

# **BASES DE DATOS I**

1º Cuatrimestre  
Curso 2018/2019

María Caseiro Arias

## Índice:

1.	Sistemas de bases de datos.....	3
2.	Planificación, diseño y administración de la base de datos.....	11
3.	Modelo Entidad-Relación.....	17
4.	Modelo Relacional.....	21
5.	Álgebra Relacional.....	25
6.	Introducción a SQL.....	29
7.	Normalización.....	31

# **TEMA 1: SISTEMAS DE BASES DE DATOS**

## **1.-CONCEPTOS BÁSICOS**

### **Base de datos:**

Colección de datos relacionados.

### **Sistema de Gestión de Base de Datos:**

Software que gestiona y controla el acceso a la base de datos.

### **Aplicación de bases de datos:**

Programa que interactúa con la base de datos en algún punto de su ejecución.

### **Sistema de base de datos:**

Colección de programas de aplicación, SQL y la propia base de datos.

## **2.-SISTEMAS TRADICIONALES BASADOS EN ARCHIVOS**

### **2.1.-Definición:**

Colección de programas de aplicación que realiza diversos servicios para los usuarios finales. Cada programa define y gestiona sus propios datos.

### **2.2.-Limitaciones:**

- Separación y aislamiento de los datos: existe una mayor dificultad para acceder a los datos debido a su distribución en archivos.
- Duplicación de datos: debido al enfoque descentralizado, puede darse la repetición del mismo dato en diferentes archivos (provoca desperdicio de recursos y perdida de integridad).
- Dependencias entre datos.
- Formatos de archivos incompatibles.
- Consultas fijas/proliferación de programas de aplicación.

## **3.-SISTEMAS DE BASES DE DATOS**

### **3.1.-Definición:**

Colección compartida de datos lógicamente relacionados, junto con una descripción de estos datos, que están diseñados para satisfacer las necesidades de información de una organización.

### **3.2.-Comparación con sistemas basados en archivos:**

Las limitaciones de los sistemas basados en archivos se deben principalmente a:

- La definición de los datos está incluida en los programas de aplicación, en lugar de almacenarse de forma separa e independiente.
- No existe ningún control sobre el acceso y manipulación de los datos, más allá del que imponen los propios programas de aplicación.

Estos problemas fueron solucionados mediante los sistemas de bases de datos que garantizan que:

- Todos los elementos de datos están integrados, manteniéndose al mínimo las posibles duplicaciones.
- Existe una independencia entre programas y datos.
- Se almacenan los datos operacionales y una descripción de los mismos.
- Los usuarios solo visualizan la información externa, no son conscientes del modo en el que está definido el objeto internamente.
- Los programas de aplicación no se ven afectados por la modificación de la base de datos.

## 4.-SISTEMA DE GESTIÓN DE BASE DE DATOS

### 4.1.-Definición:

Sistema de software que permite a los usuarios definir, crear mantener y controlar el acceso a la base de datos.

### 4.2.-Funcionalidad:

EL SGBD permite:

- Definir la base de datos
- Insertar, actualizar, borrar y extraer datos de la base de datos
- Proporcionar un mecanismo general de consulta
- Proporcionar un acceso controlado a la base de datos que garantice seguridad, integridad, control de concurrencia, control de recuperación y un catálogo accesible por el usuario.

## 5.-PROGRAMA DE APLICACIÓN

### 5.1.-Definición:

Programa informático que interactúa con la base de datos emitiendo las apropiadas solicitudes (normalmente una instrucción SQL) dirigidas al SGBD. Los programas se utilizan principalmente para introducir datos, el mantenimiento de los mismos y la generación de informes a partir de una base de datos.

### 5.2.-Vistas:

El SGBD ofrece un mecanismo de vistas que facilita la visualización de los datos que determinado usuario necesita, mostrando el subconjunto de datos de la base de datos que este solicita.

El mecanismo de vistas, además de reducir la complejidad de visualización de los datos, cuenta con otra serie de ventajas:

- Proporciona un nivel de seguridad
- Proporciona una apariencia personalizada de la base de datos
- Presenta una imagen coherente y estática de la estructura de la BD.

## 6.-COMPONENTES DE UN ENTORNO DE SGBD

### 6.1.-Hardware:

Plataforma (computadoras y cantidad de memoria) requerida por el SGBD y las aplicaciones para ejecutarse.

### 6.2.-Software:

Componente que comprende al propio SGBD y a las aplicaciones creadas, junto con el sistema operativo de una computadora, que proporciona la red necesaria para el correcto funcionamiento de la base de datos.

### 6.3.-Datos:

Estructura principal de la base de datos (información) que actúa como puente entre los componentes ligados a la máquina y los que están ligados al operador.

### 6.4.-Procedimientos:

Instrucciones y reglas que gobiernan el diseño y utilización de la base de datos.

### 6.5.-Personas:

Usuarios finales que interactúan con la BD.

## 7.-PAPELES EN UNA BASE DE DATOS (PERSONAS)

### **7.1.-Administrador:**

El administrador de datos (DA) se asocia con la gestión y control de un SGBD y de los datos almacenados en él.

El administrador de la base de datos (DBA) es responsable de la implementación física de la base de datos incluyendo la implementación y diseño físico de la base de datos, el control de la seguridad y de la integridad, el mantenimiento de la fiabilidad del sistema y la garantía de que las aplicaciones exhiban un rendimiento satisfactorio para los usuarios.

### **7.2.-Diseñador:**

El diseñador lógico de la base de datos debe identificar los datos, las relaciones entre los datos y las restricciones que hay que aplicar a los datos que se almacenen en la base de datos.

El diseñador físico de la base de datos debe materializar físicamente el diseño lógico.

### **7.3.Desarrollador de aplicaciones:**

Personas encargadas de crear programas de aplicación que proporcionen la funcionalidad requerida por los usuarios finales.

### **7.4.Usuarios finales:**

Los usuarios finales, también denominados clientes, se pueden clasificar:

#### **Usuarios inexpertos:**

No son conscientes de la existencia de un SGBD y acceden a la BD mediante programas determinados que resuelven la función que solicitan.

#### **Usuarios avanzados:**

Están familiarizados con la estructura de una BD y con las funcionalidades del SGBD, por lo que utilizan lenguajes de alto nivel como el SQL para llevar a cabo las operaciones requeridas.

## 8.-VENTAJAS Y DESVENTAJAS DE LOS SGBD

<b>VENTAJAS</b>	
Control de la redundancia de datos	Economía de escala
Coherencia de datos	Equilibrio entre requisitos conflictivos
Más información a partir de la misma cantidad de datos	Mejor accesibilidad a los datos y mayor capacidad de respuesta
Compartición de datos	Productividad mejorada
Mayor integridad de los datos	Mantenimiento más sencillo gracias a la independencia de los datos
Mayor seguridad	Mayor nivel de concurrencia
Imposición de estándares	Servicios mejorados de copia de seguridad y recuperación

<b>DESVENTAJAS</b>	
Complejidad	Tamaño
Coste del SGBD	Costes de hardware adicional
Costes de conversión	Prestaciones
Mayor impacto de fallos	

## 9.-ARQUITECTURA DE 3 NIVELES DE ANSI-SPARC

### 9.1.-Explicación:

La arquitectura ANSI-SPARC es un estándar de diseño abstracto para un sistema de gestión de bases de datos (DBMS), propuesto por primera vez en 1975, que adoptó un enfoque basado en 3 niveles cuyo objetivo es separar la vista que tiene cada usuario de la BD de la forma en la que se representa físicamente.

### 9.2.-Niveles:

#### **Externo:**

Vista que tienen los usuarios de la base de datos. Este nivel describe la parte de la base de datos relevante para el usuario.

#### **Conceptual:**

La vista comunitaria de la base de datos. Este nivel describe que datos están almacenados en la base de datos y las relaciones existentes entre los mismos. Este nivel contiene la estructura lógica de la BD y proporciona soporte a cada una de las vistas externas sin contener ningún detalle que sea dependiente del almacenamiento.

#### **Interno:**

Representación física de la BD en la computadora. Este nivel describe como están almacenados los datos en la BD.

Este nivel pretende conseguir unas prestaciones óptimas en tiempo de ejecución y una utilización óptima del espacio de almacenamiento. Utiliza métodos de acceso al sistema operativo.

Por debajo de este nivel se encuentra el NIVEL FÍSICO que puede ser gestionado por el sistema operativo bajo la dirección del SGBD.

### 9.3.-Organización:

La descripción global de la base de datos se denomina esquema de la base de datos. Existen 3 tipos de esquemas de bases de datos (externo, conceptual e interno) que se corresponden con los tres niveles de abstracción. Solo existe un esquema conceptual y esquema interno, mientras que pueden existir múltiples esquemas externos. Es el SGBD realiza la correspondencia entre los 3 tipos de esquemas.

Mediante una asignación externo/conceptual se relaciona el esquema externo y conceptual. Los datos de la base de datos en un instante concreto de tiempo se denominan instancia.

### 9.4.-Independencia de datos:

Uno de los objetivos principales de la arquitectura de 3 niveles es el de proporcionar independencia de los datos, lo que quiere decir que los niveles más altos no se vean afectados por las modificaciones en los niveles más bajos.

#### **Independencia lógica de los datos:**

Inmunidad de los esquemas externos a las modificaciones que se efectúan en el esquema conceptual.

#### **Independencia física de los datos:**

Inmunidad del esquema conceptual a los cambios que se efectúen en el esquema externo.

La correspondencia en 2 etapas que se efectúa en la arquitectura ANSI-SPARC puede ser poco eficiente, pero proporciona una mayor independencia de los datos. Sin embargo, si se quiere conseguir una asignación más eficiente, el modelo ANSI-SPARC permite establecer una correspondencia directa entre los esquemas externos y el esquema interno, evitando el esquema conceptual.

## 10.-LENGUAJES DE BASES DE DATOS

### 10.1.-DDL (lenguaje de definición de datos):

Lenguaje que permite al DBA o al usuario describir y nombrar entidades, atributos y relaciones requeridas por la aplicación, junto con cualquier restricción asociada de integridad y seguridad.

### 10.2.-DML (lenguaje de manipulación de datos):

Lenguaje que proporciona un conjunto de operadores para permitir las manipulaciones básicas de los datos contenidos en la base de datos.

### **Procedimentales:**

Lenguaje que permite al usuario decirle al sistema que datos necesita y cuál es la forma exacta de extraerlos. Manipulan los registros de forma individual.

### **No procedimentales:**

Lenguaje que permite al usuario indicar que datos necesita, en lugar de como hay que extraerlos. Manipulan conjuntos de registros.

LENGUAJE DE CONSULTA: lenguaje de propósito especial y alto nivel que se utiliza para satisfacer solicitudes diversas de extracción de datos contenidos en la BD.

### 10.3.-4GL (lenguaje de cuarta generación):

Lenguaje de programación optimizado que comprende lenguajes de presentación, lenguajes especializados, generadores de aplicaciones y lenguajes de muy alto nivel.

## 11. MODELOS DE DATOS Y MODELADO CONCEPTUAL

### 11.1.-Definiciones:

Modelo de datos: colección integrada de conceptos para describir y manipular datos, las relaciones existentes entre los mismos y las restricciones aplicables a los datos, todo ello dentro de una organización.

Componentes principales:

### **Parte estructural:**

Conjunto de reglas

### **Parte manipulativa:**

Tipos de operaciones que pueden realizarse sobre los datos.

### **Conjunto de restricciones de integridad:**

Garantiza precisión en los datos.

### **Modelado conceptual:**

Proceso de construir un modelo sobre el uso de la información dentro de una empresa, independientes de los detalles de implementación.

## 11.2.-Categorías:

### **Basado en objetos:**

Utilizan conceptos como entidades, atributos y relaciones. Este modelo amplía la definición de entidad para incluir el comportamiento y el estado de sus atributos.

### **Basado en registros:**

La base de este modelo es un conjunto de registros de formato fijo, posiblemente de distintos tipos. Existen 3 modelos lógicos basados en registros:

- a) Modelo de datos relacional: basado en el concepto de las relaciones matemáticas. Los datos y las relaciones se representan mediante tablas.
- b) Modelo de datos en red: los datos se representan como colecciones de registros y las relaciones mediante conjuntos.

