Université Sorbonne Paris Nord

IUT BOBIGNY

R213 : Développement Web

Jérôme Nobécourt

jerome.nobecourt@gmail.com

Sur le réseau pédagogique http://tomcat/Portail

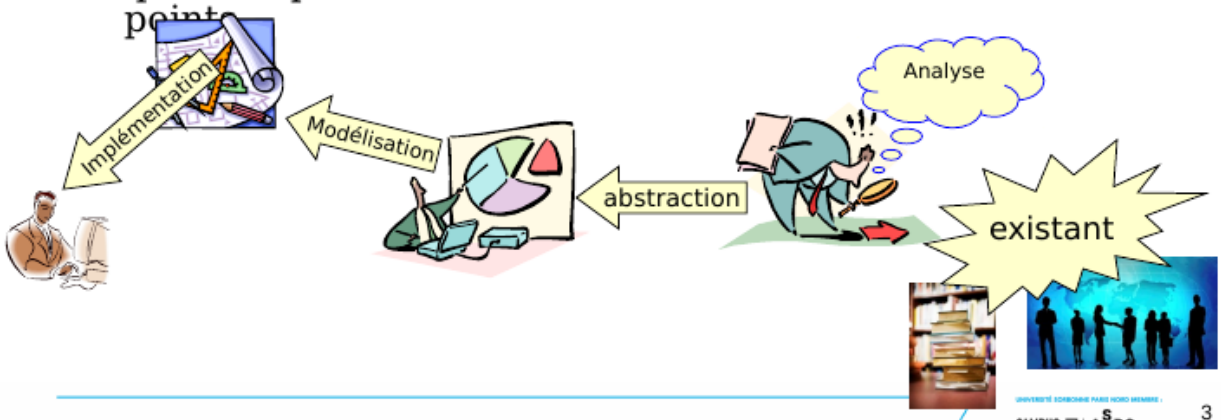
PHP : Programmation Objet, Exemple Utilisation de PDO.

Rappel : Analyse et Conception des Systèmes d'Informations

- Modéliser des données
- Implémenter ce MCD dans un modèle logique
- Savoir interroger et retrouver des données : SQL et requêtes

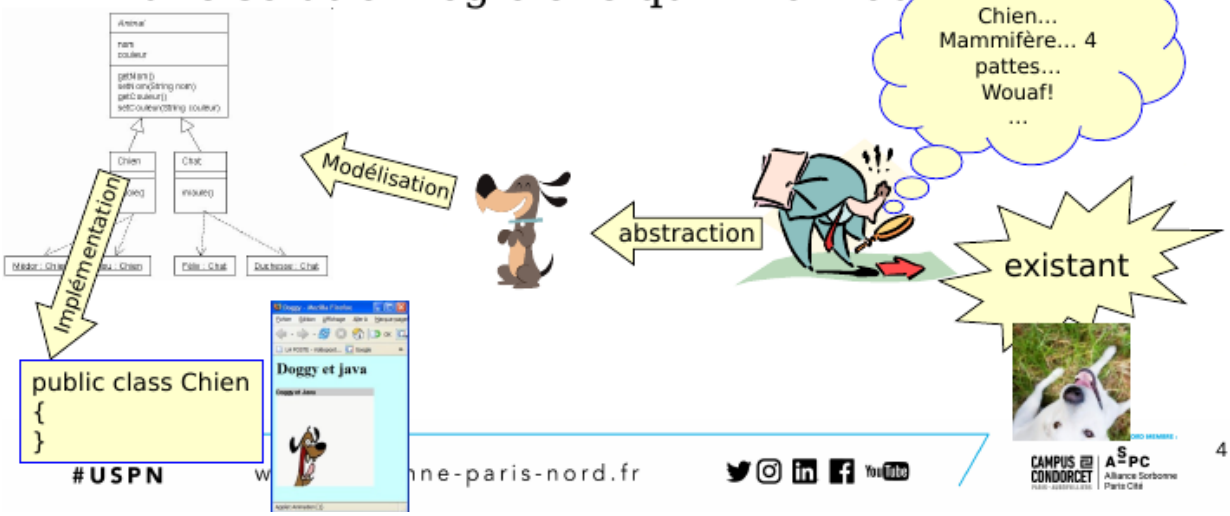
Analyser avant de concevoir

- Vous aurez généralement jamais de problème clef en main.
- Il faut toujours analyser ce que vous devez faire
- Cette analyse vous amènera a une liste de questions qui vous permettra de centrer vos efforts sur certains points



Analyser avant de concevoir (2)

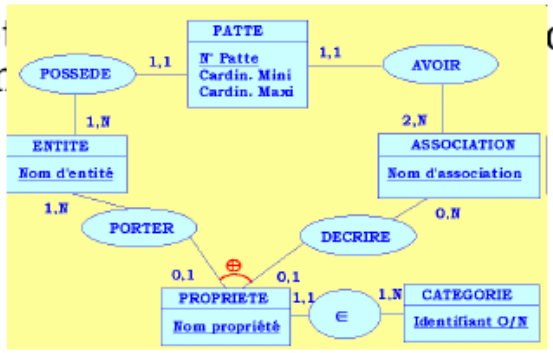
- Le processus est cyclique
- Il y a de nombreux retours arrières
- Le résultat n'est jamais la réalité, seulement une solution logicielle qui « va » au cli



MCD= Modéliser (E/A ou UML)

Libellé de la propriété	Nom du champ	Type	Dimension
Code du client	Code client	Numéro auto	3
Titre du client	Titre	C	15
Nom du client	Nom	C	30
Nom de la société	Société	C	30
Rue du client	Rue	C	50
Code postal du client	Code postal	C	5
Ville du client	Ville	C	30
Téléphone du client	Téléphone	C	15
Client particulier	Part culier	L	1
Référence de la location	Ref location	C	3
Date de début de location	Date début location	D	10
Date de fin de location	Date fin location	D	10
Date effective de retour	Date effective retour	D	10
Code du matériel	Code matériel	C	3
Catégorie du matériel	Catégorie	C	10
Désignation du matériel	Désignation	C	50
Modèle du matériel	Modèle	C	10
Marque du matériel	Marque	C	20
Tarif par jour	Tarif par jour	N	5
Caution	Caution	N	5
Quantité louée	Quantité	N	1

