UNIVERSIDAD NACIONAL DE LANÚS


DEPARTAMENTO DE DESARROLLO PRODUCTIVO Y TECNOLÓGICO

LICENCIATURA EN SISTEMAS

Materia: Sistemas Distribuidos

# Actividad Práctica N°2

Open DataBase Connectivity (ODBC)

**Docentes**

Ing Diego Andrés Azcurra

Lic. Marcos Amaro

**Alumno:**

Pizarro Maximiliano

DNI 36.771.843

# ÍNDICE

## 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 |
|---|---|
| Node 12.18.3 | https://nodejs.org/en/download/ |
| MySQL Connector/ODBC 8.0.21 | https://dev.mysql.com/downloads/connector/odbc/ |
| PM2 | https://pm2.keymetrics.io/ |
| MySQL shell 8.0* | https://dev.mysql.com/downloads/shell/ |
| Bootstrap 4.5.2 | https://getbootstrap.com/ |
| Docker Desktop* | https://www.docker.com/products/docker-desktop |
| Visual Studio Code* | https://code.visualstudio.com/download |

*Opcionales

### Node 14.9.0

Instalar distribución de Node con el instalador

Variables de entorno en PATH

*C:\Users\Max\AppData\Roaming\npm*

Variables de la solución

```
process.env.PORT
process.env.DATASOURCE
```

por defecto serán invocadas con el puerto 8080 y el datasource de conexión odbc con el nombre MySQL

Instalar componentes

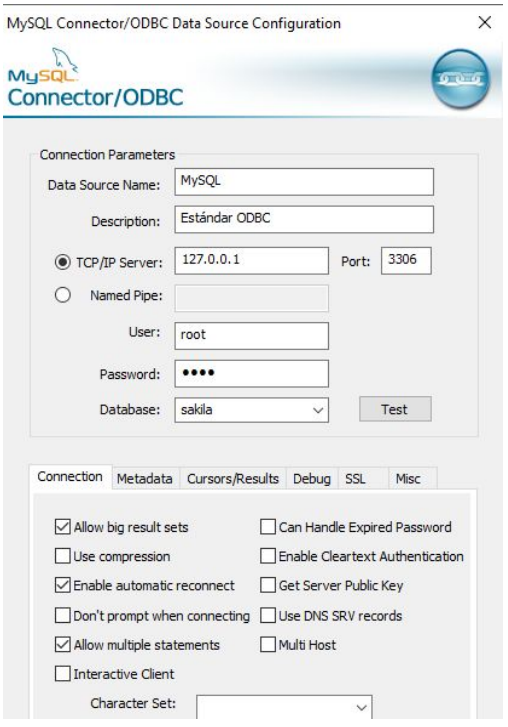
- clonar repositorio https://github.com/maximilianoPizarro/estandar-odbc
- ejecutar npm install

```
{
    "name": "estandar-odbc",
    "version": "0.0.1",
    "private": true,
    "main": "server.js",
    "scripts": {
        "start": "node server.js"
    },
    "dependencies": {
        "body-parser": "^1.19.0",
        "express": "^4.17.1",
        "odbc": "^2.3.4",
        "path": "^0.12.7"
    },
    "devDependencies": {
        "nodemon": "^1.18.11"
    }
}
```

Package.json

## MySQL Connector/ODBC 8.0.21

Instalar distribución y crear conexión agregando el driver MySQL 8.0 ANSI Driver




*Configuración DATASOURCE*

## PM2

Luego de instalar Node ejecutar **npm install pm2 -g** para la instalación, dentro del contexto del proyecto iniciar servidor con **pm2 start server**.
Por defecto la aplicación corre en http://localhost:8080

```
PS C:\Users\Max\Documents\UNLA\Sistemas Distribuidos\TP2\estandar-odbc> pm2 start server
[PM2] Applying action restartProcessId on app [server](ids: [ 0 ])
[PM2] [server](0) ✓
[PM2] Process successfully started
```

| id | name | mode | ↺ | status | cpu | memory |
|---|---|---|---|---|---|---|
| 0 | server | fork | 0 | online | 0% | 27.7mb |

iniciar servidor

```
PS C:\Users\Max\Documents\UNLA\Sistemas Distribuidos\TP2\estandar-odbc> pm2 stop server
[PM2] Applying action stopProcessId on app [server](ids: [ 0 ])
[PM2] [server](0) ✓
```

| id | name | mode | ↺ | status | cpu | memory |
|---|---|---|---|---|---|---|
| 0 | server | fork | 0 | stopped | 0% | 0b |

detener servidor

```
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /ciudades
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /clientes
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /paises
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /tiendas
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /ciudadesByI
d 1
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /ciudades
0|server   | Listening on port 8080
```

