

UNIVERSIDAD NACIONAL DE LANÚS



DEPARTAMENTO DE DESARROLLO
PRODUCTIVO Y TECNOLÓGICO

LICENCIATURA EN SISTEMAS

Materia: Sistemas Distribuidos

Actividad Práctica N°1

Common Gateway Interface (CGI)

Docentes

Ing Diego Andrés Azcurra

Lic. Marcos Amaro

Alumno:

Pizarro Maximiliano

DNI 36.771.843

1. Configuración del entorno.
2. Desarrollo.
3. Conclusión.

Configuración del Entorno

Para el desarrollo de la actividad se utilizaron las siguientes tecnologías todos instalados y configurados en Sistema Operativo Windows 10

Componente	Descarga
Python 3.8	https://www.python.org/downloads/
Apache/2.4.43 (Win64)	https://www.apachefriends.org/es/download.html
MySQL shell 8.0*	https://dev.mysql.com/downloads/shell/
MySQL Python Connector	https://dev.mysql.com/doc/connector-python/en/connector-python-installation.html
Docker Desktop*	https://www.docker.com/products/docker-desktop
Visual Studio Code*	https://code.visualstudio.com/download

*Opcionales

Python 3.8

Instalar distribución de Python en el directorio

C:\Python38

Variables de entorno en PATH

C:\Python38

C:\Python38\Scripts

La versión ya trae incorporada la librería cgi.

<https://docs.python.org/3/library/cgi.html#>

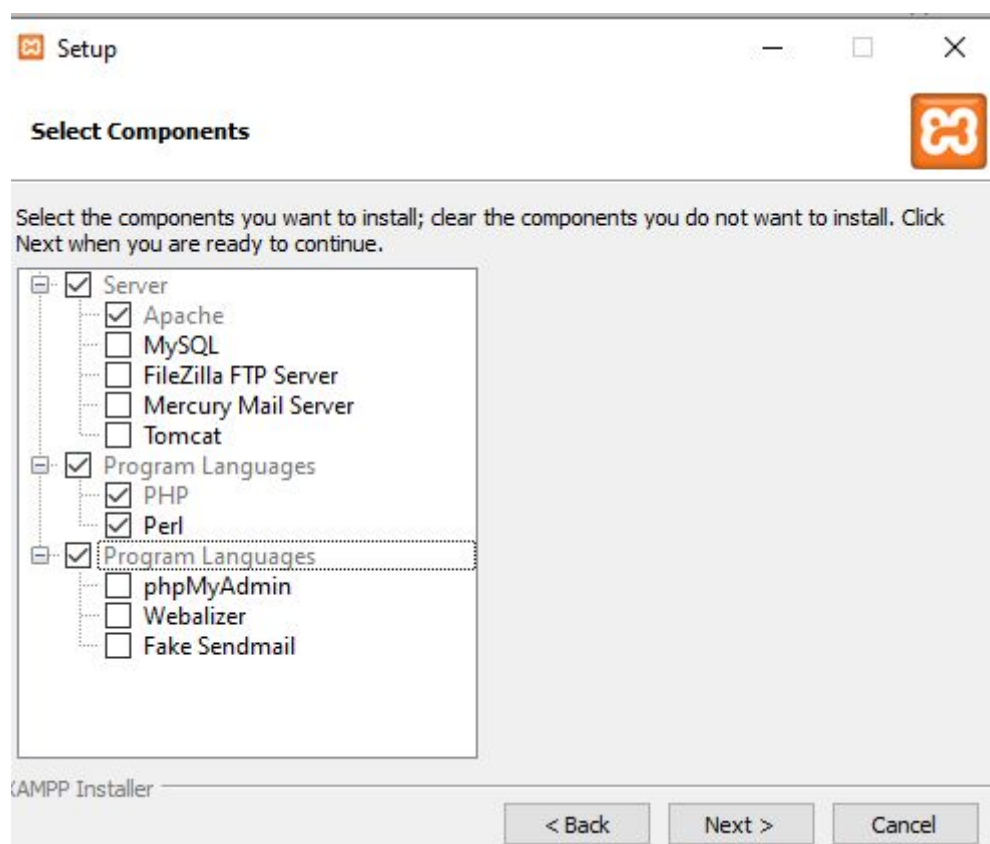
MySQL Python Connector

Instalar package desde el shell:

```
pip install mysql-connector-python
```

Apache/2.4.43 (Win64)

