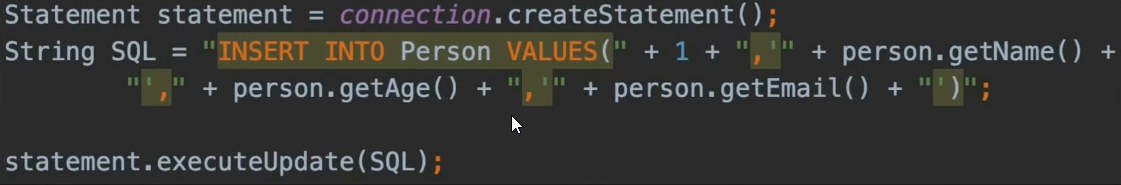
**SQL инъекции PreparedStatement**

У составления запроса вручную есть ряд минусов:

**1.** Неудобно.

**2.** Легко допустить ошибку.

**3.** Угроза SQL инъекции.

****

**SQL инъекция —** один из самых распостраненных способов взлома сайтов и программ, работающих с базами данных.

Так как строки из HTML формы напрямую конкатенируются в SQL запросе, злоумышленник может подобрать такую строку и ввести её в нашу HTML форму, которая нанесет вред, напрмер может удалиться таблица или напрмер может получить доступ ко всем записям из таблицы к которым он не должен получать.

Например:

**1.** если в качестве email`a в форме мы введем строку:

то таблица Person будет удалена из базы данных.

**2.** Если в поле ввода ввести такое значение:

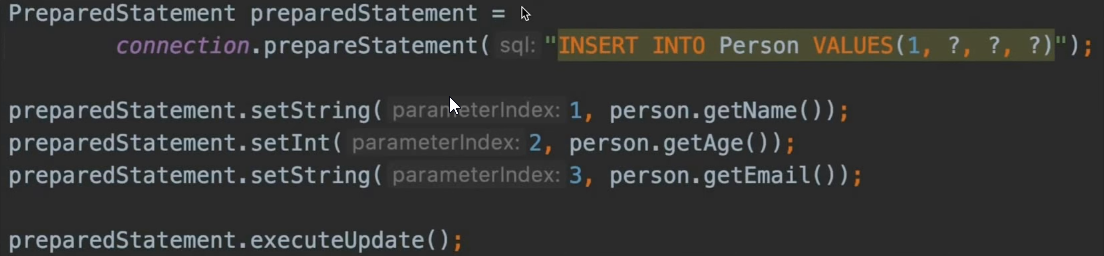
  
  
то может получится такой запрос:

из которого мы можем получить все данные из таблицы, так как условие WHERE будет для каждой строки true.

**Как защититься от SQL инъекций ?**

В JDBC API есть специальный класс который называется **PreparedStatement**. Это именно тот класс который должен использоваться в JDBC API для всех запросов, где содержатся данные, полученные от пользовател, например с формы.

Вот так выглядит запрос с помощью **PreparedStatement:**



Этот запрос добавляет строку в таблицу P*erson.*

**1.**Получаем объект **PreparedStatement** и передаем SQL запрос в аругменты.

В запросе для всех значений которые получаются с формы здесь пишутся вопросительные знаки (можно представить как некие параметры).

**2.** Потом с помощью сеттеров назначаем значения которые подходт под каждый вопросительный знак.

При этом SQL запрос компилируется один раз и не может быть изменен, тоесть, данные которые мы получаем от пользователя они могут быть вставлены лишь в места с вопросителным знаком и они никак не могут изменить SQL запрос.

**Преимущества PreparedStatemant перед Statemant:**

**1.** Удобнее использовать, чем **Statement** (не надо вручную заниматься конкатенацией).

2. Предотврощает SQL инъекции.

**3.** Быстрее, чем **Statement**. Особенно если запросов много.

**Разница в скорости PreparedStatemant и Statemant**

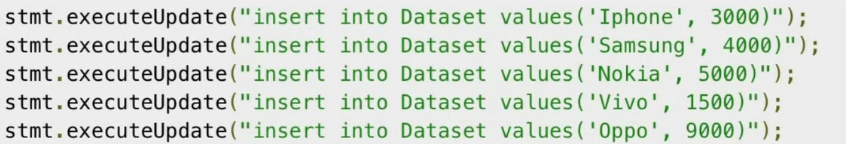
**PreparedStatemant –** SQL запрос компилируется только один раз.

**Statemant** – SQL запрос компилируется каждый раз.

**PreparedStatemant –** может кэшироваться на самой БД.

Пример:

**Statemant**

 **PreparedStatemant**

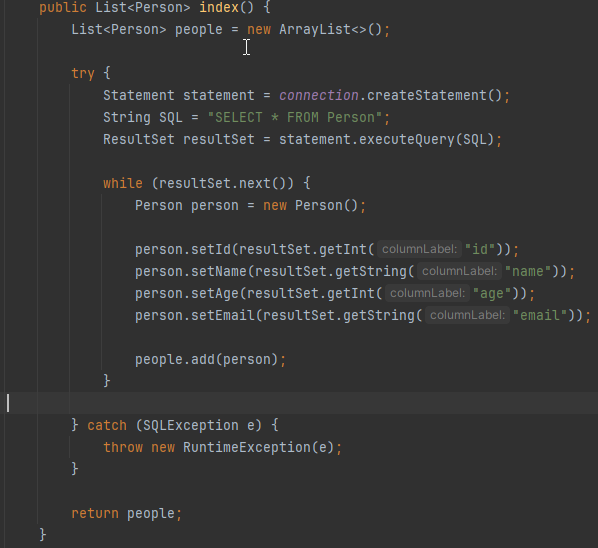


В случае с обычным **Statemant** каждй раз пишем SQL запрос, в случае с **PreparedStatemant** мы один раз написали SQL запрос, вместо конкретных значений поставили вопросительные знаки и с помощью сеттеров каждый раз вместо вопросительных знаков вставляем конкретные значения. Здесь есть большая разница в скорости выполнения запросов, чем больше строк будет внедряться в таблицу, тем больше будет разница в скорости. **PreparedStatemant** здесь работает гораздо быстрее потому-что мы написали один раз запрос и он один раз скомпилировался, в случае с **Statemant** компилируется каждый раз.

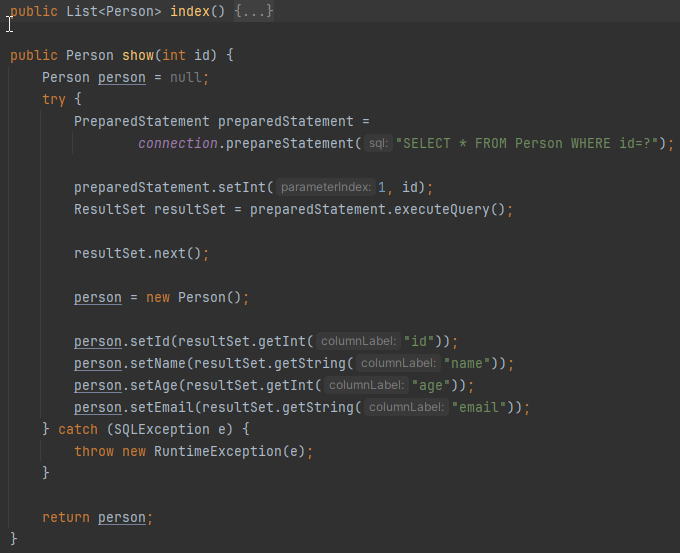
Помимо этого **PreparedStatemant** может кэшироваться на самой БД тоесть даже если мы удалим объект **PreparedStatemant** и создадим его еще раз с таким же SQL запросом база, то данных может понять, что этот запрос уже выполнялся и заново компилироваться запрос не будет, а всего лишь подставит нужные значения вместо знаков вопроса.

**Примеры кода с PreparedStatemant:**

**1.**



**2.**



**3.**

****