- c) Modelo de datos jerárquico: los datos se representan como colecciones de registros y las relaciones mediante conjuntos, pero este modelo solo permite que cada nodo tenga un parente.

**Físico:**

Describen como se almacenan los datos en la computadora.

**12.-FUNCIONES DEL SGBD**

- Almacenamiento, extracción y actualización de datos.
- Catálogo accesible por el usuario (almacena las descripciones de los elementos de datos).
- Soporte de transacciones (mecanismo que garantice que se lleven a cabo todas las actualizaciones correspondientes a una determinada transacción, o que no se lleve a cabo ninguna).
- Servicios de control de concurrencia (mecanismo que garantice que la base de datos se actualice correctamente cuando hay varios usuarios utilizándola).
- Servicios de recuperación (método para recuperar la base de datos en caso de que resulte dañada).
- Servicios de autorización (control del acceso a usuarios autorizados).
- Software para la tramitación de datos (capacidad de integración con software de comunicaciones).
- Servicios de integridad (proporcionar un medio de garantizar que tanto los datos de la base de datos como los cambios efectuados en los mismo se adecuen con ciertas reglas).
- Servicios para mejorar la independencia de los datos (incluir funcionalidades para permitir que los programas sean independientes de la estructura real de la base de datos).
- Servicios de utilidad.

**13.-COMPONENTES DEL SGBD**

**Procesador de consultas:**

Transforma las consultas en una serie de instrucciones de bajo nivel dirigidas al gestor de base de datos.

**Gestor de base de datos:**

Se comunica con las consultas enviadas por el usuario y con los programas de aplicación.

**Gestor de archivos:**

Manipula los archivos de almacenamiento subyacentes y gestiona la asignación del espacio de almacenamiento en disco.

**Preprocesador DML:**

Convierte instrucciones DML integradas en un programa de aplicación en llamada a estándar a funciones en el leguaje host.

**Compilador DDL:**

Convierte las instrucciones DDL en una serie de tablas que contienen metadatos.

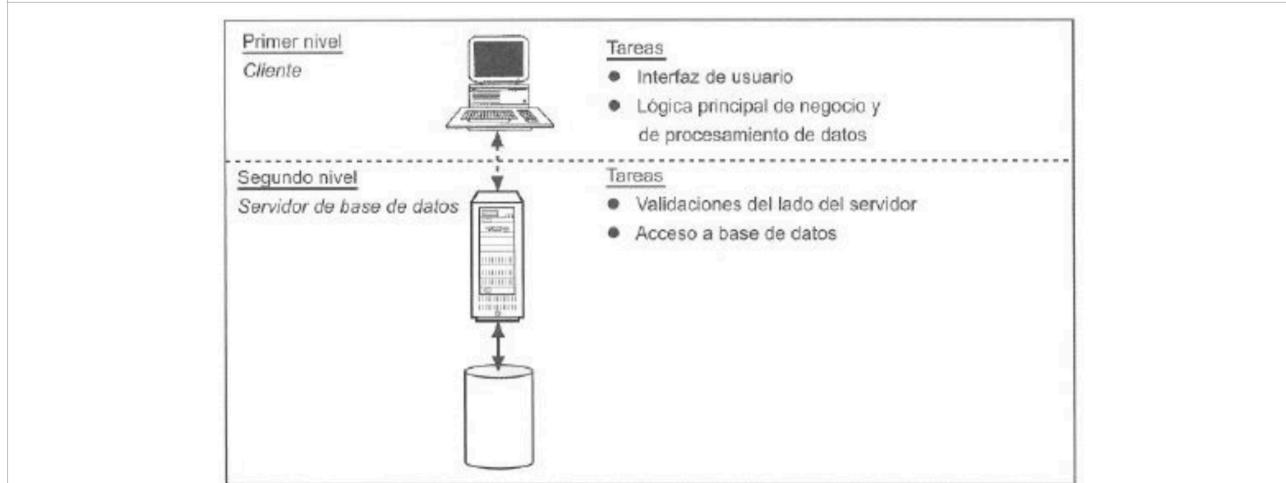
**Gestor del catálogo:**

Gestiona el acceso al catálogo del sistema y se encarga de mantenerlo.

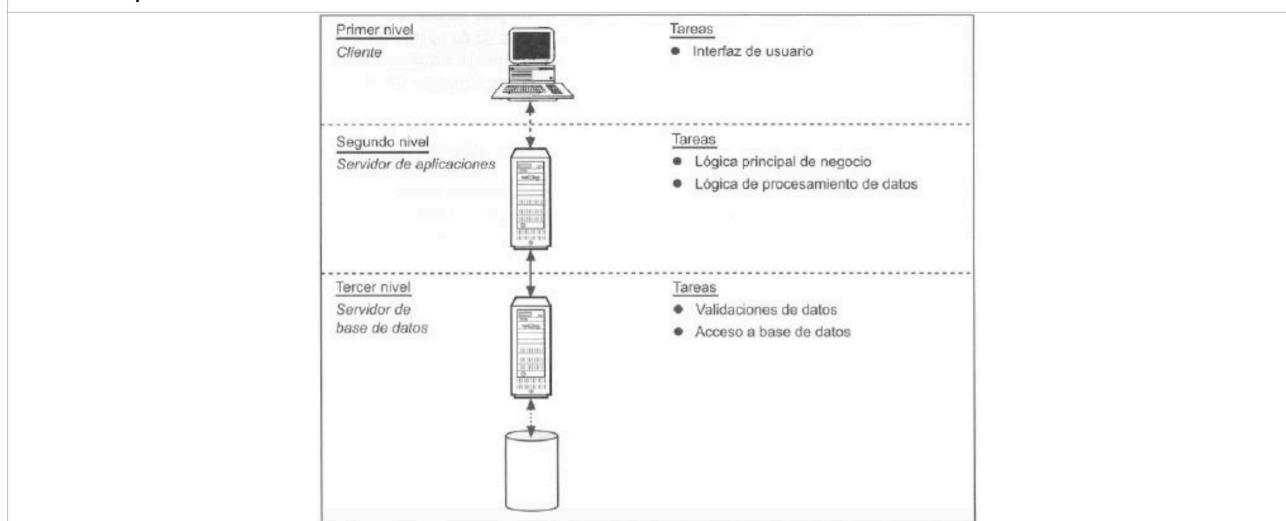
## 14.-ARQUITECTURAS DE SGBD MULTIUUSUARIO

<p><b>14.1.-Teleprocesamiento:</b></p> <pre> graph LR     S[Servidor] --- C1[Estación de trabajo 1]     S --- C2[Estación de trabajo 2]     S --- C3[Estación de trabajo 3]     S --- C4[Estación de trabajo 4]     S --- C5[Estación de trabajo 5]   </pre>	<p><b>14.2-Arquitectura de servicio de archivos:</b></p> <pre> graph TD     ET1[Estación de trabajo 1] --- LAN((LAN))     ET2[Estación de trabajo 2] --- LAN     ET3[Estación de trabajo 3] --- LAN     LAN --- SA[Servidor de archivos]     SA --- BD[Base de datos]     SA -- "Archivos devueltos" --&gt; ET1     SA -- "Solicitudes de datos" --&gt; ET2     SA -- "Solicitudes de datos" --&gt; ET3   </pre>
--	--

## **14.3.-Arquitectura cliente-servidor tradicional en dos niveles:**

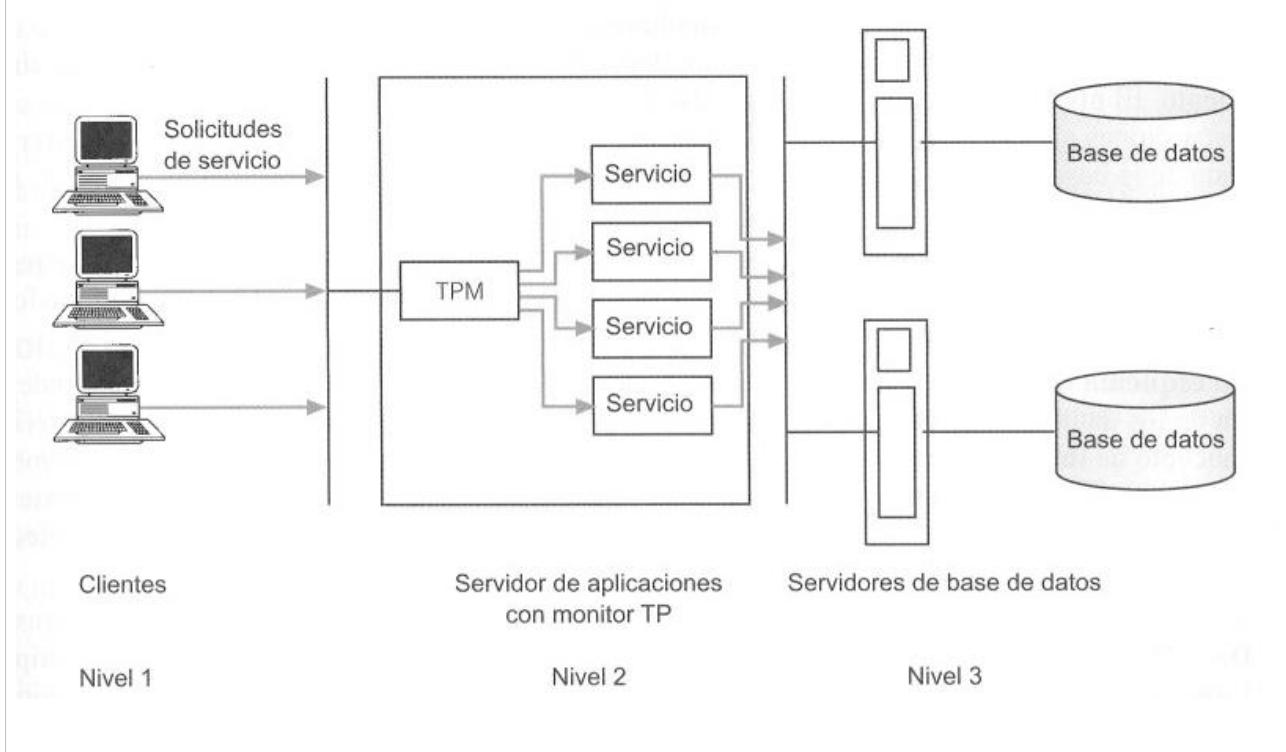


## **14.4.-Arquitectura cliente-servidor en tres niveles:**



#### 14.5.-Monitores de procesamiento de transacciones:

Un programa que controla la transferencia de datos entre clientes y servidores para proporcionar un entorno coherente, particularmente para el procesamiento de transacciones en línea (OLTP, online transaction processing).



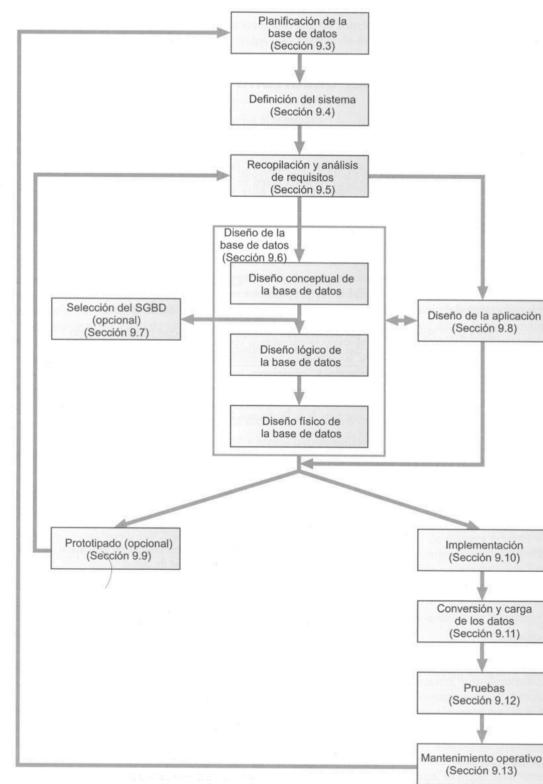
## **TEMA 2: PLANIFICACIÓN, DISEÑO Y ADMINISTRACIÓN DE LA BASE DE DATOS**

**Sistemas de información:** conjunto de recursos que permiten la recopilación, gestión, control y diseminación de la información en una determinada organización.

