CC3201-1
Bases de Datos
Otoño 2017

Clase 8: SQL (IV)
Acceso programático

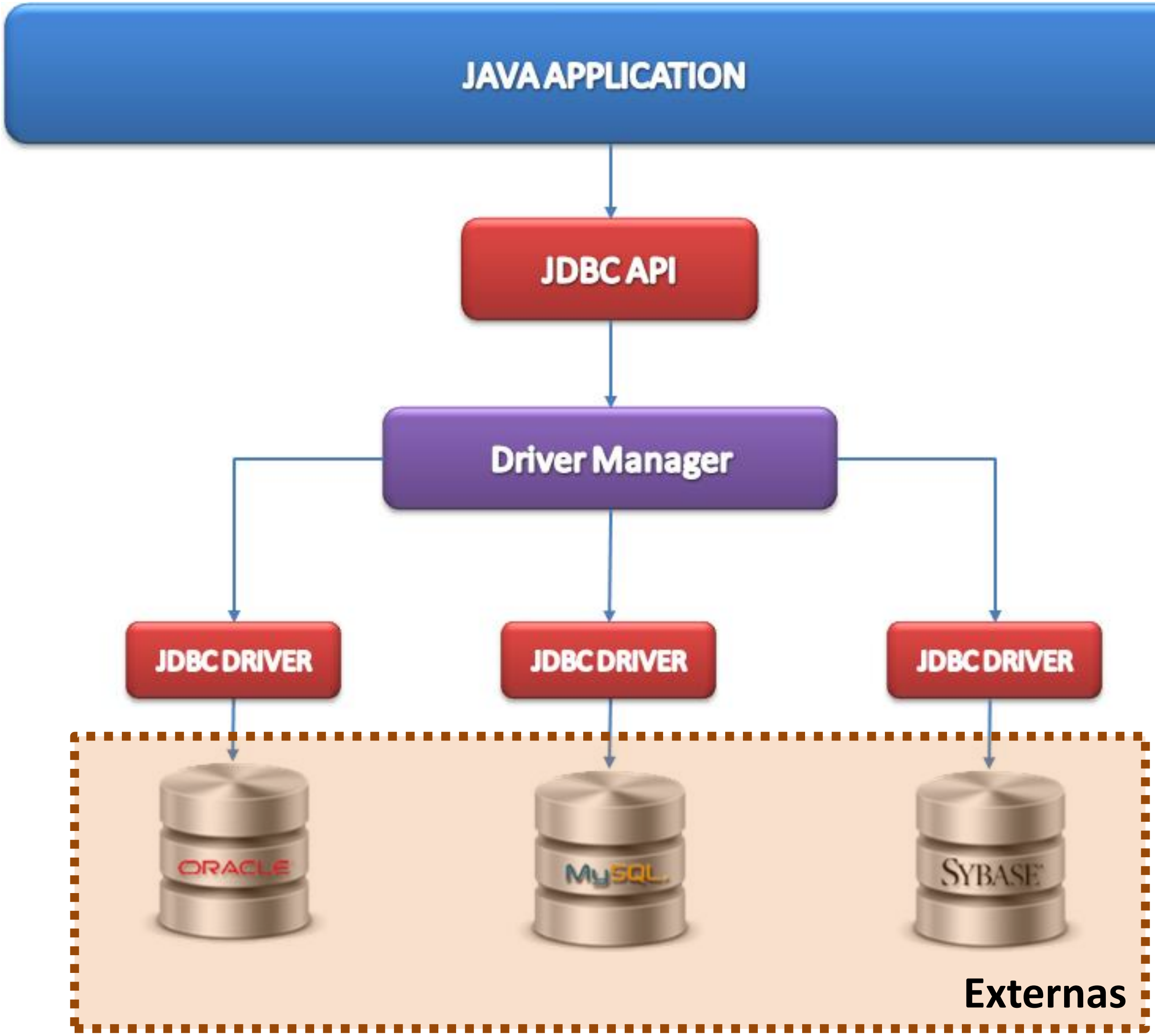
Aidan Hogan

aidhog@gmail.com

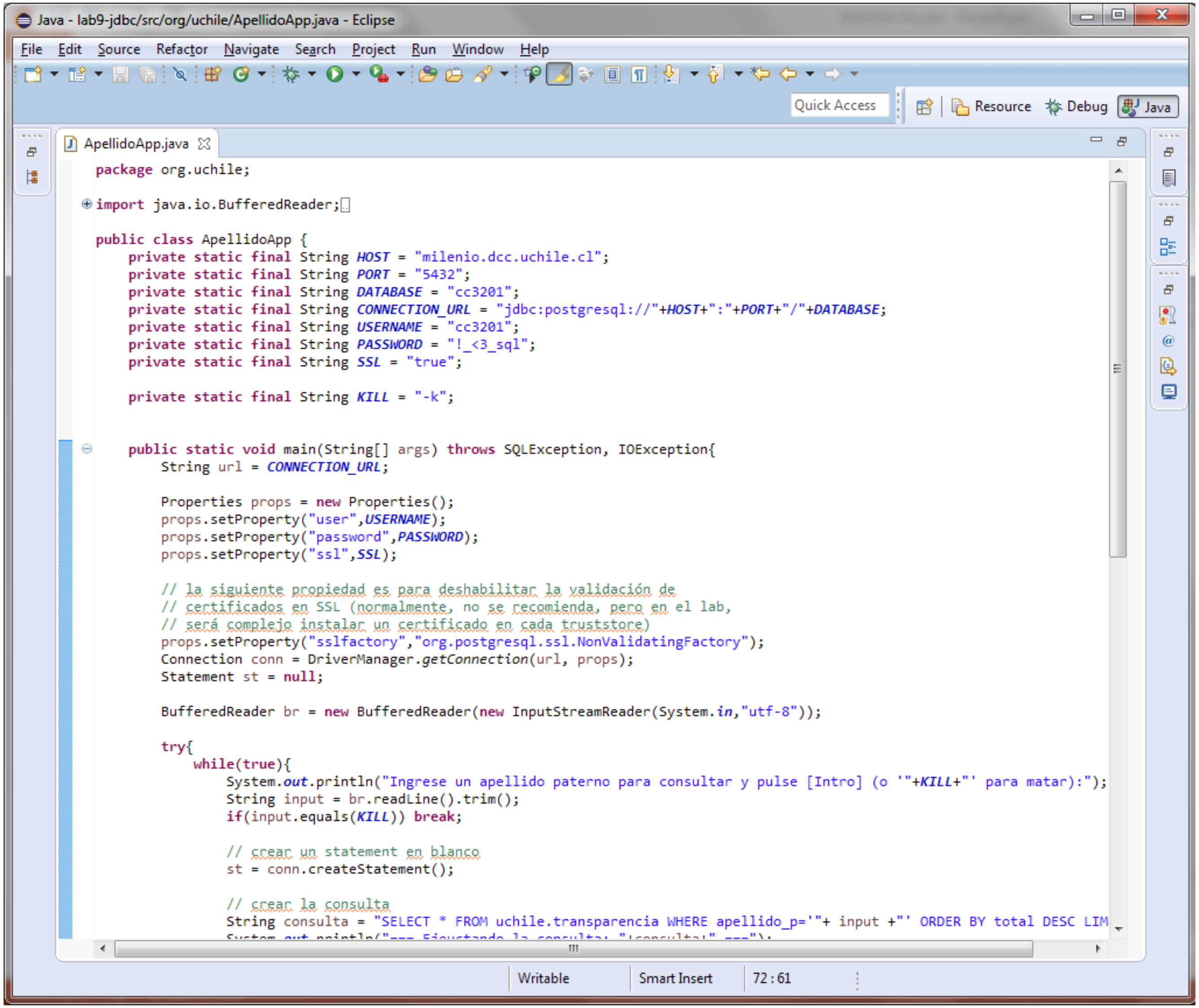
ACCESO PROGRAMÁTICO (JAVA): *JAVA DATABASE CONNECTIVITY* (JDBC)

