

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

| | | |
|------|--|----|
| 1. | <u>Configuración del Entorno</u> | 3 |
| 1.1. | <u>Node 14.9.0</u> | 3 |
| 1.2. | <u>MySQL Connector/ODBC 8.0.21</u> | 4 |
| 1.3. | <u>PM2</u> | 4 |
| 1.4. | <u>Bootstrap 4.5.2</u> | |
| 1.5. | <u>MySQL shell 8.0</u> | 5 |
| 1.6. | <u>Docker Desktop</u> | 6 |
| 1.7. | <u>Visual Studio Code</u> | 6 |
| 2. | <u>Desarrollo</u> | 5 |
| 2.1. | <u>Ejercicio 1</u> | 9 |
| 2.2. | <u>Código Fuente</u> | 11 |

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

MySQL Connector/ODBC Data Source Configuration

MySQL Connector/ODBC

Connection Parameters

Data Source Name: MySQL

Description: Estándar ODBC

☒ TCP/IP Server: 127.0.0.1 Port: 3306

☐ Named Pipe:

User: root

Password: ••••

Database: sakila

Test

Connection Metadata Cursors/Results Debug SSL Misc

☒ Allow big result sets ☐ Can Handle Expired Password

☐ Use compression ☐ Enable Cleartext Authentication

☒ Enable automatic reconnect ☐ Get Server Public Key

☐ Don't prompt when connecting ☐ Use DNS SRV records

☒ Allow multiple statements ☐ Multi Host

☐ Interactive Client

Character Set:

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): /países
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-mysqslsh-standalone>/bin

Verificamos la instalación ejecutando desde el shell:

mysqslsh

Conectar con base de datos

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

```
PS C:\Users\Max\Documents\UNLA\Sistemas Distribuidos\TP2\estandar-odbc> mysqslsh --sql --u
ri root:root@127.0.0.1/sakila
MySQL Shell 8.0.21

Copyright (c) 2016, 2020, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its affiliates.
Other names may be trademarks of their respective owners.

Type '\help' or '? for help; '\quit' to exit.
WARNING: Using a password on the command line interface can be insecure.
Creating a session to 'root@127.0.0.1/sakila'
Fetching schema names for autocompletion... Press ^C to stop.
Fetching table and column names from `sakila` for auto-completion... Press ^C to stop.
Your MySQL connection id is 43
Server version: 8.0.21 MySQL Community Server - GPL
Default schema set to `sakila`.
MySQL 127.0.0.1:3306 ssl sakila SQL > source procedimientos.sql
Query OK, 0 rows affected (0.0120 sec)
Query OK, 0 rows affected (0.0098 sec)
Query OK, 0 rows affected (0.0116 sec)
```

Ejecución de procedimientos almacenados de la solución

Bootstrap 4.5.2

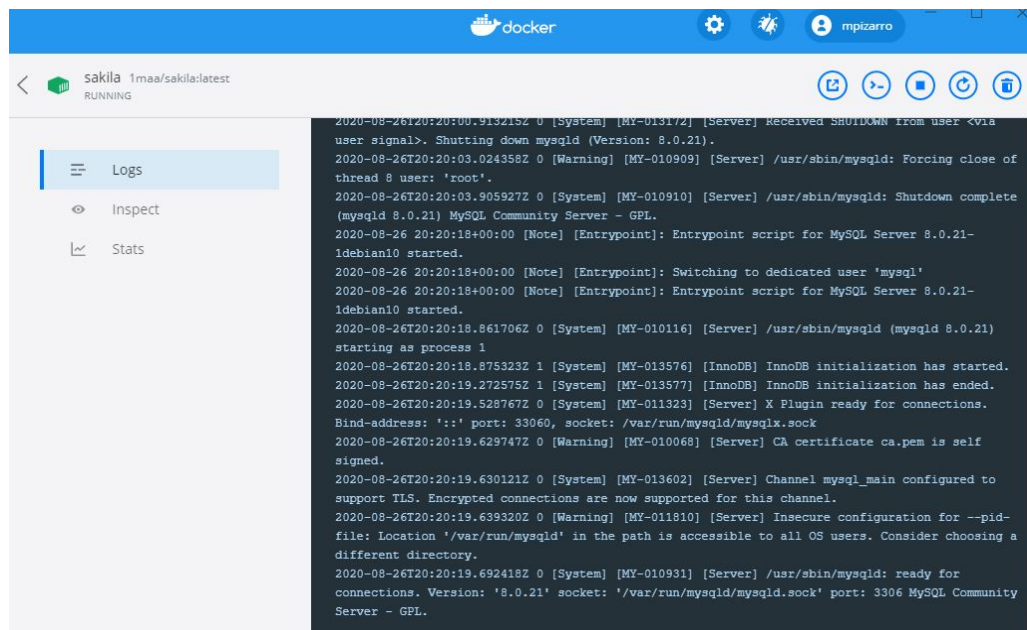
Este complemento se implementó para el material de diseño de los componentes de la UI, se encuentran importados en el archivo index.html

Docker Desktop

Este complemento es opcional y se utilizó para administrar la imagen del motor de base de datos.

