

Medicinalis

Luciana Falcon

Noviembre 2025

Índice

1. Introducción	3
2. Objetivo	4
3. Situación problemática	5
4. Modelo de negocio	6
5. Diagrama DER	7
6. Listado de tablas	9
7. Listado de Vistas	10
8. Listado de Funciones	10
9. Listado de Stored Procedures	11
10. Script SQL	13

1. Introducción

Se buscó desarrollar un sistema que relacione los resultados de estudios sanguíneos de pacientes con recomendaciones personalizadas de nutrientes y productos. Su objetivo es facilitar la interpretación de análisis clínicos y ofrecer sugerencias automáticas de productos basadas en los valores obtenidos, mejorando la orientación nutricional y el seguimiento de la salud.

2. Objetivo

El proyecto busca desarrollar una base de datos que organice y relacione información de pacientes, estudios, resultados y nutrientes, permitiendo generar recomendaciones nutricionales personalizadas de manera automática.

Además, garantiza la integridad y trazabilidad de los datos, facilita consultas rápidas y brinda soporte analítico para profesionales de salud y nutrición.

3. Situación problemática

Como respuesta a la necesidad de organizar y relacionar la información de pacientes, estudios, resultados, nutrientes y recomendaciones, surgió la necesidad de implementar una base de datos relacional SQL. Al ser un sistema estructurado, evita errores, duplicaciones y pérdida de datos frecuentes en hojas de cálculo o registros manuales. Además, permite realizar consultas rápidas, generar recomendaciones automáticas y mantener la trazabilidad de cada paciente.

Dada la gran variedad de estudios y posibles resultados, la base de datos optimiza la generación de sugerencias nutricionales personalizadas, algo difícil de lograr de forma manual o con sistemas no estructurados.

4. Modelo de negocio

La base de datos se aplicará en el contexto de una aplicación para el público general, que permite a los usuarios registrar sus estudios médicos y recibir recomendaciones nutricionales personalizadas. La información ingresada por los usuarios se almacena de manera segura y estructurada en la base de datos, lo que permite generar sugerencias automáticas basadas en los resultados de sus análisis.

El sistema que se refleja en la figura 1 está adosado a la infraestructura de un centro de salud, que gestiona la administración, respaldo y trazabilidad de los datos, permitiendo a los profesionales del centro supervisar y analizar la información de manera eficiente, lo cual asegura que los registros sean confiables.

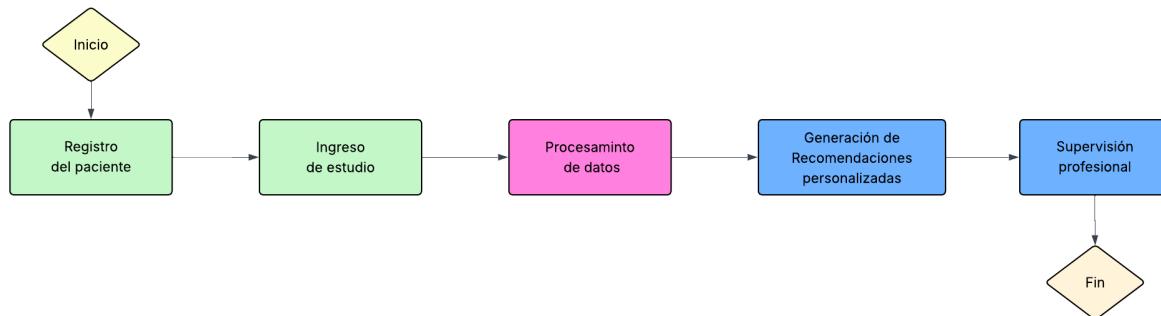


Figura 1: Diagrama de flujo del sistema Medicinalis.

5. Diagrama DER

El modelo de la figura 2 representa un sistema que gestiona estudios nutricionales de pacientes, registrando los resultados de análisis médicos y generando recomendaciones personalizadas en base a los nutrientes evaluados.

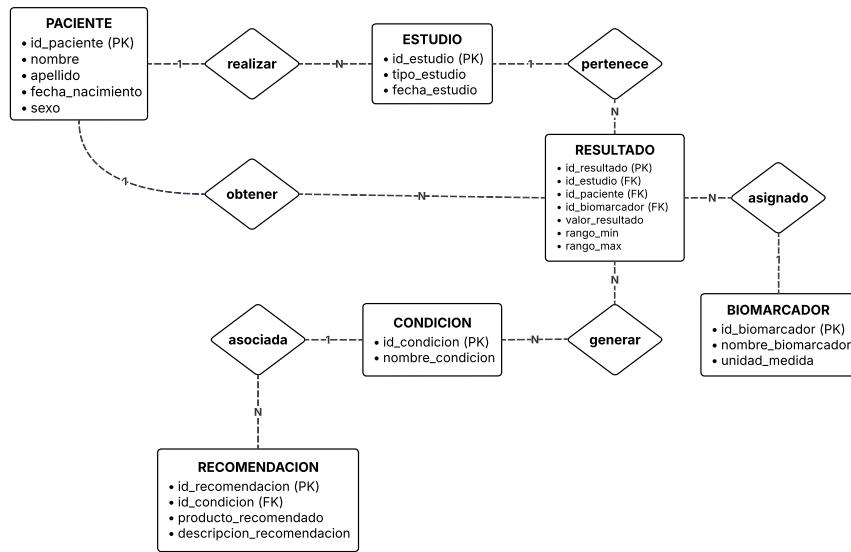


Figura 2: Diagrama Entidad-Relación.

Entidades

- **Paciente.** Contiene los datos personales de cada paciente, incluyendo su identificación, nombre, edad y sexo. Cada paciente puede realizar uno o varios estudios.
- **Estudio.** Registra los análisis clínicos realizados a los pacientes. Incluye el tipo de estudio y la fecha en que se realizó. Cada estudio puede generar múltiples resultados.
- **Resultado.** Almacena los valores obtenidos en un estudio para los distintos biomarcadores analizados. Contiene el valor medido y los rangos de referencia (mínimo y máximo) que permiten interpretar si el resultado se encuentra dentro de lo normal. Cada resultado está asociado a un único estudio y a un único biomarcador.
- **Biomarcador.** Representa los distintos componentes o sustancias evaluadas (por ejemplo: hierro, sodio, potasio, etc.). Cada biomarcador tiene un nombre y una unidad de medida. Un biomarcador puede estar asociado a muchos resultados y puede estar vinculado a una o varias condiciones.

- **Condición.** Representa la interpretación o diagnóstico derivado de los resultados de un estudio (por ejemplo: “Déficit de hierro”, “Hiperglucemia”). Cada condición puede involucrar uno o varios biomarcadores y estar asociada a múltiples recomendaciones.
- **Recomendación.** Contiene los productos o acciones sugeridas para corregir o mejorar una condición. Incluye el nombre del producto y una descripción detallada de la recomendación.

Relaciones

- **Genera** (RESULTADO – CONDICIÓN) → Relación N:N
Un resultado puede generar una o varias condiciones y una condición puede estar asociada a múltiples resultados.
Se implementa mediante la tabla intermedia `Resultado_Condicion.*`
- **Asociada** (CONDICIÓN – RECOMENDACIÓN) → Relación 1:N
Una condición puede tener varias recomendaciones, pero cada recomendación pertenece a una única condición.
- **Realizar** (PACIENTE – ESTUDIO) → Relación 1:N
Un paciente puede realizar varios estudios, pero cada estudio pertenece a un único paciente.
- **Obtener** (PACIENTE – RESULTADO) → Relación 1:N
Un paciente puede obtener varios resultados, pero cada resultado pertenece a un único paciente.
- **Pertenece** (ESTUDIO – RESULTADO) → Relación 1:N
Un estudio puede tener múltiples resultados, pero cada resultado proviene de un único estudio.

6. Listado de tablas

Tabla 1. Paciente

Campo	Tipo de dato	Restricción
id_paciente	INT AUTO_INCREMENT	PRIMARY KEY
nombre	VARCHAR(50)	NOT NULL
apellido	VARCHAR(50)	NOT NULL
fecha_nacimiento	DATE	NULL
sexo	VARCHAR(10)	NULL

Tabla 2. Estudio

Campo	Tipo de dato	Restricción
id_estudio	INT AUTO_INCREMENT	PRIMARY KEY
tipo_estudio	VARCHAR(50)	NOT NULL
fecha_estudio	DATE	NULL

Tabla 3. Biomarcador

Campo	Tipo de dato	Restricción
id_biomarcador	INT AUTO_INCREMENT	PRIMARY KEY
nombre_biomarcador	VARCHAR(50)	NOT NULL
unidad_medida	VARCHAR(10)	NULL

Tabla 4. Resultado

Campo	Tipo de dato	Restricción
id_resultado	INT AUTO_INCREMENT	PRIMARY KEY
id_paciente	INT	FOREIGN KEY → Paciente
id_estudio	INT	FOREIGN KEY → Estudio
id_biomarcador	INT	FOREIGN KEY → Biomarcador
valor_resultado	FLOAT	NOT NULL
rango_min	FLOAT	NULL
rango_max	FLOAT	NULL

Tabla 5. Condición

Campo	Tipo de dato	Restricción
id_condicion	INT AUTO_INCREMENT	PRIMARY KEY
nombre_condicion	VARCHAR(50)	NOT NULL