Descargar instalador y seleccionar el componente de apache



Variables de entorno en PATH

En el caso del instalador XAMPP ya viene configurado con los Document Root para los contenedores CGI y HTTP en los siguientes directorios

<path-instalación-apache>/apache/cgi-bin (CGI)
<path-instalación-apache>/apache/htdocs (HTML)

Verificar directorios en el archivo httpd.conf

<path-instalación-apache>/apache/apache/conf

Contenedor HTML

```
DocumentRoot "C:/Users/Max/Documents/apache/htdocs"  
<Directory "C:/Users/Max/Documents/apache/htdocs">
```

Contenedor CGI

```
<Directory "C:/Users/Max/Documents/apache/cgi-bin">  
    AllowOverride All  
    Options None  
    Require all granted  
</Directory>
```

Iniciar servidor apache

XAMPP Control Panel v3.2.4 [Compiled: Jun 5th 2019]

Service	Module	PID(s)	Port(s)	Actions
<input checked="" type="checkbox"/>	Apache	14732 4156	80, 443	<input type="button" value="Stop"/> Admin Config Logs
<input type="checkbox"/>	MySQL			<input type="button" value="Start"/> Admin Config Logs
<input type="checkbox"/>	FileZilla			<input type="button" value="Start"/> Admin Config Logs
<input type="checkbox"/>	Mercury			<input type="button" value="Start"/> Admin Config Logs
<input type="checkbox"/>	Tomcat			<input type="button" value="Start"/> Admin Config Logs

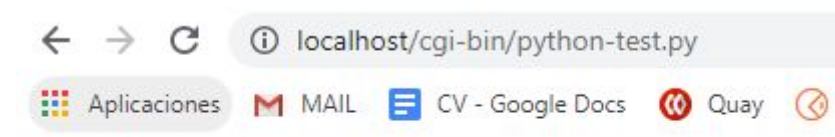
7:30:49 AM [main] The MySQL module is disabled
7:30:49 AM [main] The FileZilla module is disabled
7:30:49 AM [main] The Mercury module is disabled
7:30:49 AM [main] The Tomcat module is disabled
7:30:49 AM [main] Starting Check-Timer
7:30:49 AM [main] Control Panel Ready
7:31:32 AM [Apache] Attempting to start Apache app...
7:31:33 AM [Apache] Status change detected: running

Crear un cgi de prueba python-test.py en el contenedor CGI

```
#!/C:/Python38/python.exe"
import cgi
cgi.enable()

print ("Content-type: text/html\n\n");
print("<html>");
print("<head>");
print("<meta name='author' content='Maximiliano Pizarro'>");
print("<link href='/xampp/xampp.css' rel='stylesheet'
type='text/css'>");
print("</head>");
print("<body>&nbsp;<p><h1>GCI con Python</h1>");
print ("CGI con Python esta listo ...</body></html>");
```

Ingresar desde el browser a <http://localhost/cgi-bin/pyhon-test.py>



GCI con Python

CGI con Python esta listo ...

MySQL shell 8.0

Este componente es opcional y se utilizó para explorar la base de datos sakila, probar y testear la conexión, creación y depuración store procedure.