- Vous devez toujours analyser l'univers du discours et en faire un dictionnaire des données AVANT
- Vous pouvez alors regrouper ces données en paquet cohérent.
- Ces paquet oit des associatior



Représenter et opérationnaliser (implémentation)

- Vous devez toujours passer par un MCD **AVANT**.
- Il existe des règles de traduction automatique d'un MCD en un MLD. Regardez vos cours de SI (R214)

- Si vous avez un problème de représentation des données dans la BD et que vous avez suivie les règles de traduction, alors le problème ce situe au niveau du MCD ou du dictionnaire des données.

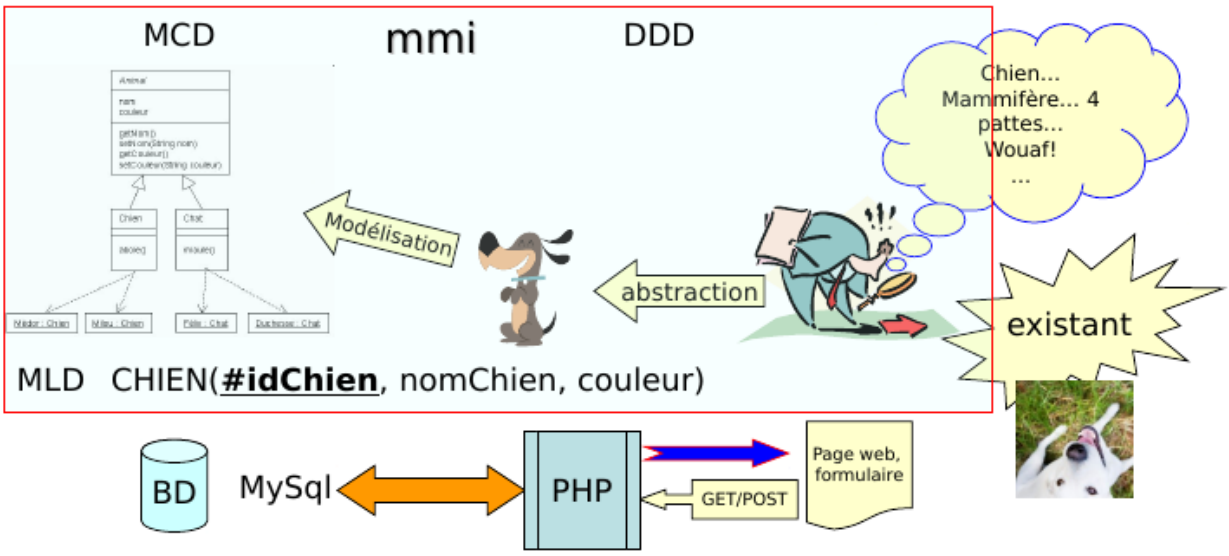
- Pour qu'une BD soit efficace elle **DOIT** respecter au moins les 3 premières formes normales (sur 6+1)
  - 1FN : les attributs sont atomiques et ne se répètent pas
  - 2FN : les attributs ne dépendent pas d'une partie de la clef primaire
  - 3FN : A est la clef, B un attribut, C un autre attribut de la même entité. Si A=>B=>C alors on n'est pas en 3FN.

Allez la pour plus de détail :
https://info.usherbrooke.ca/mlafond/IFT187/IFT187-C20-21-normalisation-b.pdf

SGBDR : MySQL

- En mmi, nous utiliserons le SGBD MariaDB. Une version totalement open source de MySQL (oracle).
- Pour faire simple, nous parlerons de MySQL même si le moteur utilisé est MariaDB.
- Il n'y a quasiment aucune différence
  - https://kinsta.com/fr/blog/mariadb-mysql/
- Vous devez savoir traduire toute question en requête MySQL.
- Vous devez connaître vos cours de R214 !
- Vous pouvez aller sur ce lien pour vous rafraichir la mémoire
  - https://mysql.developpez.com/cours/
- Vous devez savoir faire du SQL en ligne de commande, PHP permet d'interroger une BD MAIS vous devez lui donner les commandes à faire
- 95% des stages sur la programmation web utilisent MySQL ou son équivalent
- 100% des CMS que vous utiliserez en entreprise utilisent des Bds