Los sistemas de información incluyen las bases de datos, el software de la base de datos, el software de la aplicación, el hardware informático y el personal que utiliza y desarrolla al sistema.

En el desarrollo de los sistemas de información podemos identificar una serie de etapas establecidas, estas etapas no son estrictamente secuenciales, sino que existe cierta repetición (bucles de realimentación):

- Planificación: planificación del modo en que pueden llevarse a cabo las distintas etapas del ciclo de la vida de la forma más eficiente y efectiva.
- Recopilación: especificación del ámbito y los límites del sistema de la base de datos, incluyendo las principales vistas del usuario, los tipos de usuario y las áreas de aplicación.
- Análisis y requisitos.
- Diseño: diseño conceptual, lógico y físico de la base de datos.
- Selección del SGBD (opcional): escoger un SGBD adecuado a la base de datos.
- Prototipado (opcional): construcción de un modelo funcional del sistema de la base de datos que permite a los diseñadores o usuarios visualizar y evaluar el aspecto y la función del sistema final.
- Implementación: creación de las definiciones físicas de la base de datos y de los programas de aplicación.
- Conversion: carga de los datos del antiguo sistema en el nuevo.
- Pruebas: prueba de la base de datos en busca de errores y validación de la misma con respecto a los requisitos especificados por los usuarios.
- Mantenimiento operativo: el sistema de base de datos está completamente implementado, después de lo cual se monitoriza y mantiene de manera continua. Cuando sea necesario, se incorporarán nuevos requisitos al sistema de bases de datos aplicando de nuevo las etapas precedentes del ciclo de vida.



### **1.-PLANIFICACIÓN DE LA BASE DE DATOS**

Actividades de gestión que permiten llevar a cabo las distintas etapas del ciclo de vida del desarrollo de sistemas de base de datos de la forma más eficiente y efectiva posible.

Existen 3 cuestiones principales para llevar a cabo la planificación:

- Identificación de los planes y objetivos de la empresa.
- Evaluación de los sistemas de información actuales para determinar las fortalezas y debilidades existentes.
- Aprovechamiento de las oportunidades en tecnología de la información que puedan proporcionar una ventaja competitiva.

## Metodología:

- a) Enunciar claramente una misión de la base de datos.
- b) Enunciar los objetivos de esa misión (identificar una tarea concreta a la que el sistema de la base de datos debe proporcionar soporte).
- c) Información adicional que especifique en términos generales la tarea que hay que realizar, los recursos con los que hay que llevarla a cabo y el dinero que debe costar.

## 2.-DEFINICIÓN DEL SISTEMA

Describe el ámbito y los límites de la aplicación de base de datos y las principales vistas del usuario.

En esta etapa es importante la identificación de las vistas de cada uno de los usuarios. Se define qué es lo que se requiere de un sistema de bases de datos desde la perspectiva de un determinado rol o de un área de aplicación empresarial.

## 3.-VISTA DEL USUARIO

Define qué es lo que se requiere de un sistema de base de datos desde la perspectiva de un determinado rol de la organización (como, por ejemplo, Gerente o Supervisor) o de un área de aplicación empresarial (como, por ejemplo, marketing, personal o control de almacén).

Un sistema de base de datos puede tener una o más vistas de usuario. La identificación de las vistas de usuario es un aspecto de gran importancia a la hora de desarrollar un sistema de base de datos, porque ayuda a garantizar que no se deje de lado a ninguno de los usuarios principales de la base de datos a la hora de desarrollar los requisitos para el nuevo sistema.

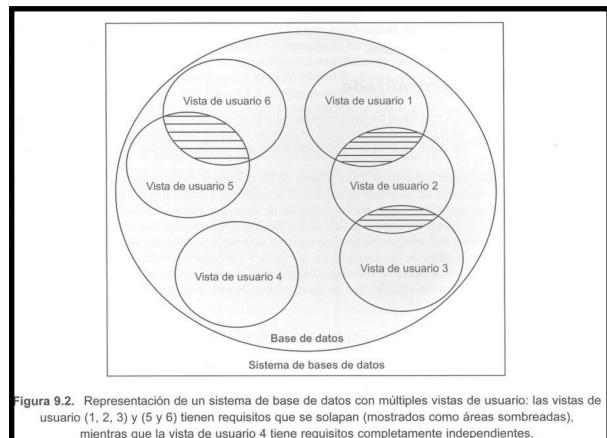


Figura 9.2. Representación de un sistema de base de datos con múltiples vistas de usuario: las vistas de usuario (1, 2, 3) y (5 y 6) tienen requisitos que se solapan (mostrados como áreas sombreadas), mientras que la vista de usuario 4 tiene requisitos completamente independientes.

## 4.-RECOPILACIÓN Y ANÁLISIS DE REQUISITOS

Proceso de recopilar y analizar información acerca de la parte de la organización a la que el sistema de bases de datos tenga que dar soporte, y utilizar esta información para identificar los requisitos relativos al nuevo sistema.

Se recopila información para cada una de las vistas de usuario principal, incluyendo una descripción de los datos utilizados y generados, los detalles acerca de cómo hay que utilizar o generar los datos y cualesquier otros requisitos que sean aplicables al nuevo sistema de base de datos.

Toda esta información es utilizada posteriormente para generar unos documentos denominados especificaciones de requisitos.

En esta etapa es necesario buscar un equilibrio de especificaciones que nos permitan crear un sistema de bases de datos completo, pero sin excesivas restricciones que compliquen su uso e implementación.

Otra actividad asociada a esta etapa es la gestión de los requisitos cuando existen múltiples vistas de usuario. Existen diferentes técnicas:

- Enfoque centralizado: todas las vistas de usuario siguen el mismo modelo.
- Enfoque de integración de vistas: los requisitos de cada vista se mantienen en listas separadas.
- Combinación de ambas técnicas.

#### 4.1.-Enfoque centralizado/de integración de las vistas

##### **Enfoque centralizado:**

Los requisitos de cada vista de usuario se combinan en un único conjunto de requisitos para el nuevo sistema de base de datos.

##### **Enfoque de integración:**

Los requisitos de cada vista de usuario se mantienen en listas separadas. Durante la etapa de diseño de la base de datos se crean y combinan los modelos de datos que representan cada una de las vistas de usuario.

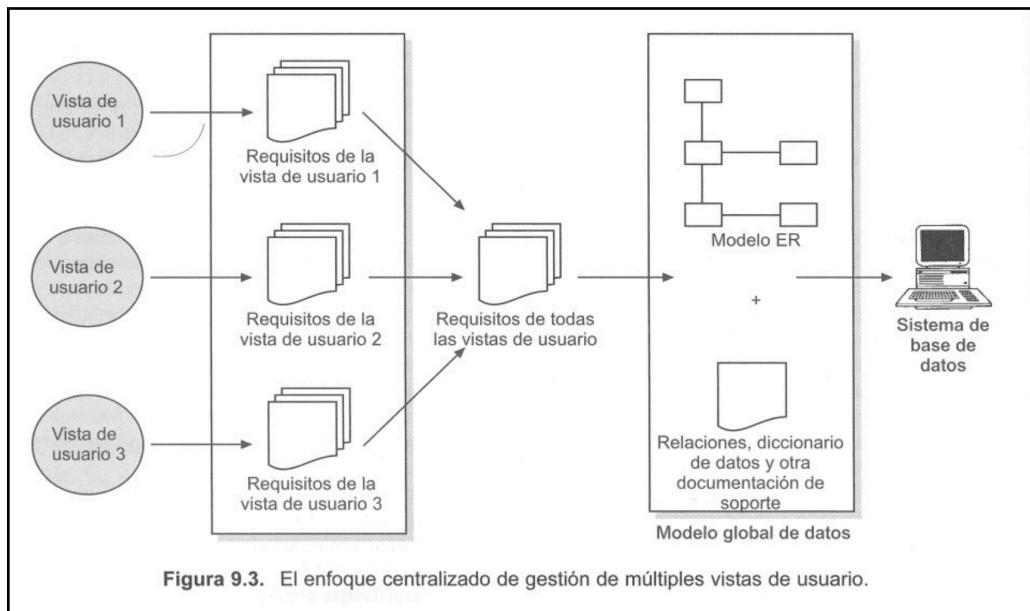


Figura 9.3. El enfoque centralizado de gestión de múltiples vistas de usuario.

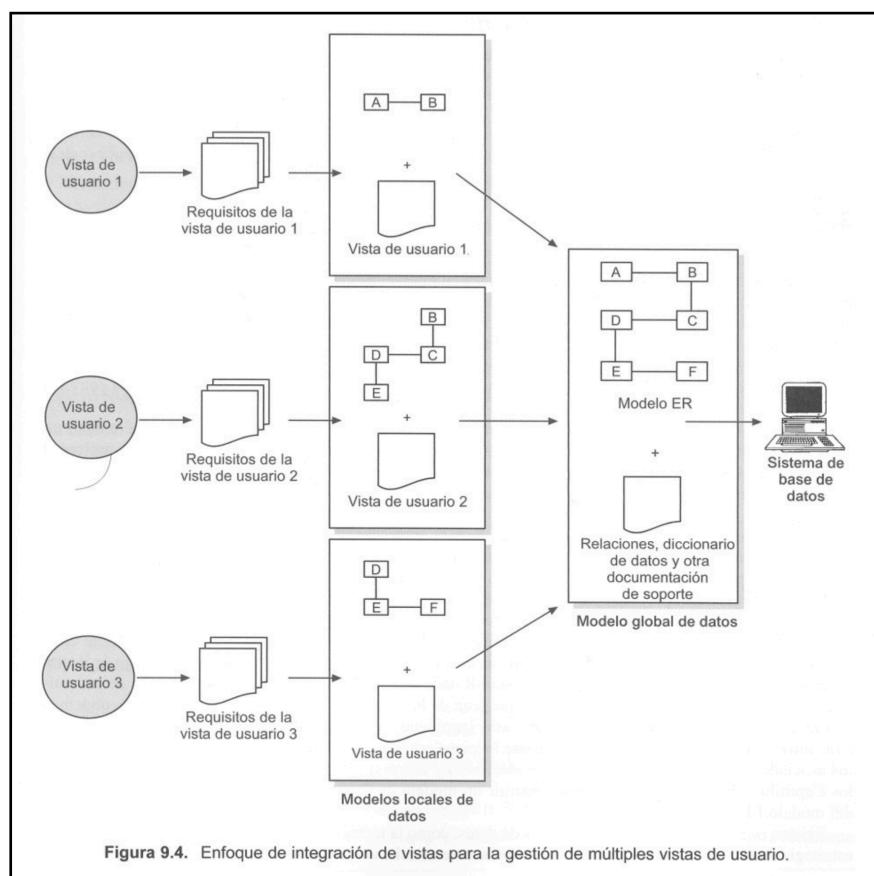


Figura 9.4. Enfoque de integración de vistas para la gestión de múltiples vistas de usuario.

Por ejemplo, los requisitos de dos o más vistas de usuario pueden primero combinarse utilizando el enfoque centralizado, lo que resulta útil para construir un modelo lógico local de los datos. Este modelo puede después combinarse con otros modelos lógicos locales de los datos utilizando el enfoque de integración de vistas, con el fin de producir un modelo lógico global de los datos.

## 5.-DISEÑO DE LA BASE DE DATOS

Proceso de creación de un diseño que dé soporte a la misión y a los objetivos de la misión de la empresa para el sistema de base de datos requerido.

### 5.1.-Técnicas de diseño:

Existen 2 técnicas principales de diseño:

‘De abajo a arriba’: diseño de bases de datos simples.

‘De arriba a abajo’: diseño de bases de datos complejas.