Descargar la distribución standalone

<https://dev.mysql.com/doc/mysql-shell/8.0/en/>

Variables de entorno en PATH:

<path-mysqlsh-standalone>/bin

Verificamos la instalación ejecutando desde el shell:

mysqlsh

Conectar con base de datos

mysqlsh c --mysql mysql://root:root@127.0.0.1/sakila

Ejecutar Script SQL

Varía dependiendo el intérprete que usemos, por defecto viene configurado con javascript pero se puede cambiar a sql o python al iniciar la conexión sumando el argumento --py o --sql. La sentencia en JavaScript es desde el contexto donde se encuentra el archivo.sql

\source archivo-store-procedure.sql

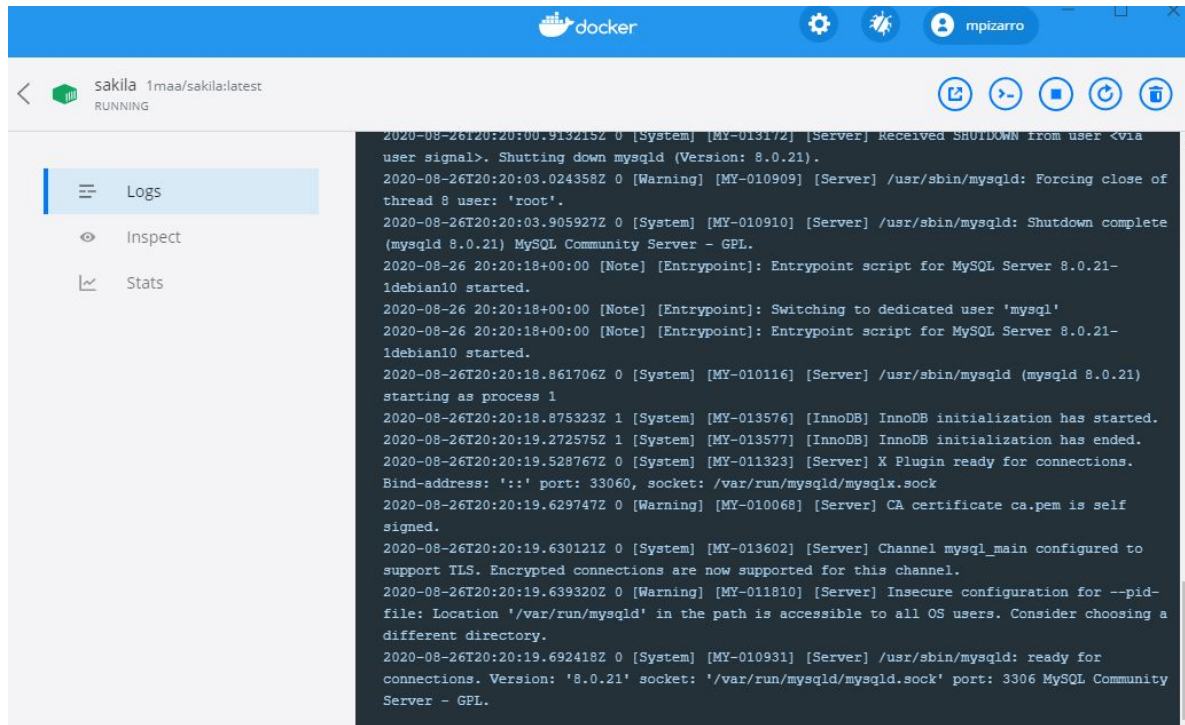
<https://dev.mysql.com/doc/mysql-shell/8.0/en/mysql-shell-commands.html>

Docker Desktop

Este complemento es opcional y se utilizó para administrar la imagen del motor de base de datos. Se combinó con las pruebas de conexión de mysqlsh. Se ejecutaron pull de las siguientes imágenes:

