Licenciatura En Ciencias Computacionales

Sexto Semestre Grupo 2

Nombre de la Materia:

Bases de Datos Distribuidas

Reporte del Proyecto

Nombre del Profesor

Eduardo Cornejo Velazquez

SIMIFIT

Equipo: Drop Database

Integrantes:

Eduardo Vázquez Bonilla

Kevin Axel Chávez Quiroz

Fernando Hernandez Morales

César Martínez Andrade

Braulio Adrian Perez Orta

Jesús Romo Guzmán

Sebastián Trejo Muñoz

1. Índice

Cronograma

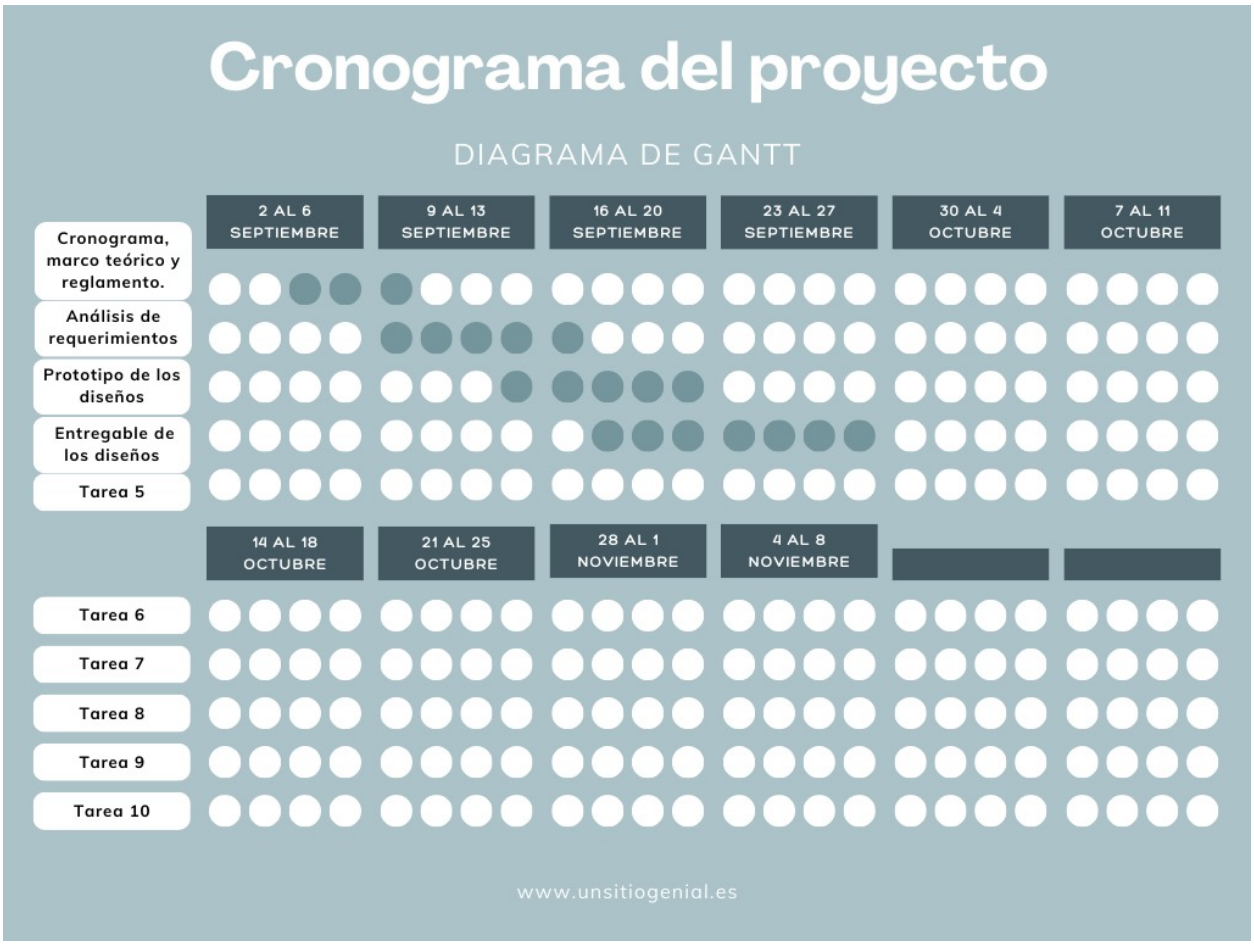


Figure 1: Cronograma de actividades.

Introducción

En este proyecto se realizará una Base de Datos Distribuida Homogénea con la finalidad de llevar el control de un gimnasio, de igual manera se espera diseñar un sistema en java con el cual se pueda gestionar la base de datos, cabe mencionar que, no solo se hará desde una computadora, si no, desde 2 de manera simultánea para poder aplicar la base de datos distribuida La finalidad de este proyecto es emplear base de datos distribuida en pro de aplicar todos los fundamentos de base de datos distribuida para llevar a cabo y observar todas los beneficios de la misma.

Se utilizará una metodología SCRUM con registro de cada avance por medio de versiones del proyecto, esperando tener una buena coordinación del equipo y de igual manera un avance colaborativo y eficiente.

Se llevará a cabo un estudio de campo a fin de tener una base de datos eficiente y adecuada para el negocio a gestionar (gimnasio) y añadiendo como mínimo una propuesta de una idea innovadora a favor del negocio.

Planteamiento del problema

El problema consiste en el manejo de datos e información de múltiples gimnasios y que haya comunicación entre ellas debido a que, es lo mas optimo para el cliente el tener su información siempre actualizada y disponible y el hecho de no contar con una BDD afecta en la transparencia de datos y la actualizaciónes de los mismos.

De manera que, buscamos el diseño adecuado para este caso, en el cual considerando todo lo anterior logrará ser una base de datos que cumpla con todos los requerimientos de nuestro cliente, cabe destacar que, La base de datos distribuida es Homogénea, entonces todos los DBMS serán iguales y por consecuencia el diseño también lo será.

Objetivo general

El objetivo principal es implementar las base de datos en 2 computadoras que funcionan simultáneamente y mantenga comunicación entre sí, asimismo, crear un diseño óptimo de la BDD para que gestione bien los datos de los gimnasios, sin dejar de lado las interfaces de las terminales, que si bien el objetivo principal es el diseño de la base de datos distribuida, se espera mejorar, optimizar e implementar un buen software para manejar las Bases de Datos.

Se cuenta con los recursos necesarios para llevar a cabo el proyecto, un equipo coordinado, responsable y que ven por el bien del proyecto, también por potencia de hardware no habrá inconvenientes ni mucho menos de tiempo disponible.

El progreso del proyecto es medible gracias a la metodología que estamos ocupando (SCRUM) y al software que hemos implementado como lo es github y LATex, que si bien no son explicitamente necesarios para el desarrollo de la base de datos distribuida, son software de mucha ayuda, para el control de versiones del proyecto.

Objetivo general

1. Registrar y gestionar la información de los clientes, el cual se creara un sistema para almacenar y actualizar los datos personales de los clientes o socios.
2. Controlar las suscripciones y pagos, implementando un sistema de seguimiento de pagos, donde registre los pagos realizados y generar advertencias para pagos pendientes.
3. Administrar el inventario de productos en venta, como suplementos, proteínas, pre-entreno, etc., crear un inventario para registrar el control de los productos, ofrece el gimnasio.
4. Registrar y gestionar la información de los empleados del gimnasio, el cual se realizará un sistema para almacenar información esencial de los empleados.
5. Administración de equipos para registrar y monitorear los equipos del gimnasio, como el estado, la ubicación y su mantenimiento.
6. Asegurar la escalabilidad del sistema, diseñando una base de datos y la infraestructura para soportar el aumento en números de clientes y en el aumento de datos sin comprometer el rendimiento, implementando prácticas de optimización y monitoreo continuo para que se pueda adaptar a expansiones y demandas futuras.
7. Mejorar la disponibilidad y fiabilidad de los datos, para que en caso de un fallo los datos sigan siendo accesibles.

Justificación

Actualmente, los gimnasios presentan desafíos para gestionar grandes volúmenes de datos, los cuales están relacionados con sus operaciones que se realizan diariamente, como información de los empleados, información de los clientes, inventario, pagos, entre otros más. La implementación para esta base de datos distribuida ofrece una solución eficiente y con escalabilidad para manejar los datos de una manera efectiva.

A medida que el gimnasio va creciendo y expandiéndose, la cantidad de datos también aumenta, permitiendo añadir nuevos nodos para tener una mejora en la disponibilidad y fiabilidad de los datos, ya que si ocurriera un fallo, los datos seguirán siendo accesibles porque al distribuir los datos en varios nodos, esto reduciría la carga en un solo servidor y genera una optimización al sistema.

Alcances y limitaciones

La base de datos distribuida ofrece sus alcances para el negocio, así como sus limitaciones.

Al estar los datos en múltiples nodos, la base de datos podrá seguir funcionando incluso si un nodo falla, dando una disponibilidad y redundancia en los datos que se almacenan en el sistema. Al manejar un volumen mayor de datos, su escalabilidad será más fácil, ya que se podrán añadir nuevos nodos, esto sin afectar el rendimiento del sistema, obteniendo un acceso más rápido y gestionando las diversas áreas del gimnasio, obteniendo una flexibilidad en la integración de diferentes tipos de datos y sistemas.

Como limitaciones nos encontramos con la configuración y el diseño de la base de datos distribuida, ya que esta es más compleja que una base de datos centralizada. Al estar dependiendo de la Red para que todos los nodos están en lugares diferentes y contar con una red mala calidad y estabilidad, podría ser un problema afectando a los rendimientos de la base de datos.

Marco teórico

Metodologia de Análisis

Uml

Se implementará UML para la modelación visual del sistema, lo cual permitirá representar de manera clara la estructura y las interacciones entre los diferentes componentes del sistema. Esto facilitará la comprensión y el análisis del proyecto, promoviendo una mejor toma de decisiones en el diseño e implementación. Al aplicar la metodología SCRUM, podremos gestionar el desarrollo de manera iterativa y ágil, permitiendo adaptarnos rápidamente a los cambios en los requisitos y mejorar la eficiencia del equipo. La combinación de estas dos herramientas proporcionará una base sólida para un desarrollo estructurado y centrado en los usuarios, asegurando un proceso dinámico y adaptable.

Historias de usuario

Para este apartado se iniciará haciendo las historias de usuario. Podemos decir que estas son una breve descripción escrita que captura lo que un usuario (actor) quiere lograr con el sistema. Las historias de usuario están estrechamente vinculadas con los requerimientos del sistema. Generalmente las historias de usuario siguen una estructura sencilla que describe:

- Quién es el usuario.
- Qué desea hacer o lograr.
- Para qué desea hacerlo (su objetivo o beneficio).

Las historias de usuario se elaboran teniendo la perspectiva del usuario final. Si debemos definir un formato sería como:

- Como [tipo de usuario], quiero [acción o funcionalidad] para [beneficio o razón].

Su objetivo es definir solamente los requisitos de alto nivel, además de servir como una visión global del potencial alcance y los beneficios esperados.
Una parte importante de las historias de usuario son los denominadas estancias o escenarios, estos estan diseñados para escribir pruebas concretas de aceptación, positivas o negativas sea en el marco de testing.

Casos de uso

Un diagrama de casos de uso nos permitirá identificar y visualizar todas las funcionalidades que el sistema debe ofrecer a los diferentes tipos de usuarios (actores). Además de que es visualmente sencilla de comprender, esto facilita la comunicación entre desarrolladores, clientes, y otros involucrados en el proyecto.

Teniendo en cuenta lo anterior hay que definir los puntos claves para la buena implementación del diagrama de casos de uso:

1. Actores.
2. Descripción
3. Precondiciones.
4. Postcondiciones.
5. Flujo normal.
6. Flujos alternativos.
7. Excepciones.
8. Prioridades
9. Frecuencia de uso.
10. Reglas.
11. Requerimientos especiales.
12. Suposiciones.

Una vez que tengamos definidas las historias de usuario, puedes crear diagramas de casos de uso para representar gráficamente cómo los actores (usuarios) interactúan con el sistema y las funcionalidades que necesitan. En SCRUM, estos diagramas se pueden usar en dos momentos clave:

- Definir el alcance: Puedes usar los diagramas de casos de uso durante la planificación inicial del proyecto para asegurarte de que todas las funcionalidades requeridas están cubiertas. Cada caso de uso debe estar vinculado a una o varias historias de usuario.
- Planificación de sprints: Al inicio de cada sprint, se seleccionan las historias de usuario más prioritarias para desarrollar. Cada historia de usuario puede estar respaldada por uno o varios casos de uso que guíen al equipo de desarrollo sobre cómo implementar la funcionalidad.

Metodología de diseño

El diseño de interfaces gráficas de usuario (GUI) es crucial para la creación de software eficiente y fácil de usar. En el contexto de un gimnasio, una interfaz bien diseñada puede mejorar la experiencia del usuario, facilitando la gestión de actividades, reservas y más.

Fundamentos de diseño de interface

El diseño de interfaces gráficas se basa en varios principios clave:

- Usabilidad: La interfaz debe ser intuitiva y fácil de navegar.
- Accesibilidad: Debe ser accesible para todos los usuarios, incluyendo aquellos con discapacidades.
- Estética: Un diseño visualmente atractivo puede mejorar la satisfacción del usuario.
- Consistencia: Mantener una apariencia y comportamiento consistentes en toda la aplicación.

Metodologías de Diseño

- Diseño Centrado en el Usuario (UCD): Este enfoque pone al usuario en el centro del proceso de diseño, asegurando que sus necesidades y preferencias sean prioritarias.
- Pensamiento de Diseño: Una metodología que fomenta la creatividad y la resolución de problemas a través de la empatía con el usuario.
- Desarrollo Ágil: Un enfoque iterativo que permite ajustes rápidos basados en la retroalimentación del usuario.

Herramientas de Diseño

- Figma: Una herramienta popular para el diseño de interfaces que permite la colaboración en tiempo real.
- Pencil: Utilizada ampliamente para el diseño de interfaces.
- Adobe XD: Ofrece herramientas robustas para el diseño y prototipado de interfaces.

Componentes Clave de la Interfaz

- Gestión de Usuarios: Debe proporcionar una visión general clara de las actividades y métricas importantes.
- Gestión de Empleados: Facilitar la creación y administración de perfiles de usuarios.
- Punto de venta: Un sistema intuitivo para reservar clases y equipos.

Consideraciones de Usabilidad

- Feedback del Usuario: Proporcionar retroalimentación inmediata para las acciones del usuario.
- Minimización de la Carga Cognitiva: Simplificar las tareas y reducir la cantidad de información que el usuario debe procesar.
- Diseño Responsivo: Asegurar que la interfaz funcione bien en diferentes dispositivos y tamaños de pantalla.

Metodologia de Desarrollo

JAVA

Java es un lenguaje orientado a objetos, lo que significa que se basa en el concepto de clases y objetos para estructurar el código. La programación orientada a objetos facilita la reutilización del código, la modularidad y el mantenimiento, lo que es especialmente útil en proyectos grandes y complejos.
La interfaz de usuario será desarrollada en Java, lo que permitirá ofrecer una experiencia interactiva y amigable para los administradores del sistema. Además, el uso de Java facilita la escalabilidad y mantenimiento del software, garantizando que la aplicación pueda adaptarse al crecimiento de la franquicia.

INTERFACES GRÁFICAS

Una gran ventaja de trabajar con java, es que se puede trabajar con ventanas o interfaces gráficas. Y se comportan de la misma forma independientemente del sistema operativo con el que trabajemos (windows, linux etc) y, en muchas ocasiones, utilizan los mismos componentes básicos para introducir órdenes (menús, botones, cuadros de texto, etc.).

Las interfaces gráficas se usan para interaccionar con el usuario, mostrando todas las opciones que el usuario puede realizar. Por lo tanto el diseño consistirá en crear objetos que den lugar a ventanas y sobre esas ventanas se dibujaran otros objetos llamados controles. Cada objeto estará ligado a un código que estará inactivo hasta que se produzca el evento que lo activa.

CLASES

El lenguaje java es un lenguaje orientado a objetos. La base de la POO (Programación orientada a objetos) es la clase.Una clase es un tipo de objeto definido por el usuario.
Una característica que aporta la POO es la herencia ya que permite la reutilización del código escrito por nosotros o otros.

VARIABLES

El lenguaje java es un lenguaje orientado a objetos. La base de la POO (Programación orientada a objetos) es la clase.Una clase es un tipo de objeto definido por el usuario. Una característica que aporta la POO es la herencia ya que permite la reutilización del código escrito por nosotros o otros.

SQL Server

SQL Server es un sistema de gestión de bases de datos relacional (RDBMS) desarrollado por Microsoft. Utiliza Transact-SQL (T-SQL), una extensión del lenguaje SQL estándar, para interactuar con los datos. Este marco teórico explorará los componentes clave, el funcionamiento y las teorías subyacentes que sustentan SQL Server.

Motor de Base de Datos (Database Engine)

El motor de base de datos es el corazón de SQL Server, responsable de gestionar, procesar y almacenar los datos. Está compuesto por varios subcomponentes:

- Motor Relacional: Administra las bases de datos, maneja las consultas, índices, transacciones, vistas, procedimientos almacenados y funciones. Este motor se encarga de ejecutar los comandos T-SQL y devolver los resultados.
- Motor de Almacenamiento: Administra la persistencia física de los datos, asegurando que la información se escriba y lea correctamente desde los discos duros u otros sistemas de almacenamiento. SQL Server utiliza un sistema de páginas de 8 KB para almacenar los datos.

- Gestión de Memoria: SQL Server usa un mecanismo avanzado para gestionar cómo los datos son cargados en memoria (RAM) desde el almacenamiento físico. El almacenamiento en caché de los datos mejora el rendimiento, ya que permite acceder rápidamente a los datos más utilizados.
- Motor de Consultas (Query Engine): Es responsable de procesar y optimizar las consultas SQL. SQL Server analiza el código SQL y elige el plan de ejecución más eficiente para recuperar los datos solicitados.
- Controlador de Transacciones: SQL Server sigue el modelo ACID (Atomicidad, Consistencia, Aislamiento, Durabilidad), asegurando que las transacciones se manejen de manera segura. Además, emplea un sistema de registros de transacciones (transaction logs) para garantizar que todas las operaciones sean reversibles en caso de fallos.

Conceptos Clave

1. Índices: Los índices mejoran el rendimiento de las consultas al permitir una recuperación de datos más rápida. Hay varios tipos de índices, como índices clustered y non-clustered.
2. Normalización: Es el proceso de organizar los datos en una base de datos para reducir la redundancia y mejorar la integridad de los datos. Las formas normales (1NF, 2NF, 3NF, etc.) son reglas que guían este proceso.
3. Procedimientos Almacenados: Son conjuntos de instrucciones SQL precompiladas que se almacenan en la base de datos y se pueden ejecutar repetidamente. Mejoran el rendimiento y la seguridad.
4. Vistas: Son consultas SQL almacenadas que actúan como tablas virtuales. Permiten simplificar consultas complejas y mejorar la seguridad al limitar el acceso directo a las tablas.
5. Transacciones: Garantizan las propiedades ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad) de las operaciones de la base de datos, asegurando que las operaciones se completen correctamente o se reviertan en caso de error.
6. Triggers: Son procedimientos almacenados que se ejecutan automáticamente en respuesta a ciertos eventos en una tabla o vista, como inserciones, actualizaciones o eliminaciones.
7. Particionamiento: Permite dividir grandes tablas en partes más pequeñas y manejables, mejorando el rendimiento y la gestión de datos.
8. Replicación: Es el proceso de copiar y distribuir datos y objetos de bases de datos de una base de datos a otra y luego sincronizar entre las bases de datos para mantener la consistencia.
9. Alta Disponibilidad: SQL Server ofrece varias soluciones para alta disponibilidad, como Always On Availability Groups, que permiten la replicación de bases de datos entre múltiples servidores para asegurar la disponibilidad continua.
10. Seguridad: Incluye autenticación, autorización, cifrado y auditoría para proteger los datos y cumplir con las normativas de seguridad.

Arquitectura SQL Server

Motor de Base de Datos: Es el componente principal que maneja el almacenamiento, procesamiento y seguridad de los datos. Se encarga de ejecutar consultas, gestionar transacciones y mantener la integridad de los datos.
SQL Server Management Studio (SSMS): Es una herramienta gráfica que permite a los administradores y desarrolladores gestionar y desarrollar bases de datos. Proporciona una interfaz para ejecutar consultas, crear y modificar objetos de base de datos, y monitorear el rendimiento
Servicios de Integración (SSIS): Plataforma para la integración de datos y la creación de soluciones de ETL (Extract, Transform, Load). Permite mover y transformar datos entre diferentes sistemas.
Servicios de Análisis (SSAS): Proporciona capacidades de análisis de datos y minería de datos. Permite crear modelos de datos multidimensionales y tabulares para análisis complejos.
Servicios de Reportes (SSRS): Herramienta para la creación, gestión y entrega de informes interactivos y paginados. Permite a los usuarios finales generar informes personalizados basados en los datos almacenados en SQL Server.

Proceso de consulta

- Conexión: Los clientes (aplicaciones o usuarios) se conectan al servidor SQL utilizando credenciales de autenticación (Windows o SQL Server).
- Envío de Consulta: El cliente envía una consulta T-SQL al servidor. Esta consulta puede ser una instrucción SELECT para recuperar datos, una instrucción INSERT para agregar datos, una instrucción UPDATE para modificar datos, o una instrucción DELETE para eliminar datos.
- Optimización: El motor de base de datos optimiza la consulta generando un plan de ejecución eficiente. Este plan determina cómo se recuperarán y procesarán los datos.
- Ejecución: El motor de base de datos ejecuta el plan de ejecución, accediendo a los datos necesarios y realizando las operaciones especificadas en la consulta.
- Resultados: El servidor devuelve los resultados de la consulta al cliente. Si la consulta modifica datos, el servidor confirma que las operaciones se han completado correctamente.

Analisis

Definición de requerimientos.

Requerimientos funcionales

- Gestión de membresías.
  Registro y actualización: La base de datos debe permitir el registro y actualización de los datos de los miembros.

  Historial de pagos: El sistema debe almacenar y gestionar los pagos realizados por los miembros detallando la fecha cuando se realizó, método de pago y monto que se pagó.

  Control de membresías: La base de datos debe tener un registro del estado de la membresías del usuarios para identificar rápidamente qué miembros tienen una membresía activa o inactiva, y el sistema debe activar o desactivar el acceso.

- Gestión de empleados.
  Registro de empleados y roles (actualización): El sistema debe gestionar los datos de los empleados, incluyendo nombre, dirección y roles.

  Nomina: La base de datos debe tener un registro de los sueldos de los empleados, teniendo en cuenta diferentes factores como su rol y turnos de los empleados.

  Turnos: Debe facilitar la planificación de horarios y turnos de los empleados.

  Estatus: Aqui debe de marcarse el estado del empleado.

- Check-IN.
  Entrada y salida: Los miembros y empleados deben poder entrar al gimnasio mediante tarjetas de identificación o un sistema biométrico.

- Clases y actividades.
  Horarios: Almacenamiento y gestión de los horarios de clases grupales y personalizadas.

  Reservación: Permitir a los miembros reservar espacio en las clases.

- Gestión de equipo.
  Estado: Seguimiento del estado y ubicación de los equipos de gimnasio.

  Mantenimiento (booleano): Almacenamiento de datos relacionados con el mantenimiento y reparaciones de los equipos.

  Registro: Especificaciones del equipo.

- Gestión de inventario.
  Registro: características de los productos además de su respectivo código de barras.

- Sistema de puntos.
  Historial de usuario: Permitir al miembro la obtención de puntos que pueden ser utilizados para distintos beneficios al hacer acciones específicas dentro del gimnasio.

Requerimientos no funcionales

- Disponibilidad: La base de datos distribuida debe estar disponible las 24 horas del día, los 7 días de la semana, permitiendo el acceso a los datos en tiempo real desde múltiples ubicaciones.
- Escalabilidad: El sistema debe ser capaz de escalar vertical y horizontalmente a medida que el gimnasio crece, además de permitir la adición de nuevos nodos o servidores sin afectar el rendimiento del sistema.
- Rendimiento: La base de datos debe manejar eficientemente las operaciones diarias, como las consultas de membresía y los pagos.
- Consistencia de datos: La base de datos debe garantizar la consistencia de los datos entre los diferentes nodos distribuidos, para asegurar que las actualizaciones y consultas reflejen la información correcta.

10.1.1 Requerimientos de almacenamiento y replicación

- Distribución geográfica: Los nodos de la base de datos deben estar distribuidos en ubicaciones estratégicas para garantizar una menor latencia.
- Replicación de datos: Implementar replicación de datos entre los nodos para garantizar que en caso de que un nodo falle, otro nodo tenga una copia actualizada de los datos.
- Tolerancia a fallos: Los nodos de la base de datos deben ser capaces de manejar fallos parciales sin comprometer la operación general del sistema. Esto implica tener mecanismos de recuperación ante desastres, como copias de seguridad automáticas y planes de recuperación de datos.

Diagrama de casos de uso

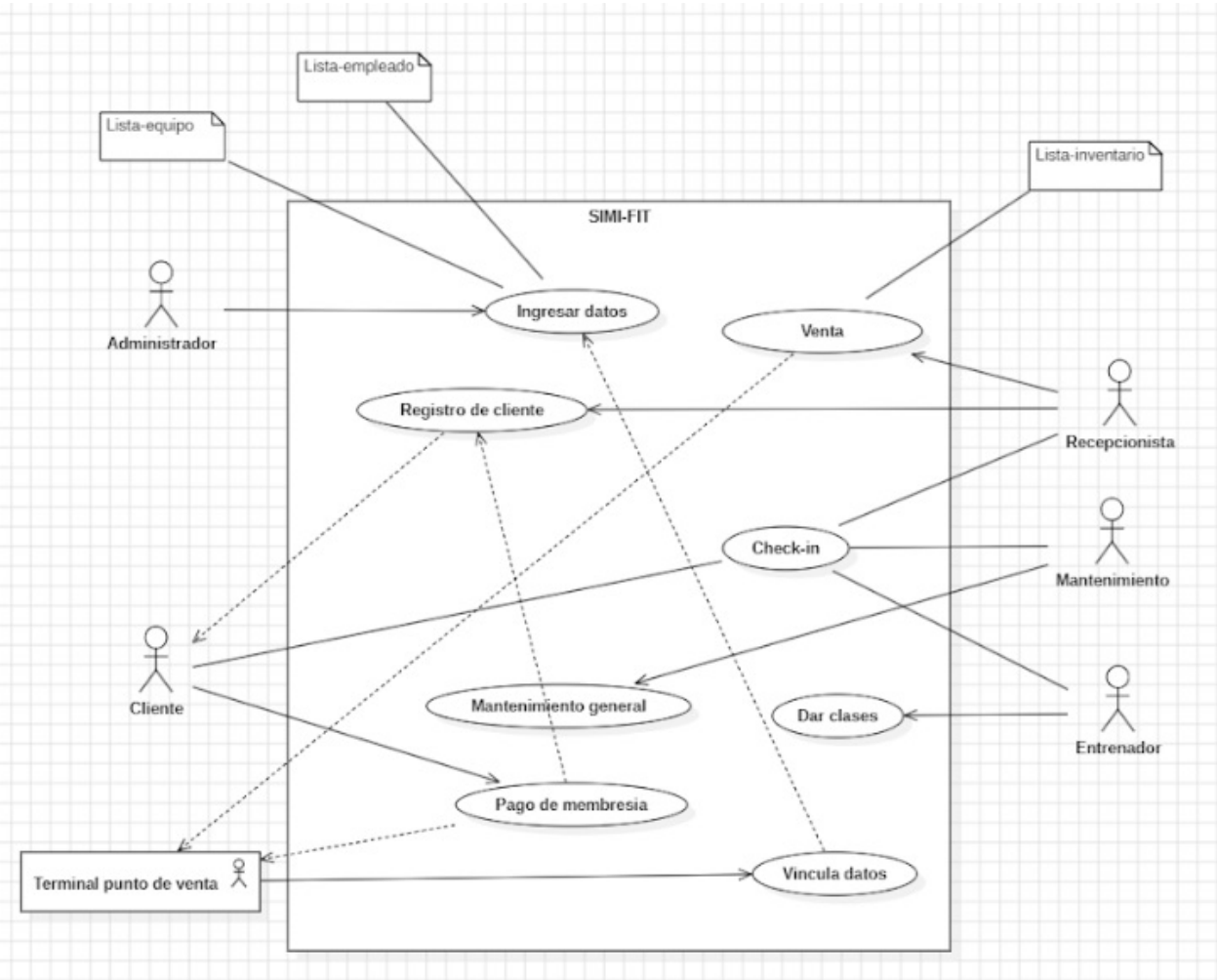


Figure 2:

Casos de uso

Table 1: Casos de uso

Casos de uso 1	Registro de cliente a Gimnasio.
Descripción	El cliente intenta registrarse en el Gimnasio, el espacio y servicio están disponibles, así que solicita al Recepcionista que lo haga y el cliente paga su registro.
Precondiciones	El Servicio debe existir y debe estar disponible y el sistema debe estar operando.
Trigger	El cliente solicita al recepcionista ser registrado en el gimnasio, el recepcionista solicita su información del cliente.
Success	El cliente será registrado correctamente y confirmará al recepcionista.
Abort	Si la información ingresada está incompleta o incorrecta, y solicitará correcciones. En cualquier caso, en particular, el registro deberá de ser eliminado.
Actores	Primary: Recepcionista. Secondary: Cliente.

Table 2: Casos de uso

Casos de uso 2	Registro de empleados
Descripción	El Administrador del sistema registra nuevos empleados en el sistema de gestión de empleados.
Precondiciones	El administrador debe de estar autorizado por el sistema y el sistema debe de estar operando.
Trigger	El Administrador actualiza o registra los datos del Empleado
Success	El nuevo Empleado queda registrado en el sistema con sus datos personales y laborales, obteniendo un identificador único de Empleado.
Abort	Cualquier campo o dato mal llenado debe ser corregido. Si ocurre algún caso, en particular, el registro y datos deben ser eliminados.
Actores	Primary: Administrador. Secondary: Sistema de Gestión de Empleados.

Table 3: Casos de uso

Casos de uso 3	Registro de productos
Descripción	El Administrador registra los nuevos productos para su venta en el Gimnasio.
Precondiciones	El administrador debe de estar autorizado en el sistema y el sistema debe de estar operando.
Trigger	El sistema solicita la información del producto y el administrador registra la información.
Success	Los productos quedan registrados y están listos para su venta dentro del Gimnasio.
Abort	En caso de que algún campo no esté lleno o un dato se invalidó, deberá de corregirse e. Si existe algún problema con el producto, su registro y sus datos deberán ser removidos y deberán contactar con su proveedor.
Actores	Primary: Administrador. Secondary: Producto, Proveedo

Table 4: Casos de uso

Caso de Uso 4	Terminal Punto de Venta
Descripción	El Cliente desea comprar un producto del Gimnasio, el producto debe de estar en existencia, así que solicita al Recepcionista para la venta del producto.
Precondiciones	El producto debe de estar en existencia para su venta.
Trigger	El cliente selecciona un producto o productos para su compra.
Success	El cliente paga y obtiene su producto.
Abort	Cualquier situación que complique la venta del producto deberá ser cancelada y regresar el dinero al cliente en caso de haber cobrado.
Actores	Primary: Cliente Secondary: Producto, Recepcionista

Table 5: Casos de uso

Casos de uso 5	Pase de Asistencia.
Descripción	El Empleado deberá de pasar asistencia al entrar y salir, el sistema Check-in deberá de estar operando.
Precondiciones	El empleado debe de estar registrado en el sistema y el sistema debe de estar operando.
Trigger	El sistema solicita la información del producto y el administrador registra la información.
Success	El empleado recibe notificación de su entrada o salida y su registro se actualiza en el sistema.
Abort	Si la identificación o pase de asistencia falla, se le notificará al empleado para intentarlo una vez más.
Actores	Primary: Empleado. Secondary: Sistema CheckIn.

Table 6: Casos de uso

Casos de uso 6	Registro de Equipo de Gimnasio.
Descripción	El Administrador y empleado registra los nuevos equipos para el Gimnasio.
Precondiciones	El administrador y empleado debe de estar autorizado en el sistema y el sistema debe de estar operando y debe de ser accesible.
Trigger	El sistema solicita la información del equipo y el administrador registra esta información.
Success	El equipo se registra de manera correcta en el sistema y se confirma su registro al administrador.
Abort	En caso de que algún campo no esté llenado o un dato sea invalidó, deberá corregirse. Si existe algún problema con el equipo, su registro y sus datos deberán ser removidos y deberán contactar con su proveedor.
Actores	Primary: Administrador, Empleado. Secondary: Producto, Proveedor.

Table 7: Casos de uso

Casos de uso 7	Pago de membresía.
Descripción	El cliente solicita pagar su membresía del Gimnasio, el servicio debe de estar disponible, así que solicita al recepcionista que lo haga y paga su membresía para tener acceso al equipo del Gimnasio.
Precondiciones	El cliente debe de estar registrado y autentificado en el sistema y el sistema debe de estar operando..
Trigger	El cliente selecciona la membresía y el sistema solicita los detalles de pago.
Success	El pago de la membresía se procesa correctamente y se actualiza el estado de la membresía del Cliente.
Abort	Si la membresía no está disponible, el sistema notificará al cliente y se cancelará el pago..
Actores	Primary: Cliente. Secondary: Recepcionista.

Table 8: Casos de uso

Casos de uso 8	Movimientos (Egresos e Ingresos).
Descripción	El administrador consulta los movimientos del gimnasio, obteniendo el tipo de movimientos junto con sus especificaciones, para reportar al contador y dueño del gimnasio.
Precondiciones	El administrador debe de estar registrado y autentificado en el sistema y el sistema debe de estar operando.
Trigger	El administrador consulta los movimientos del gimnasio y el sistema solicita los datos correspondientes.
Success	La consulta de los movimientos del gimnasio se procesa correctamente y se muestran los movimientos hechos, como el tipo de movimiento y los destalles de este mismo.
Abort	Si el administrador no está autentificado o falla la autentificación, el sistema cancelará la consulta y notificará al administrador para que lo intente de nuevo.
Actores	Primary: Administrador Secondary: Sistema de Movimientos.

Table 9: Matriz de relación

	CLIENTE	CONTACTOC	MEMBRESIA	PRODUCTO	EQUIPO	MOVIMIENTO	SUCURSAL	CONTACTOS	DIRECCION	HORARIO	ASIGNACION	EMPLEADO	CONTACTOE
CLIENTE	0	1	1	0	0	0	0	0	0	0	0	0	0
CONTACTOC	1	0	0	0	0	0	0	0	0	0	0	0	0
MEMBRESIA	1	1	0	0	0	0	0	0	0	0	0	0	0
PRODUCTO	0	0	0	0	0	0	1	0	0	0	0	0	0
EQUIPO	0	0	0	0	0	0	1	0	0	0	0	0	0
MOVIMIENTO	0	0	0	0	0	0	1	0	0	0	0	0	0
SUCURSAL	0	0	1	1	1	1	0	1	1	0	0	0	0
CONTACTOS	0	0	0	0	0	0	1	0	0	0	0	0	0
DIRECCION	0	0	0	0	0	0	1	0	0	0	0	0	0
HORARIO	0	0	0	0	0	0	0	0	0	0	1	0	0

Table 10: Matriz de relación

ASIGNACION	0	0	0	0	0	0	0	0	0	1	0	1	0
EMPLEADO	0	0	0	0	0	0	0	0	0	0	1	0	1
CONTACTOE	0	0	0	0	0	0	0	0	0	0	0	1	0

Diagrama Entidad Relacion

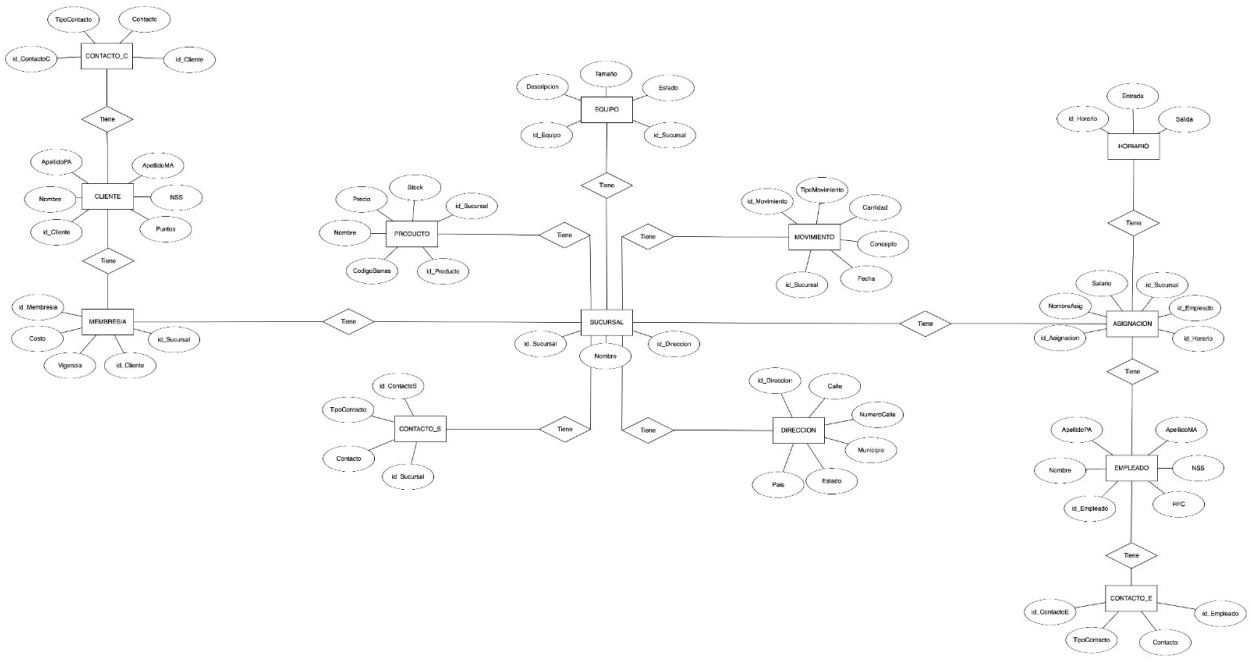


Figure 3:

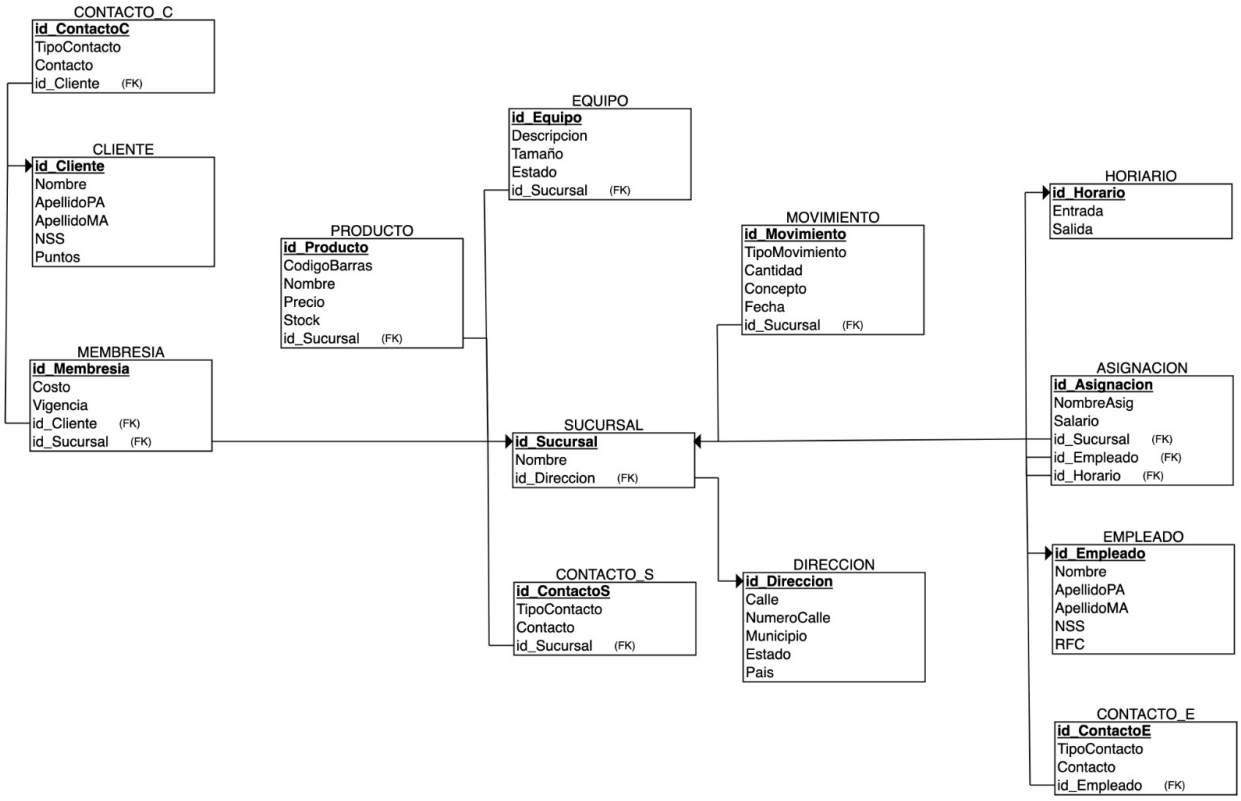


Figure 4:

Identificacion de Identidades

Table 11: Cliente

Entidad	Descripcion
idCliente (PK)	Identificador único para cada cliente.
Nombre	Nombre del cliente.
ApellidoPA	Apellido paterno del cliente.
ApellidoMA	Apellido materno del cliente
NSS	Número de Seguridad Social del cliente.
Puntos	Puntos acumulados por el cliente, posiblemente para un programa de recompensas.

Table 12: CONTACTO C

Entidad	Descripcion
idContactoC (PK)	Identificador único para cada método de contacto del cliente.
TipoContacto	ipo de contacto (por ejemplo, teléfono, correo electrónico).
Contacto	Detalle del contacto (por ejemplo, número de teléfono, dirección de correo electrónico).
idCliente (FK)	Referencia al identificador único del cliente.

Table 13: Membresia

Entidad	Descripcion
idMembresia (PK)	Identificador único para cada membresía.
Costo	Costo de la membresía.
Vigencia	Periodo de validez de la membresía.
idCliente (FK)	Referencia al identificador único del cliente.
idSucursal (FK)	Referencia al identificador único de la sucursal.

Table 14: Producto

Entidad	Descripcion
idProducto (PK)	Identificador único para cada producto.
CodigoBarras	Código de barras del producto.
Nombre6 Nombre del producto.	
Precio	Precio del producto.
Stock	Cantidad disponible del producto.
idSucursal (FK)	Referencia al identificador único de la sucursal donde se encuentra el producto.

Table 15: Equipo

Entidad	Descripcion
idEquipo (PK)	Identificador único para cada equipo.
Descripción	Descripción del equipo.
Tamaño	Tamaño del equipo.
Estado	Estado actual del equipo (por ejemplo, operativo, en reparación).
idSucursal (FK)	Referencia al identificador único de la sucursal donde se encuentra el equipo.

Table 16: Movimiento

Entidad	Descripcion
idMovimiento (PK)	Identificador único para cada movimiento (transacción).
TipoMovimiento	Tipo de movimiento (por ejemplo, ingreso, egreso).
Cantidad	Cantidad involucrada en el movimiento.
Concepto	Concepto o descripción del movimiento.
Fecha	Fecha en que se realizó el movimiento.
idSucursal (FK)	Referencia al identificador único de la sucursal donde se realizó el movimiento.

Table 17: Sucursal

Entidad	Descripcion
idSucursal (PK)	Identificador único para cada sucursal.
Nombre	Nombre de la sucursal.
idDireccion (FK)	Referencia al identificador único de la dirección de la sucursal.

Table 18: Contacto S

Entidad	Descripcion
idContactoS (PK)	Identificador único para cada método de contacto de la sucursal.
TipoContacto	Tipo de contacto (por ejemplo, teléfono, correo electrónico).
Contacto	Detalle del contacto (por ejemplo, número de teléfono, dirección de correo electrónico).
idSucursal (FK)	Referencia al identificador único de la sucursal.

Table 19: Direccion

Entidad	Descripcion
idDireccion (PK)	Identificador único para cada dirección.
Calle	Nombre de la calle.
NumeroCalle	Número de la calle.
Municipio	Municipio donde se encuentra la dirección.
Estado	Estado donde se encuentra la dirección.
Pais	País donde se encuentra la dirección.

Table 20: Horario

Entidad	Descripcion
idHorario (PK)	Identificador único para cada horario de trabajo.
Entrada	Hora de inicio del horario de trabajo.
Salida	Hora de finalización del horario de trabajo.

Table 21: Asignacion

Entidad	Descripcion
idAsignacion (PK)	Identificador único para cada asignación de empleado.
NombreAsig	Nombre de la asignación o puesto de trabajo.
Salario	Salario correspondiente a la asignación.
idSucursal (FK)	Referencia al identificador único de la sucursal donde se realiza la asignación.
idEmpleado (FK)	Referencia al identificador único del empleado.
idHorario (FK)	Referencia al identificador único del horario asignado al empleado.

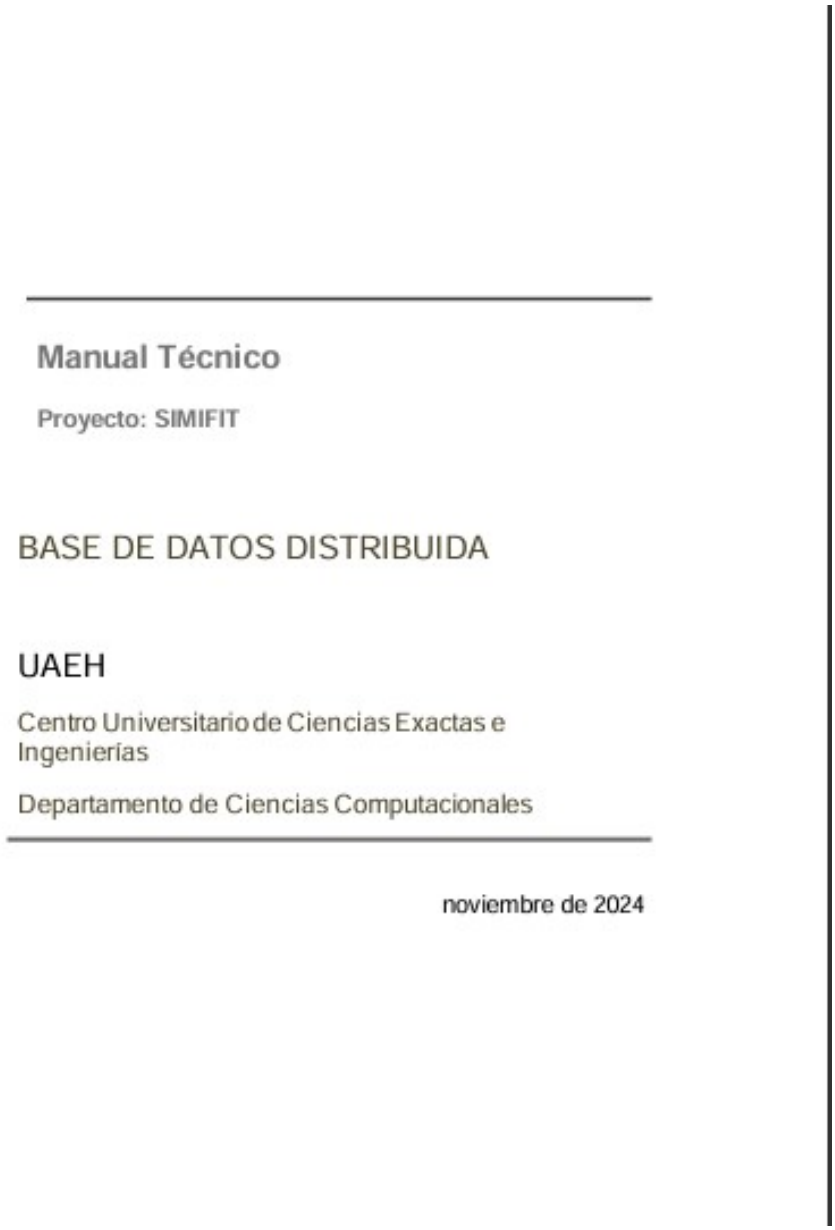
Table 22: Empleado

Entidad	Descripcion
idEmpleado (PK)	Identificador único para cada empleado.
Nombre	Nombre del empleado.
Apellido1	Primer apellido del empleado.
Apellido2	Segundo apellido del empleado.
NSS	Número de Seguridad Social del empleado.
RFC	Registro Federal de Contribuyentes del empleado (utilizado en México para propósitos fiscales).

Table 23: Contacto E

Entidad	Descripcion
idContactoE (PK)	Identificador único para cada método de contacto del empleado.
TipoContacto	Tipo de contacto (por ejemplo, teléfono, correo electrónico).
Contacto	Detalle del contacto (por ejemplo, número de teléfono, dirección de correo electrónico).
idEmpleado (FK)	Referencia al identificador único del empleado.

MANUAL TECNICO



Referencias Bibliográficas

References

[1] López, C. P. (2008). MySQL para Windows y Linux.

[2] Wiederhold, G. (1985). Diseño de bases de datos.

[3] Fontela, C. (2012). UML modelo de software para profesionales.

SIMIFIT

[4] Shaver, D. (1998). El siguiente paso en la mercadotecnia directa: cómo usar bases de datos orientándolas al consumidor.

[5] Hernandez. (2010). METODOLOGIA DE LA INVESTIGACIO.

[6] De Miguel Castaño, A., Fernandez, P. M., Galan, E. C. (2001). Diseño de bases de datos: problemas resueltos.

[7] Cuadra, D., Castro, E., Iglesias, A. M. (2013). Desarrollo de bases de datos: casos practicos desde el analisis a la implementacion.

MANUAL TÉCNICO
CONTENIDO

1. Objetivos

1.1 Objetivos Específicos.

1. Registrar y gestionar la información de los clientes, el cual se creara un sistema para almacenar y actualizar los datos personales de los clientes o socios.
2. Controlar las suscripciones y pagos, implementando un sistema de seguimiento de pagos, donde registre los pagos realizados y generar advertencias para pagos pendientes.
3. Administrar el inventario de productos en venta, como suplementos, proteínas, pre-entreno, etc., crear un inventario para registrar el control de los productos, ofrece el gimnasio.
4. Registrar y gestionar la información de los empleados del gimnasio, el cual se realizará un sistema para almacenar información esencial de los empleados.
5. Administración de equipos para registrar y monitorear los equipos del gimnasio, como el estado, la ubicación y su mantenimiento.

2. Alcance

Éste documento está dirigido a: El dueño a quien se le proporciono el software.

Conocimientos básicos en: Base de datos (formas normales, sentencias de sql para manipular datos y tipos de datos) , lenguajes de programación y consulta de datos (Java y sql respectivamente), Bases de datos distribuidas (fragmentación y arquitectura), Diseño de diagramas para la estructura de software (diagramas elaborados en UML).

3. Requerimientos Técnicos

3.1 Requerimientos Mínimos de Hardware.

Procesador: Ryzen 3 o i3 o superior

Memoria RAM (Mínimo): 4GB

Disco Duro (Mínimo) : 16GB

3.2 Requerimientos Mínimos de Software.

Privilegios de Administrador: Si

Sistema Operativo: Windows 8 o Superior

Versión de Java: 8

4. Herramientas Utilizadas para el Desarrollo.

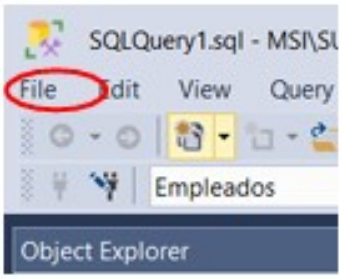
- **SQL Server**
  Descripción: Es un sistema gestor de base de datos (SGBD)
  En efectos prácticos, en consideración del enfoque que tiene el proyecto (Hacer uso de base de datos distribuidas), la conexión entre nodos de una BDD es mas fácil y practica en este SGBD en particular.

- **UMJ**
  Descripción: Lenguaje unificado de modelado.
  Es un software amigable con el usuario que permite diseñar los modelos necesarios para modelar la estructura del proyecto

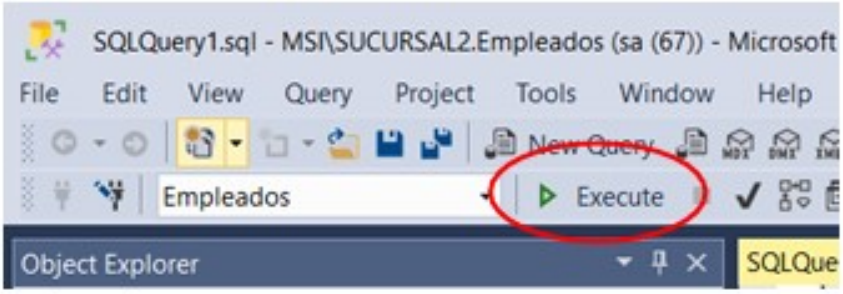
- **Neatbeans**
  Descripción: Entorno de desarrollo integrado (IDE)
  Utilizado para la programación y diseño del software del proyecto utilizando lenguaje Java y a grosso modo, las herramientas para diseñar interfaces son muy practicas y reducen considerablemente el tiempo empleado para diseñar las mismas.

5. Instalación

- Abrir sql server
- Abrir FILE---OPEN---PROJECT/SOLUTION



- Escoger archivo de tipo txt llamado Sucursal.txt
- Ejecuta las sentencias cargadas (botón execute)



- Guardar el ejecutable del software en documentos SIMIFIT.exe

NOTA: Al guardar la base de datos se recomienda no modificar las rutas predeterminadas para guardar la misma, de no ser así podría no funcionar el sistema

6. Configuración

1. **Cambiar el nombre de una tabla.**
   - **Propósito:** Renombrar una tabla para reflejar mejor su contenido.
   - **Comando:**

sql
Copiar código
EXEC sp_rename 'TablaAntigua', 'TablaNueva';

   - **Situación:** Usado cuando el nombre original ya no es representativo o se requiere una mejor organización.

---

2. **Agregar una columna a una tabla.**
   - **Propósito:** Añadir nuevos datos que no se contemplaron al crear la tabla.
   - **Comando:**

sql
Copiar código
ALTER TABLE MiTabla ADD NuevaColumna INT;

   - **Situación:** Usado cuando necesitas registrar información adicional en una tabla existente.

---

3. **Cambiar el tipo de datos de una columna.**
   - **Propósito:** Ajustar el tipo de datos de una columna para que coincida con el formato necesario.
   - **Comando:**

sql
Copiar código
ALTER TABLE MiTabla ALTER COLUMN Columna1 NVARCHAR(50);

   - **Situación:** Usado cuando un campo cambia de propósito, por ejemplo, de números a texto.

---

4. **Configurar un valor predeterminado para una columna.**
   - **Propósito:** Establecer un valor que se inserta automáticamente si no se especifica uno.
   - **Comando:**

sql
Copiar código
ALTER TABLE MiTabla ADD CONSTRAINT DF_Columna1 DEFAULT 'PorDefecto' FOR Columna1;

   - **Situación:** Usado para evitar valores nulos en columnas esenciales.

---

5. Eliminar registros duplicados en una tabla.

- **Propósito:** Limpiar datos redundantes de una tabla.
- **Comando:**

sql
Copiar código
DELETE FROM MiTabla
WHERE ID NOT IN (
SELECT MIN(ID) FROM MiTabla GROUP BY ColumnaClave
);

- **Situación:** Usado al realizar auditorías o limpieza de datos para evitar duplicados.

7. Diseño de la Arquitectura Física.

La arquitectura física presentada fueron distintos equipos ya que la base de datos puede ser conectada de N equipos distintos mientras estén conectados a la misma red LAN

- Nombre de los equipos: 7 Laptops Arquitectura 64 bits, 1 Router
- Ubicación física dentro del centro de cómputo, y en el sitio alterno de procesamiento: La ubicación puede ser dinámica mientras esten conectados los mismos equipos al router
- Direcciones IP asignadas: Asignadas de manera automática por el protocolo DCHP
- Lista de IP asignadas: Asignadas de manera automática por el protocolo DCHP en la puerta de enlace predeterminada "http://192.168.1.254/login.htm"
- Lista de IP's:

  192.168.1.0 – Router LAN

  192.168.1.1 – Sebastian

  192.168.1.2 – Cesar

  192.168.1.3 – Fernando

  192.168.1.4 – Axel

  192.168.1.5 – Jesus

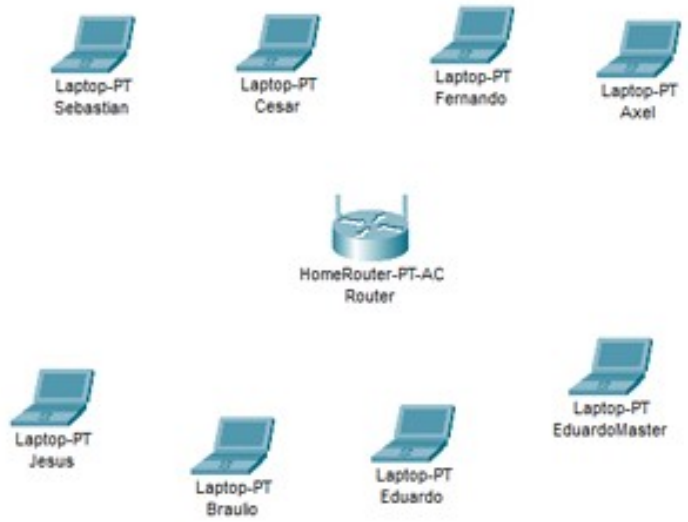
  192.168.1.6 – Braulio

  192.168.1.7 – Eduardo

  192.168.1.8 – EduardoMaster

Puertos TCP/UDP necesarios para comunicación: Puerto SQL Server 2024: 1433 Para TCP/IP

- Dependencias con otros sistemas: Seguir la instalación de manera solicitada, habilitar los puertos, abrir reglas de Firewall
- Interrelaciones de conexión entre los equipos.



8. Usuarios

Solo existe 1 solo nivel de usuario para el software, Administrador, ya que el que pueda manipular el software puede administrar, eliminar, añadir o editar productos según séase a su necesidad. Este puede ser abierto desde cualquier software Windows que cuente con la versión SQL Server 2022.

8.1 Usuarios de base de datos

Usamos en su totalidad SQL Server 2022, disponible en la pagina de Microsoft (https://www.microsoft.com/es-mx/sql-server/sql-server-downloads)

Nombre del usuario: Admin (sa), Desktop/NOMBREDELUSUARIO

Grupos a los que pertenece: Todos

Privilegios generales a nivel de base de datos: Admin

Privilegios sobre objetos de las mismas: Edicion, Actualizacion y Generacion de datos

8.2 Usuarios de sistema operativo

Usamos en su gran mayoría Windows 10, aunque hubo equipos con sistemas Windows 11.

Nombre del usuario: DESKTOP/NOMBREDEUSUARIODEINTEGRANTE.

<Para uso final>

DESKTOP/EDUARDOMASTER.

<Para uso experimental y de pruebas>

Grupos a los que pertenece: Todos

Privilegios sobre carpetas: Todos

8.3 Usuarios de aplicaciones

Nombre del usuario: Para el publico en general (O venta)

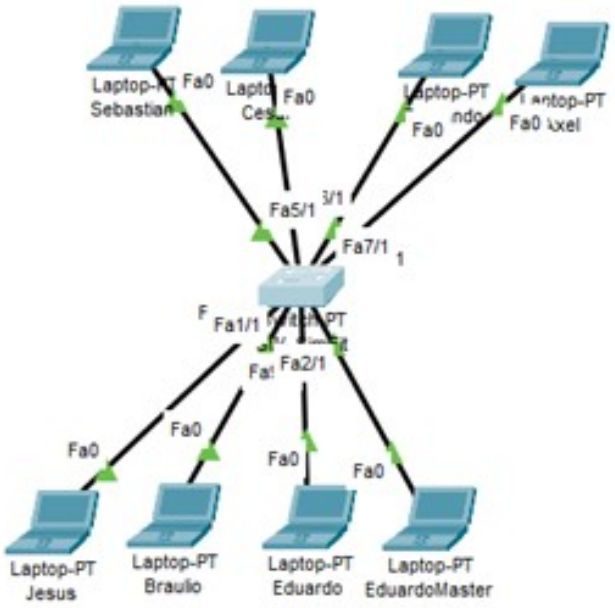
<Cualquiera que ocupe el software>

Grupos a los que pertenece: Cualquiera

Privilegios dentro de la aplicación: Todos

9. Contingencias y soluciones.

1. ***Error en el router.***
   Descripción*. Algún fallo o corte de energía que no permita la conexión LAN entre las computadoras*
   Solución*. Uso de Cableado RJ45 (Entre 2 equipos o con un switch pueden conectase N número de equipos), entre los equipos como se presenta en el diagrama.*



2. ***Pérdida de datos por falla eléctrica.***
   Descripción: *Una interrupción inesperada de energía puede causar la pérdida de datos no guardados o corrupción de los mismos.*
   Solución: *Implementar un sistema de respaldo de energía (NOBREAK) para evitar apagones repentinos y configurar copias de seguridad automáticas en SQL Server con periodicidad diaria*

3. ***Error al intentar ejecutar una consulta.***
   Descripción: *Un error de sintaxis impide que se ejecute una consulta SQL.*
   Solución: *Revisar cuidadosamente la consulta en busca de errores en palabras clave, comas, o estructuras.*

4. **Contraseña olvidada por el administrador.**
   Descripción: El administrador no puede acceder porque olvidó la contraseña del usuario sa.
   Solución: Recuperar la contraseña utilizando una cuenta con privilegios de administrador del sistema.