

SQL 101

Roadmap

- So far, so good ?
- Les bases de données
- De SGBD à SQL
- ORM
- SQL vs NoSQL

```
fs.readFile(pathToFile, "utf8", function(error, data){  
  
    var objectParsed = JSON.parse(data);  
    var customers = objectParsed.customers;  
  
    customers.push(newCustomer);  
  
    var newGlobalObject = {  
        customers : customers  
    };  
  
    var newGlobalObjectString =  
        JSON.stringify(newGlobalObject);  
  
    fs.writeFile(pathToFile, newGlobalObjectString,  
        function(err){  
            console.log("File written");  
            response.json(customers);  
        })  
    })  
})
```

Comment stocker des données ?

Comment stocker des données ?

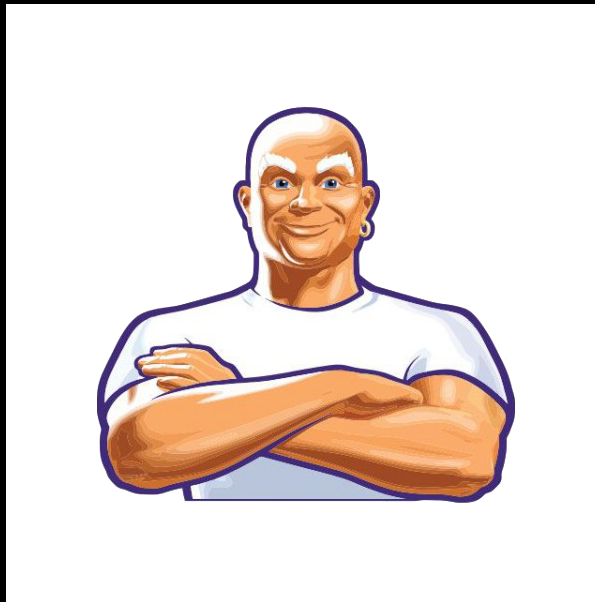
- Variable
- localStorage
- \$_Session
- \$_Cookie

Les Bases De Données - DB

Une base de données est un outil permettant de stocker et de retrouver l'intégralité des informations stockées.

Comme un fichier .txt ou .json ?

Les informations sont classées, hiérarichées, organisées ...