https://hub.docker.com/_/mysql

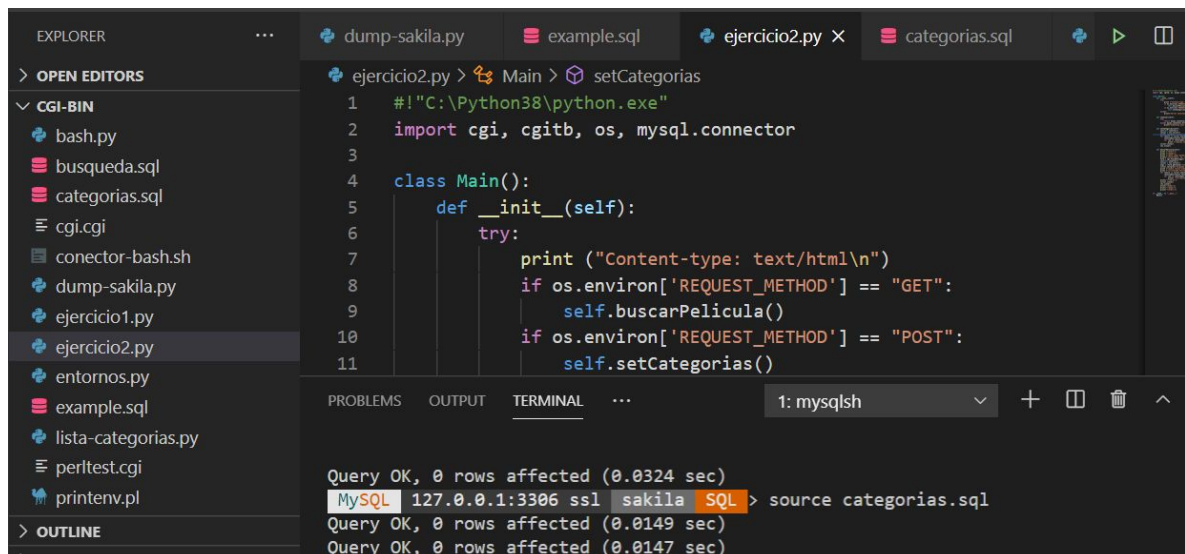
<https://hub.docker.com/r/1maa/sakila>



Logs sakila

Visual Studio Code

Este complemento es opcional y se utilizó para administrar los archivos de la actividad



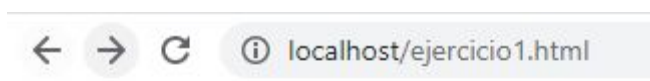
Desarrollo

Se extraen los ejercicios del enunciado de la Actividad Práctica presentada por el equipo Docente, se presentan capturas ejemplos de request y response de la solución, y se agrega el código fuente de cada ejercicio. Se omitieron los comentarios dentro del código fuente con el fin de utilizar el presente documento como espacio de comentarios y esquemas.

Ejercicio 1

1) Elegir 10 variables de entorno Crear un formulario con las 10 variables de entorno seleccionadas con un checkbox para cada una. Según las variables que el usuario tilde, enviar la petición al servidor y que devuelva una página web con una tabla como la siguiente: VARIABLE DE ENTORNO VALOR VARIABLE 1 VALOR 1 VARIABLE 2 VALOR 2 VARIABLE N VALOR N

Request checkbox completo:



Variables de Entorno

- ☒ SERVER_NAME
- ☒ SERVER_PORT
- ☒ REMOTE_PORT
- ☒ SERVER_SOFTWARE
- ☒ SERVER_PROTOCOL
- ☒ GATEWAY_INTERFACE
- ☒ REQUEST_METHOD
- ☒ REQUEST_URI
- ☒ GATEWAY_INTERFACE
- ☒ PATH

Enviar

Response CGI:

← → ↻ ⓘ localhost/cgi-bin/ejercicio1.py?SERVER_NAME=SERVER_NAME&SERVER_PORT=SERVER_PORT&REMOTE_PORT=REMOTE_PORT&SERVER_SOFTWARE=SERVER_SOFTWARE...

Variable	Valor
GATEWAY_INTERFACE	CGI/1.1
PATH	C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\Python38\Scripts;C:\Python38;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\Files\Intel\WirelessCommon;C:\Program Files\RedHat\java-1.8.0-openjdk-1.8.0.262-3\bin;C:\Program Files\RedHat\java-1.8.0-openjdk-1.8.0.262-3\jre\bin;C:\Code\bin;C:\Users\Max\AppData\Roaming\npm;C:\Program Files\RedHat\java-11-openjdk-11.0.8-2\bin;C:\Program Files\apache-maven-3.6.3\bin;C:\Pr
SERVER_NAME	localhost
SERVER_PORT	80
SERVER_PROTOCOL	HTTP/1.1
SERVER_SOFTWARE	Apache/2.4.43 (Win64) OpenSSL/1.1.1g PHP/7.4.8
REQUEST_METHOD	GET
REQUEST_URI	/cgi-bin/ejercicio1.py? SERVER_NAME=SERVER_NAME&SERVER_PORT=SERVER_PORT&REMOTE_PORT=REMOTE_PORT&SERVER_SOFTWARE=SERVER_SO
REMOTE_PORT	49186

