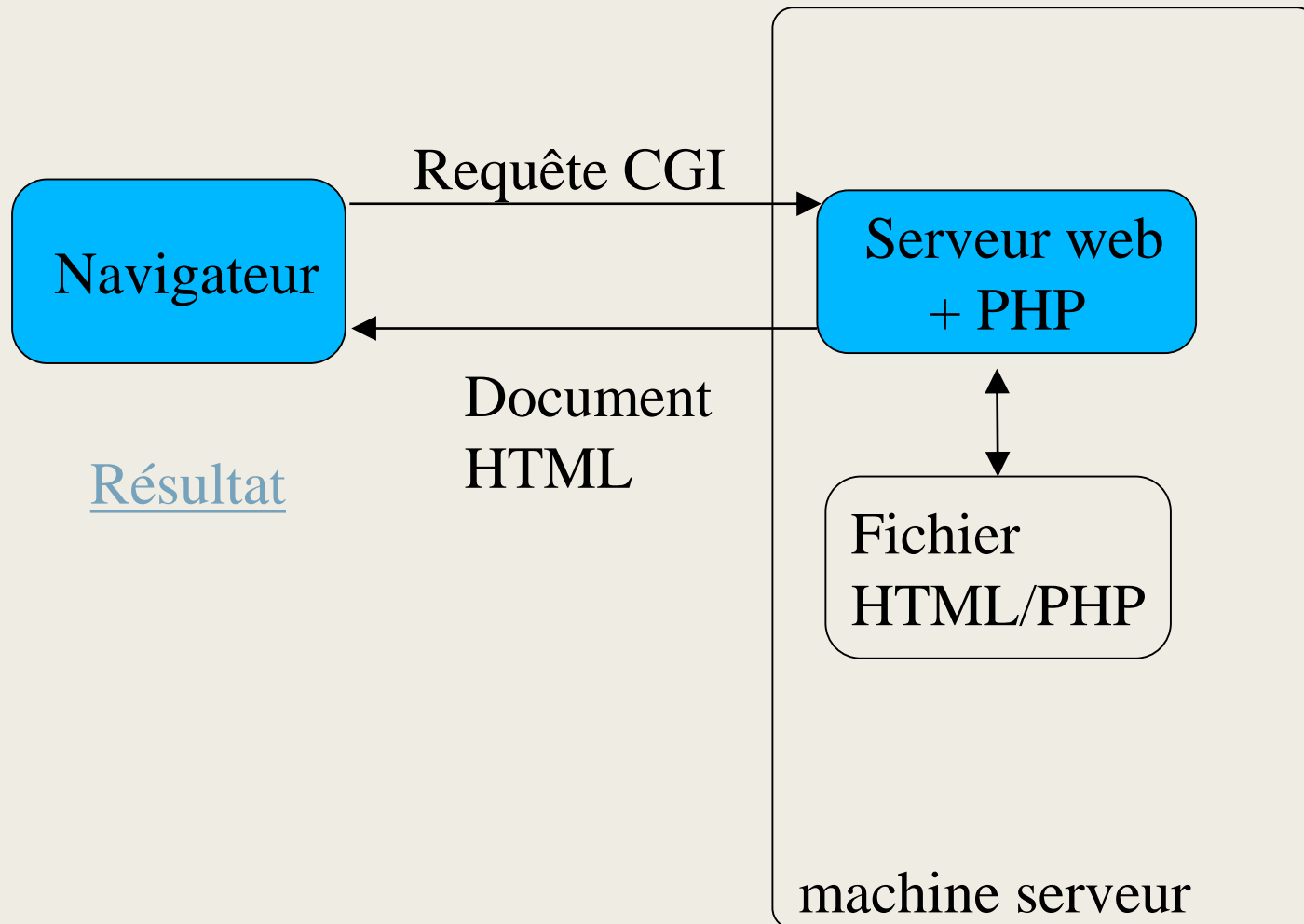
<H1>Bienvenue ! </H1>

Il est <?php echo date('h'); ?> h à ma montre.