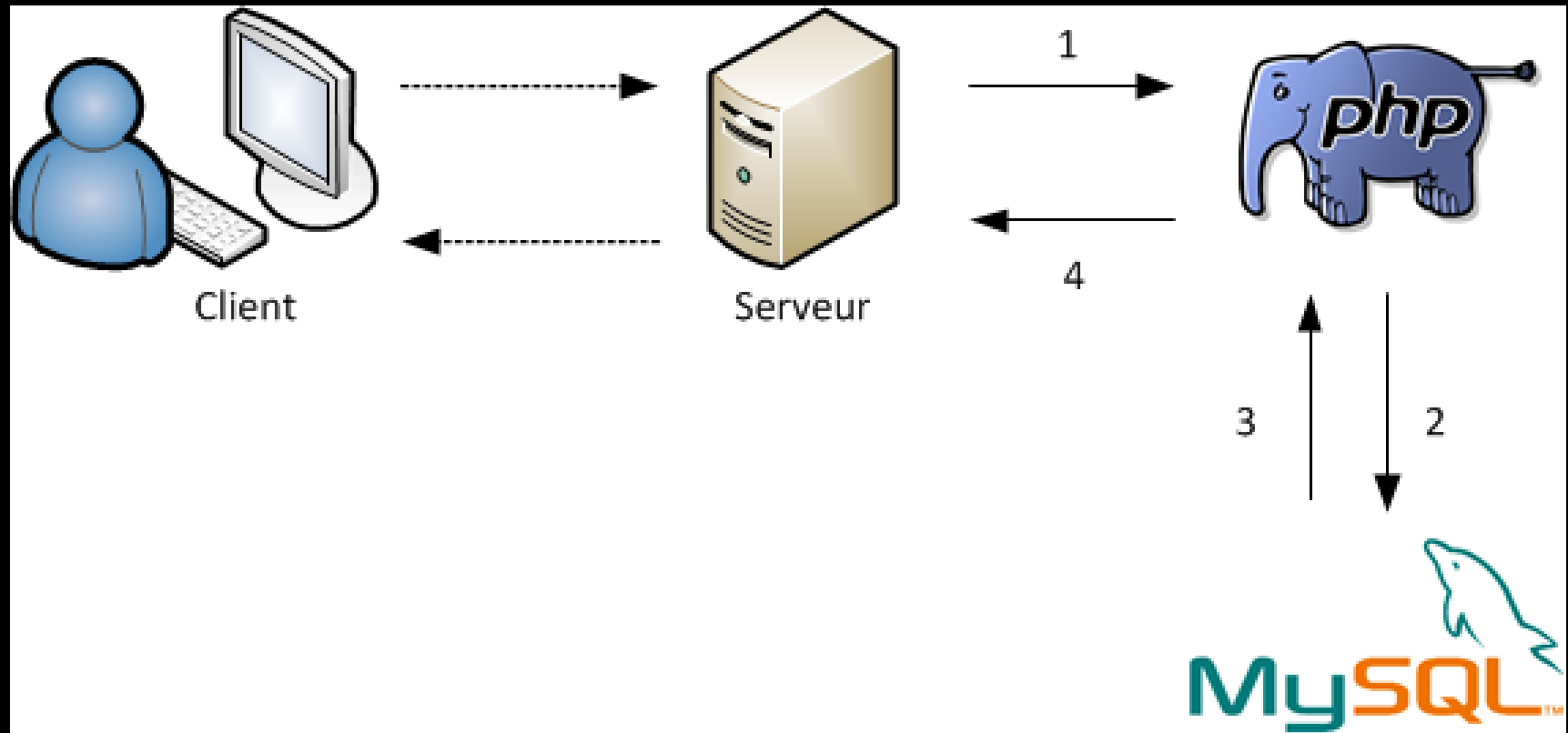
Du SGBD au SQL

SGBD

- Système de Gestion de Base de Données.
- Programme en charge de la base de données.
- Communique avec la DB grâce à un langage : le SQL

Des exemples de SGBD

- MySQL
- PostgreSQL
- SQLite
- Oracle
- Microsoft SQL Server



source : openclassrooms.com

Un peu de vocabulaire

- base
- table
- champs
- entrées

Base

L'armoire

Base

L'armoire

Table

Le tiroir

Base

L'armoire

Table

Le tiroir

Champs

Les propriétés (des chaussettes : couleur, pointure, °C de lavage...)

Base

L'armoire

Table

Le tiroir

Champs

Les propriétés (des chaussettes : couleur, pointure, °C de lavage...)

Entrées

Les chaussettes qui se trouvent dans le tiroir, ayant des propriétés différentes les unes des autres

Base de données MySQL

Table

	Champ 1	Champ 2	. . .
Entrée	xxxxxx	xxxxxx	
Entrée	xxxxxx	xxxxxx	
Entrée	xxxxxx	xxxxxx	
Entrée	xxxxxx	xxxxxx	
.	

Table

Table

source : openclassrooms.com

SQL - Structured Query Language

SQL est un langage informatique normalisé servant à exploiter des bases de données relationnelles. La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles.