Vue générale

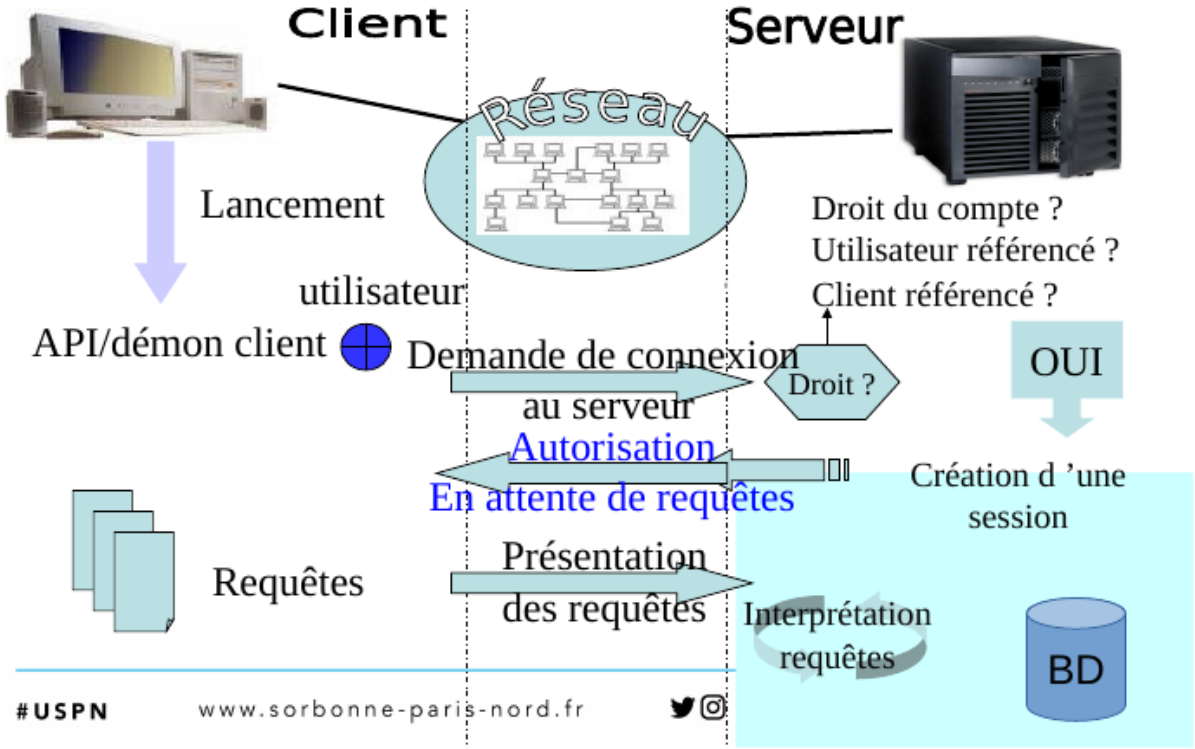


PHP : programmation de BD

- plus exactement : utilisation d'un langage de programmation pour interroger un SGBDR
- supports : API clientes d'interrogation de SGBDR

  + réseau(x)
  + serveur SGBDR
- besoins :
  - API : langage de programmation + module d'interrogation, démon client d'interrogation
  - réseau : protocole de communication entre client/serveur
  - serveur SGBDR : gestion de sessions, gestion des comptes utilisateurs et sécurité des données

Principe du client/serveur en BD



Principe de la création d'API pour l'interrogation

- Serveur
  - création d'une interface de communication entre le serveur du SGBD et les requêtes provenant des clients (p. ex. traduction de requêtes)
- Client
  - mode conversationnel : développement d'une IHM et des fonctionnalités par type de commandes + module de communication
  - mode ligne de commande : langage de commande
  - mode batch : API dédiées à un ensemble de traitements

Principe d'utilisation d'API pour l'interrogation

- Serveur
  - un démon est lancé ;
  - il teste les ports de communications ;
  - il capture les requêtes ;
  - il traite les requêtes.
- Client
  - mode conversationnel : l'application est lancée, l'utilisateur ne voit que l'interface, pas les requêtes. Exemple : réservation d'un article dans du e-commerce.
  - mode ligne de commande : un interpréteur de commandes est lancé. L'utilisateur donne les commandes grâce à ce shell.
  - mode batch : le fichier batch est lancé. Il n'y a pas d'interaction avec un utilisateur. Exemple : scripts de mise à jour.