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

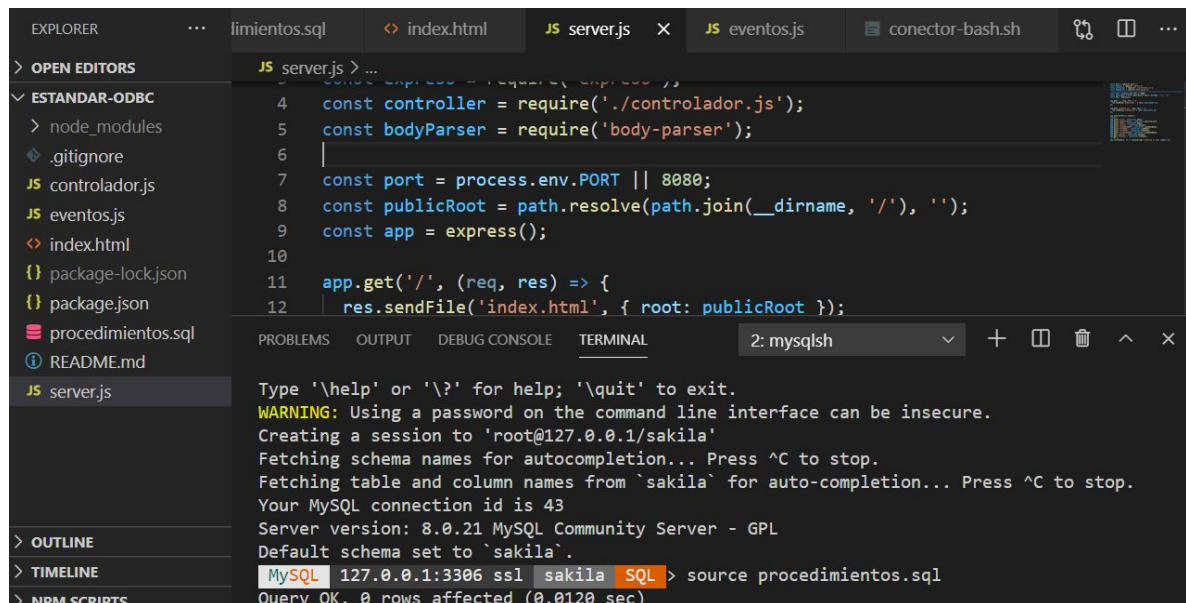
<https://hub.docker.com/r/1maa/sakila>



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

| Datos del cliente | |
|---|-------------------------------|
| <div>Alta</div> <div>Modificacion</div> <div>Baja</div> | |
| Tienda: | Pais: |
| 47 MySakila Drive | Argentina |
| | |
| Nombre: | Ciudad: |
| Maximiliano | Avellaneda |
| | |
| Apellido: | Codigo Postal: |
| Pizarro | 1824 |
| | |
| Email: | Telefono: |
| maximiliano.pizarro.5@gmail.com | 1167692039 |
| | |
| Dirección: | Ubicación |
| Ituzaingo 3168 | POINT(-34.722146099999996 -58 |
| | Consultar |
| Dirección Alternativa: | Estado |
| La Madrid 2145 | Activo |
| | Inactivo |
| Distrito: | |
| Lanús Este | |

Formulario de Alta

→ ↺ ⓘ localhost:8080

⊞ 🔍 ☆ ⚙️ 👤

Cientes

Cliente creado exitosamente!

Datos del cliente

Alta

Modificación

Baja

Tienda:

47 MySakila Drive

Nombre:

Apellido:

Email:

Búsqueda

Nombre, apellido, ciudad

A Corua (La Ci

Consultar

| # | Nombre | Apellido | Accion |
|-------|----------|--------------|----------------|
| "623" | Pizarro | Maximiliano | <div>Ver</div> |
| "622" | riquelme | juan gabriel | <div>Ver</div> |
| "614" | Casca | Felipe | <div>Ver</div> |