(\*) También hay otro tipo de técnicas similares como la ‘de dentro a fuera’ o la mixta (combina las 2 principales).

### 5.2.-Modelado:

Los propósitos principales del modelado de datos son ayudar a comprender el significado (semántica) de datos y facilitar la comunicación de los requisitos de información. Para llevar a cabo este proceso de una forma estructurada y comprensible deben seguirse una serie de criterios que permiten generar un modelo de datos óptimo: validez estructural, simplicidad, capacidad de expresión, no redundancia, capacidad de comparación, ampliabilidad, integridad y representación diagramática.

### CRITERIOS PARA GENERAR UN MODELO DE DATOS ÓPTIMO:

<i>Validez estructural</i>	Coherencia con la forma en que la empresa define y organiza la información.
<i>Simplicidad</i>	Facilidad de comprensión por parte de los profesionales de los sistemas de información y por parte de los usuarios no técnicos.
<i>Capacidad de expresión</i>	Capacidad de distinguir entre diferentes datos, relaciones entre los datos y restricciones.
<i>No redundancia</i>	Exclusión de la información no pertinente; en particular, la representación de cada elemento de información una única vez.
<i>Capacidad de compartición</i>	No específico de ninguna aplicación o tecnologías concretas y, como consecuencia, utilizable por muchas aplicaciones o tecnologías.
<i>Ampliabilidad</i>	Capacidad de evolucionar para satisfacer nuevos requisitos con un efecto mínimo sobre los usuarios existentes.
<i>Integridad</i>	Coherencia con la forma en que la empresa utiliza y gestiona la información
<i>Representación diagramática</i>	Capacidad de representar un modelo utilizando una notación diagramática fácilmente comprensible.

### 5.3.-Fases de diseño:

Se distinguen 3 fases principales:

**Fase 1:** diseño conceptual. Proceso de construcción de forma independiente a las consideraciones físicas.

**Fase 2:** diseño lógico. Proceso de construcción basándose en un modelo de datos específico, pero de forma independiente de un SGBD concreto y cualquier consideración física.

**Fase 3:** diseño físico. Proceso de generar una descripción de la implementación de la base de datos de almacenamiento secundario; describe las relaciones de la base, la organización de los archivos y los índices utilizados para conseguir acceso eficiente a los datos, así como cualesquiera medidas de seguridad y restricciones de integridad asociadas.

## 6.-SELECCIÓN DEL SGBD

Consiste en seleccionar el SGBD apropiado para soportar el sistema de la base de datos. Los pasos principales que se deben seguir en esta etapa son:

- Definición de los términos de referencia del estudio.
- Selección de 2 o 3 productos candidatos.
- Evaluación de los productos.
- Recomendación de un producto y generación del informe.

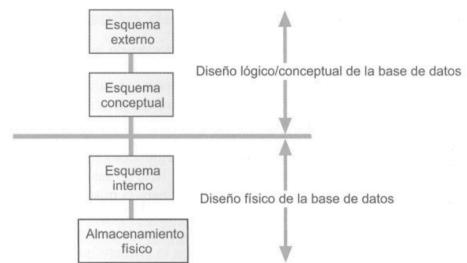


Figura 9.5. El modelado de datos en la arquitectura ANSI-SPARC.

## 7.-DISEÑO DE LA APLICACIÓN

El diseño de interfaz de usuario y de los programas de aplicación permiten utilizar y procesar la base de datos.

### 7.1.-Diseño de transacciones:

Esta actividad debe llevarse a cabo en una fase temprana del proceso de diseño, para garantizar que la base de datos que se implemente sea capaz de soportar todas las transacciones requeridas.

Una transacción es una acción o serie de acciones llevadas a cabo por un único usuario o programa de aplicación y que acceden al contenido de la base de datos o los modifican. Las transacciones pueden estar formadas por varias operaciones, pero a vista del usuario es una única tarea.

#### Transacción:

Una acción o serie de acciones llevadas a cabo por un único usuario o programa de aplicación y que acceden al contenido de la base de datos o los modifican.

Hay 3 tipos principales de transacciones:

- Transacciones de extracción: extraer datos y mostrarlo por pantalla o hacer un informe con ellos.
- Transacciones de actualización: insertar, borrar o modificar registros en la base de datos.
- Transacciones mixtas: implican tanto la extracción como la actualización.

### 7.2.-Diseño de interfaz de usuario:

Consiste en realizar un informe especificando detalladamente las características que presentará el diseño del sistema de bases de datos. Siguiendo una serie de directrices y pautas (*página 275 del Conolly y Begg*).

## 8.-PROTOTIPADO

Construcción de un modelo operativo de un sistema de bases de datos.

En este proceso el principal objetivo es construir un prototipo del sistema de bases de datos y permitir a algunos usuarios su uso con el objetivo de encontrar características inadecuadas o con mal funcionamiento.

Los prototipos deben ser relativamente baratos y rápidos de construir.

Existen dos estrategias de construcción de prototipos:

- Prototipado de requisitos: se descarta el prototipo una vez acabada esta fase.
- Prototipado evolutivo: se cambia el prototipo hasta obtener el proyecto final.

## 9.-IMPLEMENTACIÓN

Realización física del diseño de la base de datos y del diseño de aplicaciones.

La implementación de la base de datos se lleva a cabo utilizando el lenguaje de definición de datos (DDL) del SGBD seleccionado una interfaz gráfica de usuario. En esta etapa se implementan también las vistas de usuario especificadas.

## 10.-CONVERSIÓN Y CARGA DE LOS DATOS

Transferencia de los datos existentes a la nueva base de datos y conversión de las aplicaciones existentes para que se ejecuten con la nueva base de datos.

## 11.-PRUEBAS

Proceso de operar el sistema de base de datos con la intención de localizar posibles errores.

## 12.-MANTENIMIENTO OPERATIVO

Proceso de monitorizar y mantener el sistema de base de datos después de la instalación.

### **Herramientas CASE:**

El término CASE se aplica a cualquier herramienta que dé soporte a la ingeniería software y que permita que las actividades de desarrollo del sistema de base de datos se lleven a cabo de la forma más eficiente y efectiva posible. Las herramientas CASE pueden clasificarse en tres categorías: CASE de alto nivel, CASE de bajo nivel y CASE integrado.

### **Administración de datos y administración de base de datos:**

-La administración de datos es la gestión de los recursos de datos, incluyendo las tareas de planificación de base datos, desarrollo y mantenimiento de estándares, políticas y procedimientos y diseño conceptual y lógico e la base de datos.

-La administración de la base de datos es la gestión de la realización física de un sistema de base de datos, incluyendo las tareas de diseño físico de la base de datos e implementación de la misma, configuración de la seguridad y los controles de integridad, monitorización de las prestaciones del sistema y reorganización de la base de datos según sea necesario.

## **TEMA 3: MODELO ENTIDAD-RELACIÓN**

El modelado Entidad-Relación es una técnica de diseño de base de datos de tipo arriba abajo que comienza identificando los datos más importantes, denominados entidades, y las relaciones entre los datos que deben representarse en el modelo. Despues se añaden más detalles, como la información que se quiere almacenar acerca de las entidades y las relaciones, lo que se denomina atributos y sobre cualesquiera restricciones que haya que aplicar a las entidades, relaciones y atributos.

### **1.-TIPOS DE ENTIDAD**

#### **Tipo de entidad:**

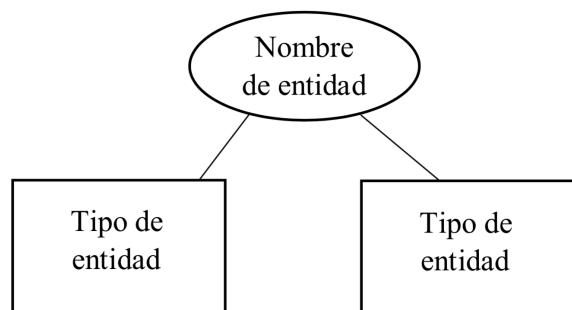
Grupo de objetos con las mismas propiedades, que la empresa identifica como poseedores de una existencia independiente.

#### **Instancia de una entidad:**

Objeto identifiable de forma unívoca dentro de un tipo de entidad.

#### **Representación diagramática de los tipos de entidad:**

Las bases de datos contienen muchos tipos diferentes de entidades, cada tipo de entidad se identifica mediante un nombre y una lista de propiedades.



### **2.-TIPOS DE RELACIÓN**

#### **Tipo de relación:**

Conjunto de asociaciones significativas entre tipos de entidad. Cada tipo de relación recibe un nombre que describe su función.

#### **Instancia de relación:**

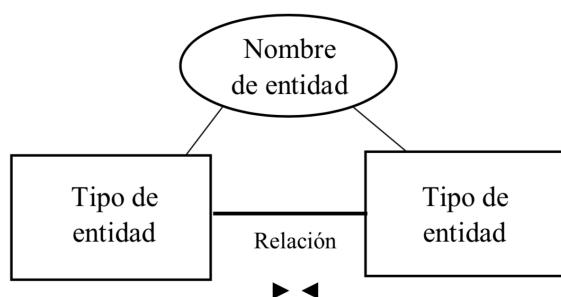
Asociación identifiable de forma unívoca que incluye una instancia de cada uno de los tipos de entidad participantes. Una instancia de relación indica las instancias de entidad concretas que están relacionadas.

#### **Red semántica:**

Modelo de nivel de objetos que utiliza el símbolo • para representar entidades y el símbolo para representar relaciones.

#### **Representación diagramática de los tipos de relación:**

Cada tipo de relación se muestra como una línea que conecta los tipos de entidad asociados, la línea está etiquetada con el nombre de la relación. Una relación solo se etiqueta en una dirección, lo que significa que solo cobra sentido en un sentido de la dirección (se indica con una flecha).

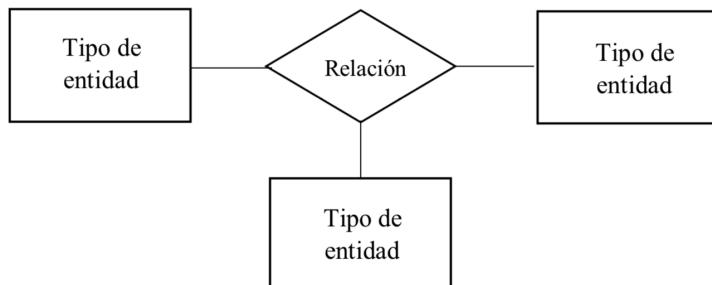


### **Grado de un tipo de relación:**

Número de tipos de entidad que participan en una relación. Las entidades implicadas en un tipo de relación se denominan participantes.

### **Representación diagramática de relaciones complejas:**

Las relaciones de grado superior a 2 (binarias) se representan mediante un rombo y se omite la dirección asociada.



### **Relación recursiva:**

Tipo de relación en el que el mismo tipo de entidad participa más de una vez en diferentes papeles.

## **3.-ATRIBUTOS**

Propiedad de un tipo de entidad o relación. Los atributos contienen valores que describen cada instancia de la entidad y representan la parte principal de los datos almacenados.

### **Dominio de atributo:**

Conjunto de valores permitidos para 1 o más atributos. Diferentes atributos pueden compartir dominio.

### **Tipos de atributos:**

#### a) Simples o compuestos:

Atributos simples: atributo compuesto de un único componente con existencia independiente, no pueden subdividirse en componentes más pequeños.

Atributos compuestos: atributo formado por múltiples componentes, cada uno de ellos con una existencia independiente.

#### b) Univaluados o multivaluados:

Atributos univaluados (monovalorados) : atributo que contiene un único valor para cada instancia de un tipo de entidad.

Atributos multivaluados: atributo que contiene múltiples valores para cada instancia de un tipo de entidad.

#### c) Atributos derivados:

Atributo que representa un valor que puede derivarse del valor de un atributo o conjunto de atributos relacionados, no necesariamente del mismo tipo de entidad.

### **Clave candidata:**

Conjunto mínimo de atributos que identifican de forma única cada instancia de un tipo de entidad. Es candidata a clave principal.