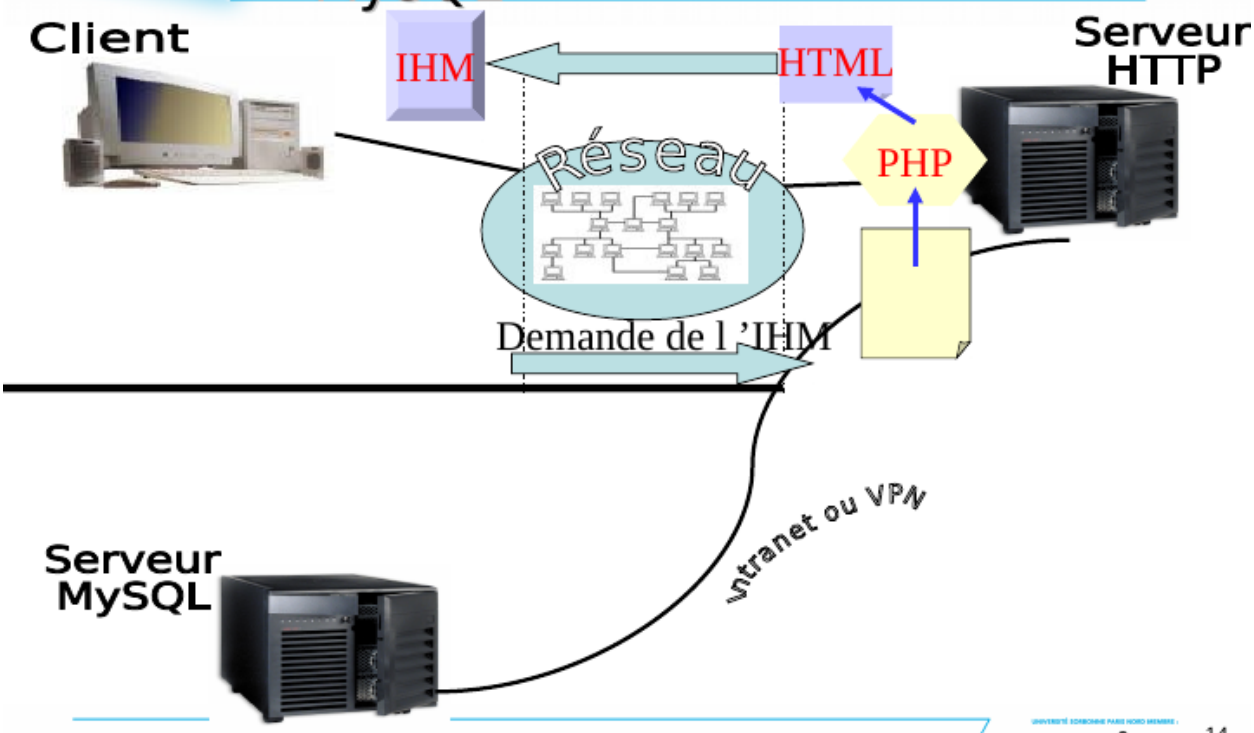
PHP/MySQL : évolution des API

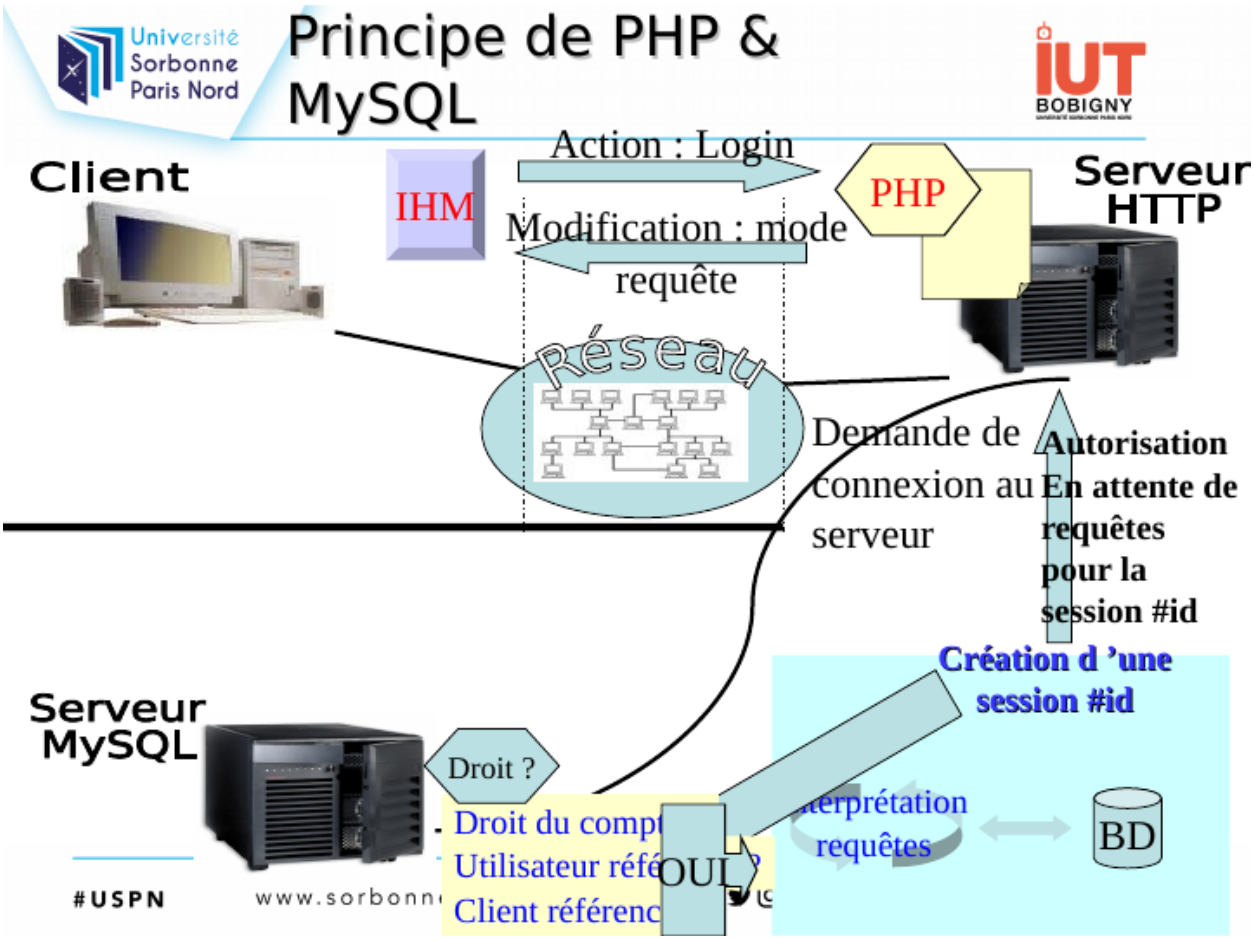
- API en PHP
  - il existe des fonctions PHP pour gérer des BD MySQL
  - ces fonctions (environ 50) commencent toutes par le prefixe **mysql_**
  - **En PHP > 5.5, il faut utiliser PDO, toutes les fonctions mysql_ sont obsolètes.**
    - **Il est possible de passer par l'API MySQli pour traduire les anciens sites réalisés avec les commandes mysql_ par des commandes mysqli_ :**