pm2 logs

## 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:

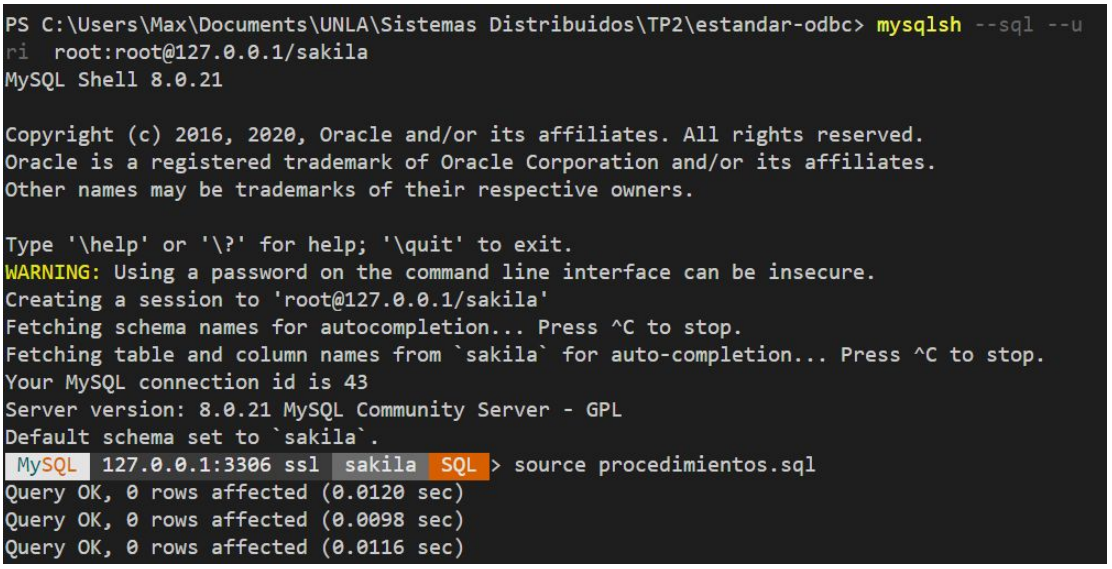
*<path-mysqlsh-standalone>**/bin***

Verificamos la instalación ejecutando desde el shell:

mysqlsh

Conectar con base de datos

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




Ejecución de procedimientos almacenados de la solución

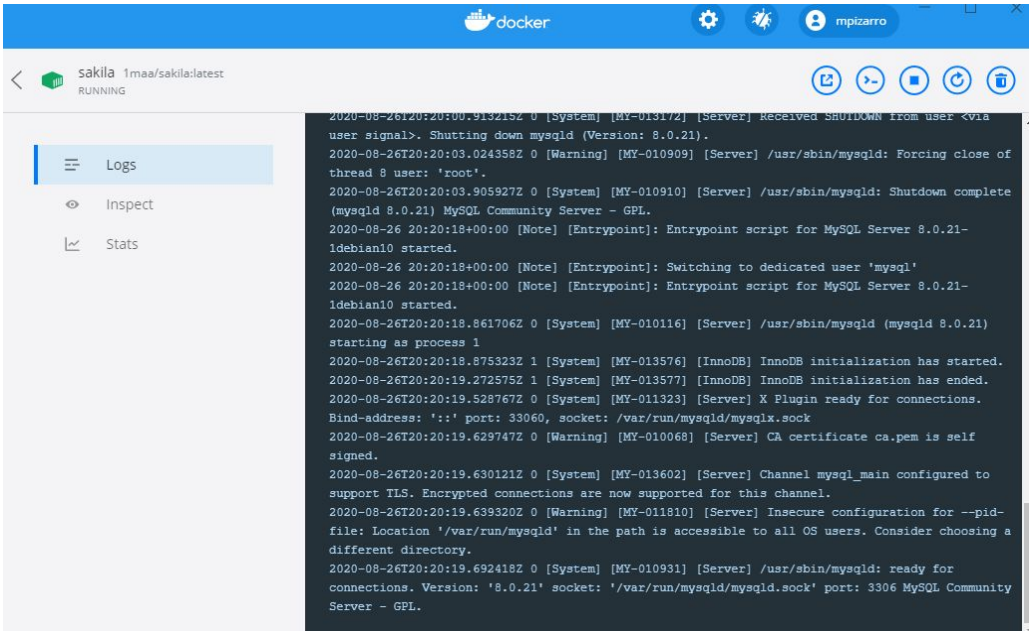
**Bootstrap 4.5.2**

Este complemento se implementó para el material de diseño de los componentes de la UI, se encuentran importados en el archivo index.html

**Docker Desktop**

Este complemento es opcional y se utilizó para administrar la imagen del motor de base de datos.
https://hub.docker.com/_/mysql

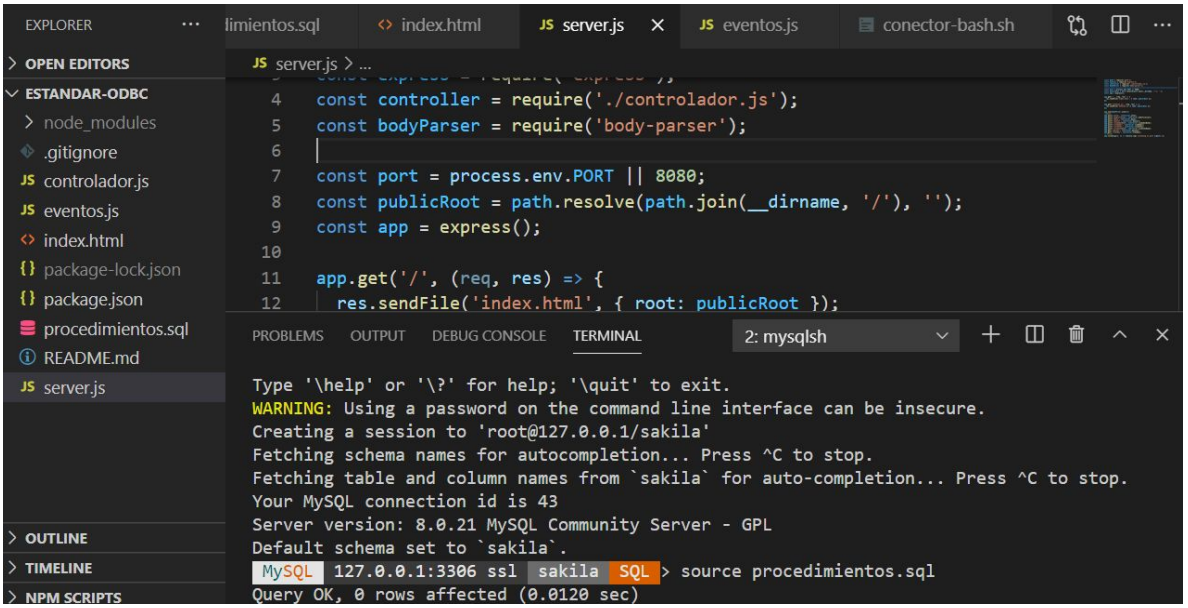
https://hub.docker.com/r/1maa/sakila




Logs sakila

## Visual Studio Code

Este complemento es opcional y se utilizó como ide para el desarrollo de la actividad