### **Clave principal (o primaria):**

Clave candidata que se selecciona para identificar de forma única cada instancia de un tipo de entidad. Un tipo de entidad puede tener más de una clave principal, en este caso, cuando se selecciona una como principal las otras pasan a denominarse claves alternativas.

### **Clave compuesta:**

Clave candidata que está formada por dos o más atributos.

Cabe destacar que las relaciones tienen claves candidatas pero parciales.

#### 4.-TIPOS DE ENTIDAD FUERTES Y DÉBILES

##### **Tipo de entidad fuerte (padres, propietarias o dominantes):**

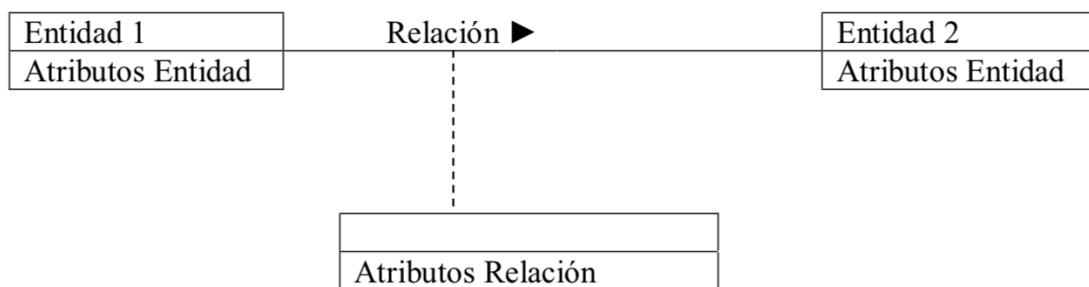
Tipo de entidad cuya existencia no depende de ningún otro tipo de entidad. Cada instancia de la identidad puede identificarse de manera única utilizando los atributos de la clave principal de dicho tipo de entidad.

##### **Tipo de entidad débil (hijas, dependientes o subordinadas):**

Tipo de entidad cuya existencia depende de algún otro tipo de entidad. Cada instancia de la identidad no puede identificarse de manera única utilizando los atributos asociados a ese tipo de entidad (no tiene clave principal).

#### 5.-ATRIBUTOS DE LAS RELACIONES

Los atributos de las relaciones se indican asociando el rectángulo de los atributos mediante una línea punteada con el nombre de la relación.



#### 6.-RESTRICCIONES ESTRUCTURALES

##### **Multiplicidad:**

El número o rango de posibles instancias de un tipo de entidad que pueden relacionarse con una única instancia de otro tipo de entidad asociado a través de una relación concreta. Restringe la forma en la que se relacionan las entidades.

##### **Tipos de relaciones binarias:**

- Relaciones uno a uno (1:1): cada entidad A binaria se asocia a lo sumo con una entidad B.
- Relaciones uno a muchos (1:\*) : cada entidad A se asocia con cualquier número (cero o más) de entidades B, no a la inversa.
- Relaciones muchos a 1 (\*:1): cada entidad A se asocia a lo sumo a cualquier entidad B, sin embargo, cada entidad B se puede asociar con cualquier entidad A.
- Relaciones muchos a muchos (\*:\*) : cada entidad A se asocia con cualquier entidad B y viceversa.

##### **Multiplicidad de relaciones complejas:**

Es el número o rango de posibles instancias de un tipo de entidad en una relación n-aria cuando los otros ( $n-1$ ) valores están fijos.

Representación de las restricciones de multiplicidad.

##### **Cardinalidad:**

Describe el número máximo de posibles instancias de relación para una entidad que participa en un tipo de relación dado (valores máximos de los rangos de multiplicidad en ambos lados de la relación).

##### **Participación:**

Determina si todas las instancias de entidad participan en una relación (participación obligatoria) o solo lo hacen algunas (participación opcional) (valores mínimos de los rangos de multiplicidad en ambos lados de la relación).

## 7.-PROBLEMAS CON LOS MODELOS ENTIDAD-RELACIÓN

### **Trampas multiplicativas:**

Cuando un modelo representa una relación entre tipos de entidad, pero la ruta entre ciertas instancias de entidad es ambigua (cruce de líneas al realizar el modelo entidad-relación)

Trampas de corte: cuando un modelo sugiere la existencia de una relación entre ciertos tipos de entidad, pero no existe ninguna ruta entre ciertas instancias de entidad (no se termina la relación entre entidades).

## 8.-MODELO ENTIDAD RELACIÓN EXTENDIDO

### **Superclase:**

Tipo de entidad que incluye uno o más subgrupos diferentes de sus instancias, los cuales es preciso representar en un modelo de datos.

### **Subclase:**

Subgrupo diferenciado de instancias de un tipo de entidad, que necesita ser representado en un modelo de datos.

Los tipos de entidad que poseen subclases diferenciadas se denominan superclases. Una superclase recoge un número de subclases, por lo que cada miembro de una subclase también lo es de una superclase, pero no ocurre lo mismo de forma inversa.

### **Especialización:**

Proceso de minimizar las diferencias entre miembros de una entidad identificando sus características distintivas (técnica de arriba-abajo)

### **Generalización:**

Proceso de minimizar las diferencias entre entidades identificando sus características comunes (técnica de abajo-arriba).

Una jerarquía de especialización es una clasificación de entidades de un TE en varias subclases. En el diagrama no se distingue de donde proviene (de una generalización o de una especialización). Una propiedad crucial en las Jerarquías es la Herencia de Atributos.

Hay dos tipos de restricciones que pueden aplicarse a la especialización/generalización:

### **Restricción de participación:**

Determina si todo un miembro de la superclase debe participar como miembro de una subclase. Puede ser obligatoria (Mandatory) u opcional (Optional).

### **Restricción de disyunción:**

Describe la relación entre miembros de las subclases e indica si es posible que un miembro de una superclase sea miembro de una subclase o más de una. Para aplicar esta restricción una superclase debe tener más de una subclase. Se utiliza la etiqueta Or para marcar la disyunción y la etiqueta And para la no disyunción.

## 9.-AGREGACIÓN

Representa una relación de tipo ‘tiene’ o ‘es parte de’ entre tipos de entidad, en la que uno de los tipos de entidad representa el ‘todo’ y el otro representa la ‘parte’. Se representa mediante un rombo en los extremos de la línea de relación que representa el ‘todo’. Una agregación es un TE de nivel conceptual superior.

## 10.-COMPOSICIÓN

Forma específica de agregación que representa una asociación entre entidades donde hay una pertenencia fuerte y una existencia coincidente entre el ‘todo’ y la ‘parte’. Se representa mediante un rombo relleno en los extremos de la línea de relación que representa el ‘todo’.

## **TEMA 4: EL MODELO RELACIONAL**

Este tipo de software representa la segunda generación de sistemas SGBD y está basado en el modelo de datos relacionales propuesto por E.F.Codd.

El modelo relacional es una técnica de diseño de BD de tipo abajo-arriba.

Todos los datos están estructurados desde el punto de vista lógico mediante relaciones (tablas).

Cada relación tiene un nombre y está compuesta de atributos (columnas) nominados de datos.

Cada tupla (fila) contiene un valor por cada atributo.

Una de las mayores fortalezas del modelo relacional es esta estructura lógica simple.

### **1.-BREVE HISTORIA DEL MODELO RELACIONAL**

Los objetivos de este modelo eran: permitir un alto grado de independencia de los datos, proporcionar una base teórica sólida que permitiera tratar con la semántica de los datos y con los problemas de coherencia y de redundancia (relaciones normalizadas) y permitir la ampliación de lenguajes de manipulación de datos orientados a conjuntos.

#### **1.1.-Proyectos de importancia en el desarrollo del modelo relacional:**

-El SGBD relacional prototipo System R (IBM), que condujo a dos desarrollos principales:

- El desarrollo de un lenguaje de consulta estructurado denominado SQL (que es el lenguaje estándar formal de ISO y de facto para los SGBD relacionales).
- El desarrollo de varios productos comerciales de SGBD relacional.

-INGRES. Fue el desarrollo de un prototipo de SGBDR, concentrándose en la investigación de los mismos objetivos globales que el proyecto System R.

-Peterlee Relational Test Vehicle. Orientación más teórica que los proyectos System R e INGRES, cuya principal importancia radica en la investigación en cuestiones tales como el procesamiento y optimización de consultas y la ampliación funcional.

### **2.-ESTRUCTURAS DE DATOS RELACIONALES**

#### **Relación:**

Es una tabla con columnas y filas.

Un SGBDR requiere sólo que la base de datos sea percibida por el usuario como una serie de tablas.

Sólo se aplica a la estructura lógica (no física).

#### **Atributo:**

Es una columna nominada de una relación.

Una relación se representa como una tabla bidimensional en la que las filas de la tabla corresponden a registros individuales y las columnas de la tabla corresponden a atributos, que pueden aparecer en cualquier orden.

#### **Dominio:**

Es un conjunto de valores permitidos para uno o más atributos.

Son una característica extremadamente potente del modelo relacional. Cada atributo de una relación está definido sobre un dominio.

El dominio permite al usuario definir en un lugar central el significado y el origen de los valores que los atributos pueden adoptar. El sistema tiene a su disposición a la hora de ejecutar una operación relacional mucha más información, pudiendo evitarse las operaciones que sean semánticamente incorrectas.

#### **Tupla:**

Es una fila de una relación.

Los elementos de una relación son las filas o tuplas de la tabla.

Pueden aparecer en cualquier orden y la relación continuará siendo la misma y transmitirá, por tanto, el mismo significado.

La estructura de una relación, junto con una especificación de los dominios y cualesquiera restricciones sobre los posibles valores se denomina en ocasiones intensión de la relación. Las tuplas se denominan extensión de una relación; la extensión cambia a lo largo del tiempo.

#### **Grado:**

Es el número de atributos que contiene una relación.

El grado de una relación es una propiedad de la intensión de la relación.

#### **Cardinalidad:**

Es el número de tuplas que contiene.

Propiedad de la extensión de la relación.

#### **Bases de datos relacional:**

Colección de relaciones normalizadas en la que cada relación tiene un nombre distintivo.

Para referirnos a que la estructura de las relaciones sea apropiada utilizamos el término normalización.

Términos formales	Alternativa 1	Alternativa 2
Relación	Tabla	Archivo
Tupla	Fila	Registro
Atributo	Columna	Campo

### 3.-RELACIONES EN UNA BASE DE DATOS

#### **Esquema de relación:**

Definida por un conjunto de parejas de atributos y nombres de dominio.

Instancia de relación: `{(branchNo:B005, street:22 Deer Rd, city: London, postcode: SW1 4EH)}`

Las relaciones, en términos matemáticos, se definen como el producto cartesiano entre 2 o más conjuntos.

Para definir estas relaciones tenemos que especificar los conjuntos, o dominios, entre los que vamos a seleccionar los valores.

#### **Esquema de la base de datos relacional:**

Conjunto de esquemas de relación, cada uno con un nombre distintivo.

### 4.-PROPIEDADES DE LAS RELACIONES

- Cada relación tiene un nombre distinto del resto de los nombres de las relaciones del esquema relacional.
- Cada celda tiene un valor atómico (único). No posee ni atributos compuestos ni multivalorados.
- Cada atributo tiene un nombre distintivo.
- Los valores de un atributo pertenecen todos al mismo dominio.
- Cada tupla es diferente.
- El orden de los atributos no tiene importancia.
- El orden de las tuplas no tiene importancia teóricamente (en la práctica, el orden puede afectar a la eficiencia de acceso a las tuplas).

Las relaciones no contienen grupos repetitivos. Una relación que satisface esta propiedad se denomina normalizada.

### 5.-CLAVES RELACIONALES

No existen tuplas duplicadas dentro de una relación, por lo tanto, vamos a tener que poder identificar uno o más atributos que identifiquen de forma única cada tupla de una relación.

**Superclave:**

Atributo o conjunto de atributos que identifica de forma única cada tupla dentro de una relación.

Una superclave puede contener atributos adicionales que no sean necesarios para la identificación única, por lo que resultaría interesante identificar aquellas superclaves que sólo contengan el número mínimo de atributos necesarios para efectuar de forma única la identificación.