Capítulo 6 | Ramakrishnan / Gehrke

Java Database Connectivity (JDBC)



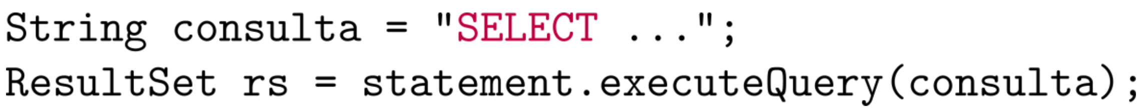
Veamos el ejemplo ApellidoApp.java



```java
package org.uchile;

import java.io.BufferedReader;

public class ApellidoApp {
    private static final String HOST = "milenio.dcc.uchile.cl";
    private static final String PORT = "5432";
    private static final String DATABASE = "cc3201";
    private static final String CONNECTION_URL = "jdbc:postgresql://"+HOST+":"+PORT+"/"+DATABASE;
    private static final String USERNAME = "cc3201";
    private static final String PASSWORD = "!_<3_sql";
    private static final String SSL = "true";

    private static final String KILL = "-k";


    public static void main(String[] args) throws SQLException, IOException{
        String url = CONNECTION_URL;

        Properties props = new Properties();
        props.setProperty("user",USERNAME);
        props.setProperty("password",PASSWORD);
        props.setProperty("ssl",SSL);

        // la siguiente propiedad es para deshabilitar la validación de
        // certificados en SSL (normalmente, no se recomienda, pero en el lab,
        // será complejo instalar un certificado en cada truststore)
        props.setProperty("sslfactory","org.postgresql.ssl.NonValidatingFactory");
        Connection conn = DriverManager.getConnection(url, props);
        Statement st = null;

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in,"utf-8"));

        try{
            while(true){
                System.out.println("Ingrese un apellido paterno para consultar y pulse [Intro] (o '"+KILL+"' para matar):");
                String input = br.readLine().trim();
                if(input.equals(KILL)) break;

                // crear un statement en blanco
                st = conn.createStatement();

                // crear la consulta
                String consulta = "SELECT * FROM uchile.transparencia WHERE apellido_p='"+ input +"' ORDER BY total DESC LIM
                System.out.println("=== Ejecutando la consulta: "+consulta+" ===");
```

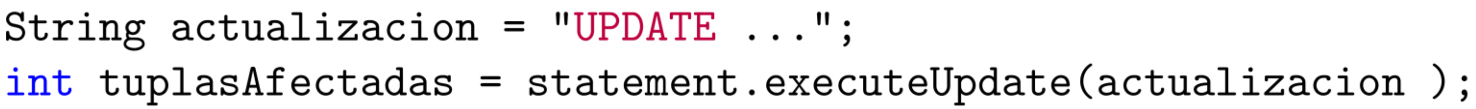
Consulta vs. Actualización

- Para hacer consultas (SELECT):



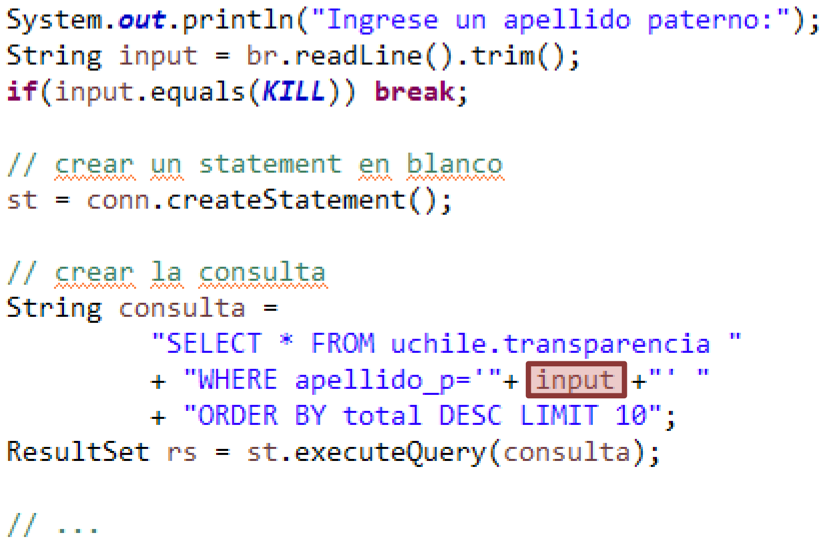
```
String consulta = "SELECT ...";
ResultSet rs = statement.executeQuery(consulta);
```

- Para hacer actualizaciones (INSERT; UPDATE, …)



```
String actualizacion = "UPDATE ...";
int tuplasAfectadas = statement.executeUpdate(actualizacion );
```

Un problema …



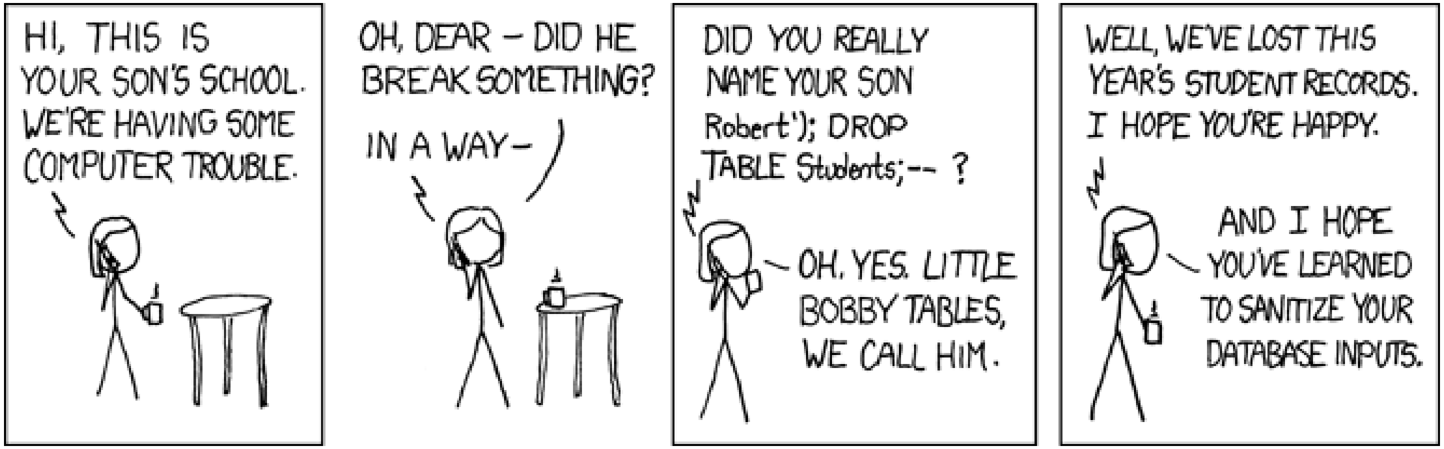
```java
System.out.println("Ingrese un apellido paterno:");
String input = br.readLine().trim();
if(input.equals(KILL)) break;

// crear un statement en blanco
st = conn.createStatement();

// crear la consulta
String consulta =
         "SELECT * FROM uchile.transparencia "
         + "WHERE apellido_p='"+ input +"' "
         + "ORDER BY total DESC LIMIT 10";
ResultSet rs = st.executeQuery(consulta);

// ...
```

¿Hay algún problema aquí?