Wikipédia

```
SELECT couleur, pointure FROM tiroir_chaussettes ORDER BY  
date_lavage DESC
```

Show Time

- installation de MySQL
- création d'une base de données `mon_armoire`

```
sudo apt-get install mysql-server  
mysql --version  
# Connexion à MySQL avec le user 'root'  
mysql -u root -p
```

- Dans MySQL :

```
SHOW DATABASES;  
CREATE DATABASE mon_armoire;  
SHOW DATABASES;
```

```
mysql> show databases;
```

```
+-----+  
| Database |  
+-----+  
| information_schema |  
| mysql |  
| performance_schema |  
| sys |  
| vetements |  
+-----+
```

```
mysql> use vetements;
mysql> show tables;
+-----+
| Tables_in_vetements |
+-----+
| Etagere_pantalon     |
| Tiroir_chaussettes   |
| Tiroir_slips        |
+-----+
```


SELECT

```
mysql> SELECT * FROM tiroir_chaussettes;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50

```
mysql> SELECT id, pointure, temp_lavage  
FROM tiroir_chaussettes;
```

id	pointure	temp_lavage
1	45	40
2	39	40
3	40	40
4	40	60
5	42	50

Show Time

- importer le script mon_armoire.sql (sortez de MySql avant)

```
mysql -u root -p < mon_armoire.sql
```

- Afficher tous les éléments de la table mes_chaussettes
- Afficher uniquement la pointure de mes_chaussettes
- Afficher uniquement la pointure et la couleur de mes_chaussettes

WHERE

```
mysql> SELECT id, pointure, temp_lavage  
FROM tiroir_chaussettes WHERE id=2;
```

```
+---+-----+-----+  
| id | pointure | temp_lavage |  
+---+-----+-----+  
| 2 | 39 | 40 |  
+---+-----+-----+
```

```
mysql> SELECT id, pointure, temp_lavage  
FROM tiroir_chaussettes WHERE pointure=40;
```

id	pointure	temp_lavage
3	40	40
4	40	60

```
mysql> SELECT id, pointure, temp_lavage  
FROM tiroir_chaussettes WHERE pointure>=40;
```

id	pointure	temp_lavage
1	45	40
3	40	40
4	40	60
5	42	50


```
mysql> SELECT *  
FROM tiroir_chaussettes WHERE pointure>=40;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50

Opérateurs de comparaison

- =
- <
- <=
- >
- >=
- <> ou !=
- <=>

```
mysql> SELECT *  
FROM tiroir_chaussettes  
WHERE pointure>=40 AND date_lavage>'2017-01-01';
```

id	couleur	description	pointure	date_lavage	temp_lavage
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50

Opérateurs WHERE

- AND
 - &&
- OR
 - ||
- XOR
- NOT
 - !
 -

Show Time

- Afficher les chaussettes de pointure supérieure ou égale à 42
- Afficher les chaussettes qui ont été lavées avant octobre 2016
- Afficher les chaussettes bleues qui se lavent à 50°C ou plus
- Afficher les chaussettes de couleur (pas noir/blanc) de pointure en dessous 43, sauf 38, et qui se lavent à 30°C

ORDER BY

```
mysql> SELECT id, pointure, temp_lavage  
FROM tiroir_chaussettes  
ORDER BY pointure;
```

id	pointure	temp_lavage
2	39	40
3	40	40
4	40	60
5	42	50
1	45	40

```
mysql> SELECT id, pointure, temp_lavage  
FROM tiroir_chaussettes  
ORDER BY pointure DESC;
```

id	pointure	temp_lavage
1	45	40
5	42	50
3	40	40
4	40	60
2	39	40


```
mysql> SELECT *  
FROM tiroir_chaussettes  
ORDER BY pointure;
```

id	couleur	description	pointure	date_lavage	temp_lavage
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
7	noir		40	2017-01-13 12:00:00	50
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40

```
mysql> SELECT *  
FROM tiroir_chaussettes  
ORDER BY pointure, temp_lavage;
```

id	couleur	description	pointure	date_lavage	temp_lavage
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
7	noir		40	2017-01-13 12:00:00	50
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40

Eliminer les doublons : DISTINCT

```
mysql> SELECT couleur  
FROM tiroir_chaussettes;
```

```
+-----+  
| couleur |  
+-----+  
| rose |  
| bleu |  
| bleu |  
| blanc |  
| jaune |  
| noir |  
+-----+
```

```
mysql> SELECT DISTINCT couleur  
FROM tiroir_chaussettes;
```

```
+-----+  
| couleur |  
+-----+  
| rose |  
| bleu |  
| blanc |  
| jaune |  
| noir |  
+-----+
```

Show Time

- Récupérer les différentes pointures (sans doublon).
- Récupérer les différentes couleurs (sans doublon).