</BODY>

</HTML>

Architecture avec base de données

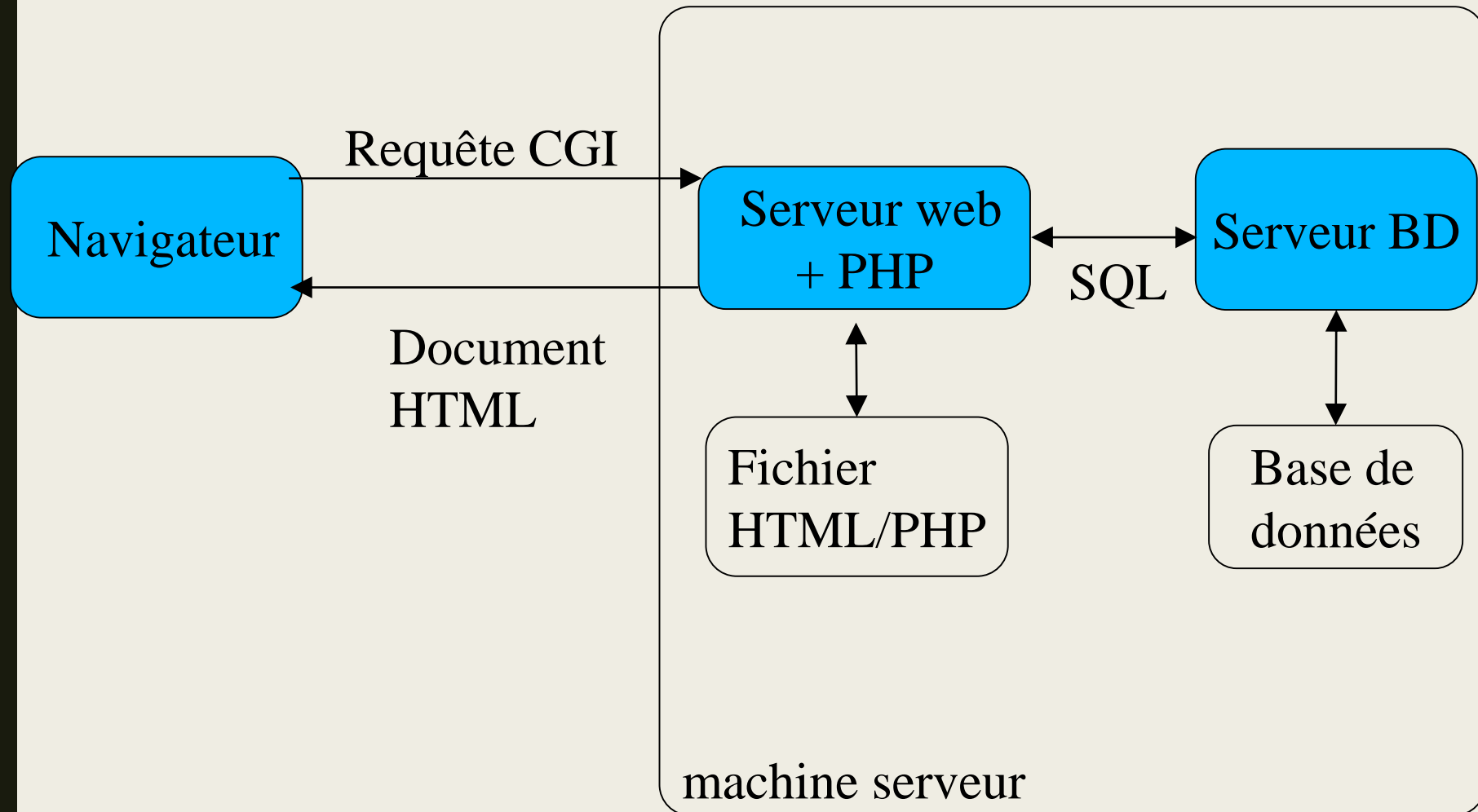


CGI/Bases de données

Application avec MySQL/PHP

- Principe : création de documents web à partir d'une BD
 - *MySQL se charge du stockage, de la protection des données, de l'interface SQL*
 - *PHP :*
 - extrait des données et les met en forme
 - reçoit des données et les stocke
 - *Le navigateur fournit l'interface graphique*
- Architecture à trois pôles, à la sauce Web

