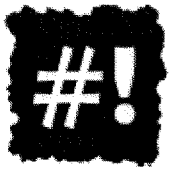


Preparador Informática

www.preparadorinformatica.com

TEMA 40. INFORMÁTICA

DISEÑO DE BASES DE DATOS RELACIONALES

TEMA 40 INF: DISEÑO DE BASES DE DATOS RELACIONALES.

1. INTRODUCCIÓN

En otros temas se ha abordado cómo es una base de datos relacional y se ha estudiado un lenguaje, el SQL, que nos proporciona mecanismos para crear estas bases de datos, así como para actualizarlas y consultarlas. Sin embargo, es necesario complementar estos conocimientos con un aspecto que es fundamental para poder utilizar adecuadamente la tecnología de las bases de datos relacionales: **el diseño**. Por ejemplo, cómo se puede decidir qué relaciones debe tener una base de datos determinada o qué atributos deben presentar las relaciones, qué claves primarias y qué claves foráneas se deben declarar, etc. La tarea de tomar este conjunto de decisiones recibe el nombre de diseñar la base de datos.

Éste será el objeto de estudio del presente tema, que tratará el diseño de bases de datos para el caso específico del modelo relacional. Concretamente, se abordará en qué consiste el diseño de una base de datos, se analizarán las etapas en las que se puede descomponer y describiremos con detalle las etapas del diseño conceptual, lógico y físico de una base de datos relacional.

2. DISEÑO DE BASES DE DATOS RELACIONALES.

El **diseño de una base de datos** consiste en definir la estructura de los datos que debe tener un sistema de información determinado. Realizar un buen diseño puede ser un proceso complejo sobre todo si queremos que nuestra base de datos sea coherente y eficiente. Por lo que necesitamos una metodología muy potente para evitar posibles futuros problemas de redundancia de datos (repetición innecesaria de información), incoherencia de datos o pérdida de dependencias funcionales. La metodología que usamos para un buen diseño de la base de datos suele seguir por regla general unas fases concretas en el proceso de diseño, definiendo para ello el modelo conceptual, el lógico y el físico.

En el **diseño conceptual** se hace una descripción de alto nivel de la estructura de la base de datos, independientemente del SGBD (Sistema Gestor de Bases de Datos) que se vaya a utilizar. Su objetivo es describir el contenido de información de la base de datos y no las estructuras de almacenamiento que se

necesitarán para manejar dicha información. En el caso de las bases de datos relacionales se corresponde con el modelo Entidad-Relación.

El **diseño lógico** parte del resultado del diseño conceptual y da como resultado una descripción de la estructura de la base de datos en términos de las estructuras de datos que puede procesar un tipo de SGBD. El diseño lógico depende del tipo de SGBD que se vaya a utilizar, se adapta a la tecnología que se debe emplear, pero no depende del producto concreto. En el caso de bases de datos convencionales relacionales, el diseño lógico consiste en definir las tablas que existirán, las relaciones entre ellas, normalizarlas, etc...

El **diseño físico** parte del lógico y da como resultado una descripción de la implementación de una base de datos en memoria secundaria: las estructuras de almacenamiento y los métodos utilizados para tener un acceso eficiente a los datos. Aquí el objetivo es conseguir una mayor eficiencia, y se tienen en cuenta aspectos concretos del SGBD sobre el que se vaya a implementar.

3. DISEÑO CONCEPTUAL: MODELO ENTIDAD-RELACIÓN (E/R).

El modelo E/R es uno de los modelos de datos conceptuales más extendidos en las metodologías de diseño de base de datos. Fue propuesto por Peter P. Chen en 1976 aunque existen otros autores que han investigado y propuesto posteriormente nuevas aportaciones sobre el mismo.

El modelo E/R puede ser usado como una base para una vista unificada de los datos adoptando el enfoque más natural del mundo real que consiste en entidades y relaciones. Se basa en llegar a un nivel de abstracción que permita definir los elementos que componen nuestra base de datos. Se componen básicamente de **Entidades** (E) y **Relaciones** (R), donde ambas contienen atributos que almacenan información.

