

LP PRISM DÉVELOPPEMENT WEB

Programmer en PHP côté serveur

UTILISATION D'UNE BASE DE DONNÉES

Requêtes sensibles – se protéger des injections SQL – préparer les requêtes

Requête sans risque

// méthode static qui retourne les voitures en un tableau d'objets

```
public static function getAllVoitures() {
```

// écriture de la requête

```
$requete = "SELECT * FROM voiture;";
```

requête figée.
Le code SQL est
sous contrôle.

// envoi de la requête et stockage de la réponse

```
$resultat = connexion::pdo()->query($requete);
```

// traitement de la réponse

```
$resultat->setFetchmode(PDO::FETCH_CLASS, 'Voiture');
```

```
$tableau = $resultat->fetchAll();
```

// réponse

```
return $tableau;
```

```
}
```

Requête risquée

```
// méthode static qui retourne la voiture immatriculée $i

public static function getVoitureByImmat($i) {
    // écriture de la requête

    $requete = "SELECT * FROM voiture WHERE immatriculation = '$i'";

    // envoi de la requête et stockage de la réponse
    $resultat = connexion::pdo()->query($requete);

    // traitement de la réponse
    $resultat->setFetchmode(PDO::FETCH_CLASS, 'Voiture');
    $v = $resultat->fetch();

    // réponse
    return $v;
}
```

requête avec un paramètre \$i.
Le code SQL n'est pas sous
contrôle.

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

Que devient cette requête si le paramètre `$i` prend des valeurs ... diverses ?

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

Que devient cette requête si le paramètre `$i` prend des valeurs ... diverses ?

- Si `$i` prend la valeur : **111AB45** la requête devient...

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

Que devient cette requête si le paramètre `$i` prend des valeurs ... diverses ?

- Si `$i` prend la valeur : `111AB45` la requête devient...

```
SELECT * FROM voiture WHERE immatriculation = '111AB45';
```

A black line originates from the parameter `$i` in the first code block, extends vertically upwards, then horizontally to the right, and finally vertically downwards to point at the value `111AB45` in the second code block. The value `111AB45` is highlighted with a green border in the second code block.

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

Que devient cette requête si le paramètre `$i` prend des valeurs ... diverses ?

- Si `$i` prend la valeur : `111AB45` la requête devient...

```
SELECT * FROM voiture WHERE immatriculation = '111AB45';
```

- Si `$i` prend la valeur : `' ; DELETE FROM voiture WHERE couleur = 'bleu';--`

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

Que devient cette requête si le paramètre `$i` prend des valeurs ... diverses ?

- Si `$i` prend la valeur : `111AB45` la requête devient...

```
SELECT * FROM voiture WHERE immatriculation = '111AB45';
```

- Si `$i` prend la valeur : `' ; DELETE FROM voiture WHERE couleur = 'bleu';--`

```
SELECT * FROM voiture WHERE immatriculation = ' ; DELETE FROM voiture WHERE couleur = 'bleu';--';
```

Requête risquée

```
SELECT * FROM voiture WHERE immatriculation = ''; DELETE FROM voiture WHERE couleur = 'bleu';--';"
```

Requête risquée

```
SELECT * FROM voiture WHERE immatriculation = '';
```

requête court-circuitée



Requête risquée

```
SELECT * FROM voiture WHERE immatriculation = '';
```

requête court-circuitée



```
DELETE FROM voiture WHERE couleur = 'bleu';
```

injection d'une requête SQL dangereuse



Requête risquée

```
SELECT * FROM voiture WHERE immatriculation = '';
```

requête court-circuitée



```
DELETE FROM voiture WHERE couleur = 'bleu';
```

injection d'une requête SQL dangereuse



```
--';";
```



La fin est commentée pour éviter les erreurs SQL

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Il ne faut jamais utiliser une requête faisant intervenir directement un paramètre si on ne contrôle pas la valeur de ce paramètre.
- Le risque est l'injection de requêtes malveillantes mettant en danger la base de données.

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Il ne faut jamais utiliser une requête faisant intervenir directement un paramètre si on ne contrôle pas la valeur de ce paramètre.
- Le risque est l'injection de requêtes malveillantes mettant en danger la base de données.



Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Il ne faut jamais utiliser une requête faisant intervenir directement un paramètre si on ne contrôle pas la valeur de ce paramètre.
- Le risque est l'injection de requêtes malveillantes mettant en danger la base de données.