      https://www.linuxtricks.fr/wiki/php-passer-de-mysql-a-mysqli-requetes-de-base

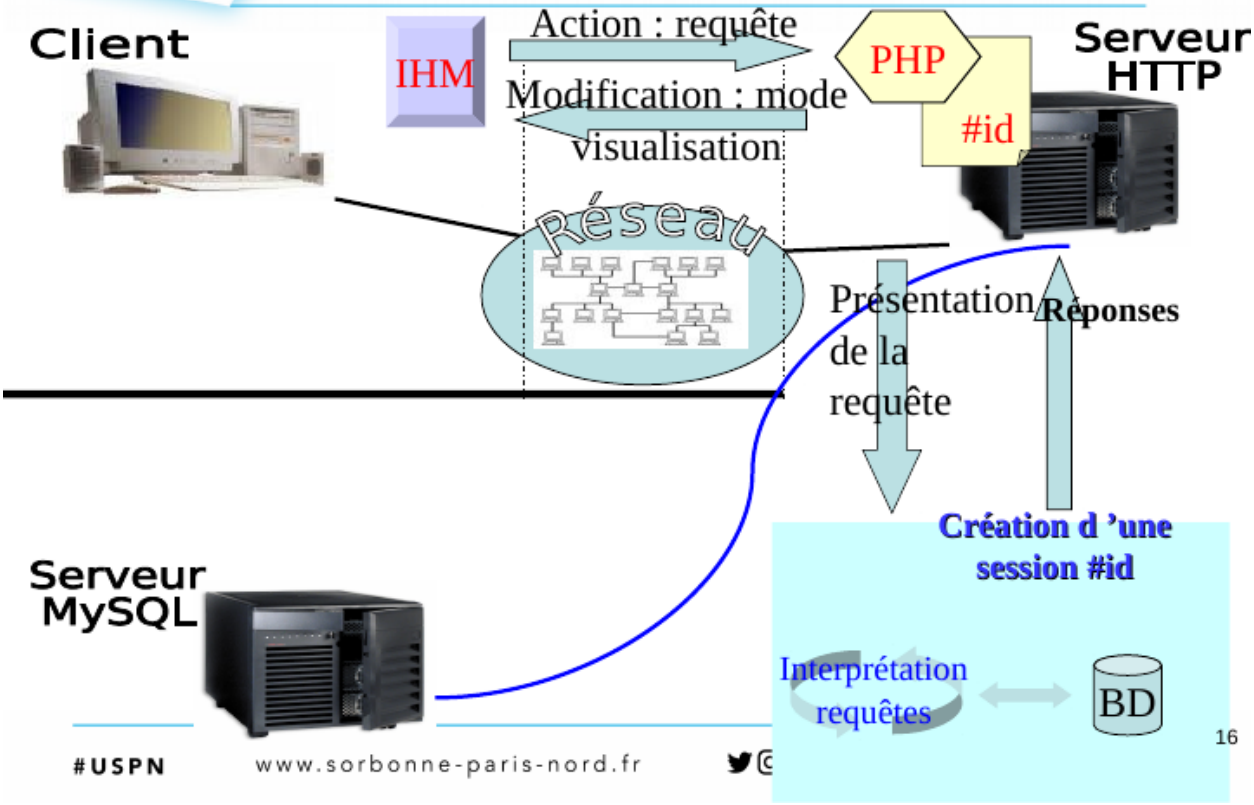
- PDO en PHP:
  - établir la communication avec le démon MySQL/SQLi/Postgresql/... ;
  - préparer les requêtes et les soumettre au démon ;
  - récupérer les réponses du SGBD ;
  - présenter les résultats sur le browser client ;
  - créer et gérer une IHM de consultations et de requêtes à partir d'un poste client (phpMyAdmin).

Principe de PHP & MySQL

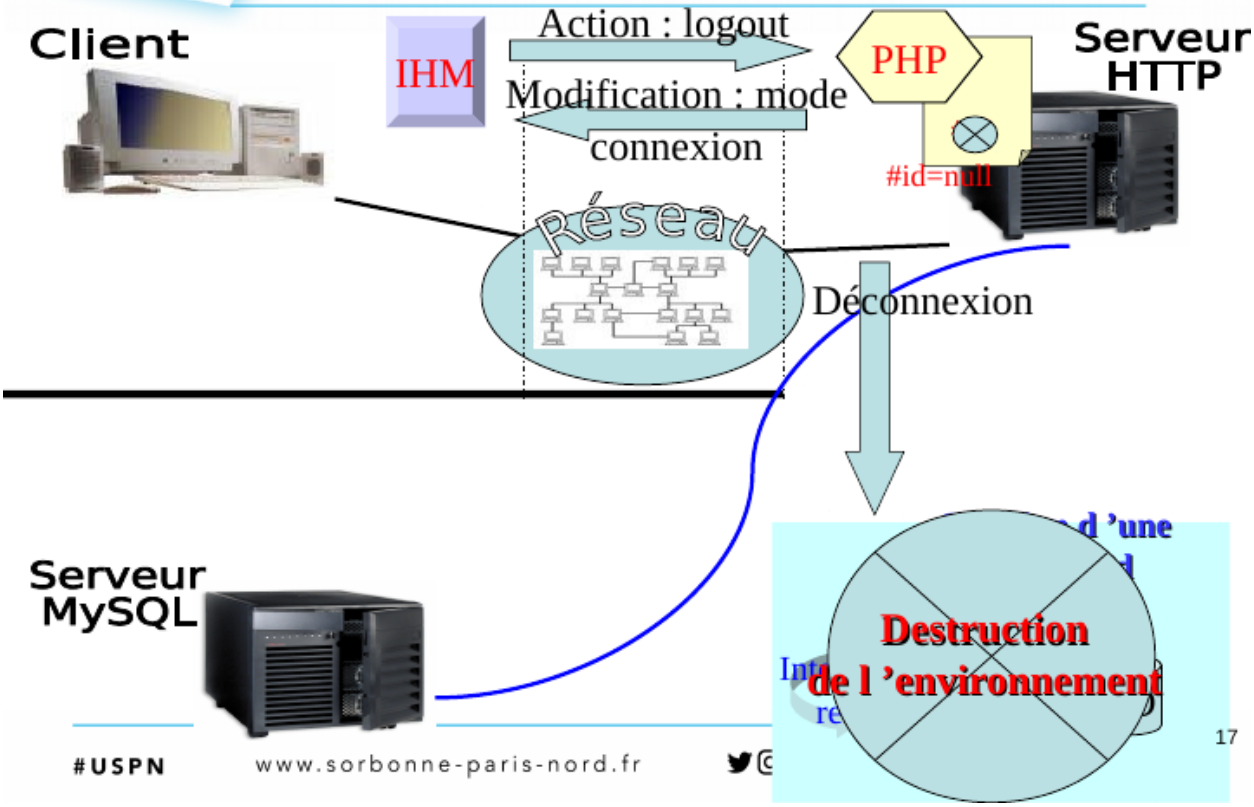



Université Sorbonne Paris Nord
Principe de PHP & MySQL
IUT BOBIGNY
Client
IHM
Action : Login
Modification : mode requête
PHP
Serveur HTTP
réseau
Demande de connexion au serveur
Autorisation En attente de requêtes pour la session #id
Création d 'une session #id
Serveur MySQL
Droit ?
Droit du compt
Utilisateur réfé
Client référenc
OUI
terprétation requêtes
BD
#USPN
www.sorbonn

Principe de PHP & MySQL



Principe de PHP & MySQL



Pensez objet !