Restreindre les résultats : LIMIT

```
mysql> SELECT *  
FROM tiroir_chaussettes;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
7	noir		40	2017-01-13 12:00:00	50


```
mysql> SELECT *  
FROM tiroir_chaussettes  
LIMIT 3;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40

```
mysql> SELECT *  
FROM tiroir_chaussettes  
LIMIT 3 OFFSET 2;
```

id	couleur	description	pointure	date_lavage	temp_lavage
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50

A little further

```
mysql> SELECT *  
FROM tiroir_chaussettes;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
7	noir		40	2017-01-13 12:00:00	50

```
mysql> SELECT *  
FROM tiroir_chaussettes  
ORDER BY temp_lavage;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
7	noir		40	2017-01-13 12:00:00	50
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60

```
mysql> SELECT *  
FROM tiroir_chaussettes  
ORDER BY temp_lavage  
LIMIT 4;
```

id	couleur	description	pointure	date_lavage	temp_lavage
1	rose	déteignent un peu	45	2010-09-13 15:02:00	40
2	bleu		39	2017-09-13 15:02:00	40
3	bleu		40	2017-01-13 12:04:00	40
7	noir		40	2017-01-13 12:00:00	50

```
mysql> SELECT *  
FROM tiroir_chaussettes  
ORDER BY temp_lavage  
LIMIT 4 OFFSET 2;
```

id	couleur	description	pointure	date_lavage	temp_lavage
3	bleu		40	2017-01-13 12:04:00	40
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
7	noir		40	2017-01-13 12:00:00	50
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60

```
mysql> SELECT *  
FROM tiroir_chaussettes  
WHERE date_lavage > '2017-01-01'  
ORDER BY temp_lavage  
LIMIT 4 OFFSET 2;
```

id	couleur	description	pointure	date_lavage	temp_lavage
5	jaune	avec des rayures	42	2017-01-10 12:00:00	50
7	noir		40	2017-01-13 12:00:00	50
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60


```
mysql> SELECT *  
FROM tiroir_chaussettes  
WHERE date_lavage > '2017-01-01' && pointure = 40  
ORDER BY temp_lavage  
LIMIT 4 OFFSET 2;
```

id	couleur	description	pointure	date_lavage	temp_lavage
4	blanc	avec des rayures	40	2017-01-10 12:00:00	60

```
mysql> SELECT description
FROM tiroir_chaussettes
WHERE date_lavage > '2017-01-01' && pointure = 40
ORDER BY temp_lavage
LIMIT 4 OFFSET 2;
```

+-----+

| description |

+-----+

| avec des rayures |

+-----+

Show Time

- Afficher les 10 chaussettes ayant été lavées le plus récemment
- Afficher les 10 chaussettes ayant été lavées le plus récemment de pointure supérieure à 39
- Afficher les 10 chaussettes bleues, de pointure entre 39 et 45 et rangées par date de lavage décroissante

Sum Up

- MySql
- Premières requêtes SQL

Insertion, Modification et suppression

```
mysql> INSERT INTO tiroir_chaussettes  
VALUES (1, "rose", "déteignent un peu", 45,  
         '2010-09-13 15:02:00', 40);
```

```
mysql> UPDATE tiroir_chaussettes  
SET couleur = 'rose'  
WHERE couleur = 'rouge;
```

```
mysql> DELETE  
FROM tiroir_chaussettes  
WHERE id = 7;
```

