Acuaponía desde la automatización y el control

# Almacenamiento de información en la base de datos estructurada MySQL

Robert Alexis Gómez Giraldo

José Luis Parra Villamizar

Oscar Andrés Meza Ortiz

Daniel Felipe Manrique Caicedo

15 de noviembre de 2025

https://github.com/RingoSinye/proyecto_acuaponia.git

## 1. INTRODUCCIÓN.

En el presente documento se describe el proceso completo para capturar, transformar y almacenar los datos provenientes del flujo de simulación de sensores y actuadores generado en Node-RED. El objetivo principal es estructurar estos datos y almacenarlos en una base de datos MySQL ubicada en un dispositivo local, con el fin de permitir su posterior consulta, análisis y replicación del sistema por parte de otros desarrolladores o investigadores.

Para ello, se detalla la lógica de procesamiento aplicada al mensaje JSON recibido, donde se realiza la categorización y extracción de variables relevantes. Asimismo, se explica la configuración del nodo MySQL dentro de Node-RED para el almacenamiento automático de los registros en las tablas correspondientes.

Finalmente, se presenta la estructura de la base de datos, junto con los scripts SQL utilizados para crear tablas, insertar datos y ejecutar consultas de verificación, de forma que cualquier usuario pueda reproducir el sistema en su propio entorno.

## 2. REQUISITOS

Para poder replicar el proceso descrito en este documento, el usuario debe contar con los siguientes elementos instalados y configurados previamente:

### 2.1. Software necesario.

- Node.js
- Node-RED
- MySQL Server
- MySQL Workbench

### 2.2. Paquetes y dependencias de Node-RED.

- **node-red-node-mysql:** Nodo necesario para la comunicación con MySQL desde Node-RED.
  Cualquier otro nodo utilizado en el flujo.

### 2.3. Archivos del repositorio de GitHub.

Se requiere clonar o descargar el repositorio oficial del proyecto, el cual contiene:

- **Flujo de Node-RED (.json)** con toda la lógica de extracción y transformación del mensaje Sparkplug B.

- **Scripts SQL** para la creación de la base de datos, tablas y consultas de prueba.

## 3. PREPARACIÓN DEL ENTORNO

### 3.1 Importación del flujo en Node-RED

Estando en la página principal de **Node-RED** (http://localhost:1880), abra el menú lateral superior derecho, seleccione **Import**, y cargue el archivo del flujo contenido en el repositorio de GitHub. Este paso permitirá desplegar toda la lógica del sistema.

### 3.2 Configuración inicial en MySQL Workbench

A continuación, abra **MySQL Workbench** y ejecute los scripts incluidos en el repositorio. Estos scripts cumplen las siguientes funciones:

- Crear la base de datos del proyecto.
- Generar las tablas necesarias para almacenar sensores analógicos y digitales.
- Mantener disponibles los scripts de consulta y eliminación de datos, útiles durante las pruebas para verificar registros o limpiar las tablas según sea necesario.

Se recomienda dejar abiertos los scripts de consulta mientras se realizan las pruebas en Node-RED, facilitando la verificación del correcto almacenamiento de los datos enviados.

## 4. DESCRIPCIÓN DEL FLUJO DE NODE-RED

Una vez que el flujo ha sido importado correctamente y el entorno se encuentra configurado, procedemos a analizar la lógica implementada en Node-RED para transformar y organizar los datos provenientes del bloque que simula el envío de datos provenientes de sensores y actuadores. El área de interés del flujo se observa en la **Figura 1**.

### 4.1. Extracción y organización del archivo JSON.

Antes de almacenar información en la base de datos, los datos provenientes del flujo deben ser interpretados y organizados correctamente. Estos datos llegan al

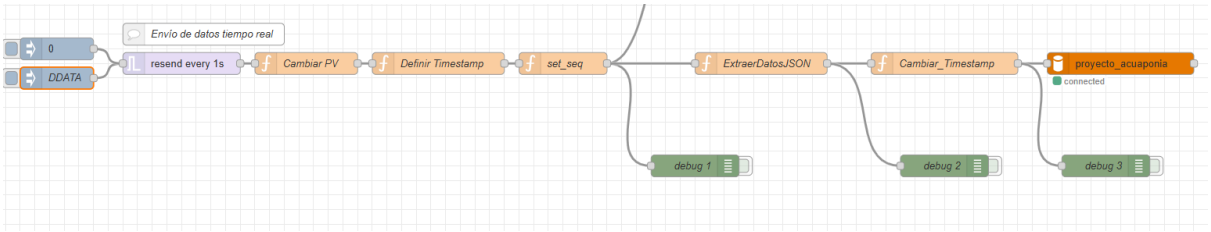


**Figura 1.** Sección del flujo de Node-red correspondiente al almacenamiento de datos en la base de datos estructurada MySQL.

nodo *ExtraerDatosJSON* en forma de un objeto JSON complejo, el cual contiene múltiples dispositivos pertenecientes a distintos submódulos del sistema acuapónico. La finalidad de este bloque es descomponer el mensaje original y transformarlo en elementos individuales que puedan clasificarse como: Sensores analógicos, sensores digitales, actuadores y materia prima. En la **Figura 2** se presenta la categorización que ejecuta el código cargado en el bloque *ExtraerDatosJSON.*