Architecture avec base de données



Connexion à MySQL avec un script PHP

- *mysql_pconnect (serveur, nom, passe)*
 - *établit une connexion*
 - *si OK, renvoie un identifiant non nul cnx*
- *mysql_select_db(base, cnx)*
 - *se place dans une base et renvoie vrai si OK*
- *mysql_query (requete, cnx)*
 - *exécute une requête et renvoie un identifiant*
- *mysql_fetch_object (resultat)*
 - *renvoie la ligne suivante sous forme d'objet*

Programmation MySQL/PHP : premier exemple

Connexion à la base MaBase (base de films)

Affichage du titre, année et metteur en scene du film

Exemple : interrogation et affichage (1)

```
<?php
$connexion = mysql_pconnect ("localhost", "MonNom",
                             "MonMDP");

if (!$connexion)
{
    echo "Désolé, connexion impossible\n";
    exit;
}

if (!mysql_select_db ("MaBase", $connexion))
{
    echo "Désolé, accès à la base impossible\n";
    exit;
}
```

Exemple : interrogation et affichage (2)

```
$resultat = mysql_query ("SELECT * FROM FilmSimple",  
                           $connexion);  
  
if ($resultat)  
    while ($film = mysql_fetch_object ($resultat))  
        echo "$film->titre, paru en $film->annee, réalisé "  
            . "par $film->prenomMES $film->nomMES.<BR>\n";  
else  
{  
    echo "<B>Erreur dans l'exécution de la requête.</B><BR>";  
    echo "<B>Message :</B> " .mysql_error($connexion);  
}
```

Programmation MySQL/PHP : exemple avec formulaire

En association avec un formulaire

<H1>Formulaire pour programme PHP</H1>

<FORM ACTION='ExMyPHP2.php' METHOD=POST>

<P>

Titre : <INPUT TYPE=TEXT SIZE=20 NAME = 'titre'> <P>

Année début : <INPUT TYPE=TEXT SIZE=4 NAME='anMin'>

Année fin : <INPUT TYPE=TEXT SIZE=4 NAME='anMax'> <P>

<P>

Comment combiner ces critères.

ET <INPUT TYPE=RADIO NAME='comb' VALUE='ET'>

OU <INPUT TYPE=RADIO NAME='comb' VALUE='OU'> ?

<INPUT TYPE=SUBMIT VALUE='Rechercher'>

</FORM>

On récupère donc les variables \$titre, \$anMin, \$anMax, et \$comb

```
<?php
```

```
echo "<B>Titre = $titre anMin = $anMin anMax= $anMax\n";
```

```
echo "Combinaison logique : $comb<P></B>\n";
```

```
if ($comb == 'ET')
```

```
    $requete = "SELECT * FROM FilmSimple "
```

```
        . "WHERE titre LIKE '$titre' "
```

```
        . "AND annee BETWEEN $anMin and $anMax";
```

```
else
```

```
    $requete = "SELECT * FROM FilmSimple "
```

```
        . "WHERE titre LIKE '$titre' "
```

```
        . "OR (annee BETWEEN $anMin and $anMax)";
```

Il reste à exécuter la requête

```
<?php
```

```
$connexion = mysql_pconnect (SERVEUR, NOM, PASSE);
```

```
mysql_select_db (BASE, $connexion);
```

```
$resultat = mysql_query ($requete, $connexion);
```

```
while ( ($film = mysql_fetch_object ($resultat)))
```

```
    echo "$film->titre, paru en $film->annee, réalisé "
```

```
        . "par $film->prenomMES $film->nomMES.<BR>\n";
```

```
?>
```

Programmation MySQL/PHP : mise à jour d'une base

Mises à jour de la base

- On utilise un formulaire de saisie, et on déclenche:
 - *Un ordre INSERT pour des insertions*
 - *Un ordre UPDATE pour une modification*
 - *Un ordre DELETE pour une destruction*
- Dans tous les cas la fonction *mysql_query* permet d'exécuter l'ordre.