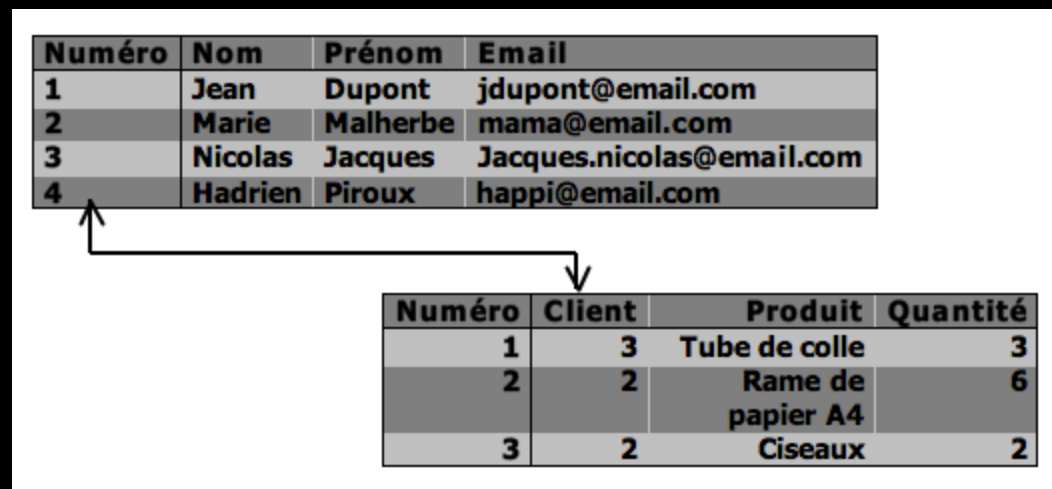

Clés primaires

- unique
- une ou plusieurs colonnes (composite)
- identifie de manière unique chaque ligne (non NULL)

```
CREATE TABLE [IF NOT EXISTS] Tiroir_chaussettes (  
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,  
    couleur VARCHAR(20),  
    description VARCHAR(80),  
    pointure SMALLINT UNSIGNED,  
    date_lavage DATETIME,  
    temp_lavage SMALLINT UNSIGNED,  
    PRIMARY KEY (id)  
  
)  
ENGINE=INNODB;
```

Clés étrangères

- Protège l'intégrité des données
- Référence d'une clé primaire d'une table A, dans une table B



source : openclassrooms.com

Jointures

```
-- On récupère l'id de Cartouche
SELECT espece_id FROM Animal WHERE nom = 'Cartouche';

-- On récupère la description grâce à l'id trouvée précéd
SELECT description FROM Espece WHERE id = 1;
```

```
-- On récupère l'id de Cartouche
SELECT espece_id FROM Animal WHERE nom = 'Cartouche';

-- On récupère la description grâce à l'id trouvée précéd
SELECT description FROM Espece WHERE id = 1;
```

```
SELECT Espece.description

FROM Espece

INNER JOIN Animal

    ON Espece.id = Animal.espece_id

WHERE Animal.nom = 'Cartouche';
```

Animal

<i>id</i>	<i>sexe</i>	<i>nom</i>	<i>race_id</i>	<i>espece_id</i>
24	M	Cartouche	NULL	1
25	M	Zambo	1	1
33	M	Caribou	4	2

Espece

<i>id</i>	<i>nom_courant</i>	<i>nom_latin</i>
1	Chien	Canis canis
2	Chat	Felix silvestris

Jointure Animal-Espece

<i>id</i>	<i>sexe</i>	<i>nom</i>	<i>race_id</i>	<i>espece_id</i>	<i>id</i>	<i>nom courant</i>	<i>nom latin</i>
24	M	Cartouche	NULL	1	1	Chien	Canis canis
25	M	Zambo	1	1	1	Chien	Canis canis
33	M	Caribou	4	2	2	Chat	Felix silvestris

source : openclassrooms.com

Alias

```
SELECT Espece.id AS id_espece,  
       Espece.description AS description_espece,  
       Animal.nom AS nom_bestiole  
  
FROM Espece  
  
INNER JOIN Animal  
       ON Espece.id = Animal.espece_id  
  
WHERE Animal.nom LIKE 'Ch%';
```


id_espece	description_espece	nom_bestiole
1	Espèce volante dans l'eau	chat botté
4	Espèce venimeuse	poisson rouge
4	Espèce venimeuse	Flipper