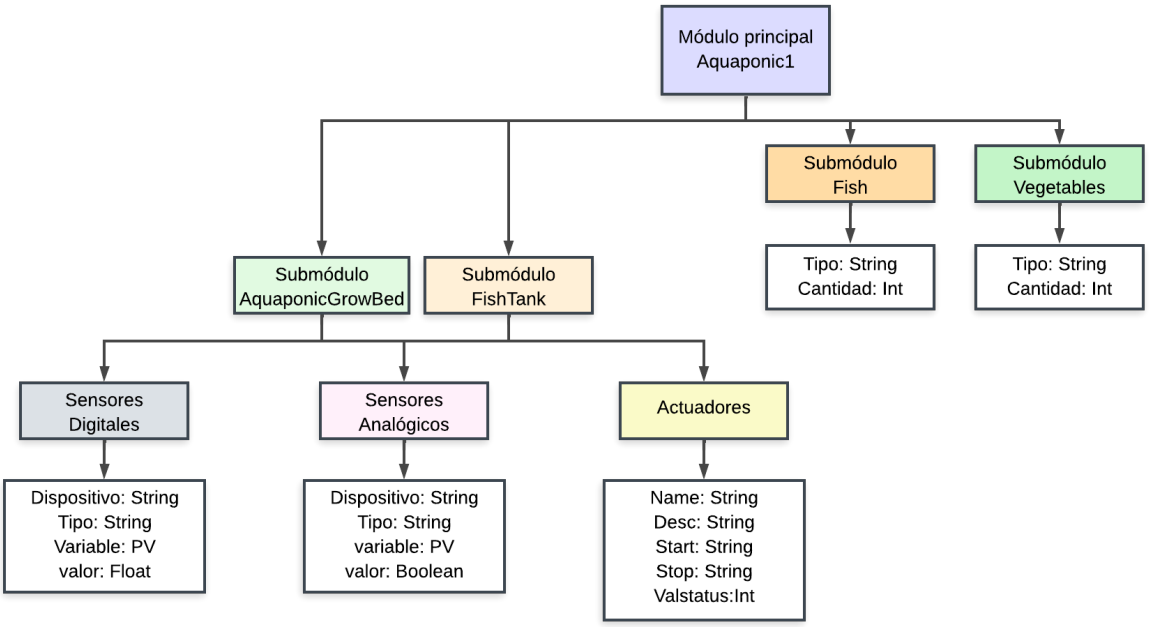


**Figura 2.** Organización y categorización de datos del objeto JSON correspondiente a la simulación de sensores y actuadores.

Como resultado de este procesamiento, el nodo ExtraerDatosJSON no entrega un único mensaje con toda la estructura original. Por el contrario, genera un mensaje independiente por cada elemento identificado, ya sea un sensor (analógico o digital), un actuador o un dato de materia prima.

Esto significa que si el sistema reporta, por ejemplo, cinco sensores y dos actuadores, el bloque producirá siete mensajes separados, cada uno con la información del dispositivo correspondiente completamente organizada y lista para ser almacenada en las tablas específicas de MySQL.

## 5. CAMBIO DEL TIMESTAMP Y ALMACENAMIENTO EN BASE DE DATOS.

El nodo CambiarTimestamp se encarga de preparar cada registro generado por *ExtraerDatosJSON* para ser almacenado correctamente en la base de datos MySQL. Su función combina dos tareas fundamentales: la conversión del timestamp original y la generación dinámica de consultas SQL para cada tipo de dato.

### 5.1. Conversión del Timestamp

Los datos provenientes de la simulación incluyen un timestamp expresado en milisegundos, sin embargo, para almacenar estos datos de tiempo en MySQL se quiere un formato de fecha estándar (AAAA-MM-DD HH:MM:SS). Por ello, el nodo implementa una función que:

- Recibe el valor original en milisegundos.
- Construye un objeto Date de JavaScript.
- Extrae año, mes, día, hora, minutos y segundos.
- Envía una fecha correctamente formateada para ser almacenada sin errores.

#### 5.2 Identificación del tipo de registro.

Cada objeto procesado contiene diferentes campos según su naturaleza. El nodo determina automáticamente si el mensaje corresponde a sensor analógico o digital, actuador y materia prima. Además, se verifica el submódulo de origen, permitiendo direccionar cada registro hacia la tabla correcta.

#### 5.3 Generación dinámica de la consulta SQL.

Según el tipo de dato detectado, el nodo construye una sentencia INSERT distinta, con la estructura correspondiente a cada tabla, cada sentencia incluye:

- El timestamp convertido,
- El módulo y submódulo,
- El nombre del dispositivo,
- Los valores específicos de cada categoría (PV, start, stop, fault, etc.).

#### 5.4 Envío de consultas agrupadas.

Una vez generadas todas las sentencias SQL, el nodo:

- las almacena temporalmente en un arreglo,

- las concatena en una sola cadena separada por “;”,
- y las envía dentro de msg.topic al nodo MySQL.