# Desarrollo

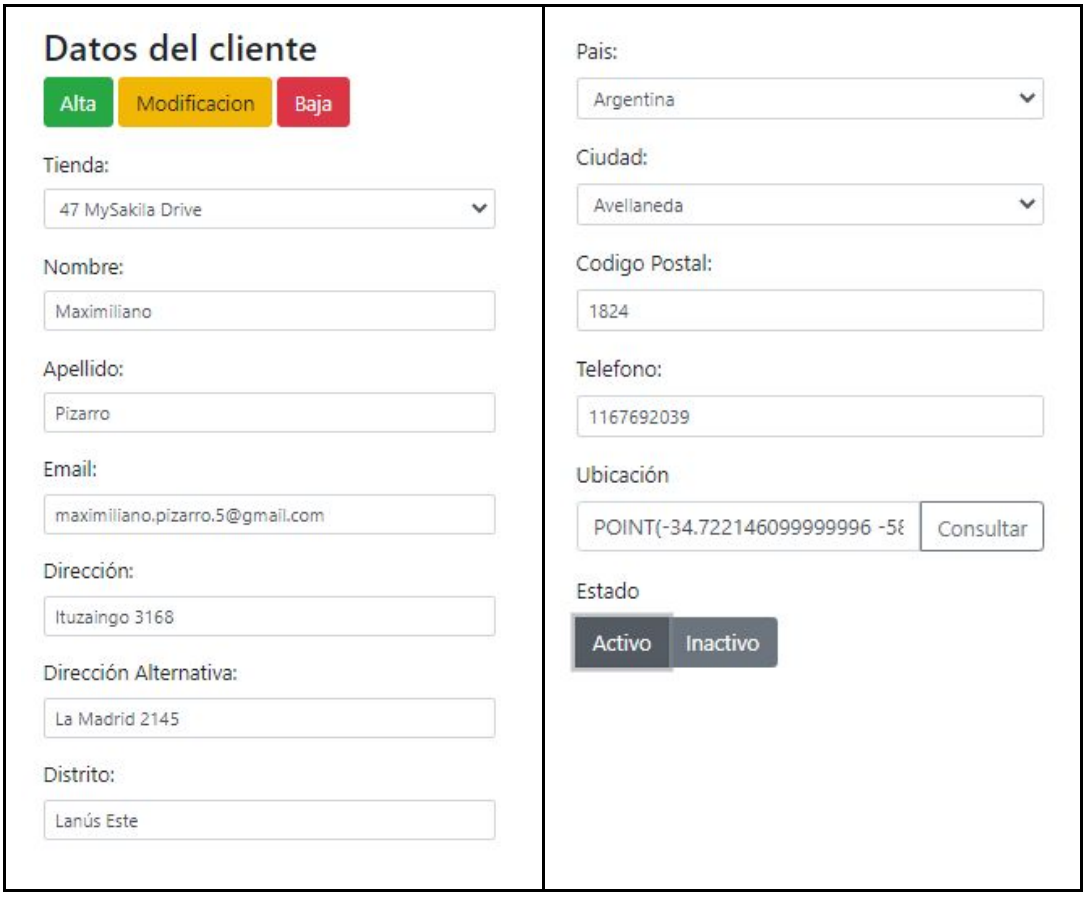
Se extraen los ejercicios del enunciado de la Actividad Práctica presentada por el equipo Docente, se presentan capturas de la solución, y se agrega el código fuente de cada ejercicio con la definición del servicio que se consultó y el procedimiento almacenado que se invocó.

## Ejercicio 1

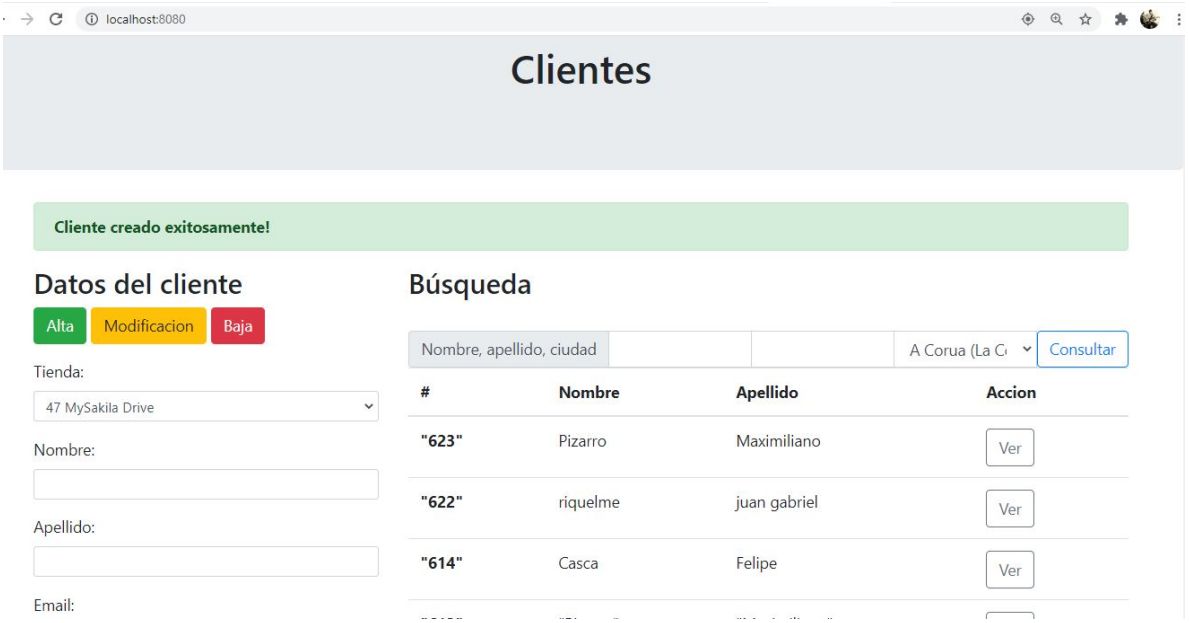
1) Utilizando la base de datos Sakila, desarrollar las pantallas necesarias para gestionar los clientes