Opérations

Fonctions scalaires

Fonctions scalaires

- CEIL / CEILING

- `SELECT CEIL(3.2), CEIL(3.7);`

- FLOOR

- `SELECT FLOOR(3.2), FLOOR(3.7);`

- ROUND

- `SELECT ROUND(3.22, 1), ROUND(3.55, 1),
ROUND(3.77, 1);`

- `SELECT ROUND(3.2), ROUND(3.5), ROUND(3.7);`

Fonctions scalaires

- TRUNCATE
- POW / POWER
 - `SELECT POW(32, 1/5);`
- SQRT
- RAND
 - `SELECT RAND();`

Manipulation de chaînes de caractères

- CHAR_LENGTH(chaine)
- SUBSTRING()
- LOCATE()
- LOWER()
- UPPER()

Agrégateurs

Agrégateurs

- COUNT

```
SELECT COUNT(*) FROM Race;
```


Agrégateurs

- SUM

```
SELECT SUM(prix)FROM Produit;
```

Agrégateurs

- AVG
















```
SELECT AVG(prix)FROM Produit;
```



UI Management

PhpMyAdmin

Serveur: localhost ▶ Base de données: test ▶ Table: news

[Afficher](#) [Structure](#) [SQL](#) [Rechercher](#) [Insérer](#) [Exporter](#) [Importer](#) [Opérations](#) [Vider](#) [Supprimer](#)

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra	Action
<input type="checkbox"/>	id	int(11)			Non	Aucun	auto_increment	       
<input type="checkbox"/>	titre	varchar(255)	latin1_swedish_ci		Non	Aucun		       
<input type="checkbox"/>	contenu	text	latin1_swedish_ci		Non	Aucun		       

↑ [Tout cocher](#) / [Tout décocher](#) Pour la sélection :      

[Version imprimable](#) [Suggérer des optimisations quant à la structure de la table](#) ?

[Ajouter](#) champ(s) ☒ En fin de table ☐ En début de table ☐ Après [Exécuter](#)

DB Ninja

The screenshot shows the DbNinja MySQL web interface. The browser address bar displays 'demo.dbninja.com'. The interface includes a sidebar with a tree view of the database structure, including tables like 'actor', 'address', 'category', 'city', 'country', 'customer', 'film', 'film_actor', 'film_category', 'film_text', 'inventory', 'language', 'payment', 'rental', 'staff', and 'store'. The main panel shows a table of data from the 'sakila' database, specifically the 'address' table. The table has columns: #, address_id, address, district, city_id, postal_code, phone, and last_update. A context menu is open over the row with address_id 6, showing options like 'Upload Value', 'Download Value', 'Filter', 'Preview As...', 'Copy to Clipboard...', 'Set to NULL', 'Set to Empty String', 'Set to Zero (0000-00-00)', 'Set to NOW()', 'Set to My Local Time', 'Set to SHA(...)', 'Set to MD5(...)', 'Set to SHA of Cell Value', 'Set to MD5 of Cell Value', and 'Delete Record'.