Tabla 6. Recomendación

Campo	Tipo de dato	Restricción
id_recomendacion	INT AUTO_INCREMENT	PRIMARY KEY
id_condicion	INT	FOREIGN KEY → Condicion
producto_recomendado	VARCHAR(255)	NULL
descripcion_recomendacion	VARCHAR(255)	NULL

Tabla 7. Resultado_Condicion (Tabla Intermedia)

Campo	Tipo de dato	Restricción
id_resultado	INT	FOREIGN KEY → Resultado
id_condicion	INT	FOREIGN KEY → Condicion
PRIMARY KEY (id_resultado, id_condicion)		

7. Listado de Vistas

- **Vista: vw_HistorialNutricional**

Descripción: Combina datos del paciente, los estudios realizados, los resultados obtenidos y las condiciones asociadas.

Objetivo de uso: Proporcionar una visión clínica completa del historial del paciente en una única vista, facilitando auditorías y consultas rápidas.

Tablas que la componen: Paciente, Estudio, Resultado, Resultado_Condicion, Condicion.

- **Vista: vw_RecomendacionesPendientes**

Descripción: Muestra todas las recomendaciones generadas para condiciones específicas, incluyendo el producto o acción sugerida.

Objetivo de uso: Identificar de manera rápida qué recomendaciones deben ejecutarse o comunicarse. Permite seguimiento eficiente desde centro de salud o por el paciente.

Tablas que la componen: Condicion, Recomendacion.

- **Vista: vw_BiomarcadoresFueraRango**

Descripción: Lista aquellos resultados en los que el biomarcador presenta valores por encima o por debajo del rango normal establecido.

Objetivo de uso: Detectar rápidamente valores críticos o desviaciones relevantes para la toma de decisiones clínicas.

Tablas que la componen: Resultado, Biomarcador.

8. Listado de Funciones

- **Función: fn_CalcularEdad(fecha_nacimiento)**

Descripción: Calcula la edad de un paciente en años, a partir de su fecha de nacimiento.

Objetivo de uso: Normalizar el dato de edad sin almacenarlo físicamente en la

base de datos (ya que cambia con el tiempo). Facilita análisis por rangos etarios.

Datos/Tablas que manipula: Campo *fecha_nacimiento* de la tabla *Paciente*.

- **Función:** `fn_EstadoResultado(id_resultado)`

Descripción: Evalúa el valor de un resultado y determina si se encuentra dentro del rango normal, por debajo o por encima. Devuelve: '*Normal*', '*Bajo*' o '*Alto*'.

Objetivo de uso: Automatizar la interpretación clínica del resultado, facilitando la generación de condiciones diagnósticas.

Datos/Tablas que manipula: Campos *valor_resultado*, *rango_min*, *rango_max* de la tabla *Resultado*.

9. Listado de Stored Procedures

Listado de Stored Procedures

- **Stored Procedure:** `sp_RegistrarNuevoEstudio`

Descripción: Inserta un nuevo registro en la tabla *Estudio* y posteriormente inserta múltiples resultados asociados a ese estudio. Se ejecuta como una única transacción para garantizar coherencia en los datos.

Objetivo y Beneficio: Asegurar la carga completa y correcta de un estudio clínico, evitando registros incompletos. Mejora la integridad referencial y reduce el margen de error humano.

Tablas que Interactúa: *Estudio*, *Resultado*.

- **Stored Procedure:** `sp_GenerarCondicion`

Descripción: Evalúa el estado de un resultado (utilizando `fn_EstadoResultado`). Si el resultado se encuentra fuera del rango normal, inserta automáticamente un vínculo entre *Resultado* y *Condición* en la tabla *Resultado_Condicion*.

Objetivo y Beneficio: Automatizar la fase diagnóstica posterior a la carga del resultado, facilitando la detección rápida de indicadores clínicos significativos. Constituye la base del sistema de recomendaciones.

Tablas que Interactúa: *Resultado*, *Condicion*, *Resultado_Condicion*. Utiliza la función `fn_EstadoResultado`.

Listado de Triggers

- **Trigger:** `tr_after_insert_resultado`

Descripción: Se ejecuta automáticamente después de insertar un nuevo registro en la tabla *Resultado*. Llama al procedimiento `sp_GenerarCondicion` para evaluar el resultado recién cargado.

Objetivo / Situación de Activación: Automatizar el diagnóstico sin requerir intervención manual, asegurando que todos los resultados sean evaluados.

Tablas que Afecta / Activa: Se dispara en *Resultado* y genera cambios en *Resultado_Condicion* a través del stored procedure correspondiente.

10. Script SQL

[tablas.sql](#)
[objetos_db.sql](#)
[insercion_datos.sql](#)