Drapé sur la plaque d'immatriculation avant de l'automobile se trouve une impression, attachée comme si elle sortait d'un rouleau de scotch. Sur l'impression se trouve une instruction SQL ; probablement la dernière chose que l'on s'attendrait à voir comme ornement de capot. Personne ne sait d'où vient la photo ni si quelqu'un essayait d'être drôle ou tentait légitimement de compromettre le système backend contrôlant la caméra de circulation sur la même photo. Mais une chose est sûre, ce coup astucieux a permis de faire la lumière sur l'insécurité des systèmes de contrôle.

Source <https://threatpost.com/ics-vulnerabilities-surface-monitoring-systems-integrate-digital-backends-032613/77670/>

- Ici le paramètre est capté par le radar, puis intégré à la requête, qui vise à supprimer la base de données entière...

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.
- En particulier, il faut contrôler ATTENTIVEMENT les informations issues d'un formulaire, dont la valeur est soumise par l'utilisateur du formulaire (l'internaute = malveillant par défaut).

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.
- En particulier, il faut contrôler ATTENTIVEMENT les informations issues d'un formulaire, dont la valeur est soumise par l'utilisateur du formulaire (l'internaute = malveillant par défaut).
- Dans la diapo suivante on simule le comportement d'un internaute malveillant :

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.
- En particulier, il faut contrôler ATTENTIVEMENT les informations issues d'un formulaire, dont la valeur est soumise par l'utilisateur du formulaire (l'internaute = malveillant par défaut).
- Dans la diapo suivante on simule le comportement d'un internaute malveillant :
 - Cet internaute a fait des hypothèses sur la façon dont le code côté serveur a été conçu

Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.
- En particulier, il faut contrôler ATTENTIVEMENT les informations issues d'un formulaire, dont la valeur est soumise par l'utilisateur du formulaire (l'internaute = malveillant par défaut).
- Dans la diapo suivante on simule le comportement d'un internaute malveillant :
 - Cet internaute a fait des hypothèses sur la façon dont le code côté serveur a été conçu
 - En particulier, il suppose que la requête envoyée est construite en récupérant les éléments du formulaire, et en les incluant dans une chaîne de caractères qui est la requête à envoyer.