(tabla CUSTOMER):

A. Insertar un nuevo cliente
B. Modificar los datos de un cliente existente
C. Baja de clientes




Formulario de Alta




alta cliente

## Código Fuente

Para la transacción del alta se precargan los selectores de tiendas, países, ciudades según país con los consumen los siguientes servicios del servidor

```
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /ciudades
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /clientes
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /paises
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /tiendas
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /ciudadesByI
d 1
0|server   | Sat Sep 12 2020 21:22:45 GMT-0300 (hora estándar de Argentina): /ciudades
0|server   | Listening on port 8080
0|server   | Sat Sep 12 2020 21:58:45 GMT-0300 (hora estándar de Argentina): /ciudadesByI
d 6
0|server   | Sat Sep 12 2020 22:05:51 GMT-0300 (hora estándar de Argentina): /alta
0|server   | {"customer_id":0,"store_id":2,"last_name":"Maximiliano","first_name":"Pizarr
o","email":"maximiliano.pizarro.5@gmail.com","address":"Ituzaingo 3168","address2":"La Ma
drid 2145","district":"Lanús Este","country_id":6,"city_id":43,"postal_code":"1824","phon
e":"1167692039","active":"1","location":"POINT(-34.722146099999996 -58.3679088)","address
_id":null}
```

Del lado del Cliente se cargan los documentos index.html y el script eventos.js que se encarga de enviar y recibir las request/response del servidor, eventos se encarga de serializar y deserializar el formulario, también de capturar la ubicación requerida en el campo location.

**eventos.js**

```
function inicio(){

    blanquearForm()

    clientes()

    paises()
```

```
    tiendas()
    ciudadesById()
    ciudades()
}

function tiendas() {
    var out = "";
    var xhttp = new XMLHttpRequest();
    xhttp.open("GET", "/tiendas", false);
    xhttp.setRequestHeader("Content-Type", "application/json");
    xhttp.send();
    var data = JSON.parse(xhttp.responseText)
    for (let i = 0; i < data.length; i++) {
        out += '<option value="' + data[i].store_id + '">' + data[i].address +
'</option>';
        }
    document.getElementById('tiendas').innerHTML=out;
}

function paises() {
    var out = "";
    var xhttp = new XMLHttpRequest();
    xhttp.open("GET", "/paises", false);
    xhttp.setRequestHeader("Content-Type", "application/json");
    xhttp.send();
    var data = JSON.parse(xhttp.responseText)
    for (let i = 0; i < data.length; i++) {
        out += '<option value="' + data[i].country_id + '">' + data[i].country +
'</option>';
        }
    document.getElementById('paises').innerHTML=out;
}

function ciudadesById() {
    var out = "";
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/ciudadesById", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
```

```
xhttp.send(JSON.stringify({"country_id":document.getElementById('paises').value})
);
    var data = JSON.parse(xhttp.responseText)
    for (let i = 0; i < data.length; i++) {
        out += '<option value="' + data[i].city_id + '">' + data[i].city +
'</option>';
        }
    document.getElementById('ciudades').innerHTML=out;
}
function ciudades() {
    var out = "";
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/ciudades", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
    xhttp.send();
    var data = JSON.parse(xhttp.responseText)
    for (let i = 0; i < data.length; i++) {
        out += '<option value="' + data[i].city_id + '">' + data[i].city +
'</option>';
        }
    document.getElementById('sciudades').innerHTML=out;
}


function clientes() {
    var out = "";
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/clientes", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
    xhttp.send();
    var data = JSON.parse(xhttp.responseText)
    for (let i = 0; i < data.length; i++) {
        out += '<tr><th scope="row">"' + data[i].customer_id + '"</th><td>' +
data[i].first_name +'</td><td>'+ data[i].last_name +'</td><td><button class="btn
btn-outline-secondary" type="button" id="button-addon2"  onclick="verCliente(' +
data[i].customer_id + ')" >Ver</button></td></tr>';
        }
    document.getElementById('clientes-body').innerHTML=out;
}
```

```
function getLocation() {
if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(showPosition);
} else {
    document.getElementById("location").innerHTML = "Geolocation is not supported
by this browser.";
}
}
function showPosition(position) {
document.getElementById("location").value = "POINT(" + position.coords.latitude +
" " + position.coords.longitude+")";
}
function alta() {
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/alta", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
    console.log(JSON.stringify(serialize()))
    xhttp.send(JSON.stringify(serialize()));
    document.getElementById("mensaje").style.display = "block";
    var ele = document.getElementById("mensaje").innerHTML=xhttp.responseText;
    inicio()
}
function modificacion() {
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/modificacion", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
    console.log(JSON.stringify(serialize()))
    xhttp.send(JSON.stringify(serialize()));
    document.getElementById("mensaje").style.display = "block";
    var ele = document.getElementById("mensaje").innerHTML=xhttp.responseText;
    inicio()
}
function baja() {
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/baja", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
```

```
    console.log(JSON.stringify(serialize()))
    xhttp.send(JSON.stringify(serialize()));
    document.getElementById("mensaje").style.display = "block";
    var ele = document.getElementById("mensaje").innerHTML=xhttp.responseText;
    inicio()
}
function serialize(){
    return {
        "customer_id":
        parseInt(document.getElementById('customer_id').value),
        "store_id":
        parseInt(document.getElementById('tiendas').value),
        "last_name":
        document.getElementById('last_name').value,
        "first_name":
        document.getElementById('first_name').value,
        "email":
        document.getElementById('email').value,
        "address":
        document.getElementById('address').value,
        "address2":
        document.getElementById('address2').value,
        "district":
        document.getElementById('district').value,
        "country_id":
        parseInt(document.getElementById('paises').value),
        "city_id":
        parseInt(document.getElementById('ciudades').value),
        "postal_code":
        document.getElementById('postal_code').value,
        "phone":
        document.getElementById('phone').value,
        "active":
        document.getElementById('active').value,
        "location":
        document.getElementById('location').value,
        "address_id":
```

```
        parseInt(document.getElementById('address_id').value)
    };
}
function deserialize(json){
    document.getElementById('customer_id').value=parseInt(json.customer_id);
    document.getElementById('tiendas').value=parseInt(json.store_id);
    document.getElementById('last_name').value=json.last_name;
    document.getElementById('first_name').value=json.first_name;
    document.getElementById('email').value=json.email;
    document.getElementById('address_id').value=json.address_id;
    document.getElementById('address').value=json.address;
    document.getElementById('address2').value=json.address2;
    document.getElementById('district').value=json.district;
    document.getElementById('paises').selectedIndex=json.country_id-1;
    ciudadesById();
    document.getElementById('postal_code').value=json.postal_code;
    document.getElementById('phone').value=json.phone;
    if(json.active==0)document.getElementById('inactivo').checked=true
    getLocation();
}
function blanquearForm()
{
    document.getElementById('customer_id').value=0;
    document.getElementById('tiendas').value=1;
    document.getElementById('last_name').value='';
    document.getElementById('first_name').value='';
    document.getElementById('email').value='';
    document.getElementById('address_id').value='';
    document.getElementById('address').value='';
    document.getElementById('address2').value='';
    document.getElementById('district').value='';
    document.getElementById('postal_code').value='';
    document.getElementById('phone').value='';
}
document.getElementById('mensaje').onload = function(){
    document.getElementById('mensaje').style.display = "none";
}
```

```
document.getElementById('activo').onclick = function (){
  document.getElementById('active').value=document.getElementById('activo').value
}

document.getElementById('inactivo').onclick = function (){

document.getElementById('active').value=document.getElementById('inactivo').value
}
function verCliente(idcliente){
    var out = "";
    var xhttp = new XMLHttpRequest();
    xhttp.open("POST", "/clientesById", false);
    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");
    xhttp.send(JSON.stringify({"customer_id":idcliente}));
    var data = JSON.parse(xhttp.responseText)
    console.log(data)
    deserialize(data[0])
}
```

