Diseño de Base de Datos - NutriGym

**Fecha:** 10/09/2025

**Autor:** Benjamin Miranda Quispe

Brandon Jair Chipana Salazar

John Manuel Villagarcia Mendoza

Victor Fernando Masias Baca

**Versión:** 1.0

# 1. Introducción

Este documento describe el modelo de base de datos relacional para la aplicación NutriGym, enfocada en la generación de dietas personalizadas para los deportistas de Enoki Gym. Se detallan las entidades principales, relaciones (incluyendo N:M), restricciones y el diccionario de datos.

# 2. Diagrama Entidad-Relación (ERD)

## 2.1. Entidades y sus Atributos

* **Usuario (nutrigym\_db\_usuarios)**
  + **id\_usuario**: Identificador único del usuario.
  + **nombre**: Nombre del usuario.
  + **email**: Correo electrónico.
  + **fecha\_nacimiento**: Fecha de nacimiento.
  + **fecha\_registro**: Momento en que se registró.
  + **contrasena**: Contraseña.
  + **deporte\_principal**: Deporte que practica habitualmente.
  + **nivel\_actividad**: Nivel de actividad física (sedentario, activo, etc.).
  + **frecuencia\_entrenamiento**: Cuántas veces entrena.
  + **nota\_text**: Notas adicionales.
  + **genero**: Género del usuario.
  + **peso\_kg**: Peso en kilogramos.
  + **altura\_cm**: Altura en centímetros.
* **Preferencias del usuario (nutrigym\_db\_preferencias)**
  + **id\_preferencia**: Identificador único.
  + **id\_usuario**: Usuario al que pertenece.
  + **presupuesto**: Presupuesto destinado a alimentación.
  + **alergia**: Alergias alimentarias.
* **Objetivos (nutrigym\_db\_objetivos)**
  + **id\_objetivo**: Identificador único.
  + **nombre**: Nombre del objetivo (ejemplo: bajar grasa, aumentar masa).
  + **descripción**: Explicación detallada.
* **Asignación de objetivos (nutrigym\_db\_usuario\_objetivos)**
  + **id\_usuario**: Usuario al que se le asigna el objetivo.
  + **id\_objetivo**: Objetivo elegido.
  + **fecha\_asignacion**: Cuándo se asignó.
  + **calificacion**: Estado o calificación del avance.
* **Progreso del usuario (nutrigym\_db\_progreso)**
  + **id\_progreso**: Identificador único.
  + **id\_usuario**: Usuario al que pertenece.
  + **fecha**: Fecha del registro.
  + **imc**: Índice de masa corporal.
  + **tmb**: Tasa metabólica basal.
  + **porcentaje\_grasa**: % de grasa corporal.
  + **porcentaje\_musculo**: % de músculo.
  + **porcentaje\_masa**: % de masa corporal.
* **Alimentos (nutrigym\_db\_alimentos)**
  + **id\_alimento**: Identificador único.
  + **nombre**: Nombre del alimento.
  + **porcion**: Tamaño o cantidad de la porción.
* **Comidas (nutrigym\_db\_comida)**
  + **id\_comida**: Identificador único.
  + **nombre**: Nombre de la comida (ejemplo: desayuno, almuerzo, cena).
* **Dietas (nutrigym\_db\_dietas)**
  + **id\_dieta**: Identificador único.
  + **fecha**: Fecha de la dieta.
  + **descripcion**: Breve explicación o detalle.
* **Asignación de dieta y comida (nutrigym\_db\_asignacion\_dieta\_comida)**
  + **id\_asignacion**: Identificador único.
  + **id\_usuario**: Usuario al que se le asigna.
  + **id\_dieta**: Dieta relacionada.
  + **id\_comida**: Comida en la que se aplica.
  + **calificacion**: Valoración de la asignación.
  + **fecha**: Fecha de asignación.
  + **prioridad**: Nivel de prioridad.
* **Administrador (nutrigym\_db\_admin)**
  + **id\_admin**: Identificador único.
  + **cargo**: Rol o puesto del administrador.

## 2.2. Relaciones

### 1. Relaciones ****Uno a Uno (1:1)****

* **Usuarios ↔ Preferencias**  
  Cada usuario tiene **un único conjunto de preferencias** (presupuesto, alergias, etc.).  
  Ejemplo: María tiene un solo registro con su presupuesto y alergias.

### 2. Relaciones ****Uno a Muchos (1:N)****

* **Usuarios ↔ Progreso**  
  Un usuario puede registrar **muchos progresos** (IMC, peso, grasa, etc. en distintas fechas).  
  Ejemplo: Juan puede tener 10 registros de progreso en el último mes.
* **Usuarios ↔ Objetivos (vía asignación)**  
  Un usuario puede tener **varios objetivos asignados a lo largo del tiempo**.  
  Ejemplo: Pedro puede primero tener el objetivo “bajar grasa” y luego “ganar masa muscular”.
* **Usuarios ↔ Asignaciones de dietas y comidas**  
  Cada usuario puede tener **múltiples dietas y comidas asignadas**.  
  Ejemplo: Carla puede tener una dieta diferente para la semana y otra para el fin de semana.

### 3. Relaciones ****Muchos a Muchos (N:M)****

* **Usuarios ↔ Objetivos**  
  Un usuario puede tener varios objetivos, y un mismo objetivo puede estar asignado a varios usuarios.  
  Esto se gestiona con la tabla intermedia **nutrigym\_db\_usuario\_objetivos**.  
  Ejemplo: “Perder grasa” puede estar asignado tanto a Pedro como a María.
* **Dietas ↔ Comidas**  
  Una dieta puede incluir muchas comidas (desayuno, almuerzo, cena…), y una comida puede aparecer en varias dietas.  
  Esto se maneja con la tabla **nutrigym\_db\_asignacion\_dieta\_comida**.  
  Ejemplo: “Desayuno saludable” puede estar en la dieta de lunes y también en la dieta de miércoles.
* **Usuarios ↔ Dietas ↔ Comidas**  
  La tabla **nutrigym\_db\_asignacion\_dieta\_comida** conecta directamente a un usuario con una dieta y una comida específica, añadiendo además datos como la **fecha**, la **prioridad** y una posible **calificación**.  
  Ejemplo: A Luis se le asigna la dieta “hipercalórica” con la comida “almuerzo” el 10 de septiembre, marcada como prioridad alta.

## 2.3. Reglas y Restricciones

### 1. ****Un alimento no puede eliminarse si está en uso dentro de una comida.****

* **Lógica:** Protege la consistencia de las dietas. Si borras un alimento usado en una comida, romperías la relación con las dietas que lo contienen.
* **Regla:** Si id\_alimento aparece en nutrigym\_db\_comida, no se puede borrar ese registro en nutrigym\_db\_alimentos.

### 2. ****El total de porcentajes corporales no puede superar el 100%.****

* **Lógica:** Garantiza que los registros fisiológicos sean válidos. Nadie puede tener 60% grasa + 50% músculo + 20% masa.
* **Regla:** La suma de porcentaje\_grasa + porcentaje\_musculo + porcentaje\_masa ≤ 100.

### 3. ****Las fechas de asignación de objetivos no pueden ser futuras.****

* **Lógica:** Un objetivo solo se asigna en el presente o en el pasado, nunca en el futuro.
* **Regla:** fecha\_asignacion ≤ fecha actual del sistema.

### 4. ****El peso y la altura del usuario deben ser valores positivos.****

* **Lógica:** Evita datos absurdos como un usuario con -70 kg o 0 cm.
* **Regla:** peso\_kg > 0 y altura\_cm > 0.

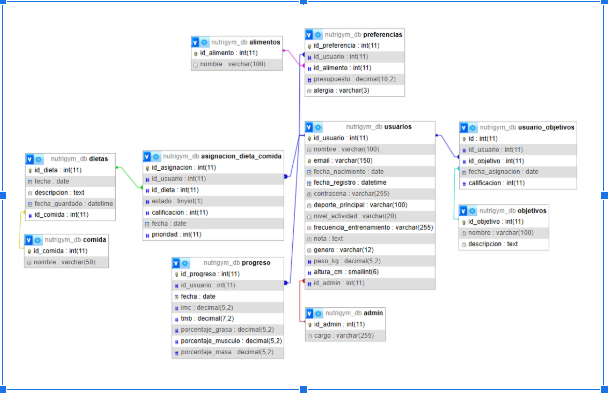
### 5. ****Un usuario no puede tener dos asignaciones de la misma dieta y comida en la misma fecha.****

* **Lógica:** Evita duplicados que confundirían el seguimiento. Un usuario no debería tener asignada dos veces el mismo almuerzo en un día.
* **Regla:** La combinación (id\_usuario, id\_dieta, id\_comida, fecha) debe ser única en nutrigym\_db\_asignacion\_dieta\_comida.

## 2.4. Diagrama de Base de Datos (E/R)

1. **Usuarios y Preferencias (1:1)**  
   nutrigym\_db\_usuarios (1) → (1) nutrigym\_db\_preferencias  
   Un usuario tiene un único conjunto de preferencias (presupuesto, alergias), y ese conjunto de preferencias pertenece a un solo usuario.
2. **Usuarios y Progreso (1:N)**  
   nutrigym\_db\_usuarios (1) → (N) nutrigym\_db\_progreso  
   Un usuario puede tener muchos registros de progreso (IMC, peso, etc.) a lo largo del tiempo, pero cada registro de progreso pertenece a un solo usuario.
3. **Dietas y Comidas (1:N)**  
   nutrigym\_db\_dietas (1) → (N) nutrigym\_db\_comida  
   Un plan de dieta puede incluir varias comidas, pero cada comida pertenece a un solo plan de dieta.
4. **Usuarios y Dietas (N:M mediante asignación)**  
   nutrigym\_db\_usuarios (N) ↔ (N) nutrigym\_db\_dietas a través de nutrigym\_db\_asignacion\_dieta\_comida  
   Un usuario puede tener asignadas muchas dietas, y una dieta puede asignarse a varios usuarios. La tabla de asignación añade detalles como fecha, prioridad y calificación.
5. **Usuarios y Objetivos (N:M mediante enlace)**  
   nutrigym\_db\_usuarios (N) ↔ (N) nutrigym\_db\_objetivos a través de nutrigym\_db\_usuario\_objetivos  
   Un usuario puede tener múltiples objetivos (bajar de peso, ganar músculo, etc.), y un objetivo puede compartirse entre varios usuarios. La tabla intermedia guarda la fecha de asignación y la calificación de cada meta.

# 3. Diagrama Relacional



# 4. Diccionario de Datos

### Tabla: nutrigym\_db\_usuarios

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_usuario | INT(11) | Clave principal, identificador único de cada usuario. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| nombre | VARCHAR(100) | Nombre completo del usuario. | NOT NULL |
| email | VARCHAR(100) | Correo electrónico del usuario. | NOT NULL, UNIQUE (asegura que cada email sea único) |
| fecha\_nacimiento | DATE | Fecha de nacimiento del usuario. | NOT NULL |
| fecha\_registro | DATETIME | Fecha y hora en que se creó la cuenta del usuario. | NOT NULL |
| contrasena | VARCHAR(255) | Contraseña encriptada del usuario. | NOT NULL, Se debe usar un algoritmo de encriptación (ej. bcrypt). |
| deporte\_principal | VARCHAR(100) | Deporte principal que practica el usuario. | Puede ser nulo |
| nivel\_actividad | VARCHAR(20) | Nivel de actividad física (e.g., sedentario, activo). | Puede ser nulo |
| frecuencia\_entrenamiento | VARCHAR(255) | Frecuencia con la que el usuario entrena. | Puede ser nulo |
| nota\_text | TEXT | Notas adicionales sobre el usuario. | Puede ser nulo |
| genero | VARCHAR(12) | Género del usuario. | Puede ser nulo |
| peso\_kg | DECIMAL(5,2) | Peso del usuario en kilogramos. | NOT NULL |
| altura\_cm | SMALLINT(6) | Altura del usuario en centímetros. | NOT NULL |

### Tabla: nutrigym\_db\_preferencias

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_preferencia | INT(11) | Clave principal para el registro de preferencias. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| id\_usuario | INT(11) | Clave foránea, vincula las preferencias a un usuario. | FOREIGN KEY (referencia a nutrigym\_db\_usuarios), NOT NULL, UNIQUE (relación 1:1) |
| presupuesto | DECIMAL(10,2) | Presupuesto del usuario para alimentos. | Puede ser nulo |
| alergia | VARCHAR(3) | Alergias alimentarias del usuario. | Puede ser nulo |

### Tabla: nutrigym\_db\_progreso

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_progreso | INT(11) | Clave principal para el registro de progreso. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| id\_usuario | INT(11) | Clave foránea, vincula el progreso a un usuario. | FOREIGN KEY (referencia a nutrigym\_db\_usuarios), NOT NULL |
| fecha | DATE | Fecha en que se tomó el registro de progreso. | NOT NULL |
| imc | DECIMAL(5,2) | Índice de Masa Corporal. | Puede ser nulo |
| tmb | DECIMAL(7,2) | Tasa Metabólica Basal. | Puede ser nulo |
| porcentaje\_grasa | DECIMAL(5,2) | Porcentaje de grasa corporal. | NOT NULL, CHECK (la suma de porcentajes no debe exceder 100) |
| porcentaje\_musculo | DECIMAL(5,2) | Porcentaje de músculo. | NOT NULL, CHECK (la suma de porcentajes no debe exceder 100) |
| porcentaje\_masa | DECIMAL(5,2) | Porcentaje de masa corporal. | NOT NULL, CHECK (la suma de porcentajes no debe exceder 100) |

### Tabla: nutrigym\_db\_alimentos

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_alimento | INT(11) | Clave principal para cada alimento. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| nombre | VARCHAR(100) | Nombre del alimento. | NOT NULL |

### Tabla: nutrigym\_db\_dietas

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_dieta | INT(11) | Clave principal para cada plan de dieta. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| fecha | DATE | Fecha de la dieta. | NOT NULL |
| descripcion | TEXT | Descripción detallada del plan de dieta. | Puede ser nulo |

### Tabla: nutrigym\_db\_comida

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_comida | INT(11) | Clave principal para cada comida. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| nombre | VARCHAR(50) | Nombre de la comida (ej. Desayuno, Almuerzo). | NOT NULL |

### Tabla: nutrigym\_db\_asignacion\_dieta\_comida

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_asignacion | INT(11) | Clave principal de la asignación. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| id\_usuario | INT(11) | Clave foránea que enlaza al usuario. | FOREIGN KEY (referencia a nutrigym\_db\_usuarios), NOT NULL |
| id\_dieta | INT(11) | Clave foránea que enlaza a la dieta. | FOREIGN KEY (referencia a nutrigym\_db\_dietas), NOT NULL |
| id\_comida | INT(11) | Clave foránea que enlaza a la comida. | FOREIGN KEY (referencia a nutrigym\_db\_comida), NOT NULL |
| calificacion | INT(11) | Calificación de la asignación. | Puede ser nulo |
| fecha | DATE | Fecha de la asignación de la dieta. | NOT NULL |
| prioridad | INT(11) | Prioridad de la asignación. | Puede ser nulo |
| fecha\_guardado | DATETIME | Fecha y hora en que se guardó la asignación. | NOT NULL |
| estado | TINYINT(1) | Estado de la asignación. | Puede ser nulo |

### Tabla: nutrigym\_db\_objetivos

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_objetivo | INT(11) | Clave principal para cada objetivo. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| nombre | VARCHAR(100) | Nombre del objetivo (e.g., Perder peso, Ganar músculo). | NOT NULL |
| descripcion | TEXT | Descripción detallada del objetivo. | Puede ser nulo |

### Tabla: nutrigym\_db\_usuario\_objetivos

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_usuario | INT(11) | Clave foránea que enlaza al usuario. | FOREIGN KEY (referencia a nutrigym\_db\_usuarios), NOT NULL |
| id\_objetivo | INT(11) | Clave foránea que enlaza al objetivo. | FOREIGN KEY (referencia a nutrigym\_db\_objetivos), NOT NULL |
| fecha\_asignacion | DATE | Fecha en que se asignó el objetivo al usuario. | NOT NULL, CHECK (fecha no puede ser futura) |
| calificacion | INT(11) | Calificación o estado del objetivo. | Puede ser nulo |

### Tabla: nutrigym\_db\_admin

| Campo | Tipo de Dato | Descripción | Restricciones |
| --- | --- | --- | --- |
| id\_admin | INT(11) | Clave principal para los administradores. | PRIMARY KEY, NOT NULL, AUTO\_INCREMENT |
| cargo | VARCHAR(255) | Rol o cargo del administrador. | Puede ser nulo |

## 4.1. Relaciones y Restricciones:

**Usuarios ↔ Preferencias:**  
Un usuario tiene un único conjunto de preferencias, y cada conjunto de preferencias pertenece a un solo usuario. La relación está representada por el campo **id\_usuario** en la tabla **nutrigym\_db\_preferencias**.  
**Restricciones:**

* id\_usuario en **nutrigym\_db\_preferencias** debe existir previamente en **nutrigym\_db\_usuarios**.
* Un usuario no puede tener más de un registro de preferencias.
* Si un usuario es eliminado, también deben eliminarse sus preferencias (**ON DELETE CASCADE**).
* Si el identificador de un usuario cambia, este cambio debe reflejarse en la tabla de preferencias (**ON UPDATE CASCADE**).

**Usuarios ↔ Progreso:**  
Un usuario puede tener muchos registros de progreso, pero cada registro de progreso pertenece a un solo usuario. La relación está representada por el campo **id\_usuario** en la tabla **nutrigym\_db\_progreso**.  
**Restricciones:**

* id\_usuario en **nutrigym\_db\_progreso** debe existir previamente en **nutrigym\_db\_usuarios**.
* No se permite un registro de progreso con fecha duplicada para el mismo usuario.
* Si un usuario es eliminado, también deben eliminarse sus registros de progreso (**ON DELETE CASCADE**).

**Usuarios ↔ Objetivos:**  
Un usuario puede tener muchos objetivos, y un objetivo puede pertenecer a muchos usuarios. La relación está representada por la tabla **nutrigym\_db\_usuario\_objetivos**, que enlaza **id\_usuario** con **id\_objetivo**.  
**Restricciones:**

* id\_usuario en **nutrigym\_db\_usuario\_objetivos** debe existir previamente en **nutrigym\_db\_usuarios**.
* id\_objetivo en **nutrigym\_db\_usuario\_objetivos** debe existir previamente en **nutrigym\_db\_objetivos**.
* Un usuario no puede tener asignado el mismo objetivo en la misma fecha más de una vez (restricción de unicidad sobre id\_usuario, id\_objetivo, fecha\_asignacion).
* La fecha de asignación no puede ser futura.

**Dietas ↔ Comidas:**  
Una dieta puede contener muchas comidas, y una comida puede pertenecer a muchas dietas. La relación está representada por la tabla **nutrigym\_db\_asignacion\_dieta\_comida**, que enlaza **id\_dieta** con **id\_comida**.  
**Restricciones:**

* id\_dieta en **nutrigym\_db\_asignacion\_dieta\_comida** debe existir previamente en **nutrigym\_db\_dietas**.
* id\_comida en **nutrigym\_db\_asignacion\_dieta\_comida** debe existir previamente en **nutrigym\_db\_comida**.
* Una misma dieta no puede tener la misma comida repetida en la misma fecha para el mismo usuario (restricción de unicidad sobre id\_usuario, id\_dieta, id\_comida, fecha).

**Usuarios ↔ Dietas (a través de Asignaciones):**  
Un usuario puede tener asignadas muchas dietas, y una dieta puede ser asignada a muchos usuarios. La relación está representada por la tabla **nutrigym\_db\_asignacion\_dieta\_comida**, que enlaza **id\_usuario** con **id\_dieta** y **id\_comida**.  
**Restricciones:**

* id\_usuario en **nutrigym\_db\_asignacion\_dieta\_comida** debe existir previamente en **nutrigym\_db\_usuarios**.
* id\_dieta en **nutrigym\_db\_asignacion\_dieta\_comida** debe existir previamente en **nutrigym\_db\_dietas**.
* Un usuario no puede tener asignada la misma dieta y comida más de una vez en la misma fecha.

# 5. Consideraciones de Escalabilidad

* Índices en claves foráneas y campos de búsqueda frecuente (id\_usuario, id\_dieta, fecha en progreso, email en usuarios).
* Uso de tablas intermedias con restricciones UNIQUE para evitar duplicidad en asignaciones de objetivos y dietas.
* Particionamiento de tablas con alto volumen de datos, como progreso, por rangos de fecha (mensual o anual) para optimizar consultas.
* Implementación de replicación maestro–esclavo para separar cargas de lectura y escritura en sistemas de gran escala.
* Uso de caché en consultas repetitivas (últimas dietas asignadas, historial de progreso) con herramientas como Redis.
* Backups incrementales automáticos y plan de recuperación ante desastres para garantizar disponibilidad y continuidad del servicio.

# 6. Conclusiones

Este modelo relacional soporta la flexibilidad necesaria para gestionar tareas colaborativas, etiquetas personalizadas y múltiples listas por usuario. Permite escalar el sistema con crecimiento de usuarios y tareas sin comprometer la integridad de los datos.