**Clave candidata:**

Superclave tal que ningún subconjunto propio de la misma es una superclave de la relación. Propiedades: unicidad e irreducibilidad.

Puede haber varias claves candidatas para una relación. Cuando una clave está compuesta de más de un atributo decimos que es una clave compuesta.

**Clave principal:**

Es la clave candidata seleccionada para identificar las tuplas de forma única dentro de la relación.

Aquellas que no son principales se denominan claves alternativas.

Cabe mencionar que los tres conceptos explicados con anterioridad se consideran equivalentes.

**Clave externa:**

Un atributo, o un conjunto de atributos, dentro de una relación que se corresponden con la clave candidata de alguna (posiblemente la misma) relación.

Cuando un atributo aparece en más de una relación, su aparición suele representar que existe algún tipo de conexión entre las tuplas de las dos relaciones.

### REPRESENTACIÓN DE ESQUEMAS DE BASES DE DATOS RELACIONAL

El convenio común para representar un esquema de relación consiste en indicar el nombre de la relación seguido por los nombres de los atributos entre paréntesis.

## 6.-RESTRICCIONES DE INTEGRIDAD

Existen restricciones (denominadas restricciones de dominio) que imponen una limitación al conjunto de valores permitidos para los atributos de las relaciones.

Las dos reglas principales para el modelo relacional se conocen con los nombres de integridad de entidad e integridad referencial.

### 6.1.-Valores nulos:

Representa un valor para un atributo que es actualmente desconocido o no es aplicable para esta tupla.

### 6.2.-Integridad de entidad:

En una relación base ningún atributo de una clave principal puede ser nulo.

Por definición, una clave principal es un identificador mínimo que se utiliza para identificar de manera única las tuplas. Ningún subconjunto de la clave principal es suficiente como para permitir una identificación única de las tuplas.

### 6.3.-Integridad referencial:

Si hay una clave externa en una relación, el valor de la clave externa debe corresponderse con el valor de una clave candidata de alguna tupla.

#### 6.4.-Restricciones generales:

Son reglas adicionales especificadas por los usuarios o administradores de la base de datos que definen o restringen algún aspecto de la organización.

El esquema de relación (asocia parejas de atributos y dominios) no es una restricción de integridad.

#### 7.-VISTAS

Una vista es una relación virtual que no tiene por qué existir necesariamente en la base de datos, sino que puede producirse cuando se solicite por parte de un usuario concreto, generándosela en el momento de la solicitud.

Las vistas son dinámicas, lo que significa que los cambios que se hagan en las relaciones base que afecten a las vistas se reflejan inmediatamente en ésta.

(Leer esta parte en el libro).

## **TEMA 5: ÁLGEBRA RELACIONAL**

Es un lenguaje teórico de manipulación con operaciones que se aplican a una o más relaciones, con el fin de definir otra relación sin modificar las relaciones originales. Por tanto, los operandos como los resultados son relaciones, de manera que la salida de una operación puede utilizarse como entrada de otra operación.

Esto permite anidar expresiones, esta propiedad de denomina **cierre**: las relaciones exhiben la propiedad de cierre en el álgebra relacional.

### **1.-OPERACIONES UNARIAS**

#### **Selección (o Restricción):**

$\sigma_{\text{predicate}}(R)$ : LA OPERACIÓN DE SELECCIÓN SE APLICA A UNA ÚNICA RELACIÓN R Y DEFINE OTRA RELACIÓN QUE CONTIENE ÚNICAMENTE AQUELLAS TUPLAS DE R QUE SATISFACEN LA CONDICIÓN (PREDICADO) ESPECIFICADA.

Ej.: Enumerar todos los miembros del personal cuyo salario sea superior a 10.000 euros.

$$\sigma_{\text{salary} > 1000}(\text{Staff})$$

#### **Proyección:**

$\Pi_{a_1, \dots, a_n}(R)$ : LA OPERACIÓN DE PROYECCIÓN SE APLICA A UNA ÚNICA RELACIÓN R Y DEFINE OTRA RELACIÓN QUE CONTIENE UN SUBCONJUNTO VERTICAL DE R, EXTRAYENDO LOS VALORES DE LOS ATRIBUTOS ESPECIFICADOS Y ELIMINANDO LOS DUPLICADOS.

Ej.: Generar una lista de salarios para todo el personal, mostrando solamente los detalles referidos a los atributos staffNo, fName, lName y salary.

$$\Pi_{\text{staffNo}, \text{fName}, \text{lName}, \text{salary}}(\text{Staff})$$

### **2.-OPERACIONES DE CONJUNTOS**

Estas operaciones combinan información de diversas relaciones.

#### **Unión:**

$R \cup S$ : LA UNIÓN DE DOS RELACIONES R Y S DEFINE UNA RELACIÓN QUE CONTIENE TODAS LAS TUPLAS DE R, DE S O TANTO DE R COMO DE S, ELIMINÁNDOSE LAS TUPLAS DUPLICADAS. R Y S DEBEN SER COMPATIBLES CON RESPECTO A LA UNIÓN.

Concatenando las relaciones en una sola, la nueva relación tendrá un máximo de I+J tuplas, siendo I y J las tuplas de las relaciones originales.

Ej.: Enumerar todas las ciudades en las que exista una sucursal o un inmueble en alquiler.

$$\Pi_{\text{city}}(\text{Branch}) \cup \Pi_{\text{city}}(\text{PropertyForRent})$$

#### **Diferencia de conjuntos :**

$R - S$ : LA OPERACIÓN DE DIFERENCIA DE CONJUNTOS DEFINE UNA RELACIÓN COMPUESTA POR LAS TUPLAS QUE SE ENCUENTRAN EN LA RELACIÓN R, PERO NO EN S. R Y S DEBEN SER COMPATIBLES RESPECTO A LA UNIÓN.

Ej.: Enumerar todas las ciudades en las que exista una sucursal, pero no haya inmuebles en alquiler.

$$\Pi_{\text{city}}(\text{Branch}) - \Pi_{\text{city}}(\text{PropertyForRent})$$

### Intersección:

R  $\cap$  S: LA OPERACIÓN DE INTERSECCIÓN DEFINE UNA RELACIÓN COMPUESTA POR EL CONJUNTO DE TODAS LAS TUPLAS QUE EXISTEN TANTO EN R COMO EN S. R Y S DEBEN SER COMPATIBLES CON RESPECTO A LA UNIÓN.

$$R \cap S = R - (R - S)$$

Ej.: Enumerar todas las ciudades en las que exista tanto una sucursal como al menos un inmueble en alquiler.

$$\Pi_{\text{city}}(\text{Branch}) \cap \Pi_{\text{city}}(\text{PropertyForRent})$$

### Producto cartesiano:

R  $\times$  S: LA OPERACIÓN DE PRODUCTO CARTESIANO DEFINE UNA RELACIÓN QUE ES LA CONCATENACIÓN DE CADA TUPLA DE LA RELACIÓN R CON CADA TUPLA DE LA RELACIÓN S.

La operación multiplica dos relaciones para definir una relación compuesta por todas las posibles parejas de tuplas. Por tanto, si una relación tiene I tuplas y N atributos y la otra tiene J tuplas y M atributos, la relación compuesta tendrá (I \* J) tuplas y (N+M) atributos.

Ej.: Enumerar los nombres y comentarios de todos los clientes que hayan visto un inmueble en alquiler.

$$(\Pi_{\text{clientNo}, \text{fName}, \text{lName}}(\text{Client})) \times (\Pi_{\text{clientNo}, \text{propertyNo}, \text{comment}}(\text{Viewing}))$$

Podemos descomponer dichas operaciones en una serie de operaciones más pequeñas de álgebra relacional y proporcionar un nombre a los resultados de las expresiones intermedias. Utilizamos la operación de asiganción, denotada por  $\leftarrow$ , para nombrar los resultados de una operación del álgebra relacional.

$\rho_S(E)$ : LA OPERACIÓN RENOMBRAR PROPORCIONA UN NUEVO NOMBRE S PARA LA EXPRESIÓN E Y OPCIONALMENTE ASIGNA A LOS ATRIBUTOS LOS NOMBRES A<sub>1</sub>, A<sub>2</sub>,..., A<sub>n</sub>.

## 3.-OPERACIONES DE COMBINACIÓN

Combinan dos relaciones para formar una nueva relación. La combinación es equivalente a realizar una operación de selección utilizando el predicado de combinación como fórmula de selección, sobre el producto cartesiano de las dos relaciones que actúan como operando.

### Combinación Theta ( $\theta$ -combinación):

R  $\bowtie_F$  S: LA OPERACIÓN DE COMBINACIÓN THETA DEFINE UNA RELACIÓN QUE CONTIENE TUPLAS DEL PRODUCTO CARTESIANO DE R Y S QUE SATISFACEN EL PREDICADO F. EL PREDICADO F TIENE LA FORMA R.a<sub>i</sub>  $\theta$  R.b<sub>j</sub>, DONDE  $\theta$  PUEDE SER UNO DE LOS OPERANDORES DE COMPARACIÓN (<, >, =,  $\leq$ ,  $\geq$ ,  $\neq$ )

$$R \bowtie_F S = \sigma_F(R \times S)$$

El grado de una combinación theta es la suma de los grados de las relaciones R y S que actúan como operandos. En caso de que el predicado F sólo contenga el operador de igualdad (=), se utiliza el término **equicombinación**.

Ej.: Enumerar los nombres y comentarios de todos los clientes que hayan visitado un inmueble en alquiler.

$$\begin{aligned} &(\Pi_{\text{clientNo}, \text{fName}, \text{lName}}(\text{Client})) \bowtie_{\text{Client.clientNo} = \text{Viewing.clientNo}} \\ &(\Pi_{\text{clientNo}, \text{propertyNo}, \text{comment}}(\text{Viewing})) \end{aligned}$$

### **Combinación natural:**

R  $\bowtie$  S: LA COMBINACIÓN NATURAL ES UNA EQUICOMBINACIÓN ENTRE LAS DOS RELACIONES R Y S SOBRE TODOS LOS ATRIBUTOS COMUNES x. DEL RESULTADO SE ELIMINA UNA DE LAS DOS APARICIONES DE CADA ATRIBUTO COMÚN.

El grado de una combinación natural es la suma de los grados de las relaciones R y S, menos le número de atributos de x.

Ej.: Enumerar los nombres y comentarios de todos los clientes que hayan visitado un inmueble en alquiler.

$$(\Pi_{clientNo,fName,IName(Client)}) \bowtie (\Pi_{clientNo,propertyNo,comment(Viewing)})$$

### **Combinación externa:**

Al combinar dos relaciones, una tupla a menudo no tiene ninguna tupla correspondiente en la otra; es decir, no existe ningún valor correspondiente en los atributos de combinación. Puede que queramos que en el resultado aparezcan las tuplas aunque no existan los valores correspondientes. En este caso llevamos a cabo una combinación externa.

R = $\bowtie$  S: LA COMBINACIÓN EXTERNA (IZQUIERDA) ES UNA COMBINACIÓN EN LA QUE TAMBIÉN SE INCLUYEN EN LA RELACIÓN RESULTANTE LAS TUPLAS DE R QUE NO TENGAN VALORES CORRESPONDIENTES EN LOS ATRIBUTOS COMUNES DE S. A LOS VALORES NO EXISTENTES EN LA SEGUNDA RELACIÓN SE LES ASIGNA UN VALOR NULO

La combinación externa preserva las tuplas que se habrían perdido si se utilizaran otros tipos de combinación.

Ej.: Generar un informe de estado sobre las visitas a los inmuebles

$$\Pi_{property,street,city(PropertyForRent)} =\bowtie Viewing$$

De forma similar, existe una **combinación externa derecha** que conserva todas las tuplas de la relación del lado derecho del resultado. También existe una **combinación externa completa** que conserva todas las tuplas de ambas relaciones, rellenando las tuplas con nulos cuando no se encuentra ninguna tupla correspondiente en la otra relación.