Procedimientos almacenados para el ABM

**procedimientos.sql**

```
DROP PROCEDURE IF EXISTS `nuevocliente`;
DELIMITER $$
CREATE DEFINER='root'@'localhost' PROCEDURE `nuevocliente` (
IN cliente   JSON )
BEGIN
    DECLARE identificador smallint;

    INSERT INTO address (address,address2,district,city_id,
    postal_code,phone,location,last_update)
    VALUES(JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.address')),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.address2')),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.district')),
     JSON_EXTRACT(cliente, '$.city_id'),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.postal_code')),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.phone')),
```

```
     ST_GeomFromText(JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.location')),0),
     CURRENT_TIMESTAMP);

    SELECT max(address_id) from address into identificador;

    INSERT INTO customer (store_id,last_name,first_name,email,active,address_id)
    VALUES(JSON_EXTRACT(cliente, '$.store_id'),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.last_name')),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.first_name')),
     JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.email')),
     JSON_EXTRACT(cliente, '$.active'),
     identificador
      );
END $$
DELIMITER ;

DROP PROCEDURE IF EXISTS `modificacliente`;
DELIMITER $$
CREATE DEFINER='root'@'localhost' PROCEDURE `modificacliente` (
IN cliente   JSON )
BEGIN
    DECLARE identificador smallint;

    UPDATE address
    SET address=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.address')),
        address2=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.address2')),
        district=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.district')),
        city_id=cliente>'$.city_id',
        postal_code=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.postal_code')),
        phone=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.phone')),
        location=ST_GeomFromText(JSON_UNQUOTE(JSON_EXTRACT(cliente,
'$.location')),0),
        last_update=CURRENT_TIMESTAMP
    where
     address_id=JSON_EXTRACT(cliente,'$.address_id');

    UPDATE customer
    SET store_id=cliente>'$.store_id',
```

```
        last_name=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.last_name')),
        first_name=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.first_name')),
        email=JSON_UNQUOTE(JSON_EXTRACT(cliente, '$.email')),
        active=cliente>'$.active'
    WHERE
    customer_id=JSON_EXTRACT(cliente,'$.customer_id');
END $$
DELIMITER ;


DROP PROCEDURE IF EXISTS `bajacliente`;
DELIMITER $$
CREATE DEFINER='root'@'localhost' PROCEDURE `bajacliente` (
IN cliente   JSON )
BEGIN
    DELETE FROM address, customer USING customer
    INNER JOIN  address
    WHERE
    customer.address_id=address.address_id
    AND customer.customer_id=JSON_EXTRACT(cliente,'$.customer_id');
END $$
DELIMITER ;
```