Exemple: mise à jour de la table FilmSimple

<FORM ACTION="ExMyPHP3.php" METHOD=POST>

Titre : <INPUT TYPE=TEXT SIZE=20 NAME="titre">

Année : <INPUT TYPE=TEXT SIZE=4 NAME="annee"> <P>

Nom : <INPUT TYPE=TEXT SIZE=20 NAME="prenom">

Prénom : <INPUT TYPE=TEXT SIZE=20 NAME="nom">

Année : <INPUT TYPE=TEXT SIZE=4 NAME="anneeNaissance">

<H1>Votre choix</H1>

<INPUT TYPE=SUBMIT VALUE='Insérer' NAME='inserer' >

<INPUT TYPE=SUBMIT VALUE='Modifier' NAME='modifier' >

<INPUT TYPE=SUBMIT VALUE='Détruire' NAME='detruire' >

</FORM>

L'action dépend du bouton choisi par l'utilisateur

```
// Test du type de mise à jour effectuée
```

```
echo "<HR><H2>\n";
```

```
if (isset($inserer))    echo "Insertion du film $titre";
```

```
elseif (isset($modifier)) echo "Modification du film $titre";
```

```
elseif (isset($detruire)) echo "Destruction du film $titre";
```

```
echo "</H2><HR>\n";
```

```
// Affichage des données du formulaire
```

```
echo "Titre: $titre <BR> annee: $annee<BR>\n";
```

```
echo "Mis en scène par $prenom $nom, né en $anneeNaiss\n";
```

Choix de la requête en fonction de l'action demandée

```
if (isset($inserer))
```

```
    $requete = "INSERT INTO FilmSimple (titre, annee, "  
        . "prenomMES, nomMES, anneeNaiss) "  
        . "VALUES ('$titre', $annee, "  
        . "'$nom', '$prenom', $anneeNaiss) ";
```

```
if (isset($modifier))
```

```
    $requete = "UPDATE FilmSimple SET annee=$annee, "  
        . "prenomMES = '$prenom', nomMES='$nom', "  
        . "anneeNaiss=$anneeNaiss WHERE titre = '$titre' ";
```

```
if (isset($detruire))
```

```
    $requete = "DELETE FROM FilmSimple WHERE titre='$titre'";
```

```
$resultat = mysql_query ($requete, $connexion);
```


Programmation MySQL/PHP : requêtes sur la structure de la base

Informations sur le schéma

- Etant donné le résultat d'une requête :
 - *mysql_num_fields(\$res)* donne le nombre d'attributs
 - *mysql_field_name(\$res, \$i)* donne le nom de l'attribut
 - *mysql_fetch_row (\$res)* renvoie une ligne du résultat sous la forme d'un tableau indicé.

Programmation MySQL/PHP : gestion de sessions

Comment associer à un utilisateur web une
grande quantité d'information

Problèmes

- Le serveur ne distingue pas les connexions successives :
comment identifier un utilisateur ?
- Comment faire pour que les données relatives à un utilisateur
soit disponibles pour tous les scripts du site web ?

Identification de l'utilisateur

- On utilise les cookies
 - *utilisateur saisit (login,password)*
 - *le navigateur enregistre les cookies (login,password)*
 - *chaque page du site teste l'existence de ces cookies -> on connaît ainsi l'utilisateur*

Données associées à l'utilisateur

- On ne peut pas mettre toutes les données dans les cookies (taille, sécurité)

-> base de données

- Solution :
 - *Identification de l'utilisateur -> numero de session unique*
 - *cookies (login,passwd,numero session)*

Données associées à l'utilisateur

Login	session	limite	donnée1	donnée2	...
dgram	552	5mn	rateau	3	
truc	127	2mn	pelle	1	
					...

