Java cours 15 SQL injection

**Diapo 33**

On peut rajouter des ‘ dans la partie login avec 1 = 1 pour être sûr que c’est toujours vrai, on finit par -- pour faire une mise en commentaire de l’évaluation du password.

Il faut alors utiliser le preparedStatement quand demande de texte entré par l’utilisateur. Car les développeurs ont mis des gardes fous pour éviter les injection SQL.

**Diapo 37**

Il faut fermer le resultSet en fin d’utilisation si on ne le met pas en try-with-ressource

**Diapo 38**

Quand on ne connait pas la table, on peut utiliser getMetaData pour avoir les informations sur le résultat de la requête.

**Diapo 39**

Une transaction, permet d’effectuer un groupe de requête en même temps. Soit on fait toutes les requêtes soit aucune. Si une seule requête du lot n’est pas possible, on arrête et on annule les requêtes du groupe déjà réalisée.

setAutoCommit(boolean), en true on a un effet direct des modifications sur une table. Si false on est en mode transaction et les opérations seront observables seulement après le commit(). On peut annuler en faisant un rollback(). Souvent on met le rollback() dans le catch pour revenir dans l’état avant l’exception et on garde une base de données cohérentes.

**Diapo 41**

Dans une transaction il peut y avoir bcp de chose, on peut donc mettre des points de sauvegarde. Le rollback(savePoint) permet de revenir au niveau de la sauvegarde savePoint.

**Diapo 43**

On peut faire une itération sur les différentes exceptions qui ont été attrapée et les afficher une à une.

**Diapo 53**

Javax.sql plus adapté pour JEE (entreprise edition)

DataSource permet d’obtenir le nom de base de données depuis le serveur, et donc pas besoin de mettre le nom de la base en dur dans le code.

Créer une connexion prend du temps. Dans une application (e-commerce par exemple) il faut ouvrir une connexion pour chaque client. Il faut donc fermer et ouvrir des connexions sans arrêt. On utilise un pool de connexion. L’appli démarre avec 100 connexions au démarrage. Quand client arrive prend une connexion ouverte dans le pool, quand fini la remet dans le pool.

**Diapo 54**

C’est la DataSource qui gère le pool de connexion. On peut configurer l’objet DataSource pour sa gestion des connexions.

**Diapo 57**

Framework de persistance d’objet dans les bases de données (ex : Hibernate)

Slide non présentes sur le net -> Sur les framework mapping.

Hibernate permet à travers JDBC de créer des classes pour le modèle objet à partir du model relationnel et vice versa.