#	address_id	address	district	city_id	postal_code	phone	last_update
1	1	47 MySakila Drive	Alberta	300			2006-02-15 04:45:30
2	2	28 MySQL Boulevard	QLD	576			2006-02-15 04:45:30
3	3	23 Workhaven Lane	Alberta	300		14033335568	2006-02-15 04:45:30
4	4	1411 Lillydale Drive	QLD	576		6172235589	2006-02-15 04:45:30
5	5	1913 Hanoi Way	Nagasaki	463	35200	28303384290	2006-02-15 04:45:30
6	6	1121 Loja Avenue	California	449	17886	83863528649	2006-02-15 04:45:30
7	7	692 Joliet Street	Atika			448477190408	2006-02-15 04:45:30
8	8	1566 Inegi Manor	Mandalay			705814003527	2006-02-15 04:45:30
9	9	53 Idfu Parkway	Nantou			10655648674	2006-02-15 04:45:30
10	10	1795 Santiago de Compostela Way	Texas			56434	2006-02-15 04:45:30
11	11	900 Santiago de Compostela Parkway	Central Serbi			0373	2006-02-15 04:45:30
12	12	478 Joliet Way	Hamilton			5970	2006-02-15 04:45:30
13	13	613 Korolev Drive	Masqat			2649	2006-02-15 04:45:30
14	14	1531 Sai Drive	Esfahan			648856936185	2006-02-15 04:45:30
15	15	1542 Tatiac Parkway	Kanagawa			635297277345	2006-02-15 04:45:30
16	16	808 Bhopal Manor	Haryana			465887807014	2006-02-15 04:45:30
17	17	270 Amroha Parkway	Osmaniye			695479687538	2006-02-15 04:45:30
18	18	770 Bydgoszcz Avenue	California			517338314235	2006-02-15 04:45:30
19	19	419 Iligan Lane	Madhya Prade			990911107354	2006-02-15 04:45:30
20	20	360 Toulouse Parkway	England			949312333307	2006-02-15 04:45:30
21	21	270 Toulon Boulevard	Kalmykia			407752414682	2006-02-15 04:45:30
22	22	320 Brest Avenue	Kaduna			747791594069	2006-02-15 04:45:30
23	23	1417 Lancaster Avenue	Northern Cap			272572357893	2006-02-15 04:45:30
24	24	1688 Okara Way	Northwest Border Prov	327	21954	144453869132	2006-02-15 04:45:30
25	25	262 A Corua (La Corua) Parkway	Dhaka	525	34418	892775750063	2006-02-15 04:45:30

603 rows selected in 0.522 sec.

PDO

```
sudo apt-get install php7.0-mysql
```

```
<?php
```

```
$bdd = new PDO(  
    'mysql:host=sql.hebergeur.com;dbname=mon_armoire;charset=  
    'pierre.durand', 's3cr3t');
```

```
?>
```

Requête avec PDO

```
<?php  
$reponse = $bdd->query('SELECT * FROM mes_chaussettes');
```



```
<?php  
$donnees = $reponse->fetch();
```

```

<?php
// On affiche chaque entrée une à une
while ($donnees = $reponse->fetch())
{
    ?>

    <p>
    Jeu : <?php echo $donnees['nom']; ?><br />
    Propriétaire : <?php echo $donnees['possesseur']; ?>,
    et il le vend à <?php echo $donnees['prix']; ?> € !<br />
    Console : <?php echo $donnees['console']; ?>
    nb joueurs max : <?php echo $donnees['nbre_joueurs_max']; ?>
    </p>
    <?php
}

// Termine le traitement de la requête
$reponse->closeCursor();

```

Fermeture du curseur d'analyse des résultats

```
<?php $reponse->closeCursor(); ?>
```

```
<?php
```

```
$reponse = $bdd->query(  
    'SELECT nom FROM jeux_video  
    WHERE possesseur=\'Patrick\'');
```

```
?>
```

```
<?php
```

```
$reponse = $bdd->query(  
    'SELECT nom FROM jeux_video  
    WHERE possesseur=\' ' . $_GET['possesseur'] . '\ '');
```

```
?>
```

Requêtes préparées

```
<?php

$req = $bdd->prepare(
    'SELECT nom FROM jeux_video
    WHERE possesseur = ? AND prix <= ?'
);

$req->execute(
    array($_GET['possesseur'], $_GET['prix_max'])
);

?>
```

Avec marqueurs nominatifs

```
<?php
$req = $bdd->prepare(
    'SELECT nom, prix FROM jeux_video
    WHERE possesseur = :possesseur AND prix <= :prixmax');

$req->execute(
    array('possesseur' => $_GET['possesseur'],
        'prixmax' => $_GET['prix_max']));
?>
```