Servidor

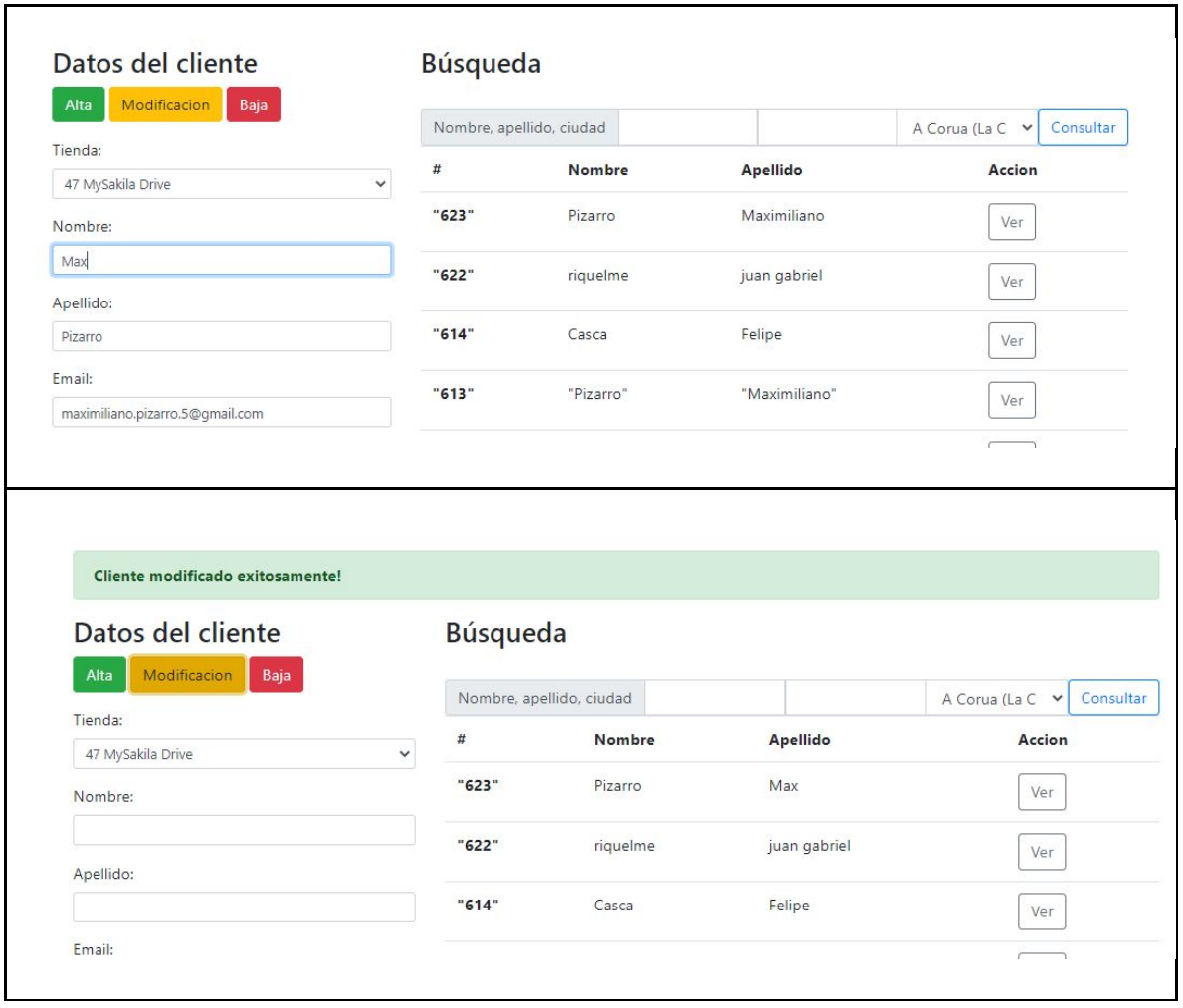
**controlador.js**

```
    alta: (req, res) => {
        console.log( Date() + ": /alta" );
        try {
            console.log(JSON.stringify(req.body, null, 0))
            const cursor = odbc.connect(datasource,(error, cursor)=>{
              cursor.query('call nuevocliente(?)',[JSON.stringify(req.body, null,
0)],
              (error, result)=>{
                if(error){
                  return res.send(JSON.stringify(error))
                }else{
                  return res.send("<strong>Cliente creado
exitosamente!</strong>")
```

```
                }
              });
            });
        } catch (e) {
            console.error( e )
            res.status( 500 )
            res.send( e )
        }
      },
      modificacion: (req, res) => {
        console.log( Date() + ": /modificacion" );
        try {
            console.log(JSON.stringify(req.body, null, 0))
            const cursor = odbc.connect(datasource,(error, cursor)=>{
              cursor.query('call modificacliente(?)',[JSON.stringify(req.body,
null, 0)],
              (error, result)=>{
                if(error){
                  return res.send(JSON.stringify(error))
                }else{
                  return res.send("<strong>Cliente modificado
exitosamente!</strong>")
                }
              });
            });
        } catch (e) {
            console.error( e )
            res.status( 500 )
            res.send( e )
        }
      },
      baja: (req, res) => {
        console.log( Date() + ": /baja" );
        try {
            console.log(JSON.stringify(req.body, null, 0))
            const cursor = odbc.connect(datasource,(error, cursor)=>{
              cursor.query('call bajacliente(?)',[JSON.stringify(req.body, null,
0)],
```

```
            (error, result)=>{
              if(error){
                return res.send(JSON.stringify(error))
              }else{
                return res.send("<strong>Cliente eliminado
exitosamente!</strong>")
              }
            });
          });
      } catch (e) {
          console.error( e )
          res.status( 500 )
          res.send( e )
      }
    }
```

Para modificar un cliente se selecciona ver que consume clientesById para traer el cliente y eventos.js lo renderiza al formulario