… no hemos “verificado” el input.

Inyección SQL

- Un usuario malintencionado puede ingresar un string de entrada para hacer algo inesperado


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

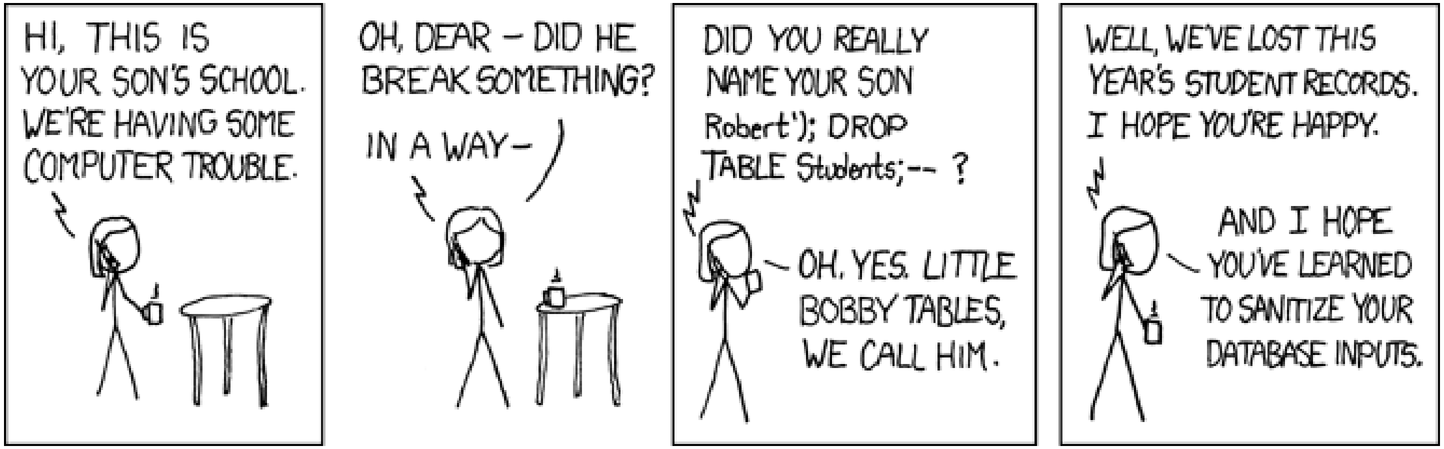
```
"SELECT nota FROM Students WHERE name='"+input+"'"
```

```
"SELECT nota FROM Students WHERE name='Robert'); DROP TABLE Students; -- '"
```

('--' empieza un comentario)

Inyección SQL (muchas formas)

- Un usuario malintencionado puede ingresar un string de entrada para hacer algo inesperado

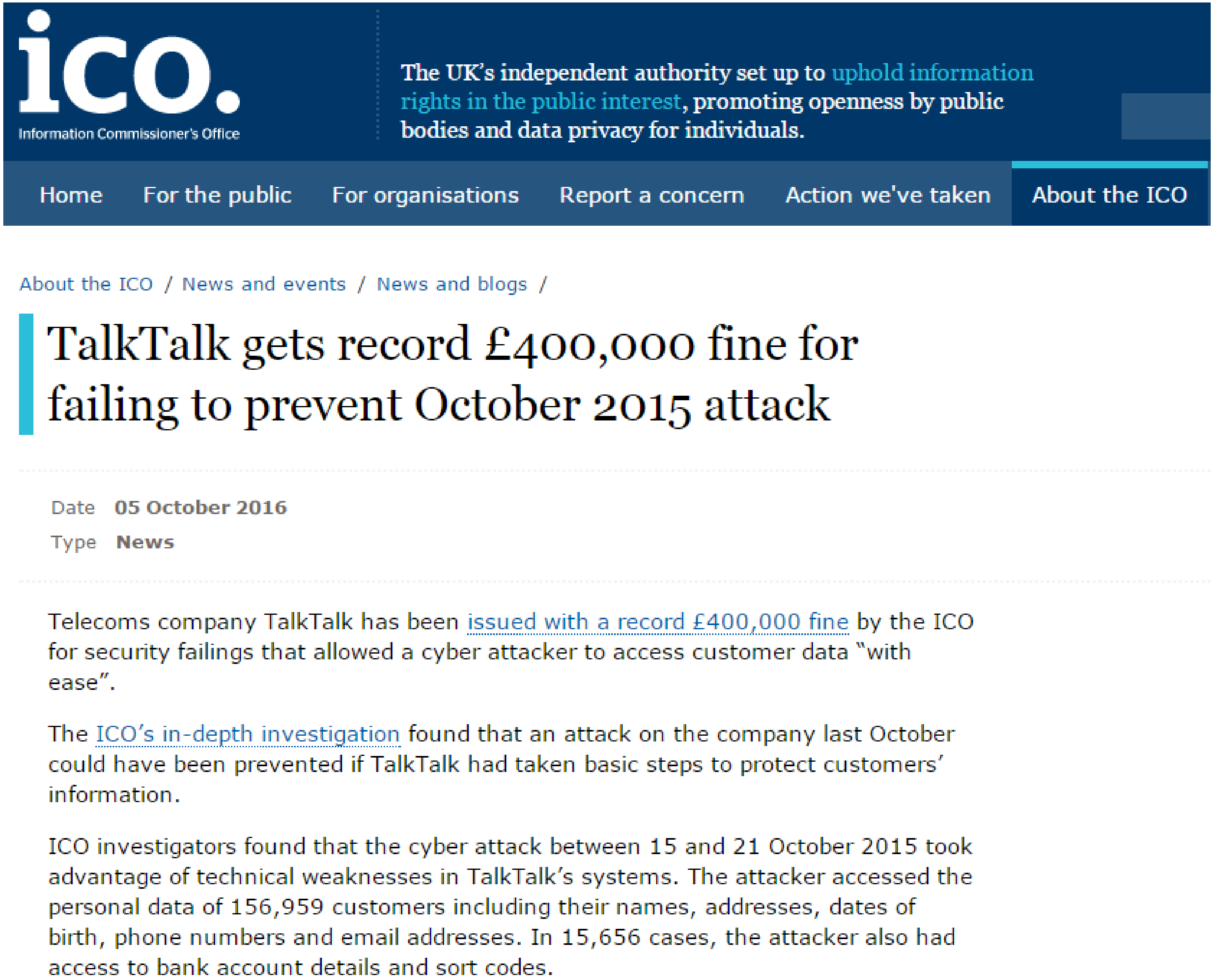

HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

```
"SELECT nota FROM Students WHERE name='"+input+"'"

"SELECT nota FROM Students WHERE name='Robert' OR 1=1;'"
```

¿Qué hace el ejemplo?

