# Documentación del Proyecto DBA (JavaScript)

Este documento contiene el código fuente necesario para ejecutar la consulta de la base de datos MySQL usando Node.js.

## 1. Dependencias (package.json)

```
{
  "name": "dbajs",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "type": "commonjs",
  "dependencies": {
    "mysql2": "^3.17.0",
    "pdfkit": "^0.17.2"
  }
}
```

## 2. Instrucciones para Windows (README_WINDOWS.md)

````
# Guía de Configuración para Windows (Versión JavaScript)

Este proyecto permite consultar una base de datos MySQL local
(`tienda_online`) utilizando Node.js.

## Requisitos Previos

1.  Tener **Node.js** instalado (LTS recomendado).
2.  Tener **MySQL Server** o similar (ej. XAMPP) instalado y corriendo.
3.  Tener la base de datos `tienda_online` creada.

## Configuración del Proyecto

Sigue estos pasos en tu terminal (CMD o PowerShell):

1.  **Abrir la terminal** en la carpeta `DBAJS` del proyecto.
2.  **Instalar dependencias**:
    ```powershell
    npm install
    ```


## Ejecución del Script

Para ver los datos de la tabla `categorias` (o la configurada en el script),
ejecuta:

```powershell
node view_data.js
```


## Estructura de Archivos

*   `db_connection.js`: Módulo para manejar la conexión a MySQL.
*   `view_data.js`: Script principal para consultar y mostrar datos.
*   `package.json`: Archivo de configuración de dependencias de Node.js.
````

## 3. Conexión a Base de Datos (db_connection.js)

```
const mysql = require('mysql2/promise');

async function createServerConnection(hostName, userName, userPassword,
dbName = null, port = 3306) {
    try {
        const connection = await mysql.createConnection({
            host: hostName,
            user: userName,
            password: userPassword,
            database: dbName,
            port: port
        });
        console.log("Conexión a MySQL exitosa");
        return connection;
    } catch (err) {
        console.error(`Error: '${err.message}'`);
        return null;
    }
}

async function executeQuery(connection, query) {
    try {
        const [results] = await connection.execute(query);
        console.log("Consulta ejecutada exitosamente");
        return results;
    } catch (err) {
        console.error(`Error: '${err.message}'`);
    }
}

async function readQuery(connection, query) {
    try {
        const [results] = await connection.execute(query);
        return results;
    } catch (err) {
        console.error(`Error: '${err.message}'`);
        return null; // Return null on error consistent with python version
intent
    }
}

if (require.main === module) {
    // Example usage
    (async () => {
        const HOST = "127.0.0.1";
        const USER = "root";
        const PASSWORD = "";
        const DB_NAME = "tienda_online";
        const PORT = 3306;

        const connection = await createServerConnection(HOST, USER, PASSWORD,
DB_NAME, PORT);
        if (connection) {
            // const query = "SELECT * FROM clientes";
            // const result = await readQuery(connection, query);
            // console.log(result);
            await connection.end();
        }
    })();
```

```
}
module.exports = { createServerConnection, executeQuery, readQuery };
```

# 4. Script de Consulta (view_data.js)

```
const { createServerConnection } = require('./db_connection');

// Configuración de conexión
const HOST = "127.0.0.1";
const USER = "root";
const PASSWORD = "";
const DB_NAME = "tienda_online"; // Base de datos
const PORT = 3306;

async function showTable(tableName) {
    const connection = await createServerConnection(HOST, USER, PASSWORD,
DB_NAME, PORT);

    if (connection) {
        try {
            const query = `SELECT * FROM ${tableName}`;
            const [results, fields] = await connection.execute(query);

            console.log(`\n--- Tabla: ${tableName} ---`);
            if (results && results.length > 0) {
                console.table(results);
            } else {
                console.log("La tabla está vacía.");
            }
            console.log("-------------------------\n");

            await connection.end();
        } catch (e) {
            console.error(`Error: ${e.message}`);
            // Ensure connection is closed even on error if it was opened
            if (connection && !connection.connection.stream.destroyed) {
                await connection.end();
            }
        }
    }
}

if (require.main === module) {
    // Puedes cambiar el nombre de la tabla aquí
    showTable("categorias");
}
```