Request checkbox parcialmente completo

← → ↻ ⓘ localhost/ejercicio1.html

Variables de Entorno

- ☒ SERVER_NAME
- ☐ SERVER_PORT
- ☐ REMOTE_PORT
- ☐ SERVER_SOFTWARE
- ☒ SERVER_PROTOCOL
- ☒ GATEWAY_INTERFACE
- ☒ REQUEST_METHOD
- ☒ REQUEST_URI
- ☐ GATEWAY_INTERFACE
- ☐ PATH

Enviar

Response CGI

← → ↻ ⓘ localhost/cgi-bin/ejercicio1.py?SERVER_NAME=SERVER_NAME&SERVER_PROTOCOL=SERVER_PROTOCOL&GATEWAY_INTERFACE=GATEWAY_INTERFACE&REQUEST_M...

Variable	Valor
GATEWAY_INTERFACE	CGI/1.1
REQUEST_METHOD	GET
REQUEST_URI	/cgi-bin/ejercicio1.py? SERVER_NAME=SERVER_NAME&SERVER_PROTOCOL=SERVER_PROTOCOL&GATEWAY_INTERFACE=GATEWAY_INTERFACE&I
SERVER_PROTOCOL	HTTP/1.1
SERVER_NAME	localhost

Código Fuente

La solución cuenta con un archivo html para el formulario y un cgi. La clase FieldStorage provee todas las claves que se envían desde un formulario y la primitiva Keys() retorna una lista completa con las claves. Se importa adicionalmente OS para la impresión del valor de la variable de entorno.

ejercicio1.html

```
<!DOCTYPE html><html><body><h2>Variables de Entorno</h2>
<form action="/cgi-bin/ejercicio1.py" method="GET">
  <input type="checkbox" id="variable1" name="SERVER_NAME" value="SERVER_NAME">
  <label for="SERVER_NAME">SERVER_NAME</label><br>
  <input type="checkbox" id="variable2" name="SERVER_PORT" value="SERVER_PORT">
  <label for="SERVER_PORT">SERVER_PORT</label><br>
  <input type="checkbox" id="variable3" name="REMOTE_PORT" value="REMOTE_PORT">
  <label for="REMOTE_PORT">REMOTE_PORT</label><br>
  <input type="checkbox" id="variable4" name="SERVER_SOFTWARE"
value="SERVER_SOFTWARE">
  <label for="SERVER_SOFTWARE">SERVER_SOFTWARE</label><br>
  <input type="checkbox" id="variable5" name="SERVER_PROTOCOL"
value="SERVER_PROTOCOL">
  <label for="SERVER_PROTOCOL">SERVER_PROTOCOL</label><br>
  <input type="checkbox" id="variable6" name="GATEWAY_INTERFACE"
value="GATEWAY_INTERFACE">
  <label for="GATEWAY_INTERFACE">GATEWAY_INTERFACE</label><br>
```

```

<input type="checkbox" id="variable7" name="REQUEST_METHOD"
value="REQUEST_METHOD">

<label for="REQUEST_METHOD">REQUEST_METHOD</label><br>

<input type="checkbox" id="variable8" name="REQUEST_URI" value="REQUEST_URI">

<label for="REQUEST_URI">REQUEST_URI</label><br>

<input type="checkbox" id="variable9" name="GATEWAY_INTERFACE"
value="GATEWAY_INTERFACE">

<label for="GATEWAY_INTERFACE">GATEWAY_INTERFACE</label><br>

<input type="checkbox" id="variable10" name="PATH" value="PATH">

<label for="PATH">PATH</label><br>

<input type="submit" value="Enviar">

</form> </body></html>