- Chaque script retrouve les données associées grâce au numéro de session

-> requête SQL

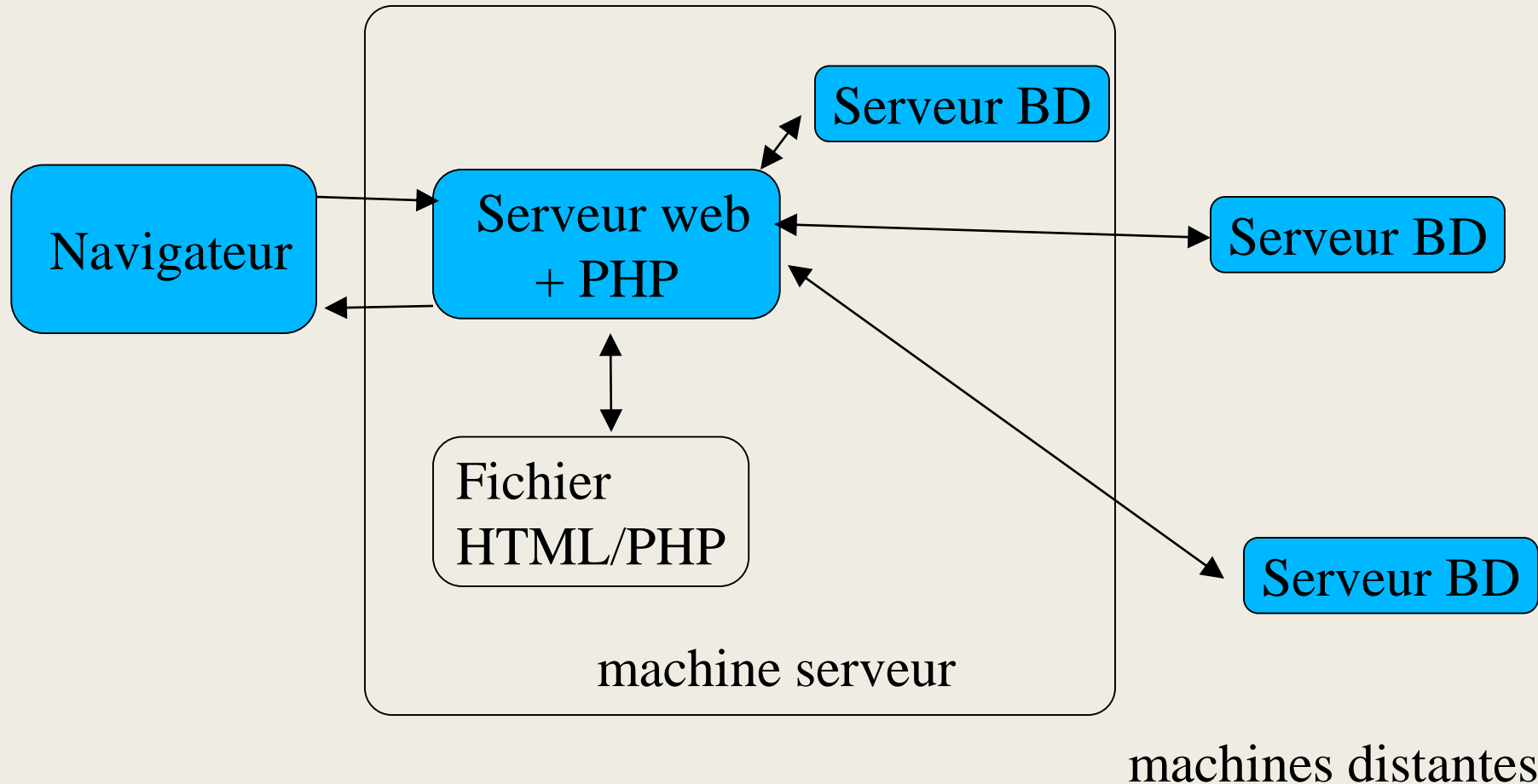
- Ajout/suppression d'information

Connexions :

- A plusieurs bases
- A plusieurs SGBD

- Intégration de données
- Vitesse de connexion

Interroger plusieurs bases Mysql



Plusieurs connexion php

```
<?php
```

```
    $connexion1 = mysql_pconnect (serveur1, nom1, pass1);  
    mysql_select_db (base1, $connexion1);
```