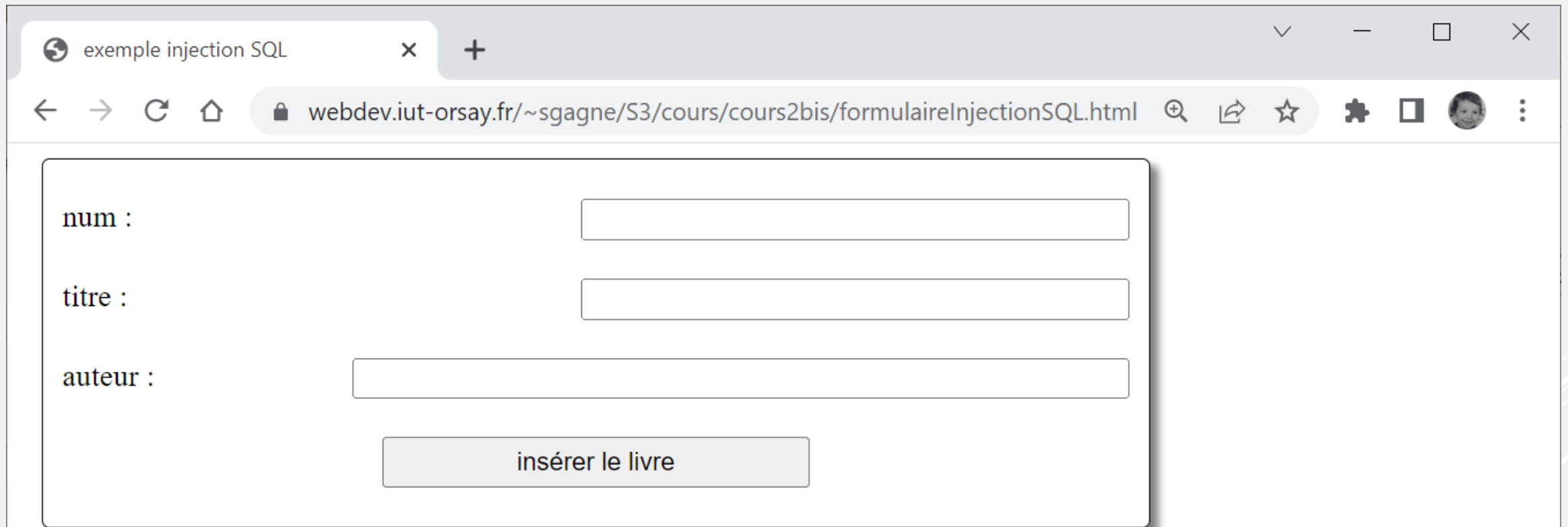
Requête risquée

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.
- En particulier, il faut contrôler ATTENTIVEMENT les informations issues d'un formulaire, dont la valeur est soumise par l'utilisateur du formulaire (l'internaute = malveillant par défaut).
- Dans la diapo suivante on simule le comportement d'un internaute malveillant :
 - Cet internaute a fait des hypothèses sur la façon dont le code côté serveur a été conçu
 - En particulier, il suppose que la requête envoyée est construite en récupérant les éléments du formulaire, et en les incluant dans une chaîne de caractères qui est la requête à envoyer.

Cette diapo va construire et montrer, par un script côté client (JavaScript), l'élaboration progressive de la requête à chaque changement des champs du formulaire...

Requête risquée – le formulaire côté client qui va être détourné par l'internaute malveillant



exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cours2bis/formulaireInjectionSQL.html

num :

titre :

auteur :

insérer le livre

requête construite à partir du formulaire

INSERT INTO livre VALUES(,"");

Requête risquée – le code serveur « deviné » par l'internaute malveillant

```
// récupération des données
```

```
$num = $_POST["num"];
```

```
$titre = $_POST["titre"];
```

```
$auteur = $_POST["auteur"];
```

```
// construction de la requête (en chaîne de caractères)
```

```
$requete = "INSERT INTO livre VALUES($num, '$titre', '$auteur');";
```



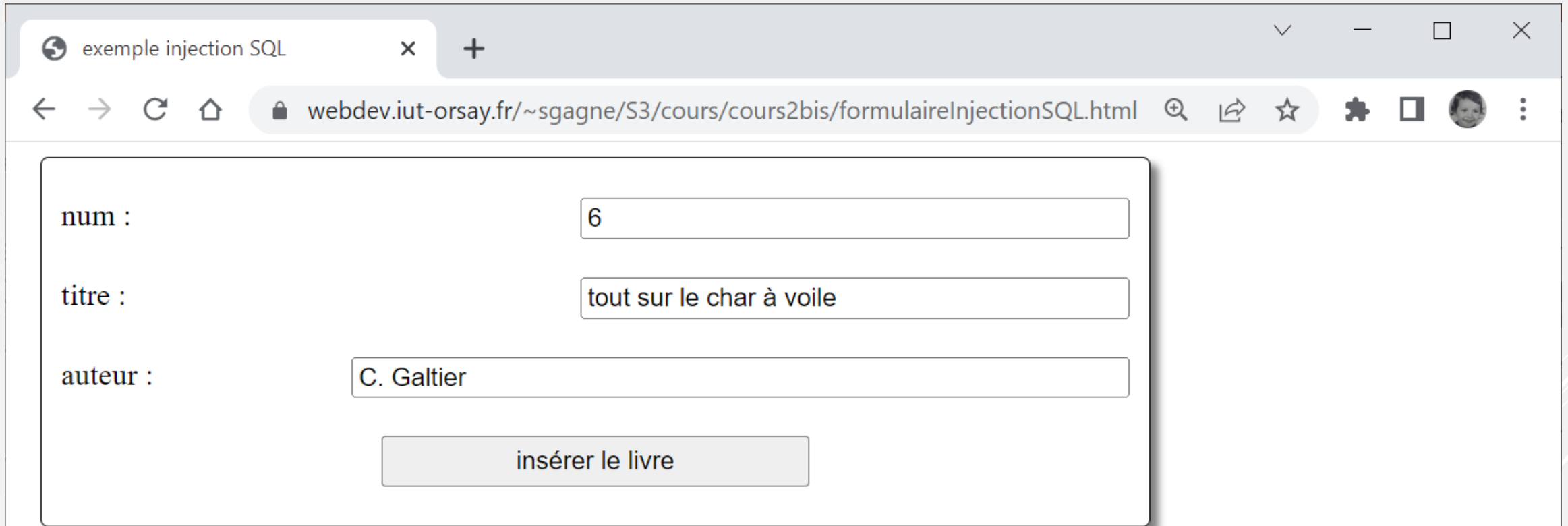
```
// Lancement de la requête sur la base de données
```

```
$resultat = Connexion::pdo()->query($requete);
```