- Vous pouvez manipuler facilement toutes bases de données en utilisant la classe PDO

Les sessions en ≥ PHP 5

Pas de caractère avant, même pas un espace

```
<?php
// page1.php

session_start();

echo 'Bienvenue à la page numéro 1';

$_SESSION['favcolor'] = 'green';
$_SESSION['animal']   = 'cat';
$_SESSION['time']     = time();

// Fonctionne si le cookie a été accepté
echo '<br /><a href="page2.php">page 2</a>';

// Ou bien, en indiquant explicitement l'identfiant d
e session
echo '<br /><a href="page2.php?' . SID . '">page
2</a>';
?>
```

L'accès/création à la mémoire partagée commence ici

Sauvegarde des données dans le tableau $_SESSION

http://php.net/manual/fr/function.session-start.php

Les sessions en ≥ PHP 5

```
<?php
// page2.php

session_start();

echo 'Bienvenue sur la page numéro 2<br />';

echo $_SESSION['favcolor']; // green
echo $_SESSION['animal'];   // cat
echo date('Y m d H:i:s', $_SESSION['time']);

// Vous pourriez utiliser la constante SID ici, tout comme dans la page
page1.php
echo '<br /><a href="page1.php">page 1</a>';
?>
```

L'accès/création à la mémoire partagée commence ici

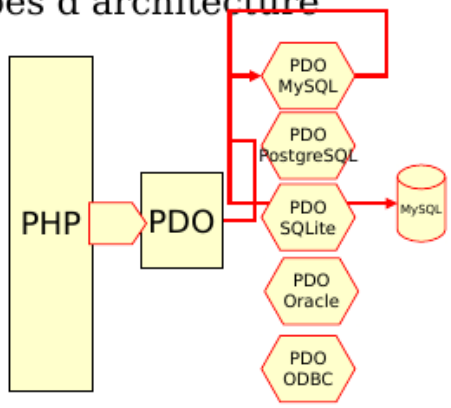
Utilisation et modification des données du tableau $_SESSION

http://php.net/manual/fr/function.session-start.php

Modifier/gérer des sessions :
http://www.php.net/manual/fr/ref.session.php

PDO

- Cette classe existe depuis la version 5.1 de PHP
- L'avantage de cette classe est de faire abstraction du type de drivers de BD
- La classe assure la compatibilité des requêtes en offrant un pont « commun » à 5 types d'architecture
  - MySQL 3, 4 et 5 (pdo_mysql)
  - PostgreSQL (pdo_pgsql)
  - SQLite 2 et 3 (pdo_sqlite)
  - Oracle (pdo_oci)
  - ODBC (pdo_odbc)
- Votre code devient donc universel
- Vous avez seulement le nom du serveur et de pilote à configurer
- Les requêtes ne changerons pas !



Sélectionner-Ouvrir-Requêter-Exploiter-Fermer

SELECTIONNER

```
$db = "nomBD"; //nom de la base de données
$host = "nomServeurMySQL"; // nom de la machine hôte
$user = "nomUser"; // nom de l'utilisateur
$pwd = "motPasse";// mot de passe
$dsn = "mysql:host=$host;dbname=$db"; // Data Source Name
```

OUVRIR

```
$cnxDB = new PDO($dsn, $user, $pwd);
                    // Objet PDO représentant la connexion
```

REQUETER

```
$sql = "SELECT * FROM loginTable";  // La requête
$resultat=$cnxDB->query($sql);  // On pose la question
                                //  et on obtient un objet PDOStatement
```

EXPLOITER

```
while ( $row = $resultat->fetch()) // pour chaque ligne du résultat
{
  print_r($row); Affiche la ligne complète
}
```

FERMER

```
$cnxDB = null; // Pensez à libérer la mémoire !
```

Pensez réutilisation : constantes

- Le nom de la BD, le nom du serveur, le nom de l'utilisateur de la BD, son pass, le DSN sont des constantes pour une application donnée.
- On factorise ces constantes soit
  - Par un fichier connexion.inc.php ou figure les déclarations via des define. Ce fichier devra être chargé par include_once à chaque fois qu'on voudra faire une connexion à la BD.
  - Par une classe, en déclarant via des const. La classe étant chargée automatiquement par la méthode magique __autoload ou spl_auloload_register.

Méthode à utiliser

Ouvrir : PROPREMENT !

```
try
{
  $cnxDB = new PDO($dsn, $user, $pwd);
                        // Objet PDO représentant la connexion
}
catch (PDOException $e)
{
  die ("Erreur à l'ouverture ! :". $e->getmessage());
}
```

Erreur à l'ouverture ! : could not find driver

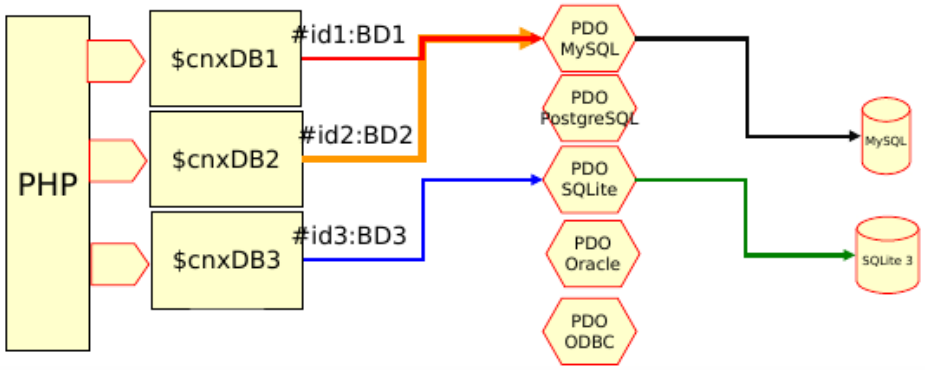
PDO n'est pas installé ou le driver

Erreur à l'ouverture ! : SQLSTATE[28000] [1045] Access denied for user 'nomUser'@'nomServeurMySQL' (using password: YES)