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

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.

```
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('países').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('países').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('activo').value=document.getElementById('activo').value
}

document.getElementById('inactivo').onclick = function (){
    document.getElementById('activo').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 `modificaccliente`;
DELIMITER $$

CREATE DEFINER='root'@'localhost' PROCEDURE `modificaccliente` (
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 modificaccliente(?)',[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

Datos del cliente

Alta

Modificacion

Baja

Tienda:

47 MySakila Drive

Nombre:

Max

Apellido:

Pizarro

Email:

maximiliano.pizarro.5@gmail.com

Búsqueda

Nombre, apellido, ciudad

A Corua (La C

Consultar

| # | Nombre | Apellido | Accion |
|-------|-----------|---------------|----------------|
| "623" | Pizarro | Maximiliano | <div>Ver</div> |
| "622" | riquelme | juan gabriel | <div>Ver</div> |
| "614" | Casca | Felipe | <div>Ver</div> |
| "613" | "Pizarro" | "Maximiliano" | <div>Ver</div> |

Cliente modificado exitosamente!

Datos del cliente

Alta

Modificacion

Baja

Tienda:

47 MySakila Drive

Nombre:

Apellido:

Email:

Búsqueda

Nombre, apellido, ciudad

A Corua (La C

Consultar

| # | Nombre | Apellido | Accion |
|-------|----------|--------------|----------------|
| "623" | Pizarro | Max | <div>Ver</div> |
| "622" | riquelme | juan gabriel | <div>Ver</div> |
| "614" | Casca | Felipe | <div>Ver</div> |

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): /ciudadesById 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","email":"maximiliano.pizarro.5@gmail.com","address":"Ituzaingo 3168","address2":"La Madrid 2145","district":"Lanús Este","country_id":6,"city_id":20,"postal_code":"1824","phone":"1167692039","active":"1","location":"POINT(-34.7209728 -58.359808)","address_id":631}
```

Logs modificación

Datos del cliente

Alta Modificación Baja

Tienda:

47 MySakila Drive

Nombre:

Felipe

Apellido:

Casca

Email:

felipe@gmail.com

Dirección:

Búsqueda

Nombre, apellido, ciudad

A Corua (La C

Consultar

| # | Nombre | Apellido | Acción |
|-------|-----------|---------------|----------------|
| "623" | Pizarro | Max | <div>Ver</div> |
| "622" | riquelme | juan gabriel | <div>Ver</div> |
| "614" | Casca | Felipe | <div>Ver</div> |
| "613" | "Pizarro" | "Maximiliano" | <div>Ver</div> |
| "612" | activo | usuario | <div>Ver</div> |

Ciente eliminado exitosamente!

Datos del cliente

Alta Modificación Baja

Tienda:

47 MySakila Drive

Nombre:

Apellido:

Búsqueda

Nombre, apellido, ciudad

A Corua (La C

Consultar

| # | Nombre | Apellido | Acción |
|-------|----------|--------------|----------------|
| "623" | Pizarro | Max | <div>Ver</div> |
| "622" | riquelme | juan gabriel | <div>Ver</div> |

```
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","email":"felipe@gmail.com","address":"ituzaingo 3168","address2":"ninguna","district":"Lanus","country_id":87,"city_id":1,"postal_code":"1824","phone":"1144569987","active":"","location":"POINT(-34.7209728 -58.359808)","address_id":622}
0|server | Sat Sep 12 2020 22:37:32 GMT-0300 (hora estándar de Argentina): /clientes
```

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

Nombre, apellido, ciudad

Abha

▼

Consultar

| # | Nombre | Apellido | Accion |
|-------|----------|--------------|--------|
| "623" | Pizarro | Max | Ver |
| "622" | riquelme | juan gabriel | Ver |

localhost:8080

Tienda:

47 MySakila Drive

Nombre:

juan gabriel

Apellido:

riquelme

Email:

juan@gmail.com

Dirección:

lamadrid

Dirección Alternativa:

no posee

Nombre, apellido, ciudad

riquelme

juan gabriel

Almirante Bro

▼

Consultar

| # | Nombre | Apellido |
|-------|----------|--------------|
| "52" | JULIE | SANCHEZ |
| "622" | riquelme | juan gabriel |
| "623" | Pizarro | Max |

A Corua (La Corua)
Abha
Abu Dhabi
Acua
Adana
Addis Abeba
Aden
Adoni
Ahmadnagar
Akishima
Akron
al-Ayn
al-Hawiya
al-Manama
al-Qadarif
al-Qatif
Alessandria
Allappuzha (Alleppey)
Allende
Almirante Brown

Búsqueda

Nombre, apellido, ciudad

casca

felipe

A Corua (La C

▼

Consultar

| # | Nombre | Apellido | Accion |
|-------|---------|----------|--------|
| "52" | JULIE | SANCHEZ | Ver |
| "609" | Casca | Felipe | Ver |
| "623" | Pizarro | Max | Ver |

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}
0|server | Sat Sep 12 2020 22:43:30 GMT-0300 (hora estándar de Argentina): /clientes
```

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":parseInt(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>