```

ejercicio1.py

```

#!/"C:\Python38\python.exe"

import cgi, cgitb, os

print ("Content-type: text/html\n\n");

print ("<html>");

print ("<head>");

print ("<meta name='author' content='Maximiliano Pizarro'>");

print("<style>table, th, td { border: 1px solid black;
border-collapse: collapse;}th, td { padding: 15px; text-align:
left;}#t01 { width: 100%; background-color: #f1f1c1;}</style>")

print("<body>");

try:

    form = cgi.FieldStorage()

    print ("<table style='width:100%'>");

    print ("<tr><th>Variable</th><th>Valor</th></tr>");

    for param in form.keys():

        print( "<tr><td><b>%20s</b></td> <td>%s</td></tr>" %
(param, os.environ[param]))

    print("</table>");

except:

```

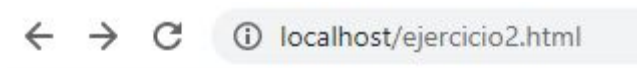
```
print("Server Internal Error");  
  
print("</body>");  
  
print("</html>");
```

Ejercicio 2

Utilizando la base de datos Sakila, disponible en <https://downloads.mysql.com/docs/sakila-db.zip>, desarrollar las siguientes pantallas:

a) Búsqueda de películas: las cuales se pueden filtrar por: título, año, categoría. El filtro de categoría será una lista desplegable con las categorías disponibles de la tabla category. Los filtros de título y año son de texto opcionales.

Request :



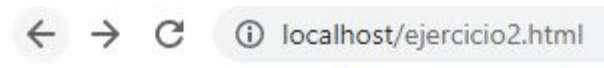
Búsqueda de películas

Título

Año

Categoría
Action ▼
Action
Animation
Children
Classics
Comedy
Documentary
Drama
Family
Foreign
Games
Horror
Music
New
Sci-Fi
Sports
Travel

Request con datos:



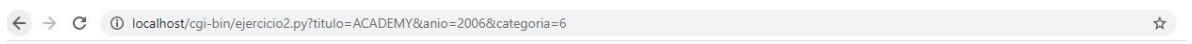
Búsqueda de películas

Título

Año

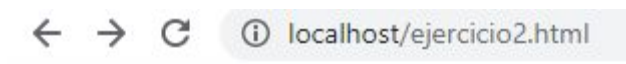
Categoría

Response CGI



ID	TÍTULO	AÑO	ELENCO
1	ACADEMY DINOSAUR	2006	CHRISTIAN GABLE
1	ACADEMY DINOSAUR	2006	CHRISTIAN None
1	ACADEMY DINOSAUR	2006	JOHNNY CAGE

Otro ejemplo de búsqueda:



Búsqueda de películas

Título

Año

Categoría

localhost/cgi-bin/ejercicio2.py?titulo=ACE+GOLDFINGER&anio=2006&categoria=1			
ID	TÍTULO	AÑO	ELENCO
2	ACE GOLDFINGER	2006	BOB FAWCETT
2	ACE GOLDFINGER	2006	BOB None
2	ACE GOLDFINGER	2006	CHRIS DEPP

Código Fuente

La solución cuenta con un archivo html para el formulario y un cgi. El archivo cgi cuenta con una Clase Main con las primitivas y su función

- **__init__(self)** para bifurcar los métodos request de acceso **GET**, **POST** y **PUT**
- **conectar(self)** para establecer la conexión con el motor de base de datos
- **setCategorias(self)** para generar generar contenido html de las categorías del componente select, será invocado por POST desde html por medio de la clase XMLHttpRequest() de JavaScript
- **buscarPelicula(self)**

ejercicio2.html

```
<!DOCTYPE html>

<html><head><meta charset="UTF-8"></head>

<body onload="setCategorias()">

<h2>Búsqueda de películas</h2>

<form action="/cgi-bin/ejercicio2.py" method="GET">

    <label for="titulo">Título</label><br>

    <input type="text" id="titulo" name="titulo"><br>

    <label for="anio">Año</label><br>

    <input type="text" id="anio" name="anio"><br>

    <label for="categoria">Categoría</label><br>

    <select id="categoria" name="categoria"> </select><br><br>

    <input type="submit" value="Enviar">

</form> </body>
```

```

<script>

    function setCategorias() {

        console.log("entro")

        var xhttp = new XMLHttpRequest();

        xhttp.open("POST", "/cgi-bin/ejercicio2.py", false);

        xhttp.send();

        var ele =
document.getElementById("categoria").innerHTML=xhttp.responseText;

    }

</script>

</html>