Soit le login n'est pas bon,
soit le pass n'est pas bon,
soit l'utilisateur n'a pas les droits nécessaire pour ouvrir cette connexion

Attention !
objetPDO = une connexion
= une et une seule BD !

- Si vous avez besoin de manipuler plusieurs BD, il faut un objet PDO par connexion
- Chaque canal peut avoir son propre triplet DSN, user, pass



fetch_style

- Le résultat retourné par fetch peut être soit
  - Un tableau associatif, le résultat est un tableau dont les colonnes sont les les noms des champs de la table
    $resultat=$cnxDB->fetch(PDO::FETCH_ASSOC);
    Syntaxe à utiliser pour le résultat :
    $resultat["nomColonne"]
  - Un tableau indexé à la fos par les noms et les numéros de colonnes
    $resultat=$cnxDB->fetch(PDO::FETCH_BOTH);
    Syntaxes à utiliser pour le résultat :
    $resultat["nomColonne"] ou $resultat[3]
  - Un objet dont les champs sont les les nor
    $resultat=$cnxDB->fetch(PDO::FETCH_
    Syntaxe à utiliser pour le résultat :
    $resultat->nomColonne

Constantes de classe

exec() ou query() ?

- $cnxDB->exec($sql) est à utiliser pour les requêtes qui modifient le contenu de la BD.

requête	Méthode à utiliser
INSERT	exec
UPDATE	exec
DELETE	exec

- $cnxDB->query($sql) est à utiliser dans les autres cas !

requête	Méthode à utiliser
SELECT	query
EXPLAIN	query
SHOW	query
DESC	query

exec et ses valeurs de retour

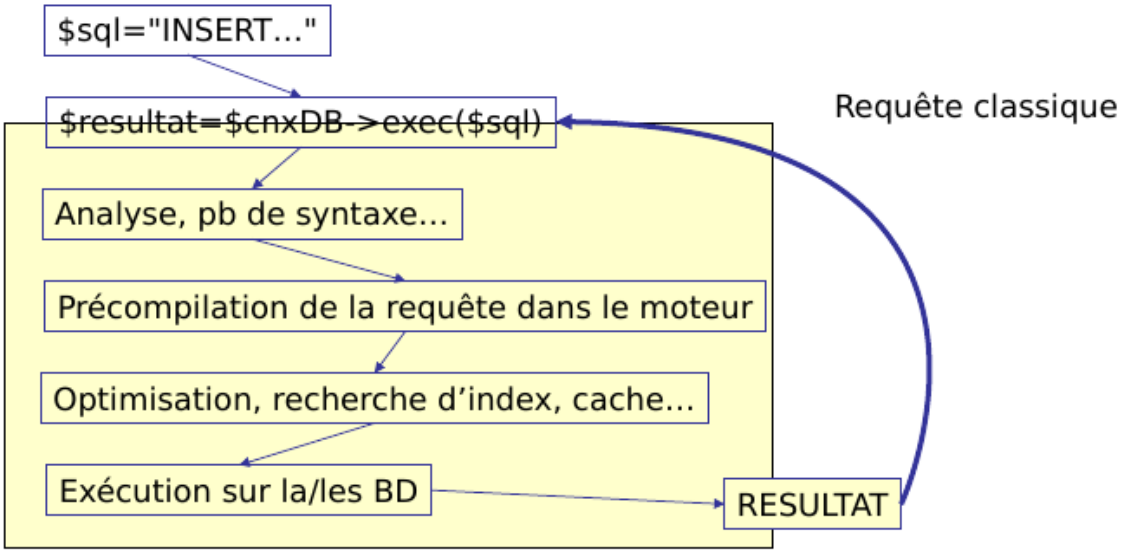
- Les requêtes de type exec ne retournent pas de ligne mais un nombre de lignes affectées par la requête
  - Si la valeur de retour est FALSE, la requête n'a pas marché
  - Si la valeur de retour est 0, la requête a été exécutée mais il n'y a pas eu de modification
  - Si la valeur de retour est supérieure à 0, elle indique le nombre de lignes qui ont été affectées.

Attention aux échappements !

