MATERIA: BASES DE DATOS
ESCUELA: ANÁLISIS DE SISTEMAS
ÚLTIMA ACTUALIZACIÓN: FEBRERO 2019

ELABORADO POR: ING. MARIA NEUS

OBJETIVO GENERAL:

-Adquirir los conocimientos necesarios que le permitan modelar, diseñar y normalizar bases de datos relaciónales.

CONTENIDO:

UNIDAD 1 CONCEPTOS GENERALES DE BASES DE DATOS

1.1Definición de Bases de Datos.

Una base de datos es una colección de información organizada de forma que un programa de ordenador pueda seleccionar rápidamente los fragmentos de datos que necesite. Una base de datos es un sistema de archivos electrónico.

Las bases de datos tradicionales se organizan por campos, registros y archivos. Un campo es una pieza única de información; un registro es un sistema completo de campos; y un archivo es una colección de registros. Por ejemplo, una guía de teléfono es análoga a un archivo. Contiene una lista de registros, cada uno de los cuales consiste en tres campos: nombre, dirección, y número de teléfono.

A veces se utiliza DB, de database en inglés, para referirse a las bases de datos

1.2 Tipos de bases de datos.

Bases de datos columnares

Estas son las bases de datos NoSQL más parecidas a las convencionales bases de datos relacionales. Almacenan datos estructurados en columnas individuales (en lugar de tablas).

Estas bases de datos utilizan grupos de columnas. Funcionan bien para datos generados por máquinas, fuentes de datos estructuradas demasiado grandes para ser manejadas por un solo ordenador, y para consultas rápidas de datos.

Si estás pensando en análisis rápido y de precisión de datos-máquina, estas pueden ser los tipos de base de datos ideales. Apache Casandra y Apache HBase son algunas de ellas.

Bases de datos documentales

Estos tipos de bases de datos se basan en el almacenamiento de documentos en lugar de datos estructurados.

Son buenas para datos no estructurados, como texto abierto de una carta o de un correo electrónico, y para datos semi-estructurados como documentos académicos.

Tendrás que fijarte en ellas si estás pensando en análisis de texto de documentos demasiado grandes para las bases de datos convencionales. Algunas de las más conocidas son MongoDB y Apache Couch DB.

Bases de datos gráficas

Estos tipos de bases de datos utilizan una estructura gráfica que es esencialmente un diagrama de las relaciones dentro de los datos, en lugar de tablas.

Son buenos motores de bases de datos para impulsar aplicaciones web que deban proporcionar información muy rápidamente, como las que se utilizan para las compras online y las plataformas de redes sociales.

Necesitarás mirar estos tipos de bases de datos si tu interés principal es una aplicación rápida, y puedes vivir con algunas aproximaciones en análisis.
Algunas de las más conocidas son Neo4J de Neo Technology's y Microsoft Horton.

Key-Value

Estas están diseñadas para desarrollo simple y fácil de aplicaciones.
Son buenas para situaciones donde necesitas trabajar con aplicaciones que se puedan desarrollar rápidamente y donde todas las demás consideraciones son secundarias.
Algunas de las más conocidas son Basho Technologies' Riak y Redis.

XML

Estos tipos de bases de datos utilizan el lenguaje XML, el cual es el lenguaje subyacente de la Web y de otros muchos sistemas de intercambio de información, para definir la estructura de datos.
Son buenas para la gestión de datos que no se puede obtener con cualquier otro tipo de bases de datos, y un buen partido cuando se tiene una gran cantidad de datos en formatos no tradicionales, como vídeo y audio.

1.3 Características.

Una base de datos es una herramienta para recopilar y organizar datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. En las bases de datos, se puede almacenar información sobre personas, productos, pedidos, o cualquier otra cosa.
Existen programas denominados sistemas gestores de bases de datos, abreviado DBMS, que permiten almacenar y posteriormente acceder a los datos de forma rápida y estructurada. Las propiedades de estos DBMS, así como su utilización y administración, se estudian dentro del ámbito de la informática.
Entre las principales características de los sistemas de base de datos podemos mencionar:
Concurrencia
Integridad
Recuperación
Seguridad

Integridad: La integridad de la base de datos se refiere a la validez y la consistencia de los datos almacenados. Normalmente, la integridad se expresa mediante restricciones o reglas que no se pueden violar. Estas restricciones se pueden aplicar tanto a los datos, como a sus relaciones, y es el SGBD quien se debe encargar de mantenerlas.
Seguridad: La seguridad de la base de datos es la protección de la base de datos frente a usuarios no autorizados. Sin unas buenas medidas de seguridad, la integración de datos en los sistemas de bases de datos hace que éstos sean más vulnerables que en los sistemas de ficheros.

Concurrencia: En algunos sistemas de ficheros, si hay varios usuarios que pueden acceder simultáneamente a un mismo fichero, es posible que el acceso interfiera entre ellos de modo que se pierda información o se pierda la integridad. La mayoría de los SGBD gestionan el acceso concurrente a la base de datos y garantizan que no ocurran problemas de este tipo.

Recuperación: Muchos sistemas de ficheros dejan que sea el usuario quien proporcione las medidas necesarias para proteger los datos ante fallos en el sistema o en las aplicaciones. Los usuarios tienen que hacer copias de seguridad cada día, y si se produce algún fallo, utilizar estas copias para restaurarlos.

1.4 Objetivos.

Un objetivo principal de un sistema de base de datos es proporcionar a los usuarios finales una visión abstracta de los datos, esto se logra escondiendo ciertos detalles de como se almacenan y mantienen los datos.
Disminuir la redundancia e inconsistencia de los datos:
Puesto que los archivos y los programas de aplicaciones fueron creados por distintos programadores en un periodo largo, es posible que un mismo dato esté repetido en varios sitios (archivos). Esta redundancia aumenta los costos de almacenamiento y acceso, además de incrementar la posibilidad de que exista inconsistencia en la información.

Reducir la dificultad para tener acceso a los datos:
Supóngase que uno de los gerentes del banco necesita averiguar los nombres de todos los clientes que viven en cierta parte de la ciudad. El gerente llama al departamento de procesamiento de datos y pide que generen la lista correspondiente. Como ésta es una solicitud fuera de lo común no existe un programa de aplicaciones para generar semejante lista. Lo que se trata de probar aquí es que este ambiente no permite recuperar la información requerida en forma conveniente o eficiente.

Evitar el aislamiento de los datos:
Puesto que los datos están repartidos en varios archivos, y éstos pueden tener diferentes formatos, es difícil escribir nuevos programas de aplicaciones para obtener los datos apropiados.

Corregir anomalías en el acceso concurrente:

Para mejorar el funcionamiento del sistema y tener un tiempo de respuesta más corto, muchos sistemas permiten que varios usuarios actualicen la información simultáneamente. En un ambiente de este tipo, la interacción de las actualizaciones concurrentes puede resultar en información inconsistente. Para prevenir estas situaciones debe mantenerse alguna forma de supervisión en el sistema.

Disminuir los problemas de seguridad:
No es recomendable que todos los usuarios del sistema de base de datos pueda tener acceso a toda la información. Por ejemplo, en un sistema bancario, una persona que prepare los cheques de nómina sólo debe poder ver la parte de la base de datos que contenga información de los empleados. No puede consultar información correspondiente a las cuentas de los clientes.

Disminuir los problemas de integridad:
Los valores que se guardan en la base de datos debe satisfacer ciertos tipos de limitantes de consistencia. El sistema debe obligar al cumplimiento de estas limitantes. Esto puede hacerse agregando el código apropiado a los distintos programas de aplicaciones. El problema se complica cuando las limitantes implican varios elementos de información de distintos archivos.
· Acceso concurrente por parte de múltiples usuarios.
· Integridad de los datos.
· Consultas complejas optimizadas.
· Seguridad de acceso y auditoria.
· Respaldo y recuperación.
· Acceso a través de lenguaje de programación estándar.

1.5Ventajas.

Una base de datos es algo más que una mera lista o tabla.
Le permite controlar de verdad los datos, recuperarlos, ordenarlos, analizarlos, resumirlos y elaborar informes. La base de datos puede combinar datos de varios archivos, por lo que nunca habrá que introducir dos veces la misma información. Incluso puede contribuir a que la entrada de datos sea más eficaz y precisa.

Obtener más información de la misma cantidad de data - La base de datos facilita al usuario obtener más información debido a la facilidad que provee esta estructura para proveer datos a los usuarios (si se tiene el privilegio). Ejemplo: comparar un Centro de Cómputos tradicional en COBOL vs uno que utilice una Base de Datos.

Compartir los Datos - Usuarios de distintas oficinas pueden compartir datos si están autorizados. Esto implica que si un dato cambia de contenido como por ejemplo la dirección de un cliente, todos los usuarios que pueden acceder ese dato, verán inmediatamente el cambio efectuado. Ejemplo: Explicar como trabajaba un Centro de Cómputos tradicional con un Sistema Estudiantil que tenga sub-sistemas de Registro, Asistencia Economica, Estudio y Trabajo, Matrícula, etc.

Balance de Requerimientos Conflictivos - Para que la Base de Datos trabaje apropiadamente, necesita de una persona o grupo que se encargue de su funcionamiento. El título para esa posición es Administrador de Base de Datos y provee la ventaja de que Diseña el sistema tomando en mente la necesidad de cada departamento de la empresa. Por lo tanto se beneficia mayormente la empresa aunque algunos departamentos podrían tener leves desventajas debido a su idiosincrasia. Tradicionalmente se diseñaba y programa según la necesidad de cada departamento por separado. Ejemplo: Explicarcomo en diferentes departamentos utilizaban diferentes herramientas y estructuras de datos para su sistema particular y como esto afectaba a los otros departamentos.
Se refuerza la estandarización - Debido a lo que se mencionó previamente, es más fácil estandarizar procesos, formas, nombres de datos, formas, etc.
Redundancia controlada - Debido al sistema tradicional de archivos independientes, los datos se duplicaban constantemente lo cual creaba mucha duplicidad de datos y creaba un problema de sincronización cuando se actualizaba un dato en un archivo en particular. Ejemplo: En el sistema de Registro y de Asistencia Económica pasaba mucho eso. El método que utilizaron para resolver el problema fue el de periódicamente actualizar el archivo de Asistencia Económica, con el archivo de registraduría (principal). Lo cual trae como consecuencia, uso innecesario de los recursos de la computadora. La redundancia se controla, no se elimina por completo.

Consistencia - Al controlarse la redundancia, cuando actualizas un dato, todos los usuarios autorizados de la Base de Datos pueden ver el cambio independientemente de que estén trabajando en distintos sistemas.

Integridad - La base de datos tiene la capacidad de validar ciertas condiciones cuando los usuarios entan datos y rechazar entradas que no cumplan con esas condiciones. El DBA (Data Base Administrator) es responsable de establecer esas validaciones.
Seguridad - El DBA al tener control central de los Datos, la Base de Datos le provee mecanismos que le permiten crear niveles de seguridad para distintos tipos de Usuarios. En COBOL esta opción tendría que programarse.

Flexibilidad y rapidez al obtener datos - Aqui el usuario puede fácilmente obtener información de la Base de Datos con tan solo escribir unas breves oraciones. Esto evita el antiguo y burocrático proceso de llenar una petición al Centro de Cómputos para poder obtener un informe. Ejemplo: Explicar como ocurría ese proceso.

Aumenta la productividad de los programadores - Debido a que los progamadores no se tienen que preocupar por la organización de los datos ni de su validación, se pueden concentrar en resolver otros problemas inmediatos, mejorando de ese modo su productividad.

Mejora el mantenimiento de los programas - Debido a que los datos son independientes de los programas (a diferencia de Cobol), si ocurre un cambio en la estructura de una tabla (archivo), el código no se afecta. Ejemplo: Explicar el problema de Cobol cuando ocurre un cambio de campo en un archivo aún con el uso de librerías.

Independencia de los Datos - Debido a lo que se mencionó previamente, los datos pueden modificarse para por ejemplo mejorar el "performance" de la Base de Datos y como consecuencia, no se tiene que modificar los programas.

EVALUACIÓN

Actividad 1: Realizar un análisis exhaustivo y detallado de las características de las bases de datos.
Actividad 2: Realizar un cuadro comparativo relacionado con los tipos de bases de datos

UNIDAD 2. MANEJADORES DE BASES DE DATOS:

2.1 Tipos

Existen diversos tipos de programas manejadores de base de datos cada uno ideal para cada situación y acorde a nuestras necesidades.
Hablemos entonces un poco sobre ellos. El primero del que hablaremos un poco es:

Microsoft SQL Server
Es un sistema para la gestión de bases de datos producido por Microsoft basado en el modelo relacional.

Características
Soporte de transacciones.
Escalabilidad, estabilidad y seguridad.
Soporta procedimientos almacenados.
Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y los terminales o clientes de la red sólo acceden a la información.
Además permite administrar información de otros servidores de datos.

PostgreSQL
Es un sistema de gestión de base de datos relacional orientada a objetos y libre, publicado bajo la licencia BSD. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyados pororganizaciones comerciales.
Características:
PostgreSQL provee nativamente soporte para:
Números de precisión arbitraria.
Texto de largo ilimitado.
Figuras geométricas (con una variedad de funciones asociadas).
Direcciones IP (IPv4 e IPv6).
Bloques de direcciones estilo CIDR.
Direcciones MAC.
Arrays.

MySQL
Es un sistema de gestión de bases de datos relacional, multi-hilo y multiusuario con más de seis millones de instalaciones. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso.
Características
Amplio subconjunto del lenguaje SQL. Algunas extensiones son incluidas igualmente.
Disponibilidad en gran cantidad de plataformas y sistemas.
Posibilidad de selección de mecanismos de almacenamiento que ofrecen diferente velocidad de operación, soporte físico, capacidad, distribución geográfica, transacciones…
Transacciones y claves foráneas.
Conectividad segura.

Replicación.
Búsqueda e indexación de campos de texto.

Oracle
Es un sistema de gestión de base de datos objeto-relacional (o ORDBMS por el acrónimo en inglés de Object-Relational Data Base Management System), desarrollado por Oracle Corporation. Se considera a Oracle como uno de los sistemas de bases de datos más completos y destacando.
Características
Soporte de transacciones
Estabilidad
Escalabilidad
Soporte multiplataforma.

Microsoft Access
Es un sistema de gestión de bases de datos relacionales para los sistemas operativos Microsoft Windows, desarrollado por Microsoft y orientado a ser usado en un entorno personal o en pequeñas organizaciones. Este programa permite manipular los datos en forma de tablas (formadas por filas y columnas), crear relaciones entre tablas, consultas, formularios para introducir datos e informes para presentar la información.
Características
Tablas para almacenar los datos.
Consultas para buscar y recuperar únicamente los datos que necesita.
Formularios para ver, agregar y actualizar los datos de las tablas.
Informes para analizar o imprimir los datos con un diseño específico.
Páginas de acceso a datos para ver, actualizar o analizar los datos de la base de datos desde Internet o desde una intranet.
Almacenar los datos una vez en una tabla y verlos desde varios lugares.

EVALUACIÓN

Actividad 1: Realizar un mapa conceptual relacionado a los tipos de bases de datos
Actividad 2: Realice un taller en clase relacionado a las características que posee cada uno de los tipos de bases de datos.

UNIDAD 3. MODELO CONCEPTUAL DE DATOS.

3.1 Definición y características

Es una representación abstracta de los datos utilizados por un sistema. Este modelo tiene un alto nivel de independencia, ya que: no presupone ninguna hipótesis sobre la utilización que se hace de ellos; no tiene en cuenta la localización de los datos en los distintos soportes; traduce las elecciones u opciones de gestión fundamentales, en términos de objetos y relaciones; no hay razón para que se modifique su diseño, a menos que se produzca un cambio radical de los requisitos del sistema. Se representa mediante el Diagrama de Entidad-Relación (DER). Algunas características son:

- Es un proceso dirigido completamente a los datos.
- Enfatiza la comprensión de los requerimientos de información del sistema.
- Permite una mejor comunicación entre usuarios, analistas y programadores durante toda la fase del diseño.

- Proporcionará las bases para diseñar una base de datos del sistema correcta, consistente, compartible y flexible.

3.2 Modelo entidad-relación.

En el modelo Entidad-Relación, una entidad agrupa un conjunto de ocurrencias de entidad del mismo tipo. Por tanto:

- Entidad: es un objeto del que se recoge información de interés de cara a la base de datos. Se representan gráficamente mediante un rectángulo. Las entidades pueden ser fuertes o débiles. Las fuertes son las que no dependen de otras entidades para existir, mientras que las entidades débiles siempre dependen de otra entidad sino no tienen sentido por ellas mismas.

- Relación: consiste en una asociación de dos o más entidades. A cada relación se le asigna un nombre para poder distinguirla de las demás y saber su función dentro del modelo entidad-relación. Las relaciones se representas gráficamente con rombos, dentro de ellas se coloca el nombre de la relación.

Conjunto de entidades

Es un grupo de entidades del mismo tipo, es decir, son objetos abstractos o concretos del sistema, con existencia propia y fácilmente identificable. Es una clase de objetos sobre los que se quiere guardar información. Puede ser de tres tipos:

1.Fundamental: Cuando es de interés por sí misma, independiente de cualquier relación.
2.Atributiva: Cuando sirve para completar la descripción de otro tipo de entidad.
3.Asociativa: Su razón de existir es relacionar otras entidades.

Cada miembro de la entidad puede identificarse de manera única por algún medio. El sistema no puede operar sin tener acceso a estos miembros. Cada uno de los miembros puede describirse con uno o más datos.

-Atributos
Es el mínimo elemento lógico de información que se puede encontrar en una entidad o asociación de entidades. El Atributo Clave de una entidad es aquel que puede ser utilizado para identificar cada ocurrencia de la entidad.

-Relaciones
Es una asociación entre entidades, cuya existencia está condicionada por las entidades que relaciona. Tiene asociada una dimensión o número de entidades que relaciona, pudiendo ser 1, 2, ... Para una relación binaria entre A y B, pueden darse de tres casos:

1.Uno a uno (1-1): en la cual una ocurrencia de A no está en relación más que con una ocurrencia de B y, cada ocurrencia de B no está en relación más que con una ocurrencia de A.
Uno a muchos (1-n): en la cual una ocurrencia de A está en relación con una o muchas ocurrencias de B y, cada ocurrencia de B no está en relación más que con una ocurrencia de A.
Muchos a muchos (n-n): en la cual una ocurrencia de A está en relación con una o muchas ocurrencias de B y, cada ocurrencia de B está en relación con una o muchas ocurrencia de A.

La relación representa la memoria del sistema, ya que representa algo que debe ser recordado por el sistema. No se muestran las relaciones que se pueden calcular o derivar.

3.3 Bases De Datos Relacionales

- Diseño de bases de datos relacionales.

A continuación, presentamos el siguiente ejemplo. Nombre de la tabla: Representantes

Atributos. Campos: Nombre, Nacionalidad, Institución. Por lo tanto, como se explicó anteriormente sobre el grado de relación, en el ejemplo presentado identificamos una relación de grado 3, ya que son 3 atributos o campos y tiene 4 tuplas o registros. Podemos observar además que la tabla (relación) debe reunir un conjunto de requisitos:

- No puede haber filas duplicadas, es decir, todas las tuplas tienen que ser distintas.
- El orden de las filas es irrelevante.
- La tabla es plana, es decir, en el cruce de una fila y una columna sólo puede haber un valor (no se admiten atributos multivaluados).

Para que pueda cumplirse el primer requisito, debe ser posible definir lo que se conoce como clave primaria. Se llama clave primaria a un campo o a una combinación de campos que identifica de forma única a cada fila de una tabla. Una clave primaria comprende de esta manera una columna o conjunto de columnas. No puede haber dos filas en una tabla que tengan la misma clave primaria. En otras palabras, una clave primaria debe identificar unívocamente a todas las posibles filas de una tabla y no solo a las filas que se encuentran en un momento determinado.

Complementando el concepto de claves, están las llamadas claves foráneas (foreignkey) que identifican una columna o grupo de columnas en una tabla (tabla hija o referendo) que se refiere a una columna o grupo de columnas en otra tabla (tabla maestra o referenciada). Las columnas en la tabla hija deben ser la clave primaria u otra clave candidata en la tabla referenciada. De esta manera es posible enlazar las diferentes tablas que definan una base de datos.

EVALUACIÓN

Una cadena de restaurantes ha relevado información acerca de los clientes y suspreferencias. De cada persona, identificada por su cédula de identidad, se conoce su nombre,el restaurant que frecuenta más y las comidas que más le gustan.
De cada restaurant, identificado por su nombre, se conoce las comidas que preparan. De cada comida se conoce su nombre, que la identifica, el tiempo de preparación y losingredientes principales.
a. Diseñar un MER que represente la realidad anterior
b. Modificar el MER anterior para representar la información anterior considerando las siguientes restricciones.
Un restaurante no vende más de 10 comidas.
Una persona frecuenta varios restaurantes.
A una persona no le gusta una comida por sí sola sino cómo la sirven en determinadosrestaurantes.

UNIDAD 4. SQL PARA EL MANEJO DE BASES DE DATOS

4.1 Comandos básicos SQL para el manejo de bases de datos

Definiendo cómo es almacenada la información.
CREATE DATABASE se utiliza para crear una nueva base de datos vacía.
DROP DATABASE se utiliza para eliminar completamente una base de datos existente.
CREATE TABLE se utiliza para crear una nueva tabla, donde la información se almacena realmente.
ALTER TABLE se utiliza para modificar una tabla ya existente.
DROP TABLE se utiliza para eliminar por completo una tabla existente.
Manipulando los datos.
SELECT se utiliza cuando quieres leer (o seleccionar) tus datos.
INSERT se utiliza cuando quieres añadir (o insertar) nuevos datos.
UPDATE se utiliza cuando quieres cambiar (o actualizar) datos existentes.
DELETE se utiliza cuando quieres eliminar (o borrar) datos existentes.

REPLACE se utiliza cuando quieres añadir o cambiar (o reemplazar) datos nuevos o ya existentes.
TRUNCATE se utiliza cuando quieres vaciar (o borrar) todos los datos de la plantilla.

Ejemplos:

```
CREATE DATABASE mydb;
USE mydb;
CREATE TABLE mitabla ( id INT PRIMARY KEY, nombre VARCHAR(20) );
INSERT INTO mitabla VALUES ( 1, 'Will' );
INSERT INTO mitabla VALUES ( 2, 'Marry' );
INSERT INTO mitabla VALUES ( 3, 'Dean' );
SELECT id, nombre FROM mitabla WHERE id = 1;
UPDATE mitabla SET nombre = 'Willy' WHERE id = 1;
SELECT id, nombre FROM mitabla;
DELETE FROM mitabla WHERE id = 1;
SELECT id, nombre FROM mitabla;
DROP DATABASE mydb;
SELECT count(1) from mitabla; da el número de registros en la tabla
```

La sintaxis básica de una consulta SELECT es la siguiente (los valores opcionales van entre corchetes):

```
SELECT [ ALL / DISTINC ] [ * ] / [ListaColumnas_Expresiones] AS [Expresion]
FROM                                                  Nombre_Tabla_Vista
WHERE                                                 Condiciones
ORDER BY ListaColumnas [ ASC / DESC ]
```

EVALUACIÓN

Realizar las tablas y relaciones de una base de datos a partir de la siguiente información
El ministerio de la salud desea mantener un sistema de información relativo a hospitales.
A continuación se detalla lo que se desea modelar:
HOSPITAL, con los datos:

- código, que lo identifica.
- nombre
- dirección
- teléfono
- cantidad de camas

SALA, con los datos:

- código
- nombre

• cantidad de camas

MEDICO, con los datos:

• cédula de identidad
• nombre
• especialidad

LABORATORIO, con los datos:

• código, que lo identifica.
• nombre
• dirección
• teléfono

PACIENTE, con los datos:

• cédula de identidad
• número de registro
• número de cama
• nombre
• dirección
• fecha de nacimiento
• sexo

DIAGNOSTICO, con los datos:

• código, que lo identifica.
• tipo
• complicaciones

Además se sabe que:

• Cada hospital tiene varias salas. Cada una de ellas pertenece a un solo hospital. En distintos hospitales puede haber salas con el mismo código, pero esto no puede ocurrir dentro de un hospital.
• Cada médico trabaja en un único hospital

UNIDAD 5 NORMALIZACION

5.1 Funciones de la normalización

Las bases de datos relacionales se normalizan para:
Evitar la redundancia de los datos.
Disminuir problemas de actualización de los datos en las tablas.
Proteger la integridad de los datos.
Para que las tablas de nuestra BD estén normalizadas deben cumplir las siguientes reglas:
Cada tabla debe tener su nombre único.
No puede haber dos filas iguales.
No se permiten los duplicados.
Todos los datos en una columna deben ser del mismo tipo.

5.2 Niveles de normalización

Existen 3 niveles de normalización que deben respetarse para poder decir que nuestra BDs, se encuentra NORMALIZADA, es decir, que cumple con los requisitos naturales para funcionar optimamente y no perjudicar el rendimiento por mala arquitectura.

Estas 3 reglas de Normalización se les conoce como las 3 FORMAS NORMALES.
Teoría sobre normalización te puedes encontrar en cualquier libro de bases de datos, lo que es difícil encontrar son ejemplos prácticos donde puedas entender los conceptos de las reglas de normalización.
A continuación te explico las formas normales con un ejemplo real.
Una de ellas es el Instituto ICONOS, donde se tienen materias en común para diferentes planes de estudios, por ejemplo en el área de web se tiene una materia para enseñar bases de datos con MySQL y otra para enseñar programación backend con PHP, ambas materias se imparten tanto en la Licenciatura en Diseño Digital, como en la Maestría en Medios Virtuales, la diferencia entre los niveles de estudios, es el número de horas y la carga de contenidos.
Pensemos que tenemos dos alumnos que cursarán ambas materias: Juanito en maestría y Pepito en licenciatura, nuestro modelo de datos podría quedar de la siguiente manera:
Sin Normalizar:
ALUMNOS

alumno	estudio_nivel	estudio_nombre	materia_1	materia_2
Juanito	Maestría	Medios Virtuales	MySQL	PHP
Pepito	Licenciatura	Diseño Digital	MySQL	PHP

Como podemos ver en la tabla anterior, tenemos los registros de ambos estudiantes con ambas materias asignadas, pero esto es poco funcional, imaginemos que cada estudiante tuviera más materias en su horario, eso significaría, agregarle más columnas a cada alumno, lo que no es muy óptimo.
Primera Forma Normal:
NO repetir campos en las tablas (atributos atómicos).
ALUMNOS

alumno_id	alumno_nombre	estudio_nivel	estudio_nombre	materia

1	Juanito	Maestría	Medios Virtuales	MySQL
1	Juanito	Maestría	Medios Virtuales	PHP
2	Pepito	Licenciatura	Diseño Digital	MySQL
2	Pepito	Licenciatura	Diseño Digital	PHP

Al aplicar la primer forma normal hemos generado un identificado para cada alumno y un registro por materia asignada, hemos duplicado información, sin embargo hemos conservado la integridad de las columnas de la información lo que es más óptimo que el modelo anterior, sin embargo podemos mejorarlo con la segunda forma normal.

Segunda Forma Normal:
Se debe aplicar la 1FN. Cada campo de la tabla debe depender de una clave única, si tuviéramos alguna columna que se repite a lo largo de todos los registros, dichos datos deberían atomizarse en una nueva tabla.

ALUMNOS

alumno_id	alumno_nombre	estudio_nivel	estudio_nombre
1	Juanito	Maestría	Medios Virtuales
2	Pepito	Licenciatura	Diseño Digital

MATERIAS

materia_id	alumno_id	materia_nombre
1	1	MySQL

2	1	PHP
3	2	MySQL
4	2	PHP

Al aplicar la segunda forma normal, hemos evitado la duplicación de registros y hemos separado la información de los alumnos de la relación que guardan con las materias generando una segunda tabla, sin embargo dicha tabla puede mejorarse con la tercer forma normal o su versión mejorada la forma de Boyce-Codd.

Tercera Forma Normal:

Se debe aplicar la 1FN y 2FN. Los campos que NO son clave NO deben tener dependencias.

Forma Normal Boyce-Codd (FNBC):

Se debe aplicar la 1FN, 2FN y 3FN. Es una versión mejorada de la 3FN. Los campos que NO son clave NO deben tener dependencias. Los campos que NO dependan de la clave se deben eliminar.

ALUMNOS

alumno_id	alumno_nombre	estudio_id
1	Juanito	1
2	Pepito	2

ESTUDIOS

estudio_id	estudio_nivel	estudio_nombre
1	Maestría	Medios Virtuales
2	Licenciatura	Diseño Digital

MATERIAS

materia_id	alumno_id	materia_nombre
1	1	MySQL
2	1	PHP
3	2	MySQL
4	2	PHP

Con la ayuda de la tercer forma hemos sacado la información de los planes de estudio de la información principal de los alumnos, lo que asegura una mejor integridad de los datos, permitiendo que el número de estudios crezcan y que los estudiante puedan matricularse a más de un estudio sin tener que desordenar el modelo de datos.

Cuarta Forma Normal:
Se debe aplicar la FNBC. La 4FN aplica únicamente para relaciones M a M, y nos ayuda a eliminar la redundancia de información generada por dicho tipo de relación.

ALUMNOS

alumno_id	alumno_nombre	estudio_id
1	Juanito	1
2	Pepito	2

ESTUDIOS

estudio_id	estudio_nivel	estudio_nombre

1	Maestría	Medios Virtuales
2	Licenciatura	Diseño Digital

MATERIAS

materia_id	materia_nombre
1	MySQL
2	PHP

MATERIAS X ALUMNO

mxa_id	alumno_id	materia_id
1	1	1
2	1	2
3	2	1
4	2	2

Con la cuarta forma hemos logrado separar la relación que guardan los alumnos con sus respectivas materias asignadas, separándolas en un catálogo independiente de materias, y guardando la relación entre alumnos y materias en otra tabla pivote que sólo guarde la relación entre ambas entidades con un registro único.

Quinta Forma Normal:

Se debe aplicar la 1FN, 2FN, 3FN y 4FN. Existe otro nivel de normalización que se aplica con poca frecuencia y en la mayoría de los casos no es necesario, para obtener la mejor funcionalidad de nuestra estructura de datos. Su principio sugiere:
La tabla original debe ser reconstruida desde las tablas resultantes en las cuales ha sido partida.
Los beneficios de aplicar la 5FN asegura que no se haya creado ninguna columna extraña en las tablas y que su estructura sea del tamaño justo que tiene que ser.
Es una buena práctica aplicar la 5FN, cuando tenemos una extensa y compleja estructura de datos, en modelos pequeños no se recomienda usar.
En síntesis la quinta forma, nos dice que en modelos muy grandes donde tenemos muchas relaciones y entidades, nos sugiere que una vez que hayamos terminado la normalización de nuestro modelo, lo revisemos una vez más en busca de posibles errores de lógica en la normalización, para efectos de nuestro ejemplo que es un modelo sencillo no aplicaremos la quinta forma normal.

Más detalles en el siguiente linkhttps://anmacosa200.webnode.com.co/curso-web-de-bases-de-datos-por-angela-sandoval/cuales-son-las-ventajas-de-utilizar-bases-de-datos/

EVALUACION

Pauta 1: Diseñe un esquema de relación que sea fácil de explicar su significado
Pauta 2: Diseñe los esquema de relaciones de base de modo que no hayan anomalías de inserción, eliminación o modificación en las relaciones.
Pauta 3: Hasta lo posible, evite incluir en una relación atributos cuyos valores puedan ser nulos.

REFERENCIAS

https://anmacosa200.webnode.com.co/curso-web-de-bases-de-datos-por-angela-sandoval/cuales-son-las-ventajas-de-utilizar-bases-de-datos/

http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro14/11_caracteristicas_de_la_base_de_datos.html

https://mariadb.com/kb/es/basic-sql-statements/

http://modeloconceptualdedatos.blogspot.com/

http://oftgu.eco.catedras.unc.edu.ar

http://unidad1grupo2551.blogspot.com/2013/10/11-objetivos-de-las-bases-de-datos.html