¡Devolverá toda la tabla!

Parece estúpido pero (por ejemplo) …



The UK's independent authority set up to uphold information rights in the public interest, promoting openness by public bodies and data privacy for individuals.

About the ICO / News and events / News and blogs /

TalkTalk gets record £400,000 fine for failing to prevent October 2015 attack

Date **05 October 2016**

Type **News**

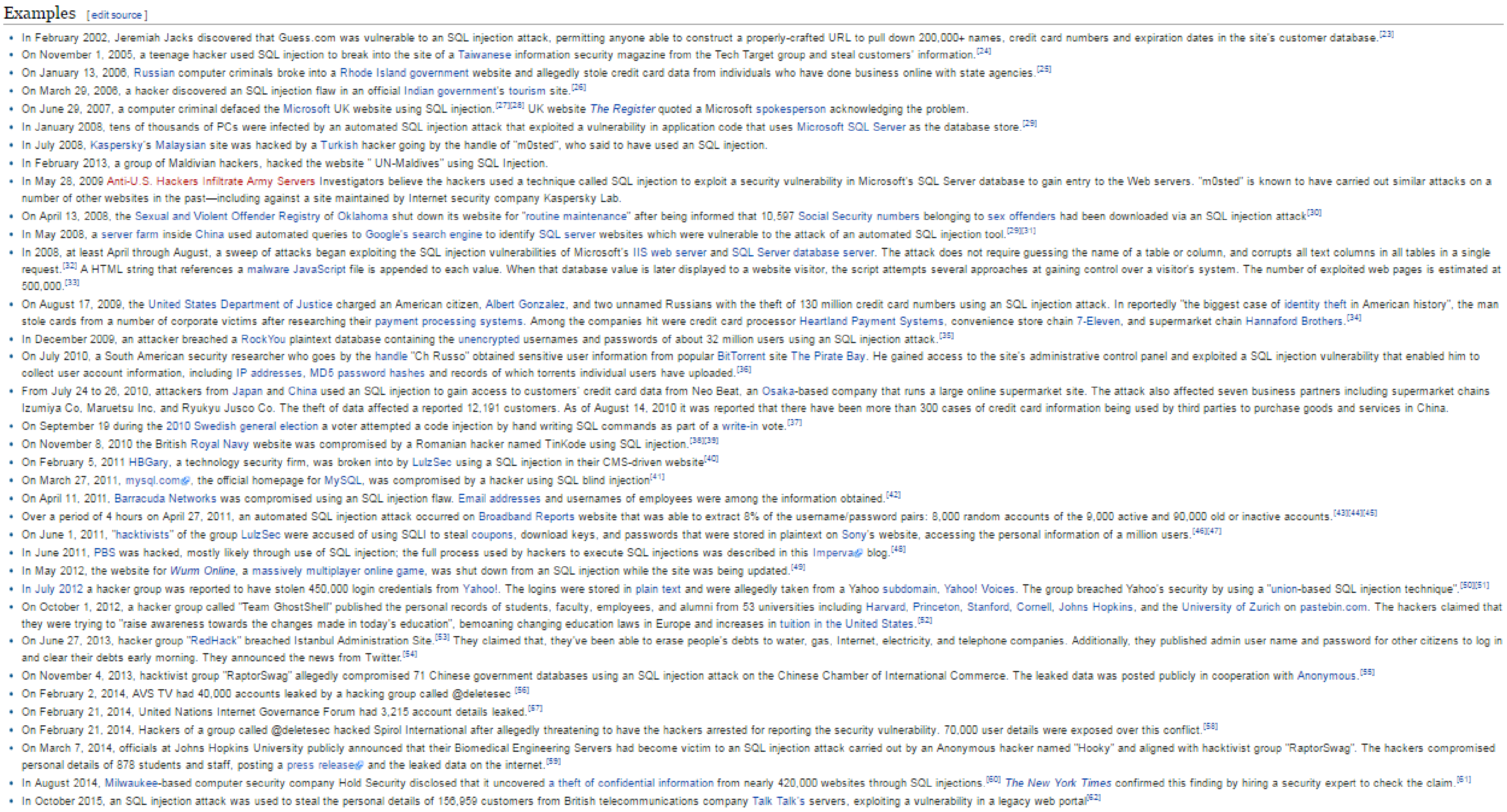
Telecoms company TalkTalk has been issued with a record £400,000 fine by the ICO for security failings that allowed a cyber attacker to access customer data "with ease".

The ICO's in-depth investigation found that an attack on the company last October could have been prevented if TalkTalk had taken basic steps to protect customers' information.

ICO investigators found that the cyber attack between 15 and 21 October 2015 took advantage of technical weaknesses in TalkTalk's systems. The attacker accessed the personal data of 156,959 customers including their names, addresses, dates of birth, phone numbers and email addresses. In 15,656 cases, the attacker also had access to bank account details and sort codes.

Más ejemplos …

https://en.wikipedia.org/wiki/SQL_injection



Examples [edit source]