### **Semicombinación:**

R  $\triangleright F$  S: LA OPERACIÓN DE SEMICOMBINACIÓN DEFINE UNA RELACIÓN QUE CONTIENE LAS TUPLAS DE R QUE PARTICIPAN EN LA COMBINACIÓN DE R Y S.

Realiza una combinación de las dos relaciones y luego la proyecta sobre los atributos del primer operando.

$$R \triangleright F S = \Pi_A(R \triangleleft F S)$$

Ej.: Enumerar los detalles completos de todos los empleados que trabajen en la sucursal de Glasgow.

$$Staff \triangleright Staff.branchNo = Branch.branchNo (\sigma_{city = "Glasgow"} (Branch))$$

## 4.-OPERACIONES DE DIVISIÓN

Supongamos que la relación R está definida sobre el conjunto de atributos A y que la relación S está definida sobre el conjunto B, de modo que B  $\subseteq$  A (B es un subconjunto de A). Se C = A - B, es decir, C es el conjunto de atributos de R que no son de S. Podemos definir la operación de división de la forma siguiente:

R  $\div$  S: LA OPERACIÓN DE DIVISIÓN DEFINE UNA RELACIÓN SOBRE LOS ATRIBUTOS C QUE ESTÁ COMPUESTA POR EL CONJUNTO DE TUPLAS DE R QUE SE CORRESPONDEN CON LA COMBINACIÓN DE **TODAS** LAS TUPLAS E S.

$$\begin{aligned} T_1 &\leftarrow \Pi_C(R) \\ T_2 &\leftarrow \Pi_C(S \times T_1 - R) \\ T &\leftarrow T_1 - T_2 \end{aligned}$$

Ej.: Identificar todos los clientes que hayan visto todos los inmuebles con tres habitaciones

$$(\Pi_{clientNo, propertyNo} (Viewing)) \div \Pi_{propertyNo} (\sigma_{rooms=3} (PropertyForRent))$$

## 5.-OPERACIONES DE AGREGACIÓN Y DE AGRUPAMIENTO

### Agregación:

$\Sigma_{AL}(R)$ : APLICA LA LISTA DE FUNCIONES DE AGREGACIÓN, AL, A LA RELACIÓN R PARA DEFINIR UNA RELACIÓN SOBRE LA LISTA DE AGREGACIÓN. AL CONTIENE UNA O MÁS PAREJAS ( $\langle$ función\_agregacion $\rangle$ ,  $\langle$ atributo $\rangle$ ).

COUNT: devuelve el número de valores en el atributo asociado.  
 SUM: devuelve la suma de los valores en el atributo asociado.  
 AVG: devuelve la media de los valores en el atributo asociado.  
 MIN: devuelve el valor más pequeño en el atributo asociado.  
 MAX: devuelve el valor máximo en el atributo asociado.

Ej.: ¿Cuántos inmuebles tienen un precio de alquiler superior a 350 euros por mes?

$$\rho_R(myCount) \Sigma COUNT \text{ propertyNo } (\sigma_{rent < 350} (PropertyForRent))$$

Localizar los valores mínimo, máximo y medio de los salarios de los empleados

$$\rho_R(myCount, myMax, myAverage) \Sigma MIN \text{ salary, MAX salary, AVG salary } (Staff)$$

### Agrupación:

$\text{GA} \Sigma_{AL}(R)$ : AGRUPA LAS TUPLAS DE LA RELACIÓN R SEGÚN LOS ATRIBUTOS DE AGRUPACIÓN, GA, Y LUEGO APLICA LA LISTA DE FUNCIONES DE AGREGACIÓN AL PARA DEFINIR UNA NUEVA RELACIÓN. AL CONTIENE UNA O MÁS PAREJAS ( $\langle$ función\_agregación $\rangle$ ,  $\langle$ atributo $\rangle$ ). LA RELACIÓN RESULTANTE CONTIENE LOS ATRIBUTOS DE AGRUPACIÓN, GA, JUNTO CON LOS RESULTADOS DE CADA UNA DE LAS FUNCIONES DE AGREGACIÓN.

Las tuplas de R se partitionan en grupos de tal forma que:

-todas las tuplas de cada grupo tienen el mismo valor de a<sub>1</sub>, a<sub>2</sub>,..., a<sub>n</sub>.

-las tuplas de los diferentes grupos tienen diferentes valores de a<sub>1</sub>, a<sub>2</sub>,..., a<sub>n</sub>.

Ej.: Calcular el número de empleados que trabajan en cada sucursal y la suma de sus salarios

$$\rho_R(brancNo, myCount, mySum) \text{branchNo} \Sigma COUNT \text{ staffNo, SUM salary } (Staff)$$

## **TEMA 6: RESUMEN BÁSICO SOBRE SQL**

Lenguaje de computación para trabajar con conjuntos de datos y las relaciones entre ellos.

### **INSTRUCCIONES SELECT:**

Describe un conjunto de datos que se quiere obtener de una base de datos. Incluye: qué tablas contienen los datos, cómo se relacionan los datos de orígenes diferentes, qué campos o cálculos proporcionarán los datos, criterios que los datos deben cumplir para ser incluidos y si se deben ordenar los datos y, en caso de ser así, cómo deben ordenarse.

### **CLAÚSULAS SQL:**

Cada cláusula realiza una función de la instrucción SQL. Algunas son necesarias en una instrucción SELECT. Las más comunes son:

**SELECT** -> muestra una lista de los campos que contienen datos de interés. *Ob.*

**FROM** -> muestra las tablas que contienen los campos de la cláusula SELECT. *Ob.*

**WHERE** -> especifica los criterios de campo que cada registro debe cumplir para poder ser incluido en los resultados. *Op.*

**ORDER BY** -> especifica la forma de ordenar los resultados. *Op.*

**GROUP BY** -> en una instrucción SQL que contiene funciones de agregado, muestra los campos que no se resumen en la cláusula SELECT. *Ob sólo si están estos campos.*

**HAVING** -> en una instrucción con funciones de agregado, especifica las condiciones que se aplican a los campos que se resumen en la instrucción SELECT. *Op.*

### **CLÁUSULAS SQL BÁSICAS: SELECT, FROM, WHERE**

**Select:** se compone de un operador (SELECT) seguido de los identificadores. Si un identificador contiene espacios se debe escribir entre [ ]. Siempre aparece antes de la cláusula FROM en una instrucción SELECT.

**From:** se compone de un operador (FROM) seguido de un identificador. No enumera los campos que se van a seleccionar.

**Where:** se compone de un operador (WHERE) seguido de los identificadores.

### **ORDER BY:**

Ordena los resultados de la consulta. Es la última consulta en la instrucción SQL. Contiene una lista de los campos que se quiere usar para ordenar, en el mismo orden en el que quiere aplicar las operaciones de ordenación (descendiente-ascendiente).

*Sintaxis:*

*SELECT lista\_de\_campos*

```
FROM tabla  
WHERE criterios_de_selección  
[ORDER BY campo1 [ASC | DESC ][, campo2 [ASC | DESC ]][, ...]]
```

Una instrucción SELECT que contiene una cláusula ORDER BY consta de las siguientes partes:

*listadecampos* = nombre del campo o campos que se van a recuperar junto con cualquier alias de nombre campo, funciones de agregado, predicados de selección...

*tabla* = nombre de la tabla de la cual se recuperan los registros.

*criteriosdeselección* = si la instrucción incluye una cláusula WHERE, los datos se ordenan después de aplicar dicha cláusula.

*campo1, campo2* = nombres de los campos por los que se quiere ordenar el registro.

#### GROUP BY:

Se deben utilizar cuando utilizamos funciones de agregado. Esta cláusula enumera todos los campos a los que no se aplica una función de agregado.

Va inmediatamente después de la cláusula WHERE, o la cláusula FROM si no hay WHERE.

*Sintaxis:*

```
SELECT listadecampos  
FROM tabla  
WHERE criterios  
[GROUP BY listadecamposdegrupo]
```

Contiene las mismas partes que ORDER BY, pero en vez de campo tenemos *listadecamposdegrupo*, que son nombres de hasta un máximo de 10 campos utilizados para agrupar registros. El orden de los nombres de campo de *listadecamposdegrupo* determina los niveles de agrupación desde el nivel más alto al más bajo.

Los valores de resumen se omiten si no hay ninguna función de agregado y los valores NULL de los campos GROUP BY se agrupan y no se omiten, pero no se evalúan en ninguna función.

#### HAVING:

Si quiere usar criterios para limitar los resultados, pero el campo al que quiere aplicar los criterios se usa en una función de agregado, no podemos usar una cláusula WHERE. En su lugar, usamos una cláusula HAVING.

Es decir, la cláusula HAVING funciona como una WHERE pero con datos agregados.

#### UNION:

Sirve para combinar los resultados de dos instrucciones SELECT. Deben tener el mismo número de campos de resultado, en el mismo orden y con el mismo tipo de datos o con tipos de datos compatibles.

Cuando se quiere trabajar con datos resumidos, se aplican **funciones de agregado** a un campo de la cláusula SELECT, como:

```
SELECT COUNT([E-mail Address]), Company
```

Funciones de agregado: AVG, Count, First y Last, Min y Max, StDev y StDevP, suma, Var y VarP...

## **TEMA 7: NORMALIZACIÓN**

Técnica para producir un conjunto de relaciones con una serie de propiedades deseables. El propósito de la normalización es identificar un conjunto adecuado de relaciones que soporte los requisitos de datos de una organización.

Las características de un conjunto adecuado incluyen:

- Número mínimo de atributos necesarios para soportar los requisitos de datos de la organización.
- Los atributos con una relación lógica fuerte se encuentran en la misma relación.
- Una redundancia mínima, estando cada atributo representado una sola vez.

### **1.-CÓMO AYUDA LA NORMALIZACIÓN AL DISEÑO DE BASES DE DATOS**

Es una técnica formal que puede utilizarse en cualquier etapa del diseño de bases de datos. Resaltamos dos técnicas principales:

- Técnica autónoma de tipo abajo-arriba para el diseño de bases de datos.
- Técnica de validación para comprobar las estructuras de las relaciones.

**Objetivo:** crear un conjunto de relaciones bien diseñadas que satisfagan los requisitos de datos de la organización.

### **2.-REDUNDANCIA DE LOS DATOS Y ANOMALÍAS DE ACTUALIZACIÓN**

Si conseguimos agrupar los atributos en relaciones de modo que se minimice la redundancia, podemos obtener diversas ventajas en la implementación de la base de datos:

- Las actualizaciones de los datos almacenados en la base de datos pueden llevarse a cabo con un número mínimo de operaciones.
- Se reduce el espacio de almacenamiento de archivos requerido.

La redundancia aparece en forma de copias de las claves primarias (o claves candidatas), copias que actúan como claves externas en las relaciones correspondientes, para permitir modelar las relaciones entre los datos.

Las relaciones con datos redundantes pueden presentar problemas que se denominan anomalías de actualización.

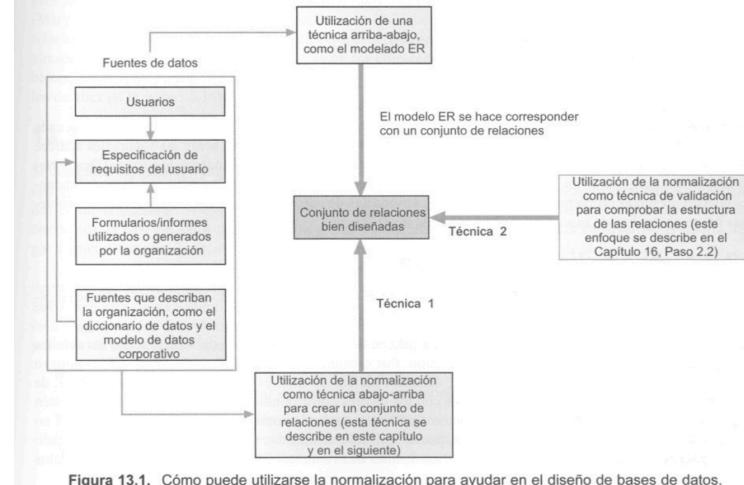


Figura 13.1. Cómo puede utilizarse la normalización para ayudar en el diseño de bases de datos.