```

ejercicio2.py

```

#!/"C:\Python38\python.exe"

import cgi, cgiib, os, mysql.connector

class Main():

    def __init__(self):

        try:

            print ("Content-type: text/html\n")

            if os.environ['REQUEST_METHOD'] == "GET":

                self.buscarPelicula()

            if os.environ['REQUEST_METHOD'] == "POST":

                self.setCategorias()

        except:

            print("Server Internal Error : request method not
allowed");

    def conectar(self):

        try:

            return mysql.connector.connect(user='root',
password='root',host='127.0.0.1',database='sakila')

        except mysql.connector.Error as err:

```



```

        print("Something went wrong: {}".format(err))

def setCategorias(self):
    cnx=self.conectar()

    cursor = cnx.cursor()

    cursor.callproc('categorias', args=())

    for result in cursor.stored_results():
        resultados=result.fetchall()

        for row in resultados:

            print("<option
value='%d'%s</option>"%(row[0],row[1]))

    cursor.close()

    cnx.close()

def buscarPelicula(self):
    print("<html>");
    print("<head>");
    print("<meta name='author' content='Maximiliano
Pizarro'>");

    print("<style>table, th, td { border: 1px solid black;
border-collapse: collapse;}th, td { padding: 15px; text-align:
left;}#t01 { width: 100%; background-color: #f1f1c1;}</style>")

    form = cgi.FieldStorage()

    cnx=self.conectar()

    cursor = cnx.cursor()

    args =
(form.getvalue('titulo'),form.getvalue('anio'),form.getvalue('categ
oria'))

    cursor.callproc('busqueda', args)

    print("<table style='width:100%'>");

    print
("<tr><th>ID</th><th>TÍTULO</th><th>AÑO</th><th>ELENCO</th></tr>");

    for result in cursor.stored_results():
        resultados=result.fetchall()

```

```

        for row in resultados:

            print( "<tr><td><b>%20s</b></td>
<td>%s</td><td>%s</td><td>%s %s</td></tr>"
%(row[0],row[1],row[2],row[3],row[4]))

        print("</table>");

        cursor.close()

        cnx.close()

        print("<body>");

        print("</body>");

        print("</html>");

if __name__ == "__main__":

    Main()

```

categorias.sql

```

DROP PROCEDURE IF EXISTS `categorias`;

DELIMITER $$

CREATE DEFINER='root'@'localhost' PROCEDURE `categorias` ()
BEGIN

    select category_id,name from category;

END $$

DELIMITER ;

```

busqueda.sql

```

DROP PROCEDURE IF EXISTS `busqueda`;

DELIMITER $$

CREATE DEFINER='root'@'localhost' PROCEDURE `busqueda` (
    IN titulo varchar(128), IN anio year, IN categoria int
)
BEGIN

    select film.film_id,

        film.title,

        film.release_year ,

```

```

        actor.first_name,actor.last_name
    from film, actor,film_actor, film_category
    where film_actor.film_id=film.film_id
    and film_actor.actor_id=actor.actor_id
    and anio like film.release_year
    and film.title LIKE CONCAT('%', titulo , '%')
    and categoria=film_category.category_id
    group by  film.film_id,
        film.title,
        film.release_year,actor.first_name,actor.last_name
WITH ROLLUP;

END $$

DELIMITER ;

```

a) Alta de actor: además de los campos necesarios para insertar en la tabla actor, se debe especificar un año. Este último dato se utilizará para asociar al actor a todas las películas que se encuentren para ese año.