3.1. ENTIDADES

Una entidad es un objeto real o abstracto sobre el que queremos almacenar información en nuestra base de datos. (Ej: una persona). La estructura genérica de un conjunto de entidades con las mismas características se denomina tipo de entidad. Existen dos clases de entidades: **regulares**, que tienen existencia por

sí mismas, y **débiles** cuya existencia depende de otra entidad. Las entidades deben cumplir las siguientes tres reglas:

- Tienen que tener existencia propia.
- Cada ocurrencia de un tipo de entidad debe poder distinguirse de las demás.
- Todas las ocurrencias de un tipo de entidad deben tener las mismas características (atributos).

La representación gráfica de un tipo de entidad regular es un rectángulo etiquetado con el nombre del tipo de entidad. Un tipo de entidad débil se representa con dos rectángulos concéntricos con su nombre en el interior.

3.2. RELACIONES

Una relación es una asociación entre entidades, sin existencia propia en el mundo real que estamos modelando, pero necesaria para reflejar las interacciones existentes entre entidades. La estructura genérica de un conjunto de relaciones denomina tipo de entidad.

La relación puede ser regular, si asocia tipos de entidad regulares, o débil, si asocia un tipo de entidad débil con un tipo de entidad regular. Dentro de las relaciones débiles se distinguen la dependencia en existencia y la dependencia en identificación.

Se dice que la dependencia es en existencia cuando las ocurrencias de un tipo de entidad débil no pueden existir sin la ocurrencia de la entidad regular de la que dependen. Se dice que la dependencia es en identificación cuando, además de lo anterior, las ocurrencias del tipo de entidad débil no se pueden identificar sólo mediante sus propios atributos, sino que se les tiene que añadir el identificador de la ocurrencia de la entidad regular de la cual dependen.

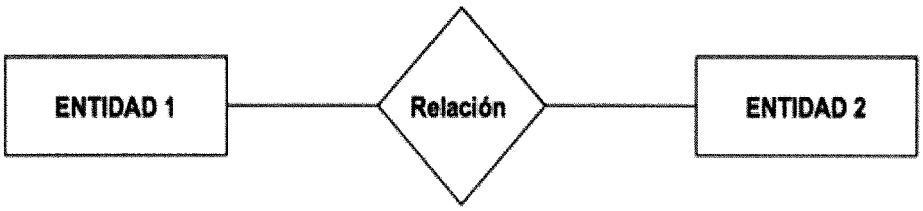
Una relación se caracteriza por:

- **Nombre:** que lo distingue unívocamente del resto de relaciones del modelo.

- **Tipo de correspondencia:** es el número máximo de ocurrencias de cada tipo de entidad que pueden intervenir en una ocurrencia de la relación que se está tratando. Conceptualmente se pueden identificar tres clases de relaciones:
  - Relaciones 1:1: Cada ocurrencia de una entidad se relaciona con una y sólo una ocurrencia de la otra entidad.
  - Relaciones 1:N: Cada ocurrencia de una entidad puede estar relacionada con cero, una o varias ocurrencias de la otra entidad.
  - Relaciones M:N: Cada ocurrencia de una entidad puede estar relacionada con cero, una o varias ocurrencias de la otra entidad y cada ocurrencia de la otra entidad puede corresponder a cero, una o varias ocurrencias de la primera.

Se representa por un rombo unido a las entidades relacionadas por dos líneas rectas a los lados. El tipo de correspondencia se representa gráficamente con una etiqueta 1:1, 1:N o M:N, cerca de alguno de los vértices del rombo.

1:1/1:N/N:



El **grado de una relación** es el número de entidades que participan en una relación. En función del grado se pueden establecer diferentes tipos de relaciones: unarias, binarias, ternarias y n-arias.

3.3. ATRIBUTOS

Es una propiedad o característica de un tipo de entidad o de un tipo de relación. Se trata de la unidad básica de información que sirve para identificar o describir la entidad. Un atributo se define sobre un dominio. Cada tipo de entidad ha de tener un conjunto mínimo de atributos que identifiquen unívocamente cada ocurrencia del tipo de entidad. Este atributo o atributos se denomina identificador principal o clave primaria. Ya que puede haber varias claves y necesitamos elegir una, lo haremos atendiendo a estas normas:

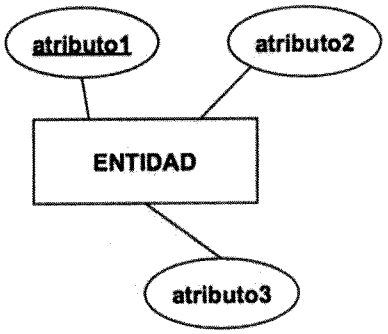
- Que sea única.
- Que se tenga pleno conocimiento de ella.