Tipos de anomalías: de inserción, de borrado y de modificación.

#### **2.1.-Anomalías de modificación:**

Hay dos propiedades importantes asociadas con la descomposición de una relación de mayor tamaño en una serie de relaciones más pequeñas:

- **Combinación sin pérdidas:** garantiza que cualquier instancia de la relación original pueda ser identificada a partir de las instancias correspondientes de las relaciones más pequeñas.
- **Preservación de la dependencia:** no necesitamos combinar las relaciones más pequeñas para comprobar si se viola o no una restricción de la relación original.

### 3.-DEPENDENCIAS FUNCIONALES

Describe la relación entre atributos de una relación. Por ejemplo, si A y B son atributos de la relación R, B será funcionalmente dependiente de A ( $A \rightarrow B$ ) si cada valor de A está asociado con exactamente un valor de B (A y B pueden consistir cada uno de ellos de uno o más atributos). Se trata de una propiedad del significado o semántica de los atributos de una relación. Cuando hay presente una dependencia funcional, la dependencia se especifica como una **restricción** entre atributos.

#### **Determinante:**

Hace referencia al atributo o grupo de atributos en el lado izquierdo de la flecha que describe una dependencia funcional.

Una característica adicional es que sus determinantes deben tener el número mínimo de atributos necesario para mantener la dependencia funcional de los atributos del lado derecho. Este requisito se denomina **dependencia funcional completa** (cuando la eliminación de cualquier atributo de A hace que la dependencia deje de existir).

En resumen: hay una relación *uno a uno* entre los atributos del lado izquierdo (determinante) y los del lado derecho de la dependencia funcional, se cumplen en todo instante de tiempo y debe haber una dependencia funcional completa entre los atributos del lado izquierdo y los del lado derecho de la dependencia.

#### **Dependencia Transitiva:**

Una condición en la que A, B y C son atributos de una relación tales que si  $A \rightarrow B$  y  $B \rightarrow C$ , entonces C depende transitivamente de A a través de B.

Una restricción de integridad importante que hay que considerar en primer lugar es la identificación de las claves candidatas, una de las cuales será seleccionada como clave principal de la relación.

### 4.-EL PROCESO DE NORMALIZACIÓN

Cuando un requisito no sea satisfecho, la relación que viole dicho requisito deberá ser descompuesta en una serie de relaciones que cumplan individualmente los requisitos de la normalización.

Formas: primera forma normal (1NF), segunda forma normal (2NF) y tercera forma normal (3NF).

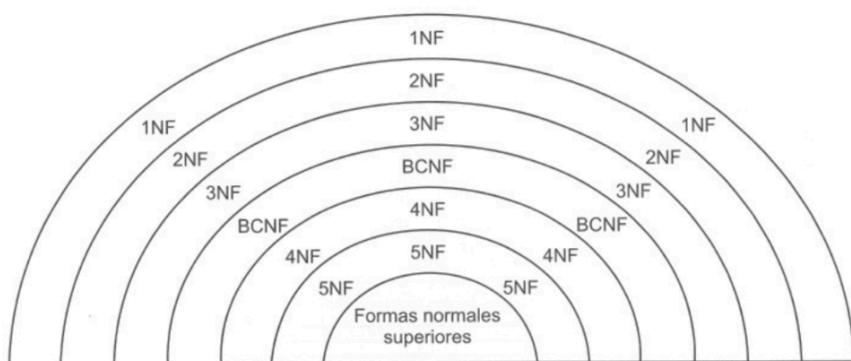


Figura 13.7. Diagrama que ilustra la relación entre las formas normales.

#### FORMA NO NORMALIZADA (UNF)

Una tabla que contiene uno o más grupos repetitivos.

## 5.-PRIMERA FORMA NORMAL (1NF)

Una relación en la que la intersección de toda fila y columna contiene un valor y sólo un valor. Para transformar la tabla no normalizada a primera forma normal, tenemos que identificar y eliminar los grupos repetitivos dentro de la tabla. Un grupo repetitivo es un atributo, o grupo de atributos, dentro de una tabla que presentan múltiples valores para un mismo valor.

Hay dos técnicas principales para eliminar los grupos repetitivos de las tablas no normalizadas:

- Llenamos los blancos duplicando los datos no repetitivos siempre que sea necesario. Esta técnica se suele denominar 'aplanamiento' de la tabla.
- Colocando los datos repetitivos, junto con una copia de los atributos clave originales en una relación independiente.

## 6.-SEGUNDA FORMA NORMAL (2NF)

La segunda forma normal está basada en el concepto de dependencia funcional completa, se aplica a las relaciones con claves compuestas, es decir, con una clave principal compuesta de dos o más atributos.

Una reacción que está en primera forma normal y en la que todo atributo que no sea de clave principal depende funcionalmente de manera completa de la clave principal.

## 7.-TERCERA FORMA NORMAL (3NF)

Ningún atributo que no sea de clave principal depende transitivamente de la clave principal. La normalización de las relaciones 2NF para pasarlas a forma 3NF implica la eliminación de las dependencias transitivas.

### **DEFINICIONES GENERALES DE LAS FORMAS 2NF Y 3NF**

Definimos el concepto de atributo de clave candidata como todo atributo que forme parte de una clave candidata y definimos las dependencias parcial, total y transitiva con respecto a todas las claves candidatas de una relación.

## 8.-REGLAS DE INFERENCIA PARA DEPENDENCIAS FUNCIONALES

Es decir, que si imponemos las restricciones de integridad definidas por las dependencias funcionales de X, estaremos imponiendo automáticamente las restricciones de integridad definidas en el conjunto de dependencias funcionales Y, que es de un tamaño mayor. Este requisito sugiere que debe haber dependencias funcionales que puedan inferirse a partir de otras dependencias funcionales.

El conjunto de todas las dependencias funcionales que pueden derivarse a partir de un conjunto dado de dependencias funcionales X se denomina cierre de X, lo que se escribe como  $X^+$ .

Un conjunto de reglas de inferencia, denominadas Axiomas de Armstrong, especifica cómo inferir nuevas dependencias funcionales a partir de otras dadas.

Reflexividad: si B es un subconjunto de A  $\Rightarrow A \rightarrow B$

Aumentación: si  $A \rightarrow B \Rightarrow A, C \rightarrow B, C$

Transitividad: si  $A \rightarrow B$  y  $B \rightarrow C \Rightarrow A \rightarrow C$

Estas reglas pueden utilizarse para hallar el cierre de  $X^+$ .

Otras reglas:

Autodeterminación: $A \rightarrow A$
Descomposición: si $A \rightarrow BC \Rightarrow A \rightarrow B$ y $A \rightarrow C$
Unión: si $A \rightarrow B$ y $A \rightarrow C \Rightarrow A \rightarrow BC$
Composición: si $A \rightarrow B$ y $C \rightarrow D \Rightarrow AC \rightarrow BD$

Una forma sistemática de determinar estas dependencias funcionales adicionales consiste en determinar primero cada conjunto de atributos A que aparezca en el lado izquierdo de alguna dependencia funcional y luego determinar el conjunto de todos los atributos que sean dependientes de A. Así, para cada conjunto de atributos A podemos determinar el conjunto  $A^+$  de atributos que están funcionalmente determinados por A basándose en F ( $A^+$  se denomina **cierre de A bajo F**).

## 9.-CONJUNTOS MÍNIMOS DE DEPENDENCIAS FUNCIONALES

En esta sección, vamos a presentar el concepto de equivalencia de conjuntos de dependencias funcionales. Un conjunto de dependencias funcionales Y está recubierto por un conjunto de dependencias funcionales X, si toda dependencia funcional de Y está también en  $X^+$ ; es decir, si toda dependencia contenida en Y puede inferirse a partir de X.

### Conjunto mínimo:

Consideramos un conjunto de dependencias funcionales X mínimo si:

- Toda dependencia contenida en X tiene un único atributo en su lado derecho.
- No podemos substituir ninguna dependencia  $A \rightarrow B$  de X por la dependencia  $C \rightarrow B$ , donde C es un subconjunto propio de A, y continuar obteniendo un conjunto de dependencias equivalente a X.
- No podemos eliminar ninguna dependencia de X y seguir teniendo un conjunto de dependencias equivalente a X.

Este conjunto debe estar en forma estándar, sin ninguna redundancia. Un recubrimiento mínimo de un conjunto de dependencias funcionales X es un conjunto mínimo de dependencias  $X_{\min}$  que sea equivalente a X. Desafortunadamente, puede haber varios recubrimientos mínimos para un conjunto de dependencias funcionales.

## 10.-FORMA NORMAL DE BOYCE-CODD

A pesar de las restricciones adicionales, pueden continuar existiendo dependencias que hagan que exista una cierta redundancia en las relaciones 3NF. Esta debilidad de la forma normal 3NF hizo que se desarrollara una forma normal más fuerte, denominada forma de Boyce-Codd.

Una relación está en BCNF, si y sólo si todo determinante es una clave candidata.

La forma normal de Boyce-Codd es mas fuerte que la forma 3NF, de modo que toda relación en BCNF está también en 3NF. La posibilidad de violar las condiciones de la forma normal BCNF puede aparecer cuando:

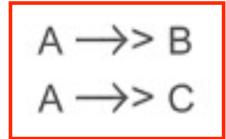
- La relación contenga dos (o más) claves candidatas compuestas: o
- Las claves candidatas se solapen, es decir, tengan al menos un atributo en común.

## 11.-CUARTA FORMA NORMAL (4NF)

La posible existencia de dependencias multivaluadas en una relación se debe a la primera forma normal, que impide que un atributo de una tupla tenga un conjunto de valores diferentes. Por ejemplo, si tenemos dos atributos multivaluados en una relación, es preciso repetir cada valor de uno de los atributos con cada uno de los valores del otro atributo, para garantizar que las tuplas de la relación sean coherentes. Este tipo de restricción se denomina dependencia multivaluada y provoca que aparezca redundancia en los datos.

### **Dependencia multivaluada:**

Representa una dependencia entre atributos (por ejemplo, A, B Y C) en una relación de modo que para cada valor de A hay un conjunto de valores de B y un conjunto de valores de C. Sin embargo, los conjuntos de valores de B y C son independientes entre sí.



Las dependencias multivaluadas pueden ser triviales (si B es un subconjunto de A o  $A \cup B = R$ ) o no triviales.

### **Cuarta forma normal:**

Una relación que está en forma normal de Boyce-Codd y no contiene dependencias multivaluadas no triviales.

La normalización de relaciones BCNF a 4NF implica la eliminación de las dependencias multivaluadas de la relación, colocando los atributos en una nueva relación junto con una copia de los determinantes.

## 12.-QUINTA FORMA NORMAL (5NF)

Cada vez que descomponemos una relación en dos relaciones, las relaciones resultantes tienen la propiedad denominada de combinación sin pérdidas. Sin embargo, hay casos en los que se necesita descomponer una relación en más de dos relaciones. Aunque son raros, estos casos están gobernados por la denominada dependencia de combinación y la quinta forma normal.

### **Dependencia de combinación sin pérdidas:**

Una propiedad de la descomposición que garantiza que no se generen tuplas espurias (relación matemática en la cual dos acontecimientos no tienen conexión lógica) al volver a combinar las relaciones mediante una operación de combinación natural.

Al dividir las relaciones mediante una operación de proyección, estamos siendo muy explícitos acerca del método de descomposición. En particular, tenemos cuidado de utilizar proyecciones que puedan invertirse mediante combinación de las relaciones resultantes, de modo que pueda reconstruirse la relación original. Dicho tipo de descomposición se denomina descomposición de combinación sin pérdidas.

### **Quinta forma normal:**

Una relación no tiene dependencias de combinación.

### **Dependencia de combinación:**

Describe un tipo de dependencia. Por ejemplo, para una relación R compuesta por una serie de subconjuntos de los atributos de R denominados A, B, ..., Z, la relación R exhibirá una dependencia de combinación si y sólo si todo valor legal de R es igual a la combinación de sus proyecciones sobre A, B, ..., Z.