Insérer des données

```
<?php
```

```
$req = $bdd->prepare('
INSERT INTO jeux_video(nom, possesseur, console, prix,
nbre_joueurs_max, commentaires)
VALUES(:nom, :possesseur, :console, :prix,
:nbre_joueurs_max, :commentaires)');
```

```
$req->execute(array(
    'nom' => $nom,
    'possesseur' => $possesseur,
    'console' => $console,
    'prix' => $prix,
    'nbre_joueurs_max' => $nbre_joueurs_max,
    'commentaires' => $commentaires
));
```

```
echo 'Le jeu a bien été ajouté !';
```

```
?>
```


ORM

Object-Relational Mapping

j4mie/idiorm

A lightweight nearly-zero-configuration object-relational mapper and fluent query builder for PHP

Setup

```
<?php
require_once 'idiorm.php';

ORM::configure('mysql:host=localhost;dbname=my_database')
ORM::configure('username', 'database_user');
ORM::configure('password', 'top_secret');
```

```
<?php
```

```
$person = ORM::for_table('person')  
    ->where('name', 'Fred Bloggs')->find_one();
```

```
<?php
$females = ORM::for_table('person')
    ->where('gender', 'female')->find_many();
```

```
<?php
$people = ORM::for_table('person')->find_many();
foreach ($people as $person) {
    $person->age = 50;
    $person->save();
}
```

Show Time

- Installer idiorm

```
composer init  
composer require j4mie/idiorm
```

- Afficher tous les id de tous les éléments

```
1  
2  
// ...  
99  
100
```

Show Time *bis*

- Afficher les id des chaussettes rouges
- Faire déteindre les chaussettes rouges en rose
- Afficher les id des chaussettes rouges
- Afficher les id des chaussettes ayant une pointure supérieure à 40

SQL vs NoSQL

NoOnly SQL

- Key-Value
 - Cassandra
 - Redis
- Document database
 - MongoDB
 - CouchDB
- Column store
 - HBase
 - BigTable
- Graph database
 - Neo4J

NoOnly SQL

- SQL Schema vs NoSQL Schemaless
- SQL Normalization vs NoSQL Denormalization
- ...

Why a SQL database ?

- ACID
 - Atomicity, Consistency, Isolation, Durability
- Données structurées et schéma défini

Why a NoSQL database ?

- Large Volume
- Rapid Development
- Geo requests (mongoDB)

Sum Up

- SGBD pour gérer et stocker de manière organiser ses données
- SQL, un vocabulaire précis
 - Base, Table, Colonnes, Lignes, Jointures, Fonctions
 - Des requêtes avancées : sélection, ajout, tri, comptage ...
- Les ORM pour nous faciliter le développement
- SQL vs NoSQL

Questions ?

Ressources

- https://fr.wikipedia.org/wiki/Base_de_données
- https://fr.wikipedia.org/wiki/Structured_Query_Language
- <https://openclassrooms.com/courses/administrez-vos-bases-de-donnees-avec-mysql>
- <https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql/presentation-des-bases-de-donnees-2>
- <https://doc.ubuntu-fr.org/mysql>
- <http://sql.sh/cours/>
- <http://idiorm.readthedocs.io>
- <https://www.sitepoint.com/sql-vs-nosql-differences/>
- <https://www.upwork.com/hiring/data/sql-vs-nosql-databases-whats-the-difference/>


```
CREATE TABLE [IF NOT EXISTS] Tiroir_chaussettes (  
    id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,  
    couleur VARCHAR(20),  
    description VARCHAR(80),  
    pointure SMALLINT UNSIGNED,  
    date_lavage DATETIME,  
    temp_lavage SMALLINT UNSIGNED,  
    PRIMARY KEY (id)  
  
)  
ENGINE=INNODB;
```

```
INSERT INTO tiroir_chaussettes  
VALUES (1, "rose", "déteignent un peu", 45,  
        '2010-09-13 15:02:00', 40);
```