- Que sea mínima, ya que será muy utilizada por el gestor de base de datos.

Se pueden definir restricciones sobre los atributos, según las cuales un atributo puede ser:

- Univaluado, atributo que sólo puede tomar un valor para todas y cada una de las ocurrencias del tipo de entidad al que pertenece.
- Multivaluado: Es aquel que tiene más de una ocurrencia para un determinado valor de la clave.
- Obligatorio, atributo que tiene que tomar al menos un valor para todas y cada una de las ocurrencias del tipo de entidad al que pertenece.

Un atributo se representa mediante una elipse, con su nombre dentro, conectada por una línea al tipo de entidad o relación.

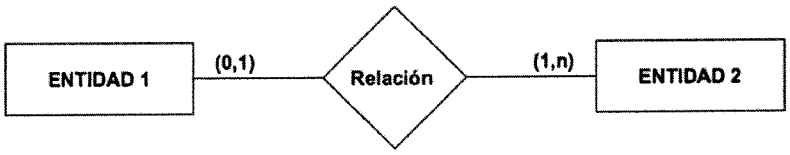


Para representar las claves primarias se subrayarán aquellos atributos que la formen.

3.4. MODELO E/R EXTENDIDO

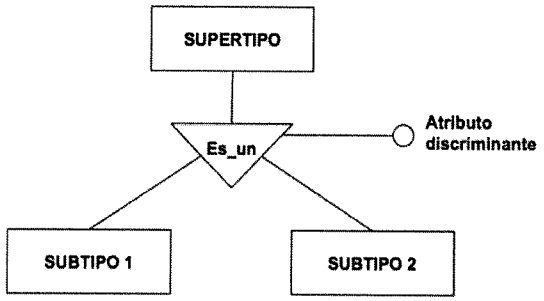
Además de los elementos comentados en apartados anteriores, existen extensiones del modelo entidad/relación que incorporan determinados conceptos o mecanismos de abstracción para facilitar la representación de ciertas estructuras del mundo real.

La **cardinalidad**: representa la participación en la relación de cada una de las entidades afectadas, es decir, el número máximo y mínimo de ocurrencias de un tipo de entidad que pueden estar interrelacionadas con una ocurrencia de otro tipo de entidad. La cardinalidad máxima coincide con el tipo de correspondencia. La representación gráfica de las cardinalidades se realiza mediante una etiqueta del tipo (0,1), (1,1), (0,n) o (1,n), que se coloca en el extremo de la entidad que corresponda. Si se representan las cardinalidades, la representación del tipo de correspondencia es redundante.



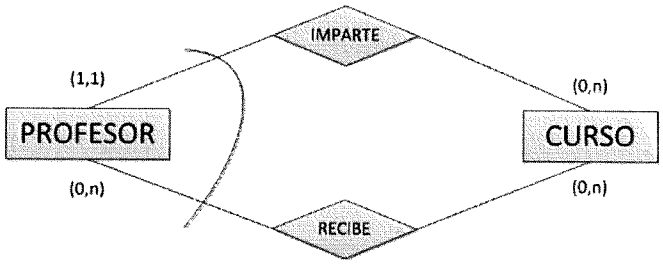
La **generalización** permite abstraer un tipo de entidad de nivel superior (supertipo) a partir de varios tipos de entidad (subtipos); en estos casos los atributos comunes y relaciones de los subtipos se asignan al supertipo. Por ejemplo: generalizar los tipos hombre y mujer obteniendo el supertipo persona.

La **especialización** es la operación inversa a la generalización, en ella un supertipo se descompone en uno o varios subtipos, los cuales heredan todos los atributos y relaciones del supertipo, además de tener los suyos propios. Un ejemplo es el caso del tipo persona, del que se pueden obtener los subtipos licenciado, ingeniero y doctor.

