Diseño Conceptual de Bases de Datos

Conceptos Fundamentales.

Marcela Varas

Contenidos.

Parte I. Conceptos Fundamentales.

I. Conceptos Fundamentales.

1 Realidades, Modelos y Lenguajes.

La realidad única, concreta y objetiva no puede ser captada como tal. Aún cuando pudiésemos asumir que esta realidad única existe, cada uno de nosotros la modifica a través del filtro de su percepción. La percepción de cada persona es algo bastante complejo, que está influido, entre otros posibles factores, por el tiempo, espacio y estado de ánimo al momento de realizar la percepción, además del impacto de experiencias previas, factores ambientales, estructura neuronal y el código genético del individuo.

Lo relevante es que para n observadores de un fenómeno, es posible obtener al menos n percepciones distintas (aunque posiblemente no “radicalmente” distintas). Las herramientas que utilizamos para poder comunicar y plasmar nuestras percepciones de realidades se denominan modelos. Los modelos son representaciones de algún fenómeno o hecho del mundo que nos interese (en el caso de la ingeniería de sistemas interesaría por ejemplo modelar organizaciones, datos o procesos de negocio). Para poder expresar estos modelos es que requerimos de los lenguajes.

Los lenguajes son herramientas creadas por el hombre (u otros seres) con el fin de comunicarse. Son imprescindibles para poder concebir modelos, pues uno expresa a lo más lo que el lenguaje le permite. Además, los lenguajes son los que permiten comunicar los modelos a otros (que comprenden dichos lenguajes), validarlos, discutirlos y ampliar la percepción del otro sobre un mismo fenómeno.

Para efectos de este curso, consideraremos los siguientes componentes de los lenguajes.

1. La sintaxis. Es el conjunto de símbolos permitidos en el lenguaje. (por ejemplo las letras del abecedario o todas las palabras del idioma español)
2. Una gramática. Son las reglas generadoras del lenguaje. (por ejemplo la gramática del español)
3. La semántica. Es el significado asociado al lenguaje (por ejemplo, el significado de las palabras y su interpretación dentro de un contexto dado).

2 Estrategia general de resolución de problemas.

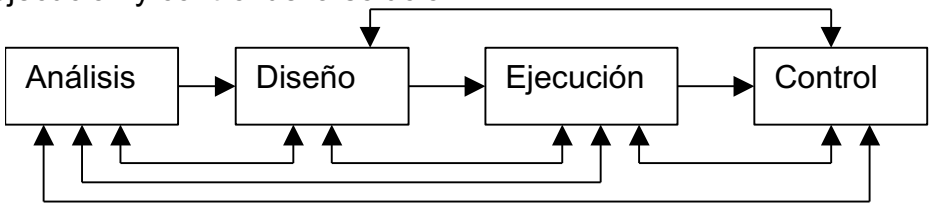
Para poder abordar un problema, primero hay que definirlo. Esto, que pudiese parecer trivial, no lo es para nada.

Definir un problema significa comprenderlo y modelarlo. Los problemas son realidades susceptibles de ser modeladas de diversas maneras, como se muestra en el siguiente ejemplo, donde distintos involucrados nos dan su visión acerca de los problemas de una organización.

INVOLUCRADOS	¿CUALES SON LOS PROBLEMAS DE ESTA ORGANIZACIÓN?
Empleado	¿Cómo aumento mis ingresos? ¿Cómo beneficio a mi organización? ¿Cómo aumento mi productividad y mi calidad de trabajo? ¿Cómo quedo mejor con mis jefes y consigo una mejora en mi situación actual?
Ejecutivo	¿Cómo aumento la productividad? ¿Cómo dejo feliz a mi gente sin ir contra las políticas de la empresa? ¿Cómo motivo a mi personal? ¿Cómo aumento la calidad de los productos? ¿Cómo puedo mejorar mis procesos y productos?
Directorio	¿Cómo aumentamos nuestras ganancias? ¿Está bien gestionada nuestra empresa? ¿Cómo aseguramos el crecimiento de la empresa?
Comercialización y Marketing	¿Cómo creo nuevos mercados? ¿Cómo dejo más satisfechos a los clientes? ¿Cómo me posiciono mejor en el mercado? ¿Dónde hay oportunidades?
Matemático	¿Cuál es la función matemática que optimiza esta organización? ¿Cuáles son las variables relevantes?
Secretaria	¿Cómo disminuyo mi carga de trabajo?
Contabilidad	¿Cómo simplifico el proceso contable? ¿Cuáles son los flujos de entrada y de salida? ¿Cómo hago la contabilidad en menor tiempo?
Recursos Humanos	¿Cómo aporto mayores beneficios al personal? ¿Cuáles son las principales técnicas para mantener al personal a gusto?
Administración Financiera	¿Cómo disminuyo costos?

Gerencia	¿Cómo aseguro la misión de la empresa? ¿Cómo aumento la productividad de esta organización? ¿Dónde hay oportunidades de mejora? ¿Cuál debe ser nuestra estrategia? ¿Cómo vamