

M10 Bases de Datos (ITC) FIN A

ACTIVIDAD 2

| **Tutor:** | **Luís Sierra Betancourt** |
| --- | --- |
| **Estudiante:** | **José Ramón Ibáñez Posadas** |
| **Matricula:** | **BNL098377** |

| Monterrey, Nuevo León | Lunes, 30 de Septiembre de 2024 |
| --- | --- |

INTRODUCCIÓN

En la presente actividad, aprenderemos sobre el proceso de normalización de una tabla, un aspecto crucial en el diseño de bases de datos que busca eliminar redundancias y mejorar la integridad de la información. Partiremos de una tabla con datos repetitivos sobre clientes, autos, servicios y registros de mantenimiento, identificando las limitaciones y los problemas que puede generar la duplicación de información, tales como inconsistencias y dificultades en la gestión. Para abordar estos problemas, utilizaremos la técnica de normalización hasta la Tercera Forma Normal (3NF), lo cual implicará dividir la tabla original en varias tablas relacionadas, cada una con un propósito bien definido. Este proceso garantizará que los datos estén organizados de forma lógica y que cada atributo dependa exclusivamente de su clave primaria, minimizando la redundancia y mejorando la eficiencia.

La normalización no solo nos ayudará a estructurar la información de manera más clara, sino que también sentará las bases para una base de datos más escalable y manejable. A través de la separación de los datos en entidades distintas, lograremos una organización que facilite la consulta, modificación y actualización de la información sin generar problemas de integridad. Así, la introducción de la normalización en el diseño de bases de datos asegurará una mayor calidad y consistencia de los datos, optimizando tanto el almacenamiento como el manejo de la información en los sistemas. Este enfoque será fundamental para cualquier proyecto que busque una gestión de datos confiable y eficiente, evitando los problemas típicos derivados de una estructura mal diseñada.

.

DESARROLLO

RESUMEN

El diseño de una base de datos es un proceso complejo que implica la planificación y estructuración de los datos para optimizar su almacenamiento, acceso y manipulación. Un diseño bien elaborado garantiza que la base de datos sea eficiente, libre de redundancias y que ofrezca la capacidad de mantener la integridad y consistencia de los datos a lo largo del tiempo. A continuación, se detallan aspectos clave relacionados con el diseño de una base de datos, incluyendo componentes fundamentales, restricciones, dependencias funcionales, clausuras y el concepto de superclases y subclases.

**1. Componentes de una Base de Datos**

Los componentes básicos de una base de datos son fundamentales para comprender cómo se estructuran y organizan los datos:

* **Tablas**: las tablas son la unidad central de almacenamiento en bases de datos relacionales. Cada tabla almacena información sobre una entidad particular, como "Clientes" o "Productos". Una tabla se compone de filas y columnas, donde cada fila representa una instancia específica de la entidad y cada columna representa un atributo de la entidad.
* **Registros (Filas)**: cada fila dentro de una tabla se conoce como un registro. Por ejemplo, en una tabla de "Clientes", cada registro representará a un cliente único con detalles como nombre, dirección y número de teléfono. Los registros deben ser únicos cuando se aplica una restricción de clave primaria, lo cual evita la duplicación innecesaria.
* **Atributos (Columnas)**: cada columna en una tabla representa un atributo de la entidad, como "Nombre", "Edad" o "Dirección". Los atributos definen las características de los datos que se almacenan en cada registro.
* **Índices**: los índices son estructuras adicionales que mejoran el rendimiento de las consultas al permitir búsquedas rápidas en una o más columnas. Aunque la creación de índices puede acelerar la recuperación de datos, también puede aumentar el tiempo necesario para insertar, actualizar o eliminar registros, ya que los índices deben mantenerse actualizados.
* **Relaciones**: en bases de datos relacionales, las tablas pueden estar conectadas entre sí mediante relaciones. Estas relaciones pueden ser:
  + **Uno a uno**: cada registro en una tabla está relacionado con un único registro en otra tabla.
  + **Uno a muchos**: un registro en una tabla puede estar relacionado con múltiples registros en otra tabla.
  + **Muchos a muchos**: múltiples registros en una tabla pueden estar relacionados con múltiples registros en otra tabla. Esto suele ser modelado mediante una tabla intermedia.
* **Vistas**: una vista es una representación lógica de datos que se deriva de una o más tablas. Las vistas permiten a los usuarios acceder a datos específicos sin tener que interactuar directamente con las tablas subyacentes, proporcionando una capa adicional de seguridad y simplificando la consulta de datos complejos.

**2. Restricciones en Bases de Datos**

Las restricciones son reglas que garantizan que los datos almacenados en una base de datos sean válidos y estén correctamente relacionados. Las restricciones más comunes incluyen:

* **Clave primaria**: una clave primaria es un identificador único para cada registro en una tabla. Asegura que no haya duplicados y que cada registro sea único. Por ejemplo, en una tabla de "Empleados", el campo "ID\_Empleado" podría ser la clave primaria, asegurando que cada empleado tenga un identificador único.
* **Clave foránea**: una clave foránea es una columna que establece una relación entre dos tablas. En una tabla de "Pedidos", una columna "ID\_Cliente" puede ser una clave foránea que hace referencia a la clave primaria "ID\_Cliente" de la tabla "Clientes". Esto asegura que todos los pedidos registrados correspondan a clientes válidos.
* **Restricción de unicidad**: esta restricción garantiza que los valores de una columna específica sean únicos en toda la tabla. Es útil para atributos que no pueden repetirse, como el número de identificación fiscal de una persona.
* **Restricciones CHECK**: permiten especificar una condición lógica que debe cumplirse para que un valor pueda ser almacenado en una columna. Por ejemplo, una restricción CHECK podría garantizar que la edad de una persona en una columna "Edad" sea mayor o igual a 18.
* **Integridad referencial**: asegura que las relaciones entre las tablas se mantengan correctamente. Si se elimina un registro que es clave primaria de otra tabla (a través de una clave foránea), la base de datos puede impedir la eliminación o aplicar acciones como la eliminación en cascada.

**3. Dependencias Funcionales**

Las dependencias funcionales son la base del proceso de normalización en las bases de datos, que busca eliminar redundancias y evitar problemas de actualización. Una **dependencia funcional** se da cuando un atributo está completamente determinado por otro atributo. Formalmente, si el atributo A determina el atributo B, se dice que B es funcionalmente dependiente de A.

Por ejemplo, en una tabla "Estudiantes", el "Número de Matrícula" determina de manera única el "Nombre del Estudiante" y su "Programa de Estudios". Aquí, "Nombre del Estudiante" y "Programa de Estudios" dependen funcionalmente del "Número de Matrícula".

La dependencia funcional también se usa para identificar la **forma normal** en la que se debe organizar una base de datos. A medida que los atributos dependen funcionalmente de la clave primaria, las tablas se descomponen para minimizar la redundancia y eliminar las anomalías de inserción, actualización y eliminación.

**4. Clausuras**

La **clausura** de un conjunto de atributos se refiere al conjunto total de atributos que se pueden derivar del conjunto dado mediante las dependencias funcionales. Se utiliza para comprobar si un esquema de base de datos satisface ciertas propiedades de normalización y para determinar claves candidatas.

Por ejemplo, si tenemos una tabla con atributos A, B y C y sabemos que A -> B y B -> C, la clausura de A ({A}+) incluirá A, B y C, ya que a partir de A se pueden determinar todos los demás atributos. La clausura se usa para identificar las claves mínimas necesarias para que se puedan derivar todos los demás atributos de la tabla.

**5. Superclases y Subclases**

El uso de superclases y subclases en el diseño de bases de datos permite modelar jerarquías de entidades. Una **superclase** es una entidad general que agrupa atributos comunes a varias subclases, mientras que las **subclases** son entidades más específicas que heredan atributos de la superclase y pueden tener atributos adicionales propios.

Por ejemplo, en un sistema de gestión de recursos humanos, "Empleado" podría ser una superclase, y "Gerente" y "Contratista" podrían ser subclases. Tanto "Gerente" como "Contratista" heredarían atributos comunes como "Nombre", "Dirección" y "Fecha de Ingreso" de la superclase "Empleado", pero podrían tener atributos específicos como "Bonificación" para "Gerente" o "Duración del Contrato" para "Contratista". Este enfoque facilita la gestión de datos jerárquicos y permite una representación más natural y organizada de las relaciones entre las entidades.

En términos de diseño, las jerarquías de superclases y subclases también ayudan a mejorar la reutilización de los datos y reducen la duplicación de atributos. Además, el uso de superclases y subclases es común en bases de datos orientadas a objetos, donde el objetivo es reflejar la estructura de clases del sistema de software directamente en el modelo de la base de datos.

**Conclusión**

El diseño de una base de datos es un proceso detallado que implica comprender y definir los diferentes componentes y sus relaciones, asegurando la integridad y eficiencia del sistema. Los componentes de una base de datos proporcionan la estructura necesaria para almacenar los datos de manera ordenada y eficiente, mientras que las restricciones garantizan la validez de la información. Las dependencias funcionales y clausuras ayudan a identificar las relaciones entre los datos y a optimizar su organización, evitando redundancias. Por último, el uso de superclases y subclases en el diseño permite una representación jerárquica de los datos, facilitando una gestión más lógica y simplificada.

EJERCICIO

Voy a normalizar la tabla para eliminar redundancias y mejorar la integridad de los datos siguiendo el proceso de normalización hasta la tercera forma normal (3NF).

**1. Análisis Inicial de la Tabla:**

En la tabla proporcionada se observa que hay varias redundancias, como:

* **Clientes (Nombre y Teléfono)**: Se repiten varias veces, lo que podría causar inconsistencias.
* **Servicios**: Los tipos de servicios también se repiten.
* **Placas y Tipo de Carro**: Cada placa está asociada a un único tipo de carro.

Para normalizar la tabla, podemos dividirla en tres o más tablas:

1. **Tabla de Clientes**
2. **Tabla de Servicios**
3. **Tabla de Autos**
4. **Tabla de Registros de Servicios**

**2. Tablas Normalizadas:**

**Tabla 1: Clientes**

Esta tabla contendrá información única de los clientes para evitar redundancia.

A screenshot of a computer

Description automatically generated

**Tabla 2: Autos**

Cada auto está identificado por su placa única.

A screenshot of a computer

Description automatically generated

**Tabla 3: Servicios**

Esta tabla contiene información única de cada tipo de servicio.

A screenshot of a computer

Description automatically generated

**Tabla 4: Registros de Servicios**

Esta tabla relaciona las placas de los autos con los clientes y los servicios que han recibido.

A screenshot of a computer

Description automatically generated

**3. Explicación del Proceso de Normalización:**

* **Primera Forma Normal (1NF)**: Se eliminaron los grupos de datos repetitivos dividiendo la información en registros únicos, asegurando que cada campo contenga solo un valor.
* **Segunda Forma Normal (2NF)**: Separamos los datos en tablas para eliminar dependencias parciales. En este caso, los datos sobre los clientes, autos y servicios fueron separados en diferentes tablas para que cada atributo dependa completamente de la clave primaria de su respectiva tabla.
* **Tercera Forma Normal (3NF)**: Se eliminaron dependencias transitivas, es decir, ninguna columna en una tabla depende de otra columna que no sea la clave primaria de esa tabla. Cada tabla está organizada para que los atributos solo dependan de la clave primaria.

De esta manera, la base de datos normalizada reduce redundancias y asegura la integridad y consistencia de los datos. Además, facilita actualizaciones, eliminaciones y consultas sin generar problemas de inconsistencia.

CONCLUSIÓN

En esta actividad, hemos abordado la normalización de una tabla para eliminar redundancias y garantizar la integridad de los datos. Al analizar la tabla inicial, identificamos la presencia de datos repetitivos, como la información de clientes y servicios, lo cual incrementaba el riesgo de inconsistencias. Para resolver esto, aplicamos el proceso de normalización hasta la Tercera Forma Normal (3NF). Dividimos la información en cuatro tablas principales: Clientes, Autos, Servicios y Registros de Servicios, logrando con ello una estructura eficiente y bien organizada.

La normalización hasta la 3NF nos permitió reducir la duplicación de datos y asegurar que cada atributo dependiera únicamente de la clave primaria de su respectiva tabla. Además, el proceso de descomposición ayudó a establecer relaciones lógicas entre los datos, facilitando su manejo y asegurando la consistencia. En particular, al dividir los datos en tablas independientes, cada tabla tiene una función específica, lo que hace que la base de datos sea más fácil de gestionar y actualizar.

Este ejercicio demostró cómo una buena práctica en el diseño de bases de datos permite mejorar la eficiencia de almacenamiento y la precisión de la información. La normalización es esencial para el desarrollo de sistemas de información robustos, ya que contribuye a mantener la calidad y coherencia de los datos, reduce problemas de actualización y evita redundancias innecesarias. De esta forma, logramos un modelo de datos más limpio, escalable y manejable, lo cual resulta crucial para cualquier sistema que requiera una gestión eficiente y segura de la información.

BIBLIOGRAFÍA

A screenshot of a computer

Description automatically generated