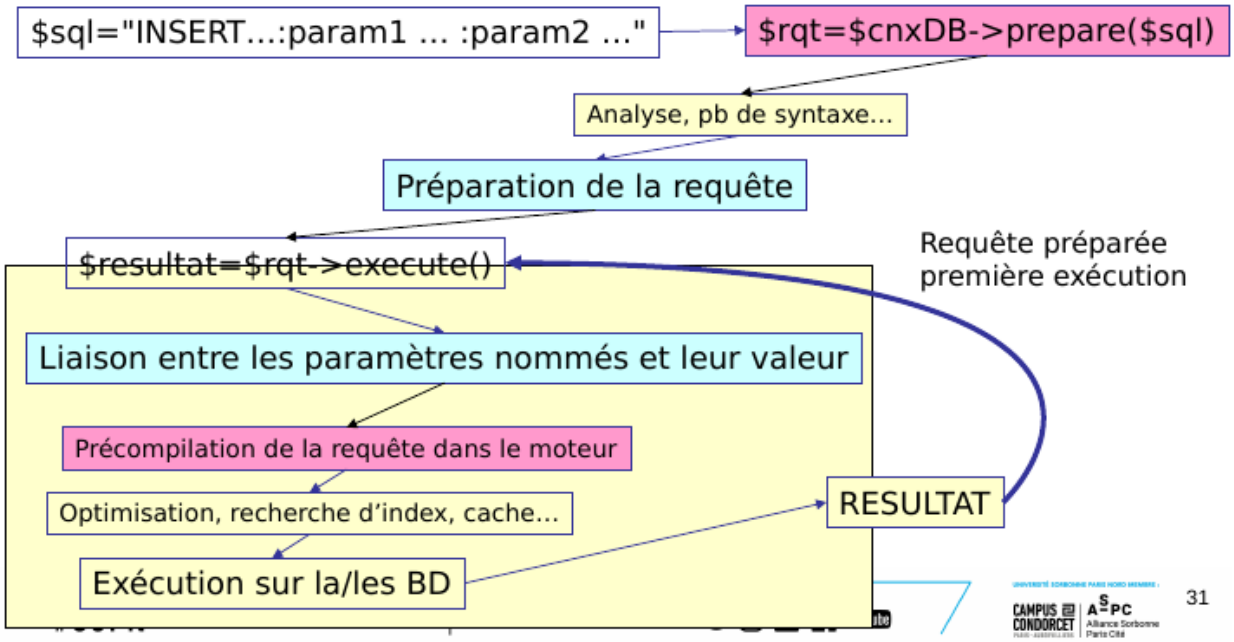
- Pour SQL, certain caractère ont un sens particulier comme l'espace, l'apostrophe...
- Ces caractères doivent être protégés si ils sont dans des chaînes de caractères

```
$nom="toto est sur l'île";
$nom=$cnxDB->quote($nom);
$sql="INSERT INTO msgTable
    (login,msg)
    VALUES ('Moi',$nom)";
```

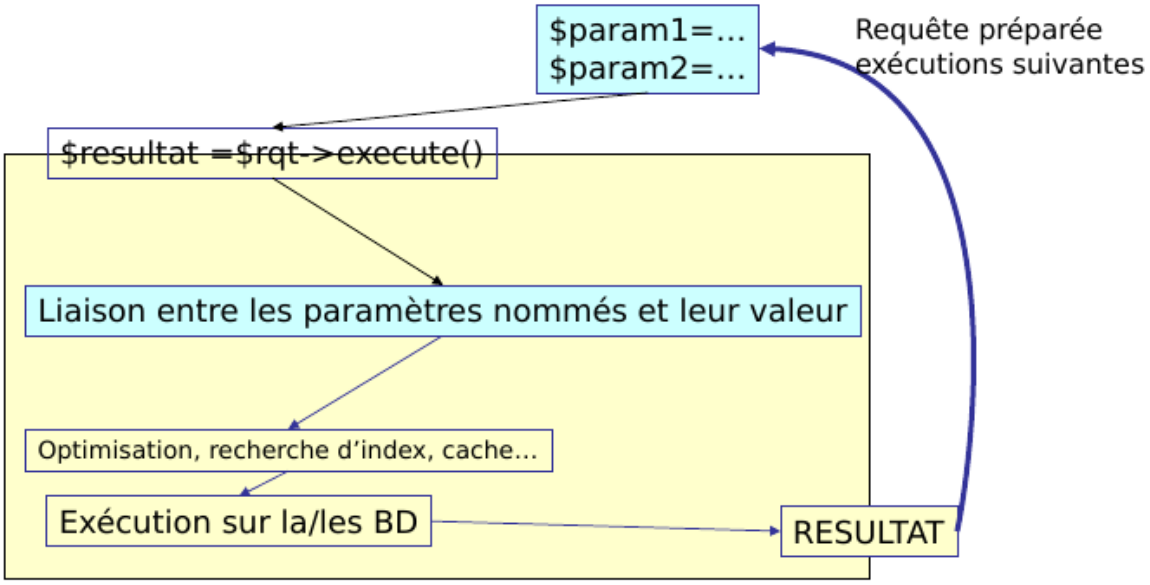
Optimiser les requêtes : utilisez des requêtes préparées



Optimiser les requêtes : utilisez des requêtes préparées



Optimiser les requêtes : utilisez des requêtes préparées



Lier les données et les paramètres nommés

```
$sql="INSERT INTO nomTable (champs1,champs2)
       VALUES (:param1,:param2)";  // requête avec des paramètres nommés

$rqt=$cnxDB->prepare($sql);  // préparation de la requête
$param1="toto";                  // initialisation de variable
$param2="titi";

$rqt->execute(array(":param1"=>$param1
                    , ":param2"=>$param2));
                                    // passage des valeurs des paramètres

$param1="tutu";
$param2="tata";

$rqt->BindParam(":param1",$param1);  // liaison dynamique entre
                                     //   les paramètres et les variables
$rqt->BindParam(":param2",$param2);

$rqt->execute();

$param1="toutou";
$rqt->execute();
```

| toto | titi |

| tutu | tata |

| toutou | tata |

Avantages et inconvénient

- Avantages
  - Gain de temps des la deuxième exécution
  - Sécurité : on se protège des injections SQL
- Inconvénient
  - Une seule requête peut être préparée à la fois

https://www.softwaretestinghelp.com/mariadb-vs-mysql/

What You Will Learn: [hide]

Conclusion

- Utilisez PDO quand vous pouvez !

•https://grafikart.fr/tutoriels/pdo-php-111

•https://fmaz.developpez.com/tutoriels/php/comprendre-pdo/

•http://sdz.tdct.org/sdz/do-interface-d-acces-aux-bdd.html

•https://openclassrooms.com/fr/courses/918836-concevez-votre-site-web-avec-php-et-mysql/914293-accedez-aux-donnees-en-php-avec-pdo

•https://www.php.net/manual/fr/book.pdo.php