### ****Basic Theory Work****

### ****What is SQL Injection?****

* **SQL:** Stands for Structured Query Language. It's a way to talk to databases and get information from them.
* **Injection:** In this context, it means putting something harmful into something else.

### ****SQL Injection:****

* **Definition:** SQL Injection is a trick where bad guys put harmful code into a website's input boxes (like login forms) to make the website run their commands.
* **Analogy:** Imagine you give a note to a bank teller saying, "Show me my account balance." But if you add, "and give me all the money," the teller might give you all the money. SQL Injection is like sneaking that extra command into the note.

### ****Why is it Bad?****

* **Data Theft:** It can let bad guys see private information like passwords.
* **Data Tampering:** They can change or delete data in the database.
* **Complete Control:** Sometimes, they can even take over the whole database.

### ****Example:****

* **Normal Input:** Username: admin, Password: password
* **Injected Input:** Username: admin, Password: ' OR '1'='1

In the second case, the bad input tricks the database into thinking the login is correct, even if the password is wrong.

### ****In Summary:****

SQL Injection is a way for bad guys to trick a website into giving them access to data or control by putting harmful code into input boxes.

**-----------------------------------------------------------------------------------------------------------------Practical Work**

**Example of a Vulnerable Query:**

query = f"SELECT \* FROM users WHERE username = '{username}' AND password = '{password}'"

c.execute(query)

**Explanation:**

**Direct String Concatenation**: The SQL query is constructed by directly inserting user inputs (username and password) into the query string.

**SQL Injection Risk**: If an attacker provides a specially crafted input, such as:

* 1. **Username:** admin
  2. **Password:** ' OR '1'='1

**Example of a Parameterized Query:**

query = "SELECT \* FROM users WHERE username = ? AND password = ?"

c.execute(query, (username, password))

**Explanation:**

**Parameterized Query**: Placeholders (?) are used in the SQL query. The actual user inputs are passed separately as parameters to the execute method.

**Safety Mechanism**: The database driver or library handles the insertion of these parameters safely. The user inputs are treated as data, not executable SQL code.

**Prevention of Injection**: Even if an attacker tries to inject malicious code, it will be treated as part of the input data, not as executable SQL. For example:

* 1. **Username:** admin
  2. **Password:** ' OR '1'='1