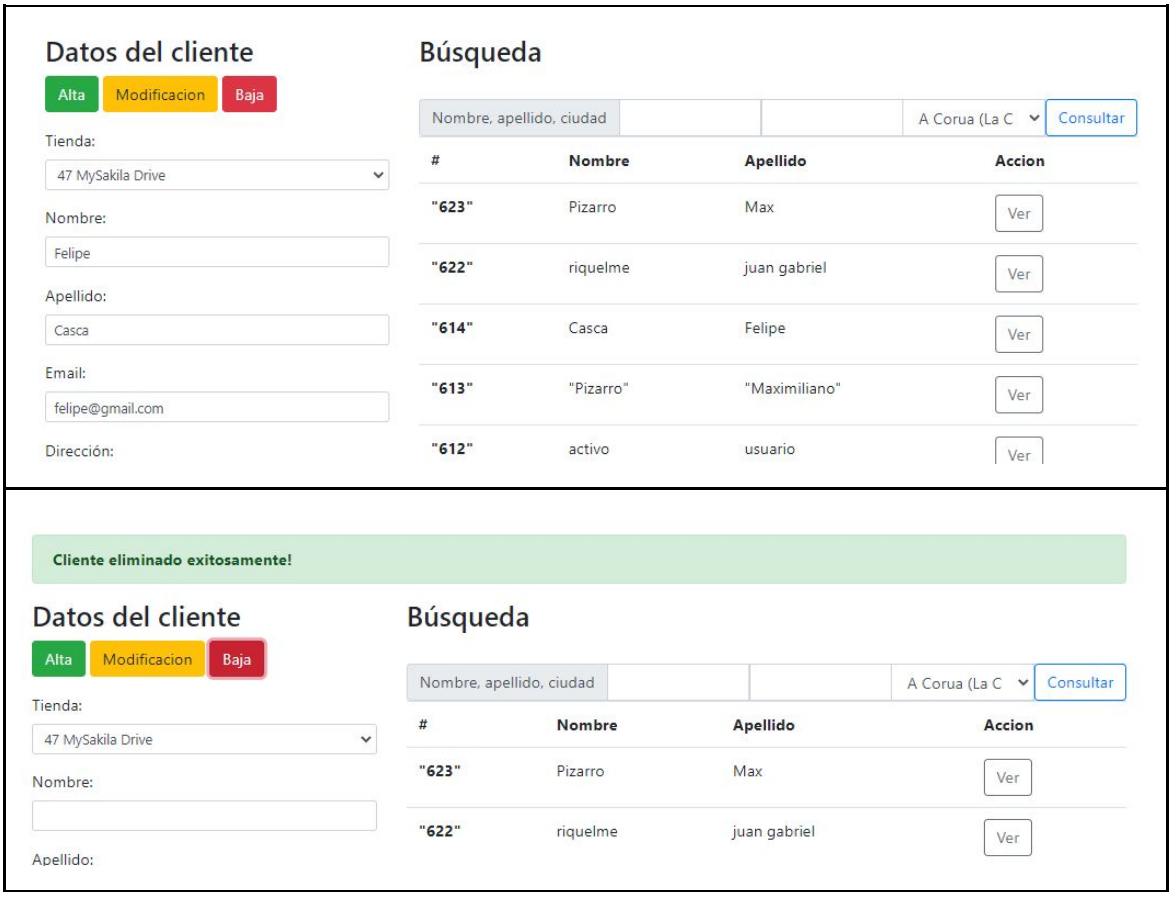


Edición Cliente

```
0|server   | Sat Sep 12 2020 22:05:51 GMT-0300 (hora estándar de Argentina): /tiendas
0|server   | Sat Sep 12 2020 22:05:51 GMT-0300 (hora estándar de Argentina): /ciudadesByI
d 1
0|server   | Sat Sep 12 2020 22:05:51 GMT-0300 (hora estándar de Argentina): /ciudades

0|server  | Sat Sep 12 2020 22:25:38 GMT-0300 (hora estándar de Argentina): /clientesById
 623
0|server  | Sat Sep 12 2020 22:25:39 GMT-0300 (hora estándar de Argentina): /ciudadesById
 6
0|server  | Sat Sep 12 2020 22:27:39 GMT-0300 (hora estándar de Argentina): /modificacion
0|server  | {"customer_id":623,"store_id":2,"last_name":"Max","first_name":"Pizarro","ema
il":"maximiliano.pizarro.5@gmail.com","address":"Ituzaingo 3168","address2":"La Madrid 21
45","district":"Lan Este","country_id":6,"city_id":20,"postal_code":"1824","phone":"116
7692039","active":"1","location":"POINT(-34.7209728 -58.359808)","address_id":631}
```

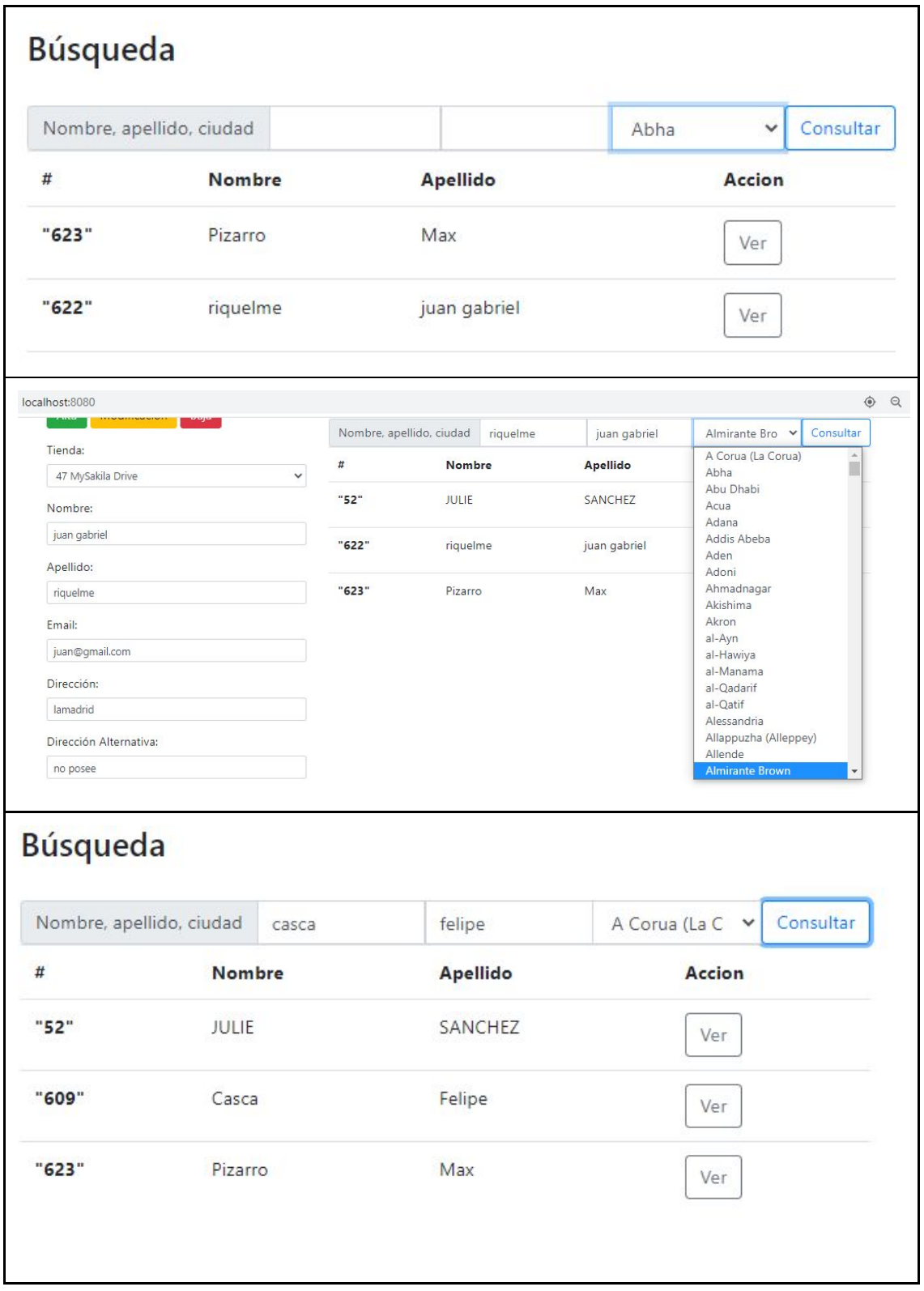
Logs modificación




```
0|server   | Sat Sep 12 2020 22:37:32 GMT-0300 (hora estándar de Argentina): /baja
0|server   | {"customer_id":614,"store_id":1,"last_name":"Felipe","first_name":"Casca","em
ail":"felipe@gmail.com","address":"ituzaingo 3168","address2":"ninguna","district":"Lanus
","country_id":87,"city_id":1,"postal_code":"1824","phone":"1144569987","active":"","loca
tion":"POINT(-34.7209728 -58.359808)","address_id":622}
```