- In February 2002, Jeremiah Jacks discovered that Guess.com was vulnerable to an SQL injection attack, permitting anyone able to construct a properly-crafted URL to pull down 200,000+ names, credit card numbers and expiration dates in the site's customer database.[23]
- On November 1, 2005, a teenage hacker used SQL injection to break into the site of a Taiwanese information security magazine from the Tech Target group and steal customers' information.[24]
- On January 13, 2006, Russian computer criminals broke into a Rhode Island government website and allegedly stole credit card data from individuals who have done business online with state agencies.[25]
- On March 29, 2006, a hacker discovered an SQL injection flaw in an official Indian government's tourism site.[26]
- On June 29, 2007, a computer criminal defaced the Microsoft UK website using SQL injection.[27][28] UK website *The Register* quoted a Microsoft spokesperson acknowledging the problem.
- In January 2008, tens of thousands of PCs were infected by an automated SQL injection attack that exploited a vulnerability in application code that uses Microsoft SQL Server as the database store.[29]
- In July 2008, Kaspersky's Malaysian site was hacked by a Turkish hacker going by the handle of "m0sted", who said to have used an SQL injection.
- In February 2013, a group of Maldivian hackers, hacked the website " UN-Maldives" using SQL Injection.
- In May 28, 2009 Anti-U.S. Hackers Infiltrate Army Servers Investigators believe the hackers used a technique called SQL injection to exploit a security vulnerability in Microsoft's SQL Server database to gain entry to the Web servers. "m0sted" is known to have carried out similar attacks on a number of other websites in the past—including against a site maintained by Internet security company Kaspersky Lab.
- On April 13, 2008, the Sexual and Violent Offender Registry of Oklahoma shut down its website for "routine maintenance" after being informed that 10,597 Social Security numbers belonging to sex offenders had been downloaded via an SQL injection attack[30]
- In May 2008, a server farm inside China used automated queries to Google's search engine to identify SQL server websites which were vulnerable to the attack of an automated SQL injection tool.[29][31]
- In 2008, at least April through August, a sweep of attacks began exploiting the SQL injection vulnerabilities of Microsoft's IIS web server and SQL Server database server. The attack does not require guessing the name of a table or column, and corrupts all text columns in all tables in a single request.[32] A HTML string that references a malware JavaScript file is appended to each value. When that database value is later displayed to a website visitor, the script attempts several approaches at gaining control over a visitor's system. The number of exploited web pages is estimated at 500,000.[33]
- On August 17, 2009, the United States Department of Justice charged an American citizen, Albert Gonzalez, and two unnamed Russians with the theft of 130 million credit card numbers using an SQL injection attack. In reportedly "the biggest case of identity theft in American history", the man stole cards from a number of corporate victims after researching their payment processing systems. Among the companies hit were credit card processor Heartland Payment Systems, convenience store chain 7-Eleven, and supermarket chain Hannaford Brothers.[34]
- In December 2009, an attacker breached a RockYou plaintext database containing the unencrypted usernames and passwords of about 32 million users using an SQL injection attack.[35]
- On July 2010, a South American security researcher who goes by the handle "Ch Russo" obtained sensitive user information from popular BitTorrent site The Pirate Bay. He gained access to the site's administrative control panel and exploited a SQL injection vulnerability that enabled him to collect user account information, including IP addresses, MD5 password hashes and records of which torrents individual users have uploaded.[36]
- From July 24 to 26, 2010, attackers from Japan and China used an SQL injection to gain access to customers' credit card data from Neo Beat, an Osaka-based company that runs a large online supermarket site. The attack also affected seven business partners including supermarket chains Izumiya Co, Maruetsu Inc, and Ryukyu Jusco Co. The theft of data affected a reported 12,191 customers. As of August 14, 2010 it was reported that there have been more than 300 cases of credit card information being used by third parties to purchase goods and services in China.
- On September 19 during the 2010 Swedish general election a voter attempted a code injection by hand writing SQL commands as part of a write-in vote.[37]
- On November 8, 2010 the British Royal Navy website was compromised by a Romanian hacker named TinKode using SQL injection.[38][39]
- On February 5, 2011 HBGary, a technology security firm, was broken into by LulzSec using a SQL injection in their CMS-driven website[40]
- On March 27, 2011, mysql.com, the official homepage for MySQL, was compromised by a hacker using SQL blind injection[41]
- On April 11, 2011, Barracuda Networks was compromised using an SQL injection flaw. Email addresses and usernames of employees were among the information obtained.[42]
- Over a period of 4 hours on April 27, 2011, an automated SQL injection attack occurred on Broadband Reports website that was able to extract 8% of the username/password pairs: 8,000 random accounts of the 9,000 active and 90,000 old or inactive accounts.[43][44][45]
- On June 1, 2011, "hacktivists" of the group LulzSec were accused of using SQLI to steal coupons, download keys, and passwords that were stored in plaintext on Sony's website, accessing the personal information of a million users.[46][47]
- In June 2011, PBS was hacked, mostly likely through use of SQL injection; the full process used by hackers to execute SQL injections was described in this Imperva blog.[48]
- In May 2012, the website for *Wurm Online*, a massively multiplayer online game, was shut down from an SQL injection while the site was being updated.[49]
- In July 2012 a hacker group was reported to have stolen 450,000 login credentials from Yahoo!. The logins were stored in plain text and were allegedly taken from a Yahoo subdomain, Yahoo! Voices. The group breached Yahoo's security by using a "union-based SQL injection technique".[50][51]
- On October 1, 2012, a hacker group called "Team GhostShell" published the personal records of students, faculty, employees, and alumni from 53 universities including Harvard, Princeton, Stanford, Cornell, Johns Hopkins, and the University of Zurich on pastebin.com. The hackers claimed that they were trying to "raise awareness towards the changes made in today's education", bemoaning changing education laws in Europe and increases in tuition in the United States.[52]
- On June 27, 2013, hacker group "RedHack" breached Istanbul Administration Site.[53] They claimed that, they've been able to erase people's debts to water, gas, Internet, electricity, and telephone companies. Additionally, they published admin user name and password for other citizens to log in and clear their debts early morning. They announced the news from Twitter.[54]
- On November 4, 2013, hacktivist group "RaptorSwag" allegedly compromised 71 Chinese government databases using an SQL injection attack on the Chinese Chamber of International Commerce. The leaked data was posted publicly in cooperation with Anonymous.[55]
- On February 2, 2014, AVS TV had 40,000 accounts leaked by a hacking group called @deletesec [56]
- On February 21, 2014, United Nations Internet Governance Forum had 3,215 account details leaked.[57]
- On February 21, 2014, Hackers of a group called @deletesec hacked Spirol International after allegedly threatening to have the hackers arrested for reporting the security vulnerability. 70,000 user details were exposed over this conflict.[58]
- On March 7, 2014, officials at Johns Hopkins University publicly announced that their Biomedical Engineering Servers had become victim to an SQL injection attack carried out by an Anonymous hacker named "Hooky" and aligned with hacktivist group "RaptorSwag". The hackers compromised personal details of 878 students and staff, posting a press release and the leaked data on the internet.[59]
- In August 2014, Milwaukee-based computer security company Hold Security disclosed that it uncovered a theft of confidential information from nearly 420,000 websites through SQL injections.[60] *The New York Times* confirmed this finding by hiring a security expert to check the claim.[61]
- In October 2015, an SQL injection attack was used to steal the personal details of 156,959 customers from British telecommunications company Talk Talk's servers, exploiting a vulnerability in a legacy web portal[62]

Más ejemplos …

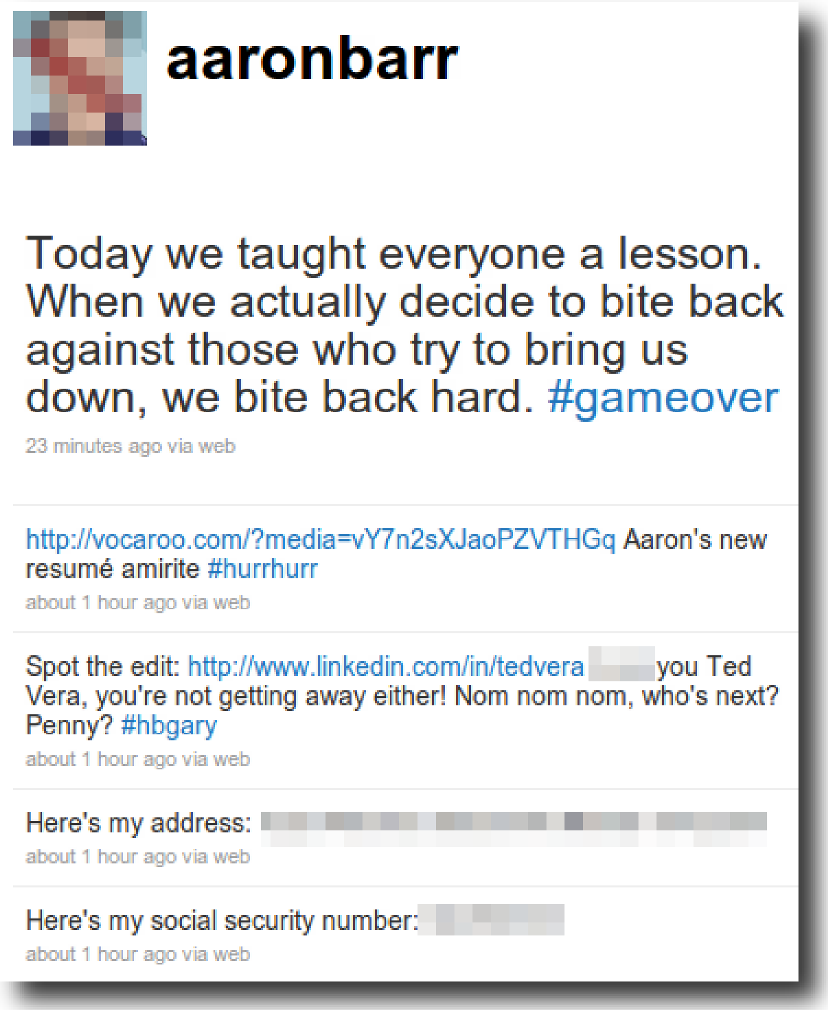
https://en.wikipedia.org/wiki/SQL_injection

