Arquitectura del Estudio de Fitness Moderno: Un Plan Maestro para una Base de Datos SQL Preparada para la Inteligencia Artificial

Parte I: El Plano Arquitectónico – Entidades Centrales y Relaciones

1.1. Introducción: La Base de Datos como Sistema Nervioso del Negocio

En la gestión de un negocio de servicios deportivos, como un gimnasio o un estudio de pilates, la base de datos es mucho más que un simple repositorio de información. Es el sistema nervioso central que conecta cada faceta de la operación, desde la gestión de clientes hasta el análisis financiero y la planificación estratégica. Una base de datos bien diseñada no solo garantiza la eficiencia operativa diaria, sino que también sienta las bases para el crecimiento futuro, la escalabilidad y, de manera crucial, la implementación de tecnologías avanzadas como la Inteligencia Artificial (IA).[1] El diseño del esquema de la base de datos es, por tanto, el paso más crítico en la construcción de la infraestructura tecnológica de su negocio. Un diseño robusto y reflexivo es un activo estratégico que facilita la innovación, mientras que un diseño deficiente puede convertirse en una barrera técnica significativa, limitando el potencial de crecimiento y la capacidad de adaptación.[3]

Este informe adopta un enfoque holístico, concibiendo la base de datos como una "única fuente de verdad" (*Single Source of Truth*). Este principio dicta que toda la información crítica del negocio —datos de miembros, registros financieros, horarios del personal, interacciones operativas— se centraliza en un ecosistema cohesivo y lógicamente estructurado.[5] Al eliminar los silos de datos, como hojas de cálculo dispersas o registros en papel, se asegura la consistencia, se reduce la probabilidad

de errores y se habilita una visión de 360 grados del negocio.[6] Esta visión unificada es indispensable para la toma de decisiones informadas y, como se detallará más adelante, es el prerrequisito fundamental para cualquier iniciativa de IA. El objetivo de este documento es proporcionar un plan maestro completo, desde el concepto arquitectónico hasta el script de implementación final, para construir una base de datos SQL que no solo satisfaga las necesidades actuales de su estudio, sino que lo posicione para liderar en la era de los datos.

1.2. Identificación de las Entidades Fundamentales del Negocio

Antes de diseñar las tablas y escribir código SQL, es imperativo identificar las entidades fundamentales que constituyen el modelo de negocio de un centro de fitness. Estas entidades son los "sustantivos" de su operación: las personas, lugares, cosas y conceptos sobre los cuales necesita almacenar información. Un análisis exhaustivo de los requisitos operativos y los estándares de la industria revela un conjunto central de entidades que formarán el esqueleto de nuestra base de datos.[7] Cada una de estas entidades se traducirá en una tabla en nuestra base de datos, con sus características representadas como columnas.

El listado de entidades principales incluye:

- **Miembros (Members):** El corazón del negocio. Representa a los clientes que se inscriben y utilizan los servicios del estudio. Almacenar sus datos de contacto, demográficos y de membresía es crucial para la gestión de relaciones con el cliente (CRM).[8]
- **Personal (Staff):** Incluye a todos los empleados del estudio, desde instructores y entrenadores personales hasta el personal de recepción y gerencia. Esta entidad es clave para la gestión de recursos humanos, horarios y nóminas.[10]
- **Planes de Membresía (Membership Plans):** Define el catálogo de "productos" o servicios que ofrece el estudio. Cada plan tendrá un nombre, descripción, precio y duración específicos (por ejemplo, "Pilates Mensual Ilimitado", "Bono de 10 Clases").[11]
- **Clases (Classes):** Representa los tipos de clases o actividades que se ofrecen, como "Pilates Reformer Nivel 1" o "Yoga Vinyasa". Esta es una definición abstracta de la oferta del estudio.[1]
- **Pagos (Payments):** Registra cada transacción de ingreso de dinero, ya sea por membresías, clases sueltas o venta de productos. Es una entidad fundamental

para el seguimiento financiero.[1]
- **Gastos (Expenses):** Registra cada transacción de egreso de dinero, como el pago de alquiler, salarios, suministros o marketing. Junto con los pagos, permite una contabilidad completa.[5]
- **Asistencia (Attendance):** Una entidad crucial que registra la participación de un miembro en una clase o sesión específica. Es un indicador clave del compromiso del cliente y una fuente de datos vital para el análisis de comportamiento.[11]

Otras entidades importantes, que se detallarán más adelante, incluyen las suscripciones de los miembros, los horarios del personal y las clases programadas, que dan vida a las operaciones diarias.

1.3. El Diagrama de Entidad-Relación (ERD): Visualizando el Ecosistema

Un Diagrama de Entidad-Relación (ERD) es una representación visual que sirve como el plano arquitectónico de la base de datos. Muestra las entidades (tablas) y, lo que es más importante, las relaciones que existen entre ellas.[2] Utilizar un ERD antes de la implementación permite identificar y resolver problemas de diseño de manera conceptual, asegurando que la estructura lógica del negocio se traduzca correctamente en la estructura de la base de datos. Para este diseño, se utilizará la notación "Pata de Gallo" (

Crow's Foot), un estándar de la industria que es intuitivo y claro para representar la cardinalidad de las relaciones.[15]

La cardinalidad define cuántas instancias de una entidad pueden estar relacionadas con cuántas instancias de otra entidad. Las relaciones clave son:

- **Uno a Uno (1:1):** Una instancia de la Entidad A se relaciona con, como máximo, una instancia de la Entidad B. Este tipo de relación es menos común.
- **Uno a Muchos (1:N):** Una instancia de la Entidad A se puede relacionar con muchas instancias de la Entidad B, pero una instancia de B solo se relaciona con una de A. Esta es la relación más común. Por ejemplo, un Miembro puede realizar muchos Pagos, pero cada Pago es realizado por un único Miembro.
- **Muchos a Muchos (M:N):** Una instancia de la Entidad A se puede relacionar con muchas instancias de la Entidad B, y viceversa. Por ejemplo, un Miembro puede asistir a muchas Clases, y una Clase puede tener muchos Miembros.[7]

Comprender y modelar correctamente estas relaciones es fundamental. Una relación 1:N se implementa típicamente mediante una clave foránea (*foreign key*) en la tabla del lado "muchos", que apunta a la clave primaria (*primary key*) de la tabla del lado "uno". Sin embargo, las relaciones M:N requieren un tratamiento especial, que es clave para la flexibilidad y escalabilidad del modelo.

1.4. El Poder de las Tablas de Unión (Junction Tables)

Una de las decisiones arquitectónicas más importantes y que distingue un diseño profesional de uno amateur es el manejo correcto de las relaciones de muchos a muchos (M:N). Intentar modelar una relación M:N directamente entre dos tablas es un error conceptual que conduce a una base de datos inflexible y con datos redundantes.

Consideremos la relación entre Miembros y ClasesProgramadas. Un miembro puede inscribirse en múltiples clases a lo largo del tiempo, y una clase programada específica (por ejemplo, la clase de Pilates del lunes a las 10:00) tendrá múltiples miembros inscritos.[1] Si intentáramos añadir una columna

clase_id a la tabla Miembros, un miembro solo podría estar inscrito en una única clase. Si, por el contrario, añadiéramos una columna miembro_id a la tabla ClasesProgramadas, cada clase solo podría tener un único miembro. Ambas soluciones son incorrectas y no reflejan la realidad del negocio.

La solución arquitectónica correcta es la creación de una tercera tabla, conocida como tabla de unión (*junction table*) o tabla asociativa. En nuestro caso, esta tabla será Asistencia (o InscripcionesClase).[11] Esta tabla de unión contiene, como mínimo, dos claves foráneas: una que apunta a la clave primaria de la tabla

Miembros (miembro_id) y otra que apunta a la clave primaria de la tabla ClasesProgramadas (clase_programada_id). La combinación de estas dos claves foráneas formará la clave primaria de la tabla de unión, asegurando que un miembro solo pueda inscribirse una vez en la misma clase programada.

La implementación de tablas de unión va más allá de ser una simple solución técnica. Transforma la forma en que se capturan los datos. La tabla Asistencia no solo resuelve la relación M:N, sino que se convierte en un registro detallado de cada interacción de

un miembro con los servicios principales del estudio. Cada fila en esta tabla es un evento: "El miembro X asistió a la clase Y en la fecha Z". Esta granularidad es oro puro para el análisis de negocio y la futura IA. Permite responder preguntas como: "¿Cuál es la frecuencia de asistencia de nuestros miembros más valiosos?", "¿Qué clases son las más populares?", "¿Qué miembros muestran una disminución en su patrón de asistencia y podrían estar en riesgo de abandonar?". Por lo tanto, se utilizarán tablas de unión para modelar todas las relaciones M:N, como la asignación de instructores a clases (Personal_Clases) o la asignación de permisos a roles (Roles_Permisos), garantizando una base de datos normalizada, escalable y, sobre todo, rica en datos de comportamiento.

Parte II: Estructuras de Datos Fundamentales – Gestión de Miembros y Suscripciones

Esta sección se adentra en el diseño de las tablas que gestionan el activo más importante del negocio: los clientes. Una gestión eficaz de los miembros y sus suscripciones no solo es vital para los ingresos, sino que también constituye la base de cualquier sistema de gestión de relaciones con el cliente (CRM).

2.1. La Tabla Miembros: El Corazón del CRM

La tabla Miembros es el repositorio central de toda la información relativa a los clientes. Su diseño debe ser exhaustivo para soportar no solo las operaciones diarias, sino también las actividades de marketing, comunicación y análisis futuro.

Un diseño robusto para la tabla Miembros irá más allá de los campos básicos de contacto.[1] Incluirá atributos que permitan una segmentación detallada y un seguimiento del ciclo de vida del cliente, elementos esenciales para la personalización y la analítica avanzada. Por ejemplo, la

fecha_nacimiento permite crear campañas de marketing específicas por edad o enviar felicitaciones de cumpleaños, mientras que el campo estado proporciona una visión clara e inmediata de la situación de cada miembro sin necesidad de complejas

consultas.[6]

Además, es de suma importancia considerar la seguridad y privacidad de los datos almacenados. Esta tabla contendrá Información de Identificación Personal (PII), y su protección es una responsabilidad legal y ética. Se deben implementar medidas de seguridad a nivel de base de datos y de aplicación para proteger esta información sensible contra accesos no autorizados.[5]

A continuación, se detalla la estructura propuesta para la tabla Miembros:

- miembro_id: INT o UUID, Clave Primaria (PK). Un identificador único para cada miembro.
- nombre: VARCHAR(100), NOT NULL.
- apellido: VARCHAR(100), NOT NULL.
- email: VARCHAR(255), UNIQUE, NOT NULL. Esencial para la comunicación y como posible nombre de usuario.
- telefono: VARCHAR(20).
- fecha_nacimiento: DATE. Permite la segmentación demográfica.
- fecha_registro: TIMESTAMP, DEFAULT CURRENT_TIMESTAMP. Registra cuándo se unió el miembro, útil para análisis de cohortes.
- estado: ENUM('activo', 'inactivo', 'congelado', 'potencial'), NOT NULL. Facilita el filtrado y la gestión del estado del cliente.
- direccion: VARCHAR(255).
- ciudad: VARCHAR(100).
- codigo_postal: VARCHAR(20).
- notas_administrativas: TEXT. Para comentarios internos del personal.

2.2. La Tabla PlanesMembresia: El Catálogo de Productos

Esta tabla funciona como el catálogo de servicios del estudio. Define todos los tipos de membresías o paquetes que un cliente puede adquirir. Centralizar esta información en una tabla separada permite una gestión sencilla de la oferta comercial: para cambiar el precio de un plan o añadir uno nuevo, solo es necesario modificar o añadir una fila en esta tabla, sin afectar a los registros de los miembros existentes.[11]

La estructura propuesta es la siguiente:

- plan_id: INT, Clave Primaria (PK).

- nombre_plan: VARCHAR(150), UNIQUE, NOT NULL. Por ejemplo, 'Acceso Total Mensual', 'Bono 10 Clases Pilates'.
- descripcion: TEXT. Detalles sobre lo que incluye el plan.
- precio: DECIMAL(10, 2), NOT NULL. El precio estándar del plan.
- duracion_dias: INT. La validez del plan en días (e.g., 30 para mensual, 365 para anual). Para bonos de clases, puede ser NULL.
- numero_clases: INT. Para planes tipo bono (e.g., 10). Para planes ilimitados, puede ser NULL.
- tipo_plan: ENUM('Solo Pilates', 'Solo Gimnasio', 'Completo', 'Bono'), NOT NULL. Para segmentar la oferta.
- activo: BOOLEAN, DEFAULT TRUE. Permite desactivar planes antiguos sin borrarlos de la base de datos.

2.3. La Tabla SuscripcionesMiembro: El Vínculo Transaccional

Esta es una tabla de unión crucial que conecta a un Miembro con un PlanMembresia en un momento específico. Cada vez que un miembro compra o renueva una membresía, se crea un nuevo registro en esta tabla. Esto crea un historial completo de la relación comercial con cada cliente.

La estructura propuesta es:

- suscripcion_id: INT, Clave Primaria (PK).
- miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id).
- plan_id: INT, Clave Foránea (FK) a PlanesMembresia(plan_id).
- fecha_inicio: DATE, NOT NULL.
- fecha_fin: DATE, NOT NULL. Calculada en el momento de la compra (fecha_inicio + duracion_dias).
- estado: ENUM('activa', 'vencida', 'cancelada', 'futura'), NOT NULL.
- precio_pagado: DECIMAL(10, 2), NOT NULL. Almacena el precio real pagado en la transacción, que podría diferir del precio estándar del plan debido a descuentos o promociones.
- clases_restantes: INT. Para planes tipo bono, se actualiza con cada asistencia.

2.4. Separar el "Qué" del "Quién y Cuándo" para la Inteligencia de Negocio

La decisión de separar la definición de los planes (PlanesMembresia) del registro de su adquisición (SuscripcionesMiembro) es una piedra angular de este diseño arquitectónico. Un enfoque más simplista, a menudo derivado de la lógica de las hojas de cálculo, podría intentar almacenar la información de la membresía (como fecha_fin o precio) directamente en la tabla Miembros.[21] Este enfoque, aunque aparentemente más sencillo, se desmorona ante escenarios del mundo real y limita drásticamente la capacidad de análisis.

Consideremos las siguientes preguntas, que un diseño simplista no puede responder fácilmente:

1. ¿Qué sucede cuando un miembro renueva su suscripción? ¿Se sobrescriben los datos antiguos, perdiendo el historial?
2. ¿Cómo se gestiona a un miembro que decide cambiar de un plan mensual a uno anual a mitad de camino?
3. Si se decide aumentar el precio del 'Plan Mensual' para los nuevos clientes, ¿cómo se evita que este cambio afecte retroactivamente a los miembros actuales que pagaron el precio antiguo?

La arquitectura de tres tablas (Miembros, PlanesMembresia, SuscripcionesMiembro) resuelve elegantemente estos problemas. La tabla SuscripcionesMiembro crea un registro histórico, inmutable y auditable de cada período de suscripción para cada miembro. Cada fila es una "instantánea" de un contrato de servicio en un momento dado.

Esta estructura no es solo una buena práctica de normalización de bases de datos; es el motor que impulsa la inteligencia de negocio y prepara el terreno para la IA. Con este historial detallado, es posible:

- **Calcular el Valor de Vida del Cliente (Customer Lifetime Value - CLV):** Sumando el precio_pagado de todas las suscripciones de un miembro a lo largo del tiempo.
- **Predecir el Abandono (Churn Prediction):** Los modelos de IA pueden analizar los patrones en SuscripcionesMiembro. Por ejemplo, un miembro que siempre renovaba anualmente y de repente cambia a un plan mensual podría ser una señal de riesgo de abandono.
- **Analizar la Rentabilidad de los Planes:** Se puede determinar fácilmente qué planes (plan_id) generan más ingresos y tienen las tasas de renovación más altas.

- **Realizar Análisis de Cohortes:** Agrupar a los miembros por su fecha_registro y analizar el comportamiento de sus suscripciones a lo largo del tiempo para entender cómo evoluciona la retención.

En resumen, esta separación deliberada entre el catálogo de productos y el historial de transacciones es lo que eleva la base de datos de ser un simple sistema de registro a una poderosa herramienta de análisis estratégico.

Parte III: Excelencia Operativa – Programación, Asistencia y Personal

Esta sección se centra en el diseño de las estructuras de datos que soportan el núcleo de las operaciones diarias del estudio: la gestión del personal, la programación de clases y el seguimiento de la asistencia. Una estructura bien definida en esta área es fundamental para la eficiencia, la comunicación interna y la calidad del servicio al cliente.

3.1. Gestión de Personal y Roles

Una gestión eficaz del personal requiere más que una simple lista de empleados. Es necesario definir claramente sus roles, responsabilidades y horarios de trabajo.

- **Tabla Personal:** Esta tabla contendrá la información básica de cada empleado del estudio.
  - personal_id: INT, Clave Primaria (PK).
  - nombre: VARCHAR(100), NOT NULL.
  - apellido: VARCHAR(100), NOT NULL.
  - email: VARCHAR(255), UNIQUE, NOT NULL.
  - telefono: VARCHAR(20).
  - fecha_contratacion: DATE, NOT NULL.
  - rol_id: INT, Clave Foránea (FK) a RolesPersonal(rol_id). Asigna un rol principal a cada empleado.[10]
  - especialidades: TEXT. Campo para describir certificaciones o áreas de especialización (ej. 'Pilates para embarazadas', 'Entrenamiento funcional').

    - activo: BOOLEAN, DEFAULT TRUE. Para gestionar altas y bajas.
- **Tabla RolesPersonal:** Esta tabla de consulta (*lookup table*) define los diferentes roles dentro de la organización. Separar los roles en su propia tabla permite una gran flexibilidad. Por ejemplo, si en el futuro se crea un nuevo puesto, como "Coordinador de Redes Sociales", simplemente se añade una nueva fila a esta tabla. Además, esta estructura es la base para un sistema de permisos basado en roles (RBAC - Role-Based Access Control), donde a cada rol se le pueden asignar permisos específicos dentro del sistema de software.[10]
    - rol_id: INT, Clave Primaria (PK).
    - nombre_rol: VARCHAR(100), UNIQUE, NOT NULL. Ejemplos: 'Instructor Pilates', 'Entrenador Personal', 'Recepción', 'Gerente'.
    - descripcion_rol: TEXT.
- **Tabla HorariosPersonal:** Esta tabla es esencial para la planificación operativa y el cálculo de la nómina, especialmente para el personal que trabaja por horas. Registra los turnos de trabajo programados para cada empleado.[26]
    - horario_id: INT, Clave Primaria (PK).
    - personal_id: INT, Clave Foránea (FK) a Personal(personal_id).
    - inicio_turno: DATETIME, NOT NULL.
    - fin_turno: DATETIME, NOT NULL.
    - notas: TEXT. Para detalles específicos del turno.

3.2. Gestión de Clases y Horarios

Para gestionar la oferta de clases de manera eficiente y flexible, es crucial distinguir entre el concepto de una clase y su instancia específica en el calendario.

- **Tabla Clases:** Esta tabla actúa como el catálogo de los tipos de clases que ofrece el estudio. Define la plantilla para cada actividad.[1]
    - clase_id: INT, Clave Primaria (PK).
    - nombre_clase: VARCHAR(150), UNIQUE, NOT NULL. Ej: 'Pilates Reformer Nivel 1', 'Yoga Vinyasa', 'HIIT'.
    - descripcion: TEXT.
    - duracion_minutos_default: INT. Duración estándar de la clase.
    - capacidad_maxima_default: INT. Capacidad estándar.
- **Tabla ClasesProgramadas:** Esta tabla representa una instancia concreta de una clase que ocurre en una fecha y hora específicas, en una sala determinada y con un instructor asignado. Es el corazón del calendario del estudio.[2]

    - clase_programada_id: INT, Clave Primaria (PK).
    - clase_id: INT, Clave Foránea (FK) a Clases(clase_id).
    - instructor_id: INT, Clave Foránea (FK) a Personal(personal_id).
    - fecha_hora_inicio: DATETIME, NOT NULL.
    - fecha_hora_fin: DATETIME, NOT NULL.
    - sala_o_ubicacion: VARCHAR(100). Ej: 'Sala Reformer', 'Estudio Yoga', 'Online'.
    - capacidad_actual: INT. Puede diferir de la capacidad por defecto.
    - estado: ENUM('Programada', 'Completada', 'Cancelada'), NOT NULL.
- **Tabla Asistencia:** Como se introdujo anteriormente, esta tabla de unión es el registro de interacción clave. Vincula a un Miembro con una ClaseProgramada específica, documentando su participación.[11]
    - asistencia_id: INT, Clave Primaria (PK).
    - clase_programada_id: INT, Clave Foránea (FK) a ClasesProgramadas(clase_programada_id).
    - miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id).
    - fecha_reserva: TIMESTAMP, DEFAULT CURRENT_TIMESTAMP.
    - fecha_check_in: DATETIME, NULL. Se rellena cuando el miembro llega físicamente.
    - estado: ENUM('Reservado', 'Asistió', 'Canceló', 'No se presentó'), NOT NULL.

3.3. La Distinción entre lo Abstracto y lo Concreto en la Operación

La decisión de diseño de separar Clases (el concepto) de ClasesProgramadas (la instancia) es fundamental para la flexibilidad y robustez del sistema de gestión. Un diseño menos experimentado podría intentar fusionar ambas ideas en una sola tabla, lo que generaría problemas operativos complejos y una estructura de datos rígida.

Este enfoque de dos tablas permite gestionar con elegancia una variedad de escenarios del mundo real:

- **Gestión de Calendario:** Se pueden programar clases recurrentes (ej. "Pilates Mat" todos los martes a las 18:00) creando múltiples entradas en ClasesProgramadas que apuntan a la misma entrada en Clases.
- **Instructores Sustitutos:** Si el instructor habitual de una clase se enferma, solo es necesario actualizar el campo instructor_id en la fila correspondiente de ClasesProgramadas para esa fecha específica, sin afectar a las demás clases programadas ni a la definición de la clase en sí.

- **Eventos y Talleres Especiales:** Se puede crear un nuevo tipo de clase en la tabla Clases (ej. "Taller de Inversiones de Yoga") y luego programar una o varias instancias de este en ClasesProgramadas para fechas concretas.
- **Cancelaciones:** Si una clase específica debe ser cancelada (ej. por falta de inscritos o un imprevisto), simplemente se cambia el estado en ClasesProgramadas a 'Cancelada'. La definición de la clase en el catálogo (Clases) permanece intacta y disponible para futuras programaciones.

Más allá de la eficiencia operativa, esta estructura granular crea los datos necesarios para un análisis profundo del negocio. La tabla Asistencia une a un Miembro con una ClasePrograma específica, creando un log detallado del comportamiento del cliente. Este registro permite analizar la popularidad de clases específicas, instructores, y horarios, así como el nivel de compromiso de cada miembro. Esta información es la materia prima indispensable para cualquier modelo de IA que busque optimizar horarios, personalizar recomendaciones o predecir el comportamiento del cliente.[13]

Parte IV: El Motor Financiero – Una Visión Holística de Ingresos y Gastos

Esta sección está diseñada para abordar directamente el requisito fundamental de obtener un balance financiero de manera rápida y precisa. Para lograrlo, la base de datos debe capturar de forma exhaustiva y estructurada todas las transacciones financieras, tanto los ingresos como los gastos. Una contabilidad clara y detallada es la base para la toma de decisiones estratégicas y la evaluación de la salud financiera del negocio.

4.1. Gestión de Ingresos (Ingresos)

La tabla Pagos es el libro mayor de todos los ingresos. Cada fila representa una entrada de dinero en el negocio. Es crucial que esta tabla no solo registre el monto y la fecha, sino que también vincule cada ingreso a su origen (un miembro, una suscripción, un producto), permitiendo un análisis de rentabilidad granular.[1]

- **Tabla Pagos:**
  - pago_id: INT, Clave Primaria (PK).
  - miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id), NULLABLE. La mayoría de los pagos estarán vinculados a un miembro, pero podría haber ventas anónimas.
  - suscripcion_id: INT, Clave Foránea (FK) a SuscripcionesMiembro(suscripcion_id), NULLABLE. Vincula el pago a una membresía específica.
  - venta_producto_id: INT, Clave Foránea (FK) a una futura tabla VentasProducto, NULLABLE. Para registrar la venta de artículos como bebidas, ropa o suplementos.
  - monto: DECIMAL(10, 2), NOT NULL.
  - fecha_pago: TIMESTAMP, NOT NULL, DEFAULT CURRENT_TIMESTAMP.
  - metodo_pago: ENUM('Tarjeta de Crédito', 'Transferencia', 'Efectivo', 'Otro'), NOT NULL.
  - concepto: VARCHAR(255), NOT NULL. Una descripción clara de la transacción (ej. 'Renovación Membresía Mensual', 'Compra botella de agua').
  - id_transaccion_externa: VARCHAR(255). Para almacenar el ID de la transacción de la pasarela de pago (ej. Stripe, Square).[31]

4.2. Gestión de Egresos (Gastos)

Para una contabilidad completa, el seguimiento de los gastos es tan importante como el de los ingresos. La estructura propuesta utiliza dos tablas para organizar los egresos de manera eficiente y facilitar la reportería.

- **Tabla CategoriasGasto:** Esta es una tabla de consulta fundamental que permite clasificar cada gasto. Una categorización consistente es la clave para generar informes financieros útiles y comprensibles.[32]
  - categoria_gasto_id: INT, Clave Primaria (PK).
  - nombre_categoria: VARCHAR(100), UNIQUE, NOT NULL. Ejemplos: 'Alquiler', 'Nómina', 'Marketing', 'Suministros', 'Mantenimiento de Equipo'.
  - tipo_gasto: ENUM('Fijo', 'Variable'). Ayuda a analizar la estructura de costes.
- **Tabla Gastos:** Esta tabla registra cada gasto individual.
  - gasto_id: INT, Clave Primaria (PK).
  - categoria_gasto_id: INT, Clave Foránea (FK) a CategoriasGasto(categoria_gasto_id).

- monto: DECIMAL(10, 2), NOT NULL.
- fecha_gasto: DATE, NOT NULL.
- proveedor: VARCHAR(150). El destinatario del pago.
- descripcion: TEXT. Detalles específicos del gasto.
- personal_id_relacionado: INT, Clave Foránea (FK) a Personal(personal_id), NULLABLE. Para vincular gastos de nómina a un empleado específico.

4.3. Categorías Comunes de Gastos Operativos de un Gimnasio

Para acelerar la configuración del sistema financiero y asegurar que no se omita ninguna partida de gasto importante, se proporciona a continuación una tabla con categorías de gastos operativos comunes, sintetizada a partir de análisis de la industria del fitness.[32] Implementar estas categorías desde el principio garantiza un seguimiento financiero robusto y completo, transformando una tarea de configuración potencialmente tediosa en un proceso estructurado. Esto es crucial para cumplir con el objetivo de generar un "balance rápido", ya que la precisión del balance depende de la exhaustividad de los datos de entrada.

Tipo de Gasto	Categoría de Gasto	Descripción y Ejemplos
Costos Fijos	Alquiler / Hipoteca	Pagos mensuales por el local comercial.
	Seguros	Responsabilidad civil, propiedad, compensación de trabajadores.[33]
	Salarios Base	Sueldos del personal administrativo y gerencial con contrato fijo.
	Licencias de Software	Costos recurrentes de software de gestión, contabilidad, marketing, etc..[32]
	Impuestos y Licencias	Impuestos sobre la propiedad,

		licencias de negocio, permisos sanitarios.[37]
	Licencias de Música	Tarifas a sociedades de derechos de autor (ej. BMI, ASCAP) para reproducir música legalmente.[35]
	Servicios Públicos Base	Costo mínimo de electricidad, agua, gas, internet.
Costos Variables	Nómina por Horas/Clases	Salarios de instructores y entrenadores pagados por clase o por hora.
	Suministros y Servicios	Consumo de electricidad, agua, productos de limpieza, toallas.
	Marketing y Publicidad	Campañas en redes sociales, publicidad online, material impreso.
	Mantenimiento de Equipos	Reparaciones, revisiones periódicas, piezas de repuesto.
	Comisiones de Pago	Tarifas cobradas por las pasarelas de pago por cada transacción con tarjeta.
	Inventario y Suministros	Compra de productos para la venta (bebidas, suplementos, ropa).

4.4. Generación del Balance: Vistas (Views) de SQL para Reportería Financiera

Con los ingresos y los gastos debidamente registrados y categorizados, el paso final para habilitar un "balance rápido" es crear una herramienta de consulta simplificada. En SQL, una VIEW (Vista) es una tabla virtual basada en el resultado de una consulta

SQL. Actúa como un atajo guardado, permitiendo ejecutar consultas complejas de manera sencilla.[11]

Se propone la creación de una vista llamada VistaGananciasYPerdidas. Esta vista pre-calculará y unirá los datos de las tablas Pagos y Gastos, agregándolos por período (mes y año).

La lógica de la vista sería la siguiente:

1. Sumar todos los Pagos por mes y año para obtener los IngresosTotales.
2. Sumar todos los Gastos por mes y año para obtener los GastosTotales.
3. Unir estos dos resultados y calcular la GananciaNeta (IngresosTotales - GastosTotales).

Una vez creada la vista, obtener el balance para un período específico sería tan simple como ejecutar una consulta:
SELECT * FROM VistaGananciasYPerdidas WHERE mes = 10 AND anio = 2024;
Esta consulta devolvería instantáneamente un resumen financiero claro y conciso para octubre de 2024, cumpliendo con el requisito principal de una reportería financiera ágil y accesible directamente desde la base de datos.

Parte V: A Prueba de Futuro – Arquitectura para la Inteligencia Artificial

Esta sección final y más estratégica se enfoca directamente en el objetivo de diseñar una base de datos que no solo gestione el presente, sino que esté explícitamente preparada para ser el motor de futuras aplicaciones de Inteligencia Artificial. La arquitectura propuesta va más allá del simple registro de transacciones para capturar la riqueza de datos de comportamiento, preferencias y metas que son esenciales para la IA.

5.1. Por Qué una Buena Base de Datos es el Prerrequisito para la IA

Es fundamental entender que los sistemas de Inteligencia Artificial, especialmente los modelos de aprendizaje automático (*machine learning*), no son mágicos. Su

capacidad para generar predicciones, recomendaciones y análisis se basa enteramente en el reconocimiento de patrones complejos dentro de los datos con los que son entrenados. La máxima "Garbage In, Garbage Out" (basura entra, basura sale) es la ley fundamental en este campo: la calidad de los resultados de un modelo de IA está directamente limitada por la calidad, la cantidad y la riqueza de sus datos de entrada.[38]

Por lo tanto, una base de datos bien estructurada, normalizada, limpia y, sobre todo, rica en datos contextuales, no es solo un complemento para una estrategia de IA; es el activo más valioso y el prerrequisito indispensable. Sin datos de alta calidad, cualquier esfuerzo por implementar IA será ineficaz y costoso. El diseño que se presenta a continuación está concebido para ser esa fuente de datos de alta calidad.

5.2. Extendiendo el Esquema para la Recopilación de Datos de IA

El cambio de paradigma de una base de datos transaccional a una preparada para la IA reside en pasar de registrar *qué* sucedió (una transacción) a registrar el *contexto completo* de lo que sucedió (un comportamiento). Una base de datos estándar registra que un miembro pagó una cuota y asistió a una clase. Una base de datos preparada para IA va mucho más allá: registra *qué clase* específica era, *cómo* se desempeñó el miembro en esa clase (ejercicios, series, repeticiones), cuál era su *meta* al unirse al gimnasio, y si la clase cumplió sus *expectativas*.

Este enfoque transforma los datos de ser bidimensionales (quién, qué) a ser multidimensionales (quién, qué, cómo, por qué, con qué sentimiento). Esta riqueza contextual proporciona las "características" (*features*) que los modelos de machine learning necesitan para hacer predicciones y recomendaciones verdaderamente personalizadas y valiosas. Por ejemplo, un modelo de predicción de abandono (churn) no se limitará a mirar la fecha_ultima_asistencia. Buscará patrones mucho más sutiles y complejos, como: "Los miembros cuyo objetivo principal es 'Perder Peso', que no han registrado un ejercicio de tipo 'Cardio' en las últimas 3 semanas y cuya frecuencia de asistencia ha disminuido un 20% con respecto a su media histórica, tienen un 85% de probabilidad de no renovar su suscripción el próximo mes". Este nivel de análisis predictivo es imposible sin las tablas granulares y centradas en el comportamiento que se proponen a continuación.

Para capturar esta información, se proponen las siguientes tablas adicionales:

- **Tabla MetasMiembro:** Captura los objetivos y motivaciones de cada miembro, el "porqué" de su viaje de fitness.
    - meta_id: INT, Clave Primaria (PK).
    - miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id).
    - tipo_meta: ENUM('Pérdida de peso', 'Ganancia muscular', 'Mejora de flexibilidad', 'Aumento de resistencia', 'Bienestar general').
    - valor_objetivo: VARCHAR(50). Ej: '10 kg', 'Correr 5 km', 'Tocar los pies'.
    - fecha_inicio: DATE.
    - fecha_objetivo: DATE, NULLABLE.
    - estado_meta: ENUM('Activa', 'Alcanzada', 'Abandonada').[20]
- **Tabla LogsEntrenamiento:** Esta es la tabla más granular y potente para la IA. Registra cada detalle de los entrenamientos de un miembro.
    - log_id: INT, Clave Primaria (PK).
    - miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id).
    - asistencia_id: INT, Clave Foránea (FK) a Asistencia(asistencia_id), NULLABLE. Vincula el ejercicio a una clase o sesión específica.
    - fecha_ejercicio: DATE.
    - nombre_ejercicio: VARCHAR(100). Ej: 'Sentadilla con barra', 'Plancha'.
    - series: INT.
    - repeticiones: INT.
    - peso_kg: DECIMAL(6, 2).
    - duracion_segundos: INT.
    - distancia_km: DECIMAL(6, 2).
    - notas_personales: TEXT.[42]
- **Tabla FeedbackMiembro:** Captura la opinión y el sentimiento del cliente, datos cualitativos de gran valor.
    - feedback_id: INT, Clave Primaria (PK).
    - miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id).
    - clase_programada_id: INT, Clave Foránea (FK) a ClasesProgramadas(clase_programada_id), NULLABLE.
    - instructor_id: INT, Clave Foránea (FK) a Personal(personal_id), NULLABLE.
    - calificacion: INT (de 1 a 5).
    - comentario: TEXT.
    - fecha_feedback: TIMESTAMP, DEFAULT CURRENT_TIMESTAMP.
- **Tablas de PreferenciasUsuario:** Para almacenar configuraciones personalizadas, se implementará el elegante diseño default/override.[44] Este enfoque es altamente escalable y evita tener que alterar la estructura de la tabla de usuarios cada vez que se añade una nueva preferencia.
    - **Tabla PreferenciasDefault:**

        - nombre_preferencia: VARCHAR(100), Clave Primaria (PK).
        - valor_default: TEXT.
        - tipo_dato: VARCHAR(50). Ej: 'BOOLEAN', 'STRING'.
        - descripcion: TEXT.
    - **Tabla PreferenciasUsuarioOverride:**
        - miembro_id: INT, Clave Foránea (FK) a Miembros(miembro_id).
        - nombre_preferencia: VARCHAR(100), Clave Foránea (FK) a PreferenciasDefault(nombre_preferencia).
        - valor_personalizado: TEXT.
        - (Clave Primaria compuesta por miembro_id y nombre_preferencia).

5.3. El ERD Preparado para IA: El Plan Maestro Completo

El Diagrama de Entidad-Relación final integra estas nuevas tablas centradas en el comportamiento con el núcleo transaccional y operativo del sistema. El resultado es un ecosistema de datos cohesivo donde las transacciones (Pagos, Suscripciones) se enriquecen con datos de comportamiento (Asistencia, LogsEntrenamiento), motivaciones (MetasMiembro) y sentimientos (FeedbackMiembro). Este ERD completo representa el plano final de una base de datos diseñada no solo para la eficiencia actual, sino para la inteligencia futura.

5.4. Casos de Uso de IA Habilitados por este Esquema

La riqueza y estructura de este modelo de datos habilitan una amplia gama de aplicaciones de IA que pueden transformar la experiencia del cliente y la eficiencia del negocio:

- **Sistemas de Recomendación Personalizados:** Utilizando los datos de LogsEntrenamiento, MetasMiembro, Asistencia y FeedbackMiembro, un modelo de IA puede sugerir a los miembros nuevas clases que se alineen con sus objetivos y preferencias, proponer ejercicios alternativos si detecta un estancamiento en su progreso, o incluso recomendar instructores cuyo estilo haya sido bien calificado por miembros con perfiles similares.[20]
- **Predicción de Abandono (Churn Prediction):** Un modelo de machine learning

puede ser entrenado para analizar patrones complejos en los datos de Asistencia, SuscripcionesMiembro y FeedbackMiembro. Puede identificar a los miembros que están en alto riesgo de no renovar su membresía, permitiendo al personal tomar acciones proactivas de retención, como ofrecer una sesión de entrenamiento personal gratuita o un descuento en la renovación.[46]

- **Optimización de Horarios y Precios:** Al analizar los datos agregados de Asistencia, la IA puede identificar las clases, horarios e instructores más populares. Esto permite optimizar la parrilla de clases para maximizar la ocupación y la satisfacción. Incluso podría sentar las bases para un sistema de precios dinámicos, donde las clases con menor demanda tengan un precio ligeramente inferior para incentivar la asistencia.
- **Gestión Inteligente de Equipos:** El análisis de los datos en LogsEntrenamiento puede revelar qué máquinas y equipos se utilizan con más frecuencia. Esto permite planificar de manera proactiva el mantenimiento preventivo, anticipar la necesidad de reemplazo y tomar decisiones de compra de nuevos equipos basadas en la demanda real de los usuarios, en lugar de en la intuición.

Conclusión y Script de Implementación Completo

Resumen Arquitectónico

Este informe ha detallado la arquitectura de una base de datos SQL robusta, escalable y preparada para el futuro para un estudio de fitness o pilates. Las decisiones de diseño clave se han basado en principios fundamentales para garantizar la integridad, la flexibilidad y, sobre todo, la capacidad de generar inteligencia de negocio. Se ha priorizado la **normalización** para eliminar la redundancia y asegurar la consistencia de los datos. Se ha implementado una **separación clara de conceptos**, como la distinción entre catálogos (PlanesMembresia, Clases) e instancias transaccionales (SuscripcionesMiembro, ClasesProgramadas), lo que permite una gestión flexible y un registro histórico completo. Finalmente, el diseño se ha extendido deliberadamente para la **recopilación de datos ricos en contexto**, capturando no solo transacciones, sino también comportamientos, metas y preferencias, sentando así una base sólida para la

futura integración con sistemas de Inteligencia Artificial.

Script SQL Final

A continuación, se proporciona el script SQL completo y comentado para crear todas las tablas descritas en este informe. El script está ordenado para respetar las dependencias de las claves foráneas. Contiene la definición de todas las tablas, claves primarias, claves foráneas (con reglas ON DELETE y ON UPDATE para mantener la integridad referencial), y restricciones como NOT NULL y UNIQUE. Este script es el entregable final y tangible que le permitirá construir la base de datos de su sistema de gestión.

SQL

```sql
-- Arquitectura de Base de Datos para Estudio de Fitness Moderno
-- Versión 1.0
-- Este script crea la estructura completa de la base de datos,
-- incluyendo tablas para gestión de miembros, personal, operaciones, finanzas y datos para IA.

-- PARTE I & II: ESTRUCTURAS FUNDAMENTALES - MIEMBROS Y SUSCRIPCIONES

-- Tabla para los tipos de planes de membresía (Catálogo)
CREATE TABLE PlanesMembresia (
    plan_id INT AUTO_INCREMENT PRIMARY KEY,
    nombre_plan VARCHAR(150) NOT NULL UNIQUE,
    descripcion TEXT,
    precio DECIMAL(10, 2) NOT NULL,
    duracion_dias INT COMMENT 'Para planes con duración definida (ej. 30, 365)',
    numero_clases INT COMMENT 'Para planes tipo bono de clases',
    tipo_plan ENUM('Solo Pilates', 'Solo Gimnasio', 'Completo', 'Bono') NOT NULL,
    activo BOOLEAN NOT NULL DEFAULT TRUE
);

-- Tabla central de Miembros (CRM)
CREATE TABLE Miembros (
```

```
    miembro_id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    apellido VARCHAR(100) NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE,
    telefono VARCHAR(20),
    fecha_nacimiento DATE,
    fecha_registro TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    estado ENUM('activo', 'inactivo', 'congelado', 'potencial') NOT NULL DEFAULT 'potencial',
    direccion VARCHAR(255),
    ciudad VARCHAR(100),
    codigo_postal VARCHAR(20),
    notas_administrativas TEXT
);

-- Tabla de unión para las suscripciones de los miembros
CREATE TABLE SuscripcionesMiembro (
    suscripcion_id INT AUTO_INCREMENT PRIMARY KEY,
    miembro_id INT NOT NULL,
    plan_id INT NOT NULL,
    fecha_inicio DATE NOT NULL,
    fecha_fin DATE NOT NULL,
    estado ENUM('activa', 'vencida', 'cancelada', 'futura') NOT NULL,
    precio_pagado DECIMAL(10, 2) NOT NULL,
    clases_restantes INT,
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE CASCADE,
    FOREIGN KEY (plan_id) REFERENCES PlanesMembresia(plan_id) ON DELETE RESTRICT
);

-- PARTE III: EXCELENCIA OPERATIVA - PERSONAL, CLASES Y ASISTENCIA

-- Tabla para los roles del personal
CREATE TABLE RolesPersonal (
    rol_id INT AUTO_INCREMENT PRIMARY KEY,
    nombre_rol VARCHAR(100) NOT NULL UNIQUE,
    descripcion_rol TEXT
);

-- Tabla de Personal
CREATE TABLE Personal (
    personal_id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nombre VARCHAR(100) NOT NULL,
    apellido VARCHAR(100) NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE,
    telefono VARCHAR(20),
    fecha_contratacion DATE NOT NULL,
    rol_id INT,
    especialidades TEXT,
    activo BOOLEAN NOT NULL DEFAULT TRUE,
    FOREIGN KEY (rol_id) REFERENCES RolesPersonal(rol_id) ON DELETE SET NULL
);

-- Tabla para los horarios de trabajo del personal
CREATE TABLE HorariosPersonal (
    horario_id INT AUTO_INCREMENT PRIMARY KEY,
    personal_id INT NOT NULL,
    inicio_turno DATETIME NOT NULL,
    fin_turno DATETIME NOT NULL,
    notas TEXT,
    FOREIGN KEY (personal_id) REFERENCES Personal(personal_id) ON DELETE CASCADE
);

-- Tabla para los tipos de clases (Catálogo)
CREATE TABLE Clases (
    clase_id INT AUTO_INCREMENT PRIMARY KEY,
    nombre_clase VARCHAR(150) NOT NULL UNIQUE,
    descripcion TEXT,
    duracion_minutos_default INT,
    capacidad_maxima_default INT
);

-- Tabla para las clases programadas en el calendario
CREATE TABLE ClasesProgramadas (
    clase_programada_id INT AUTO_INCREMENT PRIMARY KEY,
    clase_id INT NOT NULL,
    instructor_id INT,
    fecha_hora_inicio DATETIME NOT NULL,
    fecha_hora_fin DATETIME NOT NULL,
    sala_o_ubicacion VARCHAR(100),
    capacidad_actual INT,
```

```
    estado ENUM('Programada', 'Completada', 'Cancelada') NOT NULL DEFAULT 'Programada',
    FOREIGN KEY (clase_id) REFERENCES Clases(clase_id) ON DELETE RESTRICT,
    FOREIGN KEY (instructor_id) REFERENCES Personal(personal_id) ON DELETE SET NULL
);

-- Tabla de Asistencia (Junction Table)
CREATE TABLE Asistencia (
    asistencia_id INT AUTO_INCREMENT PRIMARY KEY,
    clase_programada_id INT NOT NULL,
    miembro_id INT NOT NULL,
    fecha_reserva TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    fecha_check_in DATETIME,
    estado ENUM('Reservado', 'Asistió', 'Canceló', 'No se presentó') NOT NULL DEFAULT 'Reservado',
    FOREIGN KEY (clase_programada_id) REFERENCES
ClasesProgramadas(clase_programada_id) ON DELETE CASCADE,
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE CASCADE,
    UNIQUE (clase_programada_id, miembro_id) -- Un miembro no puede reservar dos veces la
misma clase
);

-- PARTE IV: MOTOR FINANCIERO - INGRESOS Y GASTOS

-- Tabla de Pagos (Ingresos)
CREATE TABLE Pagos (
    pago_id INT AUTO_INCREMENT PRIMARY KEY,
    miembro_id INT,
    suscripcion_id INT,
    monto DECIMAL(10, 2) NOT NULL,
    fecha_pago TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    metodo_pago ENUM('Tarjeta de Crédito', 'Transferencia', 'Efectivo', 'Otro') NOT NULL,
    concepto VARCHAR(255) NOT NULL,
    id_transaccion_externa VARCHAR(255),
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE SET NULL,
    FOREIGN KEY (suscripcion_id) REFERENCES SuscripcionesMiembro(suscripcion_id) ON
DELETE SET NULL
);

-- Tabla de categorías de gastos
CREATE TABLE CategoriasGasto (
    categoria_gasto_id INT AUTO_INCREMENT PRIMARY KEY,
```

```
    nombre_categoria VARCHAR(100) NOT NULL UNIQUE,
    tipo_gasto ENUM('Fijo', 'Variable') NOT NULL
);

-- Tabla de Gastos (Egresos)
CREATE TABLE Gastos (
    gasto_id INT AUTO_INCREMENT PRIMARY KEY,
    categoria_gasto_id INT NOT NULL,
    monto DECIMAL(10, 2) NOT NULL,
    fecha_gasto DATE NOT NULL,
    proveedor VARCHAR(150),
    descripcion TEXT,
    personal_id_relacionado INT,
    FOREIGN KEY (categoria_gasto_id) REFERENCES CategoriasGasto(categoria_gasto_id) ON DELETE RESTRICT,
    FOREIGN KEY (personal_id_relacionado) REFERENCES Personal(personal_id) ON DELETE SET NULL
);

-- PARTE V: A PRUEBA DE FUTURO - DATOS PARA IA

-- Tabla para las metas de los miembros
CREATE TABLE MetasMiembro (
    meta_id INT AUTO_INCREMENT PRIMARY KEY,
    miembro_id INT NOT NULL,
    tipo_meta ENUM('Pérdida de peso', 'Ganancia muscular', 'Mejora de flexibilidad', 'Aumento de resistencia', 'Bienestar general') NOT NULL,
    valor_objetivo VARCHAR(50),
    fecha_inicio DATE NOT NULL,
    fecha_objetivo DATE,
    estado_meta ENUM('Activa', 'Alcanzada', 'Abandonada') NOT NULL DEFAULT 'Activa',
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE CASCADE
);

-- Tabla para los logs de entrenamiento detallados
CREATE TABLE LogsEntrenamiento (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    miembro_id INT NOT NULL,
    asistencia_id INT,
    fecha_ejercicio DATE NOT NULL,
```

```
    nombre_ejercicio VARCHAR(100) NOT NULL,
    series INT,
    repeticiones INT,
    peso_kg DECIMAL(6, 2),
    duracion_segundos INT,
    distancia_km DECIMAL(6, 2),
    notas_personales TEXT,
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE CASCADE,
    FOREIGN KEY (asistencia_id) REFERENCES Asistencia(asistencia_id) ON DELETE SET NULL
);

-- Tabla para el feedback de los miembros
CREATE TABLE FeedbackMiembro (
    feedback_id INT AUTO_INCREMENT PRIMARY KEY,
    miembro_id INT NOT NULL,
    clase_programada_id INT,
    instructor_id INT,
    calificacion INT NOT NULL CHECK (calificacion BETWEEN 1 AND 5),
    comentario TEXT,
    fecha_feedback TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE CASCADE,
    FOREIGN KEY (clase_programada_id) REFERENCES
ClasesProgramadas(clase_programada_id) ON DELETE SET NULL,
    FOREIGN KEY (instructor_id) REFERENCES Personal(personal_id) ON DELETE SET NULL
);

-- Tablas para preferencias de usuario (diseño default/override)
CREATE TABLE PreferenciasDefault (
    nombre_preferencia VARCHAR(100) PRIMARY KEY,
    valor_default TEXT NOT NULL,
    tipo_dato VARCHAR(50) NOT NULL,
    descripcion TEXT
);

CREATE TABLE PreferenciasUsuarioOverride (
    miembro_id INT NOT NULL,
    nombre_preferencia VARCHAR(100) NOT NULL,
    valor_personalizado TEXT NOT NULL,
    PRIMARY KEY (miembro_id, nombre_preferencia),
```

```
    FOREIGN KEY (miembro_id) REFERENCES Miembros(miembro_id) ON DELETE CASCADE,
    FOREIGN KEY (nombre_preferencia) REFERENCES
PreferenciasDefault(nombre_preferencia) ON DELETE CASCADE
);

-- Fin del script.
```

Obras citadas

1. How to Build a Database Schema for a Gym Management System? - Tutorials - Back4app, fecha de acceso: julio 24, 2025, https://www.back4app.com/tutorials/how-to-build-a-database-schema-for-a-gym-management-system
2. Database Management for Gym Management System - Prezi, fecha de acceso: julio 24, 2025, https://prezi.com/p/1pdxiq2pwo7v/database-management-for-gym-management-system/
3. Database schema: SQL schema examples and best practices - CockroachDB, fecha de acceso: julio 24, 2025, https://www.cockroachlabs.com/blog/database-schema-beginners-guide/
4. Complete Guide to Database Schema Design - Integrate.io, fecha de acceso: julio 24, 2025, https://www.integrate.io/blog/complete-guide-to-database-schema-design-guide/
5. Fitness Database Template (Free Example) - Exercise.com, fecha de acceso: julio 24, 2025, https://www.exercise.com/grow/fitness-database-template/
6. Membership Database Made Easy: Your Solution for Efficient Management - Raklet, fecha de acceso: julio 24, 2025, https://www.raklet.com/blog/membership-database/
7. Gym Management ER Diagram - Wondershare EdrawMax, fecha de acceso: julio 24, 2025, https://edrawmax.wondershare.com/templates/er-diagram-gym-management.html
8. Gym Management System ER Diagram | Academic Projects, fecha de acceso: julio 24, 2025, https://www.freeprojectz.com/entity-relationship/gym-management-system-er-diagram
9. Gym Database Management System [classic] - Creately, fecha de acceso: julio 24, 2025, https://creately.com/diagram/example/i2gntons1/gym-database-management-system-classic
10. Gym Membership Management System | PDF | Relational Database - Scribd, fecha de acceso: julio 24, 2025, https://www.scribd.com/document/399905122/Gym-membership-Management-

System
11. abdullah-zero9/SQL_Gym-Management-System: This project created an efficient SQL database for Gym Management System software, employing normalized tables and SQL queries for robust data management. - GitHub, fecha de acceso: julio 24, 2025, https://github.com/abdullah-zero9/SQL_Gym-Management-System
12. Gym and Fitness Club Management Software - Daftra, fecha de acceso: julio 24, 2025, https://www.daftra.com/en/gym-fitness-club/
13. FITNESS TRACKING SYSTEM FOR GYMS ADMINISTRATION FRAMEWORK - IRJMETS, fecha de acceso: julio 24, 2025, https://www.irjmets.com/uploadedfiles/paper//issue_4_april_2024/52680/final/fin_irjmets1714503155.pdf
14. Gym Management System Entity Relationship Diagram - YouTube, fecha de acceso: julio 24, 2025, https://www.youtube.com/watch?v=BbLg3gJzSO8
15. Entity-Relationship Diagram (ERD) | Gym and Spa Area Plans | How To Explain Gym Management Using Er Diagram - Conceptdraw.com, fecha de acceso: julio 24, 2025, https://www.conceptdraw.com/examples/how-to-explain-gym-management-using-er-diagram
16. Entity-Relationship Diagram (ERD) | Gym and Spa Area Plans | Gym Management System Erd - Conceptdraw.com, fecha de acceso: julio 24, 2025, https://www.conceptdraw.com/examples/gym-management-system-erd
17. Gym Management Mini Project | PDF | Computer Programming | Data Model - Scribd, fecha de acceso: julio 24, 2025, https://www.scribd.com/document/637236553/Gym-Management-Mini-Project
18. Create a Membership Database In 3 Steps - Five, fecha de acceso: julio 24, 2025, https://five.co/blog/create-a-membership-database-in-3-steps/
19. Mastering Gym Membership Database Management: A Comprehensive Guide for Gym Owners - Felvedere - Fitness Flow, fecha de acceso: julio 24, 2025, https://www.joinfitnessflow.com/blog/mastering-gym-membership-database-management-a-comprehensive-guide-for-gym-owners
20. Ai Driven Personalized Fitness Analytics Database, fecha de acceso: julio 24, 2025, https://www.databasesample.com/database/ai-driven-personalized-fitness-analytics-database
21. Free Excel Template for a Member Database - Join It, fecha de acceso: julio 24, 2025, https://joinit.com/blog/free-excel-template-for-a-member-database
22. The Complete Membership Database Selection Guide - Wild Apricot, fecha de acceso: julio 24, 2025, https://www.wildapricot.com/blog/membership-database-selection-guide
23. Employee Management System Database Structure and Schema, fecha de acceso: julio 24, 2025, https://www.databasesample.com/database/employee-management-system-database
24. Database schema for Roles and permissions - best approach - Laracasts, fecha de acceso: julio 24, 2025, https://laracasts.com/discuss/channels/site-improvements/database-schema-for-

roles-and-permissions-best-approach
25. Best user role permissions database design practice? [closed] - Stack Overflow, fecha de acceso: julio 24, 2025, https://stackoverflow.com/questions/46016139/best-user-role-permissions-database-design-practice
26. How to Store Employees' Schedules in a Database - Vertabelo, fecha de acceso: julio 24, 2025, https://vertabelo.com/blog/how-to-store-employees-schedules-in-a-database/
27. database design - Scheduling Employees - what data structure to use? - Stack Overflow, fecha de acceso: julio 24, 2025, https://stackoverflow.com/questions/1634248/scheduling-employees-what-data-structure-to-use
28. Gym Appointments database schema - Stack Overflow, fecha de acceso: julio 24, 2025, https://stackoverflow.com/questions/34658353/gym-appointments-database-schema
29. Gym Attendance Sheet Template (PDF, DOC, XLS – FREE) | Exercise.com, fecha de acceso: julio 24, 2025, https://www.exercise.com/grow/gym-attendance-sheet-template/
30. Top Attendance Management Software For Fitness Studios - Kenko's, fecha de acceso: julio 24, 2025, https://www.gokenko.com/articles/fitness-studio-attendance-management-software
31. Top Gym Accounting Software for Gyms | Gymdesk, fecha de acceso: julio 24, 2025, https://gymdesk.com/blog/top-gym-accounting-software-gyms/
32. What is the monthly cost of running a gym? - Exercise.com, fecha de acceso: julio 24, 2025, https://www.exercise.com/grow/what-is-the-monthly-cost-of-running-a-gym/
33. Gym Expenses: How Much Does it Cost to Run a Gym? - Arbox, fecha de acceso: julio 24, 2025, https://www.arboxapp.com/blog/gym-expenses-guide
34. Gym expenses : r/Entrepreneur - Reddit, fecha de acceso: julio 24, 2025, https://www.reddit.com/r/Entrepreneur/comments/14eznlg/gym_expenses/
35. How Much Does it Cost to Open a Gym? (A Breakdown) - Hevy Coach, fecha de acceso: julio 24, 2025, https://hevycoach.com/cost-to-open-a-gym/
36. How Much Does Opening a Gym Cost? - PushPress, fecha de acceso: julio 24, 2025, https://www.pushpress.com/blog/how-much-does-it-cost-to-start-a-fitness-or-gym-business
37. How To Create a Gym Budget | 9 Steps - Fitness On Demand, fecha de acceso: julio 24, 2025, https://www.fitnessondemand247.com/news/gym-budget
38. How to Create a Workout Tracker with AI - Bricks, fecha de acceso: julio 24, 2025, https://www.thebricks.com/resources/how-to-create-a-workout-tracker-with-ai
39. AI Fitness App with Personalized Recommendation Function - InData Labs, fecha de acceso: julio 24, 2025, https://indatalabs.com/resources/ai-fitness-app
40. How to Design a Database for Health and Fitness Tracking Applications -

GeeksforGeeks, fecha de acceso: julio 24, 2025, https://www.geeksforgeeks.org/dbms/how-to-design-a-database-for-health-and-fitness-tracking-applications/
41. Revolutionizing Fitness with Mobile Apps - Number Analytics, fecha de acceso: julio 24, 2025, https://www.numberanalytics.com/blog/ultimate-guide-mobile-apps-exercise-science
42. Database Schema for a Gym Exercise Log App - Stack Overflow, fecha de acceso: julio 24, 2025, https://stackoverflow.com/questions/54220956/database-schema-for-a-gym-exercise-log-app
43. How to Build a Database Schema for a Fitness Tracking Application? - Tutorials - Back4app, fecha de acceso: julio 24, 2025, https://www.back4app.com/tutorials/how-to-build-a-database-schema-for-a-fitness-tracking-application
44. default/override: An Elegant Schema for User Settings - Double Finance, fecha de acceso: julio 24, 2025, https://double.finance/blog/default_override
45. Building a personalized workout recommender using content-based filtering with ML.NET, fecha de acceso: julio 24, 2025, https://dev.to/gbengelebs/building-a-personalized-workout-recommender-with-mlnet-a-step-by-step-guide-1nji
46. AI for Gyms 101: Enhancing Member Experience - Personalized Fitness with AI | Keepme, fecha de acceso: julio 24, 2025, https://www.keepme.ai/blog/ai-for-gyms-101-enhancing-member-experience-personalized-fitness-with-ai/
47. Building an AI-Powered Exercise Recommendation System | by Asfand Yar - Medium, fecha de acceso: julio 24, 2025, https://medium.com/@asfandyar5133/building-an-ai-powered-exercise-recommendation-system-9852fb84335d