Logs baja

D. Búsqueda de clientes: por nombre, apellido o ciudad. Los tres campos son opcionales, en el caso de la ciudad, es una lista que irá sugiriendo opciones a medida que el usuario escriba.



Búsqueda de Clientes

```
0|server   | Sat Sep 12 2020 22:41:55 GMT-0300 (hora estándar de Argentina): /buscar
0|server   | {"first_name":"casca","last_name":"felipe","city_id":20}
0|server   | Sat Sep 12 2020 22:42:13 GMT-0300 (hora estándar de Argentina): /buscar
0|server   | {"first_name":"Casca","last_name":"Felipe","city_id":20}
0|server   | Sat Sep 12 2020 22:42:16 GMT-0300 (hora estándar de Argentina): /buscar
0|server   | {"first_name":"Casca","last_name":"Felipe","city_id":20}
0|server   | Sat Sep 12 2020 22:42:26 GMT-0300 (hora estándar de Argentina): /buscar
0|server   | {"first_name":"WILLIE","last_name":"MARKHAM","city_id":20}
```

Logs búsqueda

**eventos.js**

```
function buscar(){

    var out = "";

    var xhttp = new XMLHttpRequest();

    xhttp.open("POST", "/buscar", false);

    xhttp.setRequestHeader("Content-Type", "application/json;charset=UTF-8");


xhttp.send(JSON.stringify({"first_name":document.getElementById('sfirst_name')
.value,"last_name":document.getElementById('slast_name').value,"city_id":parse
Int(document.getElementById('sciudades').value)}));

    if (xhttp.status == 500)

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

    var data = JSON.parse(xhttp.responseText)

    for (let i = 0; i < data.length; i++) {

        out += '<tr><th scope="row">"' + data[i].customer_id + '"</th><td>' +
data[i].first_name +'</td><td>'+ data[i].last_name +'</td><td><button
class="btn btn-outline-secondary" type="button" id="button-addon2"
onclick="verCliente(' + data[i].customer_id + ')" >Ver</button></td></tr>';

        }

    document.getElementById('clientes-body').innerHTML=out;


}
```

Procedimientos almacenados para la búsqueda

**procedimientos.sql**

```
DROP PROCEDURE IF EXISTS `busquedanombreciudad`;

DELIMITER $$

CREATE DEFINER='root'@'localhost' PROCEDURE `busquedanombreciudad` (

    IN cliente JSON)

BEGIN

    select customer.customer_id as customer_id,

           customer.store_id as store_id,

           customer.last_name as last_name,

           customer.first_name as first_name

    from

        customer

    inner join  address on address.address_id=customer.address_id

    inner join city on city.city_id=address.city_id

    where

    customer.last_name LIKE CONCAT('%',JSON_UNQUOTE(JSON_EXTRACT(cliente,
'$.last_name')))

    OR customer.first_name LIKE CONCAT('%',JSON_UNQUOTE(JSON_EXTRACT(cliente,
'$.first_name')))

    OR address.city_id = JSON_EXTRACT(cliente, '$.city_id');

END $$

DELIMITER ;
```

**controlador.js**

```
      buscar: (req, res) => {

        console.log( Date() + ": /buscar ");

        try {

            console.log(JSON.stringify(req.body, null, 0))

            const cursor = odbc.connect(datasource,(error, cursor)=>{

              cursor.query('call
busquedanombreciudad(?)',[JSON.stringify(req.body, null, 0)],

              (error, result)=>{

                if(error){

                  return res.send(JSON.stringify(error))

                }else{
```

```
                return res.send(result)
            }
          });
        });
    } catch (e) {
        console.error( e )
        res.status( 500 )
        res.send( e )
    }
}
```

Repositorio github

https://github.com/maximilianoPizarro/estandar-odbc