Con esto, el sistema ejecuta la inserción de múltiples registros de forma eficiente y ordenada.

## 6. VERIFICACIÓN NODO MYSQL

Antes de ejecutar el flujo completo, es necesario confirmar que el nodo **MySQL** se encuentra configurado con los mismos parámetros utilizados al crear el servidor de base de datos. En la **Figura 3** se muestra la configuración correspondiente, donde es posible identificar el nombre de la conexión, el host, el puerto, el usuario, la contraseña y la base de datos objetivo.

Es importante verificar que:

- **Host** coincide con la ubicación del servidor MySQL (por defecto *localhost*).
- **Port** corresponde al puerto configurado para MySQL (por defecto *3306*).
- **User** y **Password** sean las mismas credenciales utilizadas en Workbench.
- **Database** apunte a la base de datos creada mediante los scripts previamente cargados.

Una vez confirmados estos parámetros, el nodo estará listo para recibir las consultas generadas por el bloque *Cambiar_Timestamp* y almacenar los datos procesados en sus respectivas tablas.

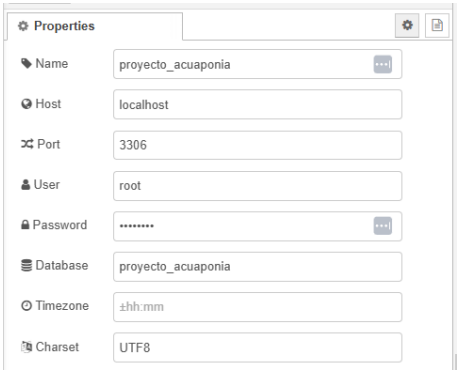


**Figura 3**. Configuración del nodo MySQL utilizado para la conexión con la base de datos proyecto_acuaponia.

## 7. CONSULTA DE TABLAS EN EL WORKBENCH

Para validar que el almacenamiento de datos se está realizando correctamente, es necesario iniciar primero el envío de información desde Node-RED. Para ello, se debe presionar el botón DDATA, ubicado en la sección izquierda del flujo (ver Figura 1). Este botón activa la simulación y envía los datos procesados hacia el nodo MySQL.

Una vez iniciado el flujo, se procede a MySQL Workbench y se ejecuta el script consultar_tablas.sql, incluido en el repositorio del proyecto. Este script permite visualizar el contenido de todas las tablas creadas previamente y confirmar que los registros están siendo insertados con el formato y la estructura adecuada.

Al ejecutar este script, se debe observar un resultado similar al de la Figura 4, donde se muestran las tablas pobladas con datos provenientes de los sensores, actuadores y módulos del sistema acuapónico.

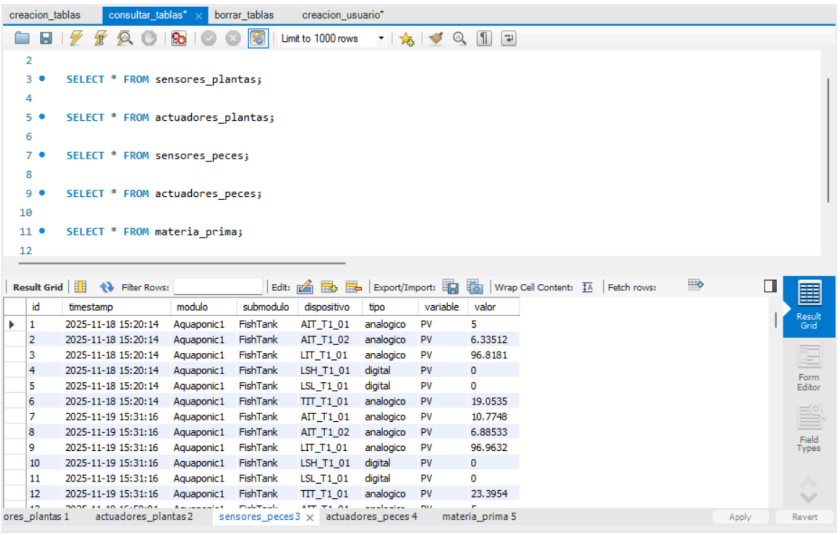


**Figura 4**. Vista de resultados obtenidos tras consultar las tablas del proyecto en MySQL Workbench.

## 8. CIERRE

El proceso desarrollado en este documento permite establecer un flujo completo y replicable para la adquisición, transformación, categorización y almacenamiento de datos provenientes de un sistema acuapónico simulado en Node-RED.

Asimismo, la integración con MySQL mediante el nodo correspondiente permite registrar la información de forma ordenada, persistente y compatible con futuros análisis. La creación de tablas específicas para cada tipo de dato facilita la trazabilidad, la depuración y la consulta histórica, mientras que las pruebas de verificación en MySQL Workbench aseguran la integridad del proceso.

Con esta documentación, cualquier persona puede replicar el entorno, importar el flujo, ejecutar los scripts de base de datos y verificar el funcionamiento del sistema sin ambigüedades.