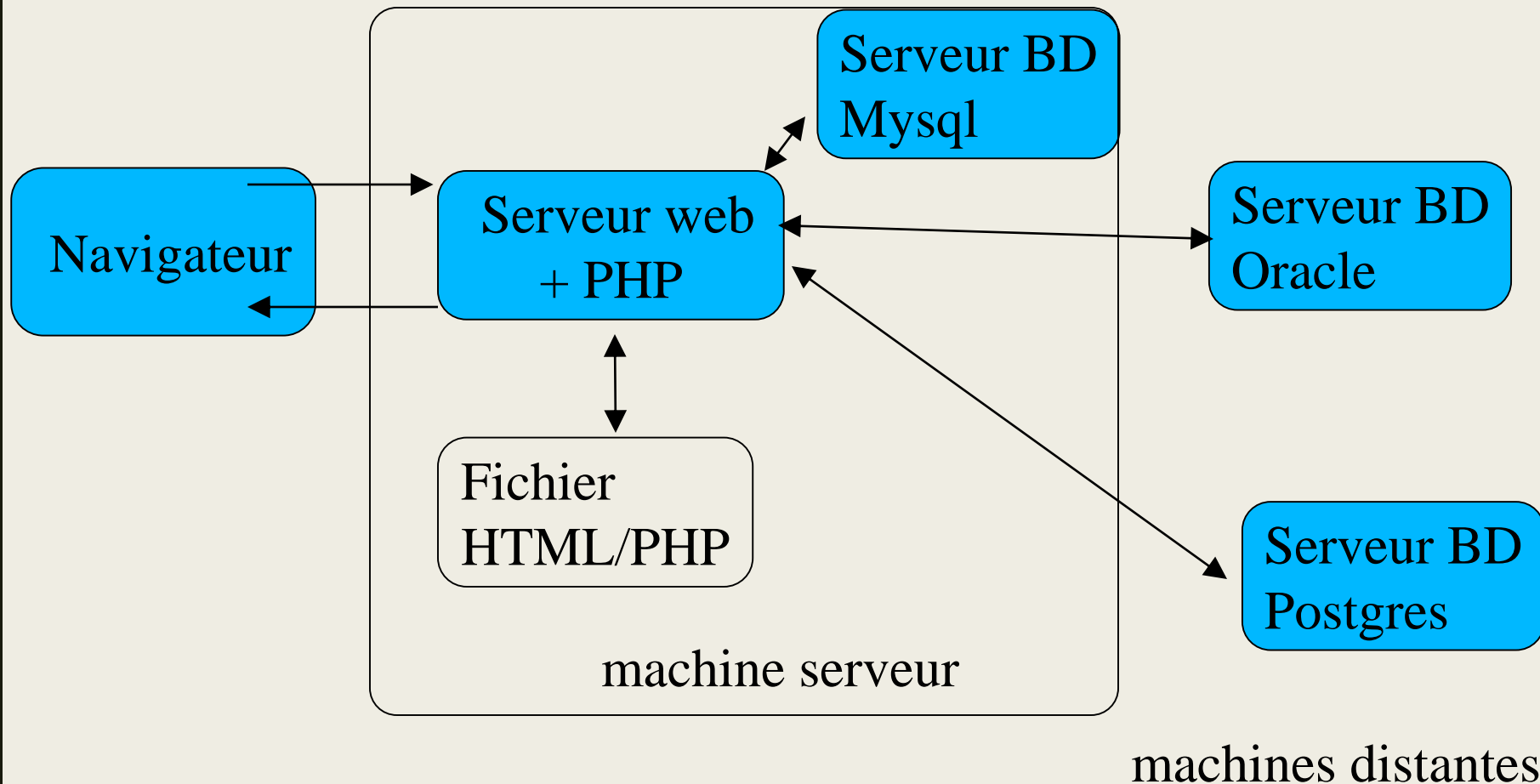
```
    $connexion2 = mysql_pconnect (serveur2, nom2, pass2);  
    mysql_select_db (base2, $connexion2);
```

```
    $resultat1 = mysql_query ($requete1, $connexion1);  
    $resultat2 = mysql_query ($requete2, $connexion2);
```

```
    traitement($resultat1,$resultat2);
```

```
?>
```

Et si la base n'est pas Mysql ?



Interface commune

- ODBC (Open Database Connectivity)
- Fonctions php valable pour tout SGBD
- `odbc_connect(db,login,pass);`
- `odbc_exec(requete,connexion);`
- `odbc_fetch_row(connexion);`

Descripteur (aperçu)

```
<?php
```

```
$connexion1 = odbc_connect ("mysql:localhost",...);
```

```
$connexion2 = odbc_connect ("oracle:db.fournisseur.fr"...);
```

```
$connexion3 = odbc_connect ("postgres:128.176.212.3",...);
```

```
?>
```

Conclusion

- Mysql : sgbd libre, puissant, répandu
- Interconnexion avec Php :
 - *production de documents HTML à partir de requêtes SQL*
 - *gestion des interactions avec l'utilisateur (requêtes/données de formulaires)*
- Odbc