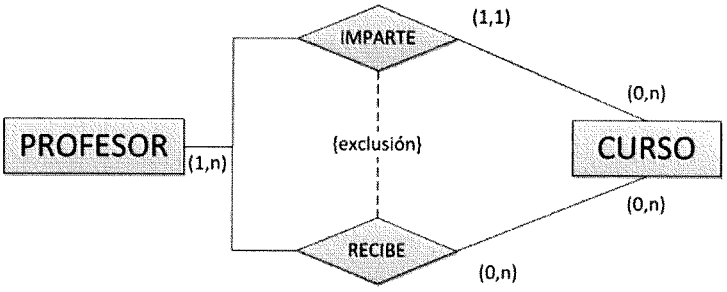


La existencia de supertipos y subtipos, en uno o varios niveles, da lugar a una jerarquía, que permitirá representar una restricción del mundo real. La representación de las jerarquías se realiza mediante un triángulo invertido, con la base paralela al rectángulo que representa el supertipo y conectando a éste y a los subtipos. Si la división en subtipos viene determinada en función de los valores de un atributo discriminante, éste se representará asociado al triángulo que representa la relación.

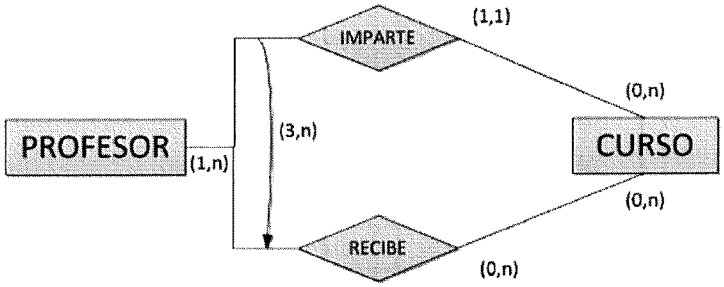
Exclusividad: Entre dos tipos de entidades puede existir más de un tipo de relación. Aunque se dice que una relación es exclusiva cuando la existencia de una relación entre dos tipos de entidades implica la no existencia de las otras relaciones.



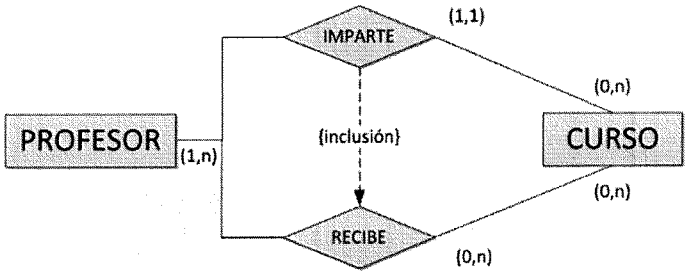
Exclusión: Entre dos tipos de relaciones R1 y R2. Significa que E1 está relacionada con E2 bien mediante R1, o bien mediante R2 pero que no pueden darse ambas relaciones simultáneamente.



Inclusividad entre dos tipos de relaciones R1 y R2 respecto a la entidad E1. Para que la entidad E1 participe en la relación R2 debe participar previamente en la relación R1.



Inclusión entre dos tipos de relaciones R1 y R2. Para que la entidad E1 participe en la relación R2 con E2 debe participar previamente en la relación R1.



4. DISEÑO LÓGICO

Una vez que nuestro esquema conceptual haya sido revisado, modificado y probado para verificar que se cumplen adecuadamente todos y cada uno de los requerimientos del problema a modelar entonces nos encontramos ante el punto de partida para la siguiente fase, el diseño lógico de la base de datos.

El **diseño lógico** consistirá en la construcción de un esquema de la información relativa al problema, basado en un modelo de base de datos concreto. El esquema conceptual se transformará en un esquema lógico que utilizará los elementos y características del modelo de datos en el que esté basado el SGBD, para implementar nuestra base de datos.

Para esta transformación será necesario realizar una serie de pasos que resumiendo son simplificar nuestro diagrama E/R, transformarlo al modelo relacional, luego aplicaremos normalización y obtendremos lo que se conoce como el paso a tablas del esquema conceptual o, lo que es lo mismo, el esquema lógico. Desde ese momento, basándonos en este esquema, podremos llevarnos nuestra base de datos a cualquier SGBD basado en el modelo relacional e implementarla físicamente.

4.1. SIMPLIFICACIÓN PREVIA DE DIAGRAMAS.

Existe un conjunto de procedimientos y normas que es necesario aplicar a nuestros diagramas E/R para que su transformación al modelo lógico basado en el modelo relacional, sea correcta y casi automática. Si se aplican correctamente estas pautas, conseguirás que el proceso de transformación sea fácil y fiable. Las transformaciones de las que estamos hablando son las siguientes:

- Transformación de relaciones n-arias en binarias.
- Eliminación de relaciones cíclicas.
- Reducción a relaciones jerárquicas (uno a muchos).
- Conversión de entidades débiles en fuertes.

4.2. PASO DEL DIAGRAMA E/R AL MODELO RELACIONAL

Para realizar el paso al modelo de datos Relacional se deben tener en cuenta:

Toda entidad se transforma en una relación o tabla: Cada tipo de entidad se debe convertir a una relación, es decir, será necesario crear una tabla para cada entidad que parezca en el diagrama E/R.

Todo atributo se transforma en columna dentro de una tabla: Cada atributo de una entidad se debe transformar en una columna en la relación a la que ha dado lugar la entidad. El atributo o atributos principales pasan a ser la clave primaria de la relación y, por tanto, se debe especificar que son no nulos. El resto pasan a ser otras columnas de la tabla pudiendo ser o no ser nulos.

Entidad débil a fuerte: Convertir una entidad débil en fuerte consiste en hacer que cada entidad débil genere una tabla que incluirá todos sus atributos, añadiéndose a ésta los atributos que son clave primaria de la entidad fuerte con la que esté relacionada. Estos atributos añadidos se constituyen como clave foránea que referencia a la entidad fuerte. Seguidamente, se escogerá una clave primaria para la tabla creada.

Transformación de relaciones: Dependen del tipo de relación que se trate:

- Relaciones **N:M.** Se transforman en una relación que tendrá como clave primaria la concatenación de los atributos principales de cada una de las entidades que relacionan que serán clave ajena respecto a cada una de las tablas donde ese atributo es clave primaria.

- Relaciones **1:N.** Existen dos soluciones y siempre teniendo en cuenta en ambas soluciones las referencias ajenas:
  - Propagar el atributo principal de la entidad que tiene cardinalidad máxima 1 a la que tienen N, y hacer desaparecer la relación como tal.
  - Transformarla en una relación como si fuese una de tipo N:M. Sólo es recomendable en tres casos, 1) si es posible que aparezcan muchos nulos porque existen pocos elementos relacionados, 2) cuando se prevé que dicha relación pasará en un futuro a ser de tipo N:M y 3) cuando la relación tiene atributos propios.
- Las relaciones **1:1**. Puesto que se trata de una particularización de cualquiera de los dos casos anteriores, se pueden igualmente o generar una nueva tabla o propagar la clave en función de la cardinalidad de las entidades.

Atributos de relaciones: Si la relación se transforma en una tabla, todos sus atributos pasar a ser columnas de la tabla. Si alguno de los atributos de la relación sea principal, entonces deberá ser incluido como parte de la clave primaria en dicha tabla.

4.3. NORMALIZACIÓN. FORMAS NORMALES.

La teoría de la normalización tiene como fundamento el concepto de formas normales diciendo que una relación está en una determinada forma normal si satisface un cierto conjunto de restricciones.

Es necesario que, al realizar la estructura de una base de datos, esta sea flexible. La flexibilidad está en el hecho que se puedan agregar datos al sistema posteriormente sin tener que rescribir lo que ya se tiene.

La eficiencia se refiere al hecho de que no se tiene duplicación de datos, y tampoco tienen grandes cantidades de "celdas vacías". El objetivo principal del diseño de bases de datos es generar tablas que modelan los registros en los que se guardará la información. Es importante que esta información se almacene sin redundancia para que se pueda tener una recuperación rápida y eficiente de los datos. Se puede decir que los principales objetivos de la normalización son:

- Controlar la redundancia de la información.
- Evitar pérdidas de información.

- Capacidad para representar toda la información.
- Mantener la consistencia de los datos.

La normalización se realiza en una etapa posterior a la correspondencia entre el esquema conceptual y el esquema lógico, para que se eliminen las dependencias entre atributos no deseadas.

Las ventajas de la normalización son las siguientes:

- Evita anomalías en inserciones, modificaciones y borrados.
- Mejora la independencia de datos.
- No establece restricciones artificiales en la estructura de los datos.

Uno de los conceptos fundamentales en la normalización es el de **dependencia funcional.** Una dependencia funcional es una relación entre atributos de una misma relación (tabla). La dependencia funcional es una noción semántica. Si hay o no dependencias funcionales entre atributos lo determinan los modelos mentales del usuario y las reglas de negocio de la organización o empresa para la que se desarrolla el sistema de información. Cada dependencia funcional es una clase especial de regla de integridad y representa una relación de uno a muchos. Por ejemplo: DNI → Nombre y Apellidos.

También es necesario conocer los conceptos de: (X, Y, Z atributos.)

Dependencia funcional reflexiva: Si "Y" está incluido en "X" entonces X → Y
Por ejemplo: Si dirección y nombre están incluidos en DNI entonces con el DNI se puede recuperar la dirección y el nombre.

Dependencia Funcional Aumentativa: Si "X" → "Y" entonces "XZ" → «YZ"
Por ejemplo: DNI → Nombre

DNI, Dirección → Nombre, Dirección

Dependencia Funcional Transitiva: Si "X" → "Y" → Z" entonces "X" → "Z"
Fecha de nacimiento → Edad
Edad → Conducir
Fecha de nacimiento → Edad → Conducir

Dependencia Funcional Completa: Se dice que el atributo Y de la relación R es por completo dependiente funcionalmente del atributo X de R si depende

funcionalmente de X y no depende funcionalmente de ningún subconjunto propio de X.

La normalización se lleva a cabo en una serie de pasos. Cada paso corresponde a una forma normal. Conforme se va avanzando en la normalización, las relaciones tienen un formato más estricto (más fuerte) y, por lo tanto, son menos vulnerables a las anomalías de actualización. El modelo relacional sólo requiere un conjunto de relaciones en primera forma normal. Las restantes formas normales son opcionales. Sin embargo, para evitar las anomalías de actualización, es recomendable llegar al menos a la tercera forma normal.

4.3.1. PRIMERA FORMA NORMAL (1FN)

Una tabla está en Primera Forma Normal (1FN o FN1) sí, y sólo sí, todos los atributos de la misma contienen valores atómicos, es decir, no hay grupos repetitivos. Dicho de otra forma, estará en 1FN si los atributos no clave, dependen funcionalmente de la clave.

Por lo general la mayoría de las relaciones cumplen con estas características, así que podemos decir que la mayoría de las relaciones se encuentran en la primera forma normal.

4.3.2. SEGUNDA FORMA NORMAL (2FN)

Una relación está en 2FN si y sólo si está en 1FN y todos los atributos no clave dependen por completo de la clave primaria. De acuerdo con esta definición, cada tabla que tiene un atributo único como clave, está en segunda forma normal.

4.3.3. TERCERA FORMA NORMAL (3FN)

Una relación R está en 3FN si y solo si está en 2FN y todos sus atributos no clave dependen no transitivamente de la clave primaria. Dicho de otro modo, si y sólo si los atributos no clave son mutuamente independientes y son dependientes por completo de la clave primaria.

4.3.4. TERCERA FORMA NORMAL DE BOYCE/CODD (FNBC)

El problema de la 3FN es que no maneja relaciones que tenga varias claves candidatas, donde las claves candidatas sean compuestas y donde las claves

candidatas se solapan. Por ello, se define la FNBC para el caso de que existan más de una clave candidata y que se cumplan las condiciones anteriores.

Para esto debemos conocer el concepto de determinante que se define como el atributo del cual depende funcionalmente algún otro atributo. Así pues, se dice que una relación está en FNBC si y sólo si todo determinante es una clave candidata. En el caso en el que no se den las condiciones dichas anteriormente, o no exista más de una clave candidata la FNBC es equivalente a la 3FN.

4.3.5. CUARTA FORMA NORMAL (4FN)

Para ello necesitamos la definición de dependencia multivaluada: "Dada una relación R con los atributos A, B y C, la DMV R.A →R.B se cumple en R si y sólo si el conjunto de valores de B correspondiente a un par dado (valor de A, valor de C) en R depende del valor de A y es independiente del valor de C. Como siempre, A, B y C pueden ser compuestos."

Una relación está en 4FN si y sólo si, siempre que existe una dependencia multivaluada en R, digamos A→B, todos los atributos de R dependen también funcionalmente de A.

4.3.6. QUINTA FORMA NORMAL (5FN)

En este caso necesitamos la definición de dependencia de reunión (DR): "La relación R satisface la dependencia de reunión (DR) *(X, Y, ..., Z), si y sólo si R es igual a la reunión de sus proyecciones según X, Y, ..., Z, donde X, Y, ..., Z son subconjuntos del conjunto de atributos de R."

Una relación está en 5FN, también llamada forma normal de proyección/reunión (PJ/NF) si y sólo si toda dependencia de reunión en R es una consecuencia de las claves candidatas de R.

5. DISEÑO FÍSICO

La última etapa de la metodología de diseño de bases de datos es el diseño físico, cuyo objetivo general es satisfacer los requisitos del sistema optimizando la relación costes/beneficios. Esto se concreta en los siguientes objetivos: Disminuir los tiempos de respuesta, minimizar el espacio de almacenamiento,

evitar las reorganizaciones periódicas, proporcionar la máxima seguridad y optimizar el consumo de recursos.

Para conseguir esto es necesario conocer la carga que el sistema debe soportar; esta carga consiste en una serie de consultas y actualizaciones de datos de nuestra base de datos. Además, los usuarios suelen tener ciertos requisitos sobre la rapidez que deben tener los procesos de consulta y actualización. La descripción de la carga y los requisitos de los usuarios sobre el rendimiento de las consultas son la base para la toma de decisiones de un buen diseño físico.

Los fabricantes de SGBDR abordan el problema del diseño físico desde tres perspectivas diferentes:

A) El SGBD impone una estructura interna y deja muy poca flexibilidad al diseñador. Esto suele suponer una mayor independencia físico/lógica a costa de menor eficiencia.

B) El administrador diseña la estructura interna. Esto supone más trabajo y un perjuicio para la independencia de datos, aunque puede mejorar la eficiencia.

C) El SGBD proporciona una estructura interna inicial a partir de algunos parámetros dados por el diseñador. El administrador puede ir afinándolos (tunning) para mejorar el rendimiento.

En general, la mejor opción es la C porque así la base de datos puede empezar a funcionar inmediatamente, La eficiencia va aumentando al ir realizando ajustes posteriores, la independencia físico/lógica se mantiene y porque es la estrategia que mejor se adapta a la metodología propuesta.

No existe un modelo formal general para el diseño físico, depende mucho de cada producto comercial concreto, aunque sí podemos comentar algunos de las técnicas más importantes que se pueden considerar en este diseño:

- Determinación de los índices secundarios y sus características.
- Tipo de registros físicos.
- Uso de punteros.
- Direccionamiento calculado (Hashing).
- Agrupamientos (Clustering) de tablas.
- Bloqueos (Locking) y compresión de datos.
- Definición de tamaños de memorias intermedias (Buffers).

- Asignación de conjuntos de datos a particiones y/o a dispositivos físicos.
- Redundancia de datos.

De las nombradas anteriormente podemos considerar como muy importante la creación adecuada de índices sobre las estructuras que definen la base de datos. Pero, al mismo tiempo, hay que tener en cuenta que la creación de índices no siempre va a ser beneficiosa para mejorar el rendimiento de nuestro sistema.

6. CONCLUSIÓN

En este tema se ha presentado una visión global del diseño de bases de datos relacionales. En primer lugar, se ha indicado qué se entiende por diseñar una base de datos y se han enumerado las etapas en las que se puede descomponer el proceso de diseño: etapa del diseño conceptual, etapa del diseño lógico y etapa del diseño físico.

Posteriormente, se ha tratado más en profundidad el diseño conceptual, el diseño lógico y el diseño físico de la base de datos. Para el diseño conceptual se ha adoptado el enfoque del modelo ER extendido, un modelo de datos muy utilizado y comprensible. Se han descrito las diversas construcciones que proporciona y se han dado ejemplos de aplicación a casos prácticos. En lo que respecta al diseño lógico, se ha centrado en el caso de utilización de la tecnología relacional. De este modo, se ha explicado cómo se puede transformar un modelo conceptual expresado mediante el modelo ER en una estructura de datos del modelo relacional.

7. BIBLIOGRAFÍA

- Date D.J.: **Introducción a los sistemas de bases de datos**. Editorial Addison-Wesley
- De Miguel A,y Piattini M:**Fundamentos y modelos de BBDD**. Edit. Ra-Ma
- Korth H. y Silberschatz: **Fundamentos de bases de datos**. Editorial McGraw-Hill