Sncode & database

- By default, Extenso's site are always connected to a specific database.
- Multiple functions in library:
  - sql : general sql function
  - insert : insert in a table
  - update : update tables
  - delete : delete from table
  - lastuid : return the lastuid after an insert
  - quote : quote a string
  - connect : connect to a different database
  - disconnect : disconnect from database

Sncode & database

- By default Extenso is connected to a mariadb database
- SQL command uses normal SQL language
- For example to create a table:

```
sql("create table clients(name varchar (20) not null,
age int not null, address varchar (25) );");


insert(table:"clients",fields:{
   "name": "pierre","age":32,"address":"439"
});
```

Exercice

- Create a widget to use the table
  - select * from clients;
- Build widget with a form to save in this table
- Build a new widget to display the information in this new table

SQL injection

- From Wikipedia:

  "S**QL injection** is a code injection technique used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution (e.g. to dump the database contents to the attacker).[1] SQL injection must exploit a security vulnerability in an application's software, for example, when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and unexpectedly executed. SQL injection is mostly known as an attack vector for websites but can be used to attack any type of SQL database."

Sql Injection example

```
{{
    cgidata = cgidata();
    sql("select * from table where uid = " .+ cgidata.uid);
}}
```

How to fix (example 1)?

```
{{
        cgidata = cgidata();
        sql("select * from table where uid = '?' ", cgidata.uid);
}}
```

Sql injection

Example 2:

```
{{
        cgidata = cgidata();
        uid = esc(filter:"sql", cgidata.uid);
        sql("select * from table where uid = '" .+ uid .+ "');
}}
```

Sql injection

Example 4

Make sure the data you receive is verified

```
{{
        cgidata = cgidata();
        uid = cgidata.uid;
        if name !~ "^[a-z]+$" then error("..."); endif
        if ! isidigit(uid) then error("uid is invalid"); endif
        sql...
}}
```

Extenso : How we build table

- Go to CMS / dev / Database managment / List tables
- Click add to add a new table
- When you generate
  - a new table is created
  - multiple pages are created to
    - add/edit items to the table
    - list data in the table
    - delete data in the table
    - export data in the data
    - import data in the table

Exercice

- Create a new table call test1
  - firstname varchar
  - lastname charchar
  - salary double
- Insert data in this table
- Create a widget to display the data in this table

```
sql("select * from test1");
```

SEDNOVE

Sncode/Extenso

pierre.Laplante@sednove.com

2020-07-17

Cours #7

Course #7

- What we have seen in course 6
  - Create a table with a program
  - Create a table with Extenso
  - Insert data in the tables
  - More on widgets

Widget : form

- We will build a form to save data in a table
- We use form from boostrap: https://getbootstrap.com/docs/4.0/components/forms/
- We will build a form with the following fields:
  - email, password, checkbox
- We will use the sncode command cgidata to get the form data
- We will use the sncode command insert to save the data in a table
- We will display the content of the table

Exercice

- Create a table call my_first_form to save the form
  - email varchar(255)
  - password varchar(255)
  - checkbox enum(yes,no)

Widget : build a form to save data

```
<form method="POST" action="?">
  <div class="form-group">
    <label for="email">Email address</label>
    <input type="email" class="form-control" name="email" aria-describedby="emailHelp" placeholder="Enter email">
    <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>
  </div>
  <div class="form-group">
    <label for="password">Password</label>
    <input type="password" class="form-control" name="password" placeholder="Password">
  </div>
  <div class="form-check">
    <input type="checkbox" class="form-check-input" name="checkbox" value="yes">
    <label class="form-check-label" for="checkbox">Check me out</label>
  </div>
  <button type="submit" class="btn btn-primary">Submit</button>
</form>
```

Sncode : cgidata

- cgidata read the data from the form and return it in json

```
\{{
     cgidata = cgidata();
     "cgidata="; cgidata;
}}
```

which will return

{"email":"laplante@sednove.ca","password":"helloWorld"}

Sncode : insert

- Function insert will be used to save data from cgidata using

```
\{{
cgidata = cgidata();
if cgidata.email ne "" then
      insert(table:"my_first_form",snc:true,fields:cgidata);
endif

res = sql("select * from my_first_form");
res;

}}
```

Widget : Display information in the table

- We use boostrap table:
  - https://getbootstrap.com/docs/4.0/content/tables/
- We use a Sncode sql to get all the informations from the table:

```
\{{
        res = sql("select * from my_first_form");
        res;
}}
```

- We use a for loop to display the information
  - \{{ for row in res.rows do }}
  - \{{ endfor }}

Widget : display data

```
<h2>Data</h2>
\{{ res = sql("select * from my_first_form"); }}
<div id="table">
  <table class="table">
    <thead>
      <tr>
        <th scope="col">Email</th>
        <th scope="col">Password</th>
        <th scope="col">checkbox</th>
      </tr>
    </thead>
    <tbody>
      \{{ for row in res.rows do }}
      <tr>
        <td>\{{ row.email }}</td>
        <td>\{{ row.password }}</td>
        <td>\{{ row.checkbox }}</td>
      </tr>
      \{{ endfor }}
    </tbody>
  </table>
</div>
```

Exercice : try to reproduce this widget