Requête risquée – utilisation « normale » du formulaire

 ▼				num	titre	auteur
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	moi, ma vie, mon oeuvre	F. Ribéry
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	comment réussir sa vie	P. Balkany
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	ma vie en jaune	L. Armstrong
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	4	sur un coup de tête	Z. Zidane
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	5	la loi du préau	ZEP

Requête risquée – utilisation « normale » du formulaire



exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cours2bis/formulaireInjectionSQL.html

num : 6

titre : tout sur le char à voile

auteur : C. Galtier

insérer le livre

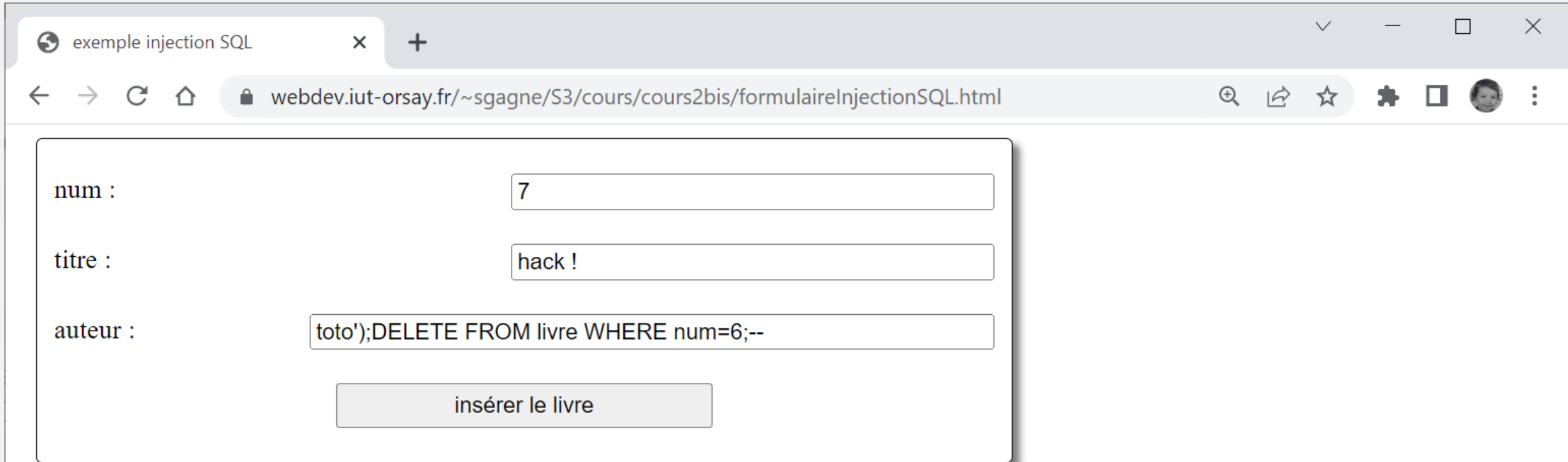
requête construite à partir du formulaire

```
INSERT INTO livre VALUES(6,'tout sur le char à voile','C. Galtier');
```

Requête risquée – utilisation « normale » du formulaire

 ▼				num	titre	auteur
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	moi, ma vie, mon oeuvre	F. Ribéry
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	comment réussir sa vie	P. Balkany
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	ma vie en jaune	L. Armstrong
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	4	sur un coup de tête	Z. Zidane
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	5	la loi du préau	ZEP
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	6	tout sur le char à voile	C. Galtier

Requête risquée – utilisation « malveillante » du formulaire



exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cours2bis/formulaireInjectionSQL.html

num : 7

titre : hack !

auteur : toto');DELETE FROM livre WHERE num=6;--

insérer le livre

requête construite à partir du formulaire

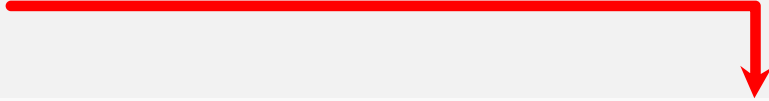
INSERT INTO livre VALUES(7,'hack !','toto');DELETE FROM livre WHERE num=6;--');

Requête risquée – utilisation « malveillante » du formulaire

 ▼				num	titre	auteur
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	1	moi, ma vie, mon oeuvre	F. Ribéry
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	2	comment réussir sa vie	P. Balkany
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	3	ma vie en jaune	L. Armstrong
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	4	sur un coup de tête	Z. Zidane
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	5	la loi du préau	ZEP
<input type="checkbox"/>	 Éditer	 Copier	 Supprimer	7	hack !	toto

Requête risquée – en résumé

```
$requete = "SELECT * FROM voiture WHERE immatriculation = '$i';";
```

- Tout paramètre (comme `$i`) introduit dans la requête doit faire l'objet d'un contrôle strict.
- En particulier, il faut contrôler ATTENTIVEMENT les informations issues d'un formulaire, dont la valeur est soumise par l'utilisateur du formulaire (l'internaute = malveillant par défaut).
- L'idéal est un **contrôle côté serveur !**
- On ne fait **jamais** comme ça ! 

// construction de la requête (en chaîne de caractères)

```
$requete = "INSERT INTO livre VALUES($num, '$titre', '$auteur');";
```

UTILISATION D'UNE BASE DE DONNÉES

Requêtes préparées

Requêtes préparées

- L'idée est de fournir un squelette modèle de requête pour le SQL, en utilisant des « tags » pour les différentes variables utilisées.

```
$requetePrepreee = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur);";
```


Requêtes préparées

- L'idée est de fournir un squelette modèle de requête pour le SQL, en utilisant des « tags » pour les différentes variables utilisées.

```
$requetePrepree = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur);";
```

```
$req_prep = Connexion::pdo()->prepare($requetePrepree);
```

Requêtes préparées

- L'idée est de fournir un squelette modèle de requête pour le SQL, en utilisant des « tags » pour les différentes variables utilisées.

```
$requetePrepreee = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur);";
```

```
$req_prep = Connexion::pdo()->prepare($requetePrepreee);
```

- On construit ensuite un tableau associatif permettant de lier les différents tags aux valeurs à passer (par exemple en provenance d'un formulaire).

Requêtes préparées

- L'idée est de fournir un squelette modèle de requête pour le SQL, en utilisant des « tags » pour les différentes variables utilisées.

```
$requetePrepreee = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur);";
```

```
$req_prep = Connexion::pdo()->prepare($requetePrepreee);
```

- On construit ensuite un tableau associatif permettant de lier les différents tags aux valeurs à passer (par exemple en provenance d'un formulaire).

```
$valeurs = array(  
    "tag_num" => $num,  
    "tag_titre" => $titre,  
    "tag_auteur" => $auteur  
);
```

Requêtes préparées

- L'idée est de fournir un squelette modèle de requête pour le SQL, en utilisant des « tags » pour les différentes variables utilisées.

```
$requetePrepreee = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur);";
```

```
$req_prep = Connexion::pdo()->prepare($requetePrepreee);
```

- On construit ensuite un tableau associatif permettant de lier les différents tags aux valeurs à passer (par exemple en provenance d'un formulaire).

```
$valeurs = array(  
    "tag_num" => $num,  
    "tag_titre" => $titre,  
    "tag_auteur" => $auteur  
);
```

- Puis les valeurs des variables sont insérées comme du texte pur, à la place préparée par les tags. Cette phase d'analyse prévient de tout risque d'injection SQL.

Requêtes préparées

- L'idée est de fournir un squelette modèle de requête pour le SQL, en utilisant des « tags » pour les différentes variables utilisées.

```
$requetePrepreee = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur);";
```

```
$req_prep = Connexion::pdo()->prepare($requetePrepreee);
```

- On construit ensuite un tableau associatif permettant de lier les différents tags aux valeurs à passer (par exemple en provenance d'un formulaire).

```
$valeurs = array(  
    "tag_num" => $num,  
    "tag_titre" => $titre,  
    "tag_auteur" => $auteur  
);
```

- Puis les valeurs des variables sont insérées comme du texte pur, à la place préparée par les tags. Cette phase d'analyse prévient de tout risque d'injection SQL.

```
try {  
    $req_prep->execute($valeurs);  
} catch (PDOException $e) {  
    echo $e->getMessage();  
}
```

Requêtes préparées

```
// recueil des données
$num = $_POST["num"];
$titre = $_POST["titre"];
$auteur = $_POST["auteur"];

// préparation de la requête à partir des données
$requetePrepree = "INSERT INTO livre VALUES(:tag_num,:tag_titre,:tag_auteur)";
$req_prep = Connexion::pdo()->prepare($requetePrepree);

// construction d'un tableau donnant les valeurs des différents tags de la requête
préparée
$valeurs = array(
    "tag_num" => $num,
    "tag_titre" => $titre,
    "tag_auteur" => $auteur
);

// exécution de la requête préparée
try {
    $req_prep->execute($valeurs);
} catch (PDOException $e) {
    echo $e->getMessage();
}
```

Requêtes préparées

num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto

avant

Requêtes préparées

exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cour...

num :

8

titre :

hack 2 !

auteur :

');DELETE FROM livre WHERE num=7;--

insérer le livre

num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto

avant

Requêtes préparées

exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cour...

num : 8

titre : hack 2 !

auteur : ');DELETE FROM livre WHERE num=7;--

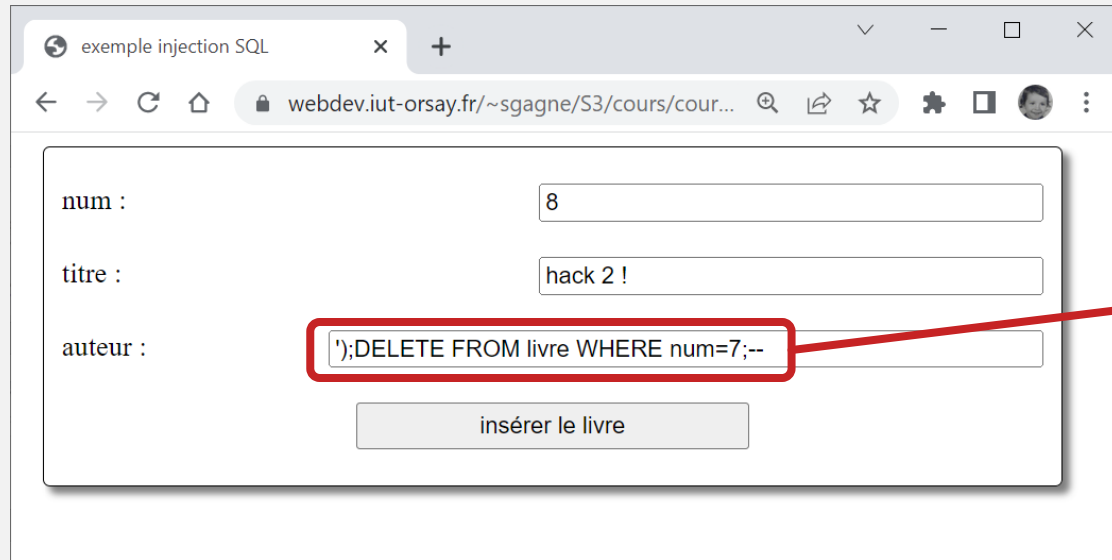
insérer le livre

tentative d'injection SQL

num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto

avant

Requêtes préparées



exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cour...

num : 8

titre : hack 2 !

auteur : ');DELETE FROM livre WHERE num=7;--

insérer le livre

tentative d'injection SQL

num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto

avant



num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto
8	hack 2 !	');DELETE FROM livre WHERE num=7;--

après

Requêtes préparées

exemple injection SQL

webdev.iut-orsay.fr/~sgagne/S3/cours/cour...

num :

8

titre :

hack 2 !

auteur :

'');DELETE FROM livre WHERE num=7;--

insérer le livre

tentative d'injection SQL
qui a échoué

num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto

avant

num	titre	auteur
1	moi, ma vie, mon oeuvre	F. Ribéry
2	comment réussir sa vie	P. Balkany
3	ma vie en jaune	L. Armstrong
4	sur un coup de tête	Z. Zidane
5	la loi du préau	ZEP
7	hack !	toto
8	hack 2 !	'');DELETE FROM livre WHERE num=7;--

après