Examples [edit source]

- request.[32] A HTML string that references a malware JavaScript file is appended to each value. When that data
  500,000.[33]
- hand writing SQL commands as part of a write-in vote.[37]
- named TinKode using SQL injection.[38][39]
- injection in their CMS-driven website[40]
- ker using SQL blind injection[41]
- resses and usernames of employees were among the information obtained.[42]
- band Reports website that was able to extract 8% of the username/password pairs: 8,000 random accounts
- On June 1, 2011, "hacktivists" of the group LulzSec were accused of using SQLI to steal coupons, download keys, and passwords that were stored in plaintext on Sony's website, accessing the personal i
- In June 2011, PBS was hack ; the full process use s described in this Imperva blog.[48]
- In May 2012, the website for game, was shut dow being updated.[49]
- In July 2012 a hacker group dentials from Yahoo! ere allegedly tak
- On October 1, 2012, a hacke e personal records of from 53 univers
  they were trying to "raise awa education", bemoan increases in tui
- On June 27, 2013, hacker gro on Site.[53] They clai e's debts to wat
  and clear their debts early mo r.[54]
- On November 4, 2013, hackt ed 71 Chinese gover ttack on the Chi
- On February 2, 2014, AVS TV roup called @deletes
- On February 21, 2014, United 15 account details le
- On February 21, 2014, Hacke l International after rested for repor
- On March 7, 2014, officials a d that their Biomedic to an SQL inject
  personal details of 878 studen e leaked data on the
- In August 2014, Milwaukee-based computer security company Hold Security disclosed that it uncovered a theft of confidential information from nearly 420
- In October 2015, an SQL injection attack was used to steal the personal details of 156,959 customers from British telecommunications company Talk Talk's servers, exploiting a vulnerability in a legacy web portal[62]

SONY
YAHOO!
The Pirate Bay
Microsoft
MySQL
7 ELEVEn
ROYAL NAVY
HBGary
Detecting Tomorrow's Threats Today
Part of ManTech International Corporation

El Jefe de HBGary …



aaronbarr

Today we taught everyone a lesson. When we actually decide to bite back against those who try to bring us down, we bite back hard. #gameover

23 minutes ago via web

http://vocaroo.com/?media=vY7n2sXJaoPZVTHGq Aaron's new resumé amirite #hurrhurr

about 1 hour ago via web

Spot the edit: http://www.linkedin.com/in/tedvera [redacted] you Ted Vera, you're not getting away either! Nom nom nom, who's next? Penny? #hbgary

about 1 hour ago via web

Here's my address: [redacted]

about 1 hour ago via web

Here's my social security number: [redacted]

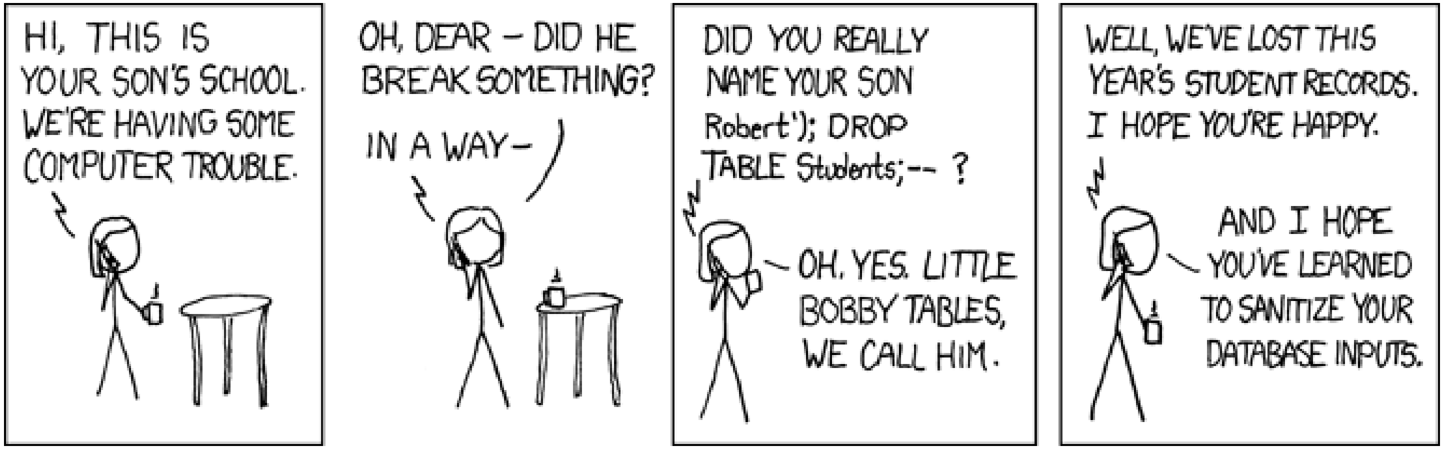
about 1 hour ago via web

HBGary

Detecting Tomorrow's Threats Today

Part of ManTech International Corporation

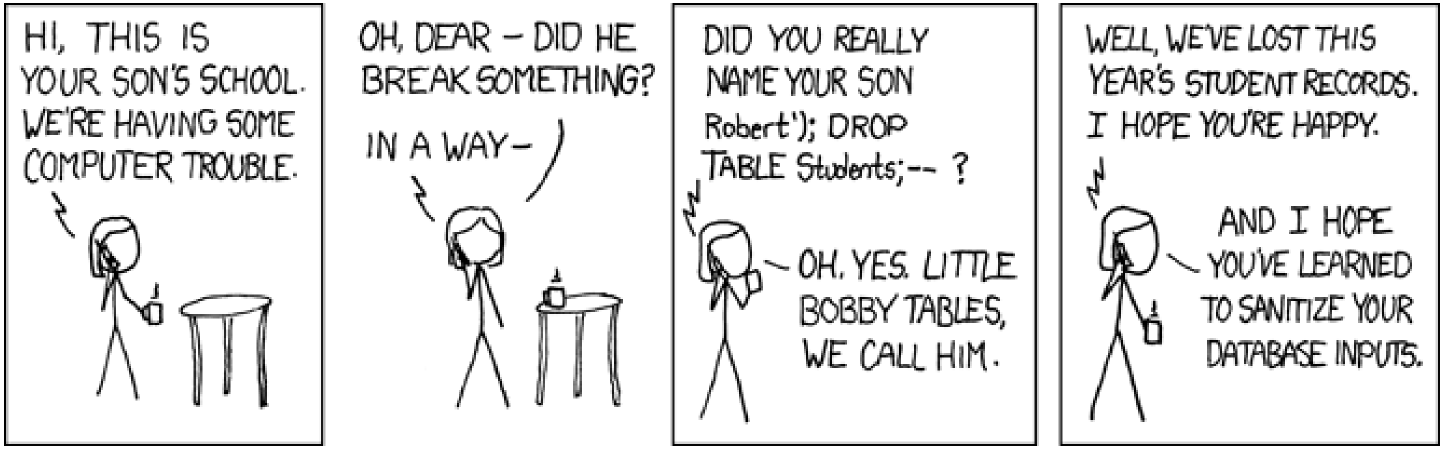
Inyección SQL


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

```
String consulta = "SELECT nota FROM Students WHERE name='"+input+"'";
ResultSet rs = statement.executeQuery(consulta);
```

¿Cómo podemos resolver el problema?

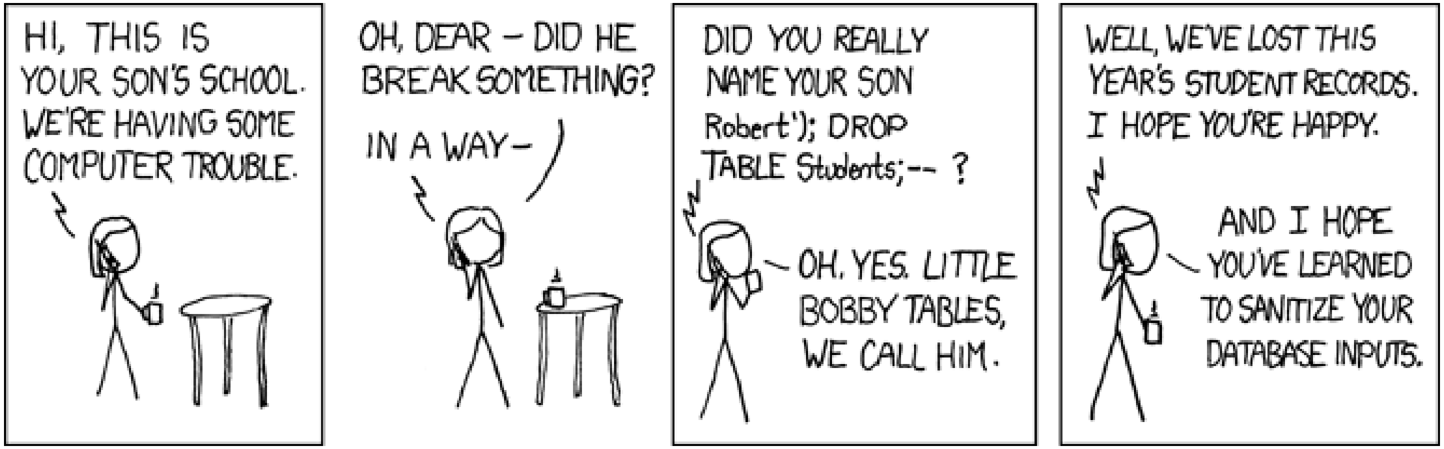
Inyección SQL: ¿escapar los strings?


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

```
String consulta = "SELECT nota FROM Students WHERE name='"+escapar(input)+"'";
ResultSet rs = statement.executeQuery(consulta);
```

Mejor, pero sería complicado implementar la función escapar en un lenguaje de programación general y garantizar que prevente cada tipo de inyección en cada versión (futura) de cada sistema de BdD dado cualquier tipo de consulta y entrada!

Inyección SQL: *¡sentencias precompiladas!*


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

```
String consulta = "SELECT nota FROM Students WHERE name='?'";
// donde ?  es un parámetro que reemplezaremos con la entrada del usuario
PreparedStatement ps = conn.prepareStatement(consulta);
ps.setString(1, input);
ResultSet rs = ps.executeQuery();
```

Mandamos la consulta al sistema de bases de datos y después reemplazar los parámetros con la entrada del usuario

Inyección SQL: *¡sentencias precompiladas!*

```
String consulta = "SELECT nota FROM Students WHERE name=?";

PreparedStatement ps = conn.prepareStatement(consulta);      // 1
ps.setString(1, input);                                      // 2
ResultSet rs = ps.executeQuery();                            // 3
```

```
// 1 :  El sistema de bases de datos compila la sentencia
SELECT nota FROM Students WHERE name=?
                                      QUERY PLAN
          -------------------------------------------------------------------
           Seq Scan on Students  (cost=0.00..9654.67 rows=57 width=132)
             Filter: ((name)::text = ?::text)
```

El sistema de base de datos

La consulta es compilada por el sistema ***sin*** *la entrara*

Inyección SQL: *¡sentencias precompiladas!*

```
String consulta = "SELECT nota FROM Students WHERE name=?";

PreparedStatement ps = conn.prepareStatement(consulta);      // 1
ps.setString(1, input);                                      // 2
ResultSet rs = ps.executeQuery();                            // 3
```

```
// 2 :  El sistema de bases de datos reempleza el parametro en el plan
SELECT nota FROM Students WHERE name=?
                                  QUERY PLAN
          -----------------------------------------------------------------
           Seq Scan on Students  (cost=0.00..9654.67 rows=57 width=132)
             Filter: ((name)::text = 'Robert'::text)
```

El sistema de base de datos

Se reemplaza el parámetro en la sentencia precompilada

(que es un plan en memoria, no un string)

Inyección SQL: *¡sentencias precompiladas!*

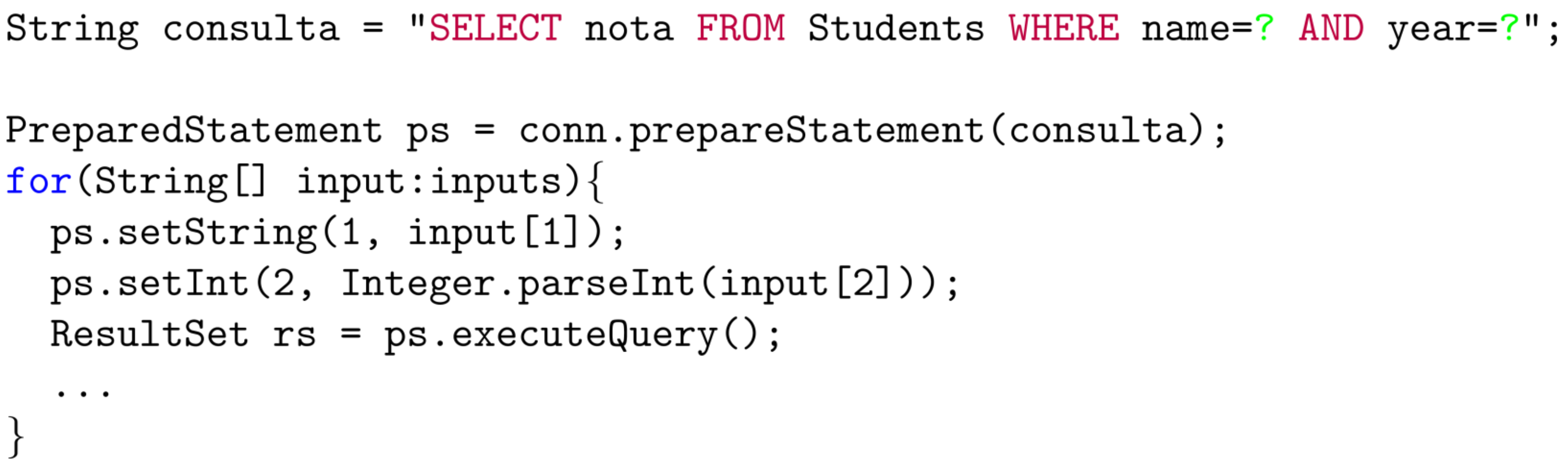
```
String consulta = "SELECT nota FROM Students WHERE name=?";

PreparedStatement ps = conn.prepareStatement(consulta);      // 1
ps.setString(1, input);                                      // 2
ResultSet rs = ps.executeQuery();                            // 3
```

```
// 3 :  El sistema de bases de datos ejecuta el plan
SELECT nota FROM Students WHERE name=?
                                       QUERY PLAN
         -------------------------------------------------------------------
          Seq Scan on Students  (cost=0.00..9654.67 rows=57 width=132)
            Filter: ((name)::text = 'Robert'::text)
```

El sistema de base de datos

nota
3,7

Sentencias precompiladas



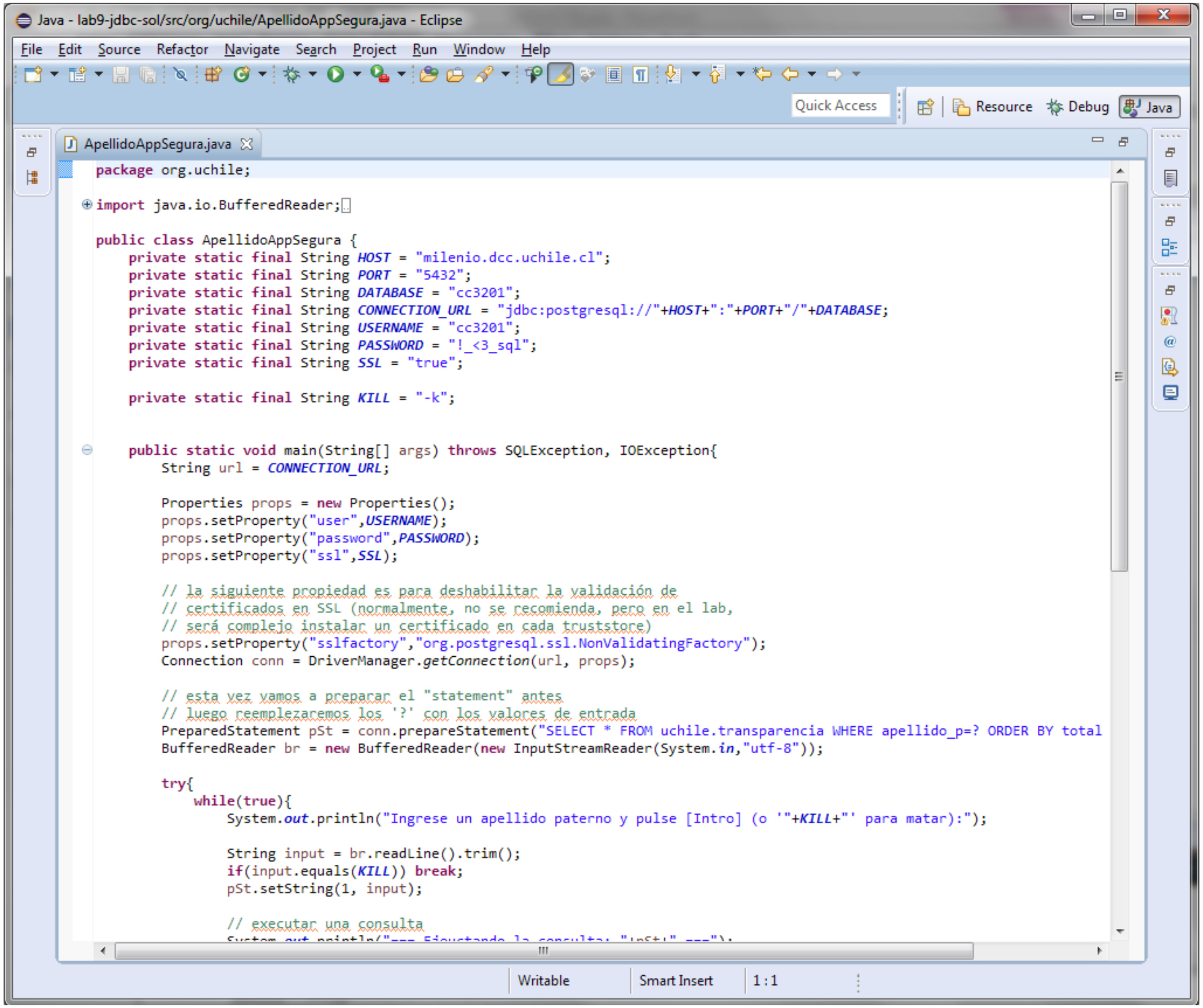
```
String consulta = "SELECT nota FROM Students WHERE name=? AND year=?";

PreparedStatement ps = conn.prepareStatement(consulta);
for(String[] input:inputs){
  ps.setString(1, input[1]);
  ps.setInt(2, Integer.parseInt(input[2]));
  ResultSet rs = ps.executeQuery();
  ...
}
```

Se puede reutilizar el PreparedStatement varias veces

(es más eficiente también: se compila la sentenica sólo una vez

Se puede tener varios parámetros con varios tipos

Veamos el ejemplo ApellidoAppSegura.java


Java - lab9-jdbc-sol/src/org/uchile/ApellidoAppSegura.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Resource
Debug
Java
ApellidoAppSegura.java
Writable
Smart Insert
1 : 1

```java
package org.uchile;

import java.io.BufferedReader;

public class ApellidoAppSegura {
    private static final String HOST = "milenio.dcc.uchile.cl";
    private static final String PORT = "5432";
    private static final String DATABASE = "cc3201";
    private static final String CONNECTION_URL = "jdbc:postgresql://"+HOST+":"+PORT+"/"+DATABASE;
    private static final String USERNAME = "cc3201";
    private static final String PASSWORD = "!_<3_sql";
    private static final String SSL = "true";

    private static final String KILL = "-k";


    public static void main(String[] args) throws SQLException, IOException{
        String url = CONNECTION_URL;

        Properties props = new Properties();
        props.setProperty("user",USERNAME);
        props.setProperty("password",PASSWORD);
        props.setProperty("ssl",SSL);

        // la siguiente propiedad es para deshabilitar la validación de
        // certificados en SSL (normalmente, no se recomienda, pero en el lab,
        // será complejo instalar un certificado en cada truststore)
        props.setProperty("sslfactory","org.postgresql.ssl.NonValidatingFactory");
        Connection conn = DriverManager.getConnection(url, props);

        // esta vez vamos a preparar el "statement" antes
        // luego reemplazaremos los '?' con los valores de entrada
        PreparedStatement pSt = conn.prepareStatement("SELECT * FROM uchile.transparencia WHERE apellido_p=? ORDER BY total
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in,"utf-8"));

        try{
            while(true){
                System.out.println("Ingrese un apellido paterno y pulse [Intro] (o '"+KILL+"' para matar):");

                String input = br.readLine().trim();
                if(input.equals(KILL)) break;
                pSt.setString(1, input);

                // executar una consulta
                System.out.println("=== Ejecutando la consulta: "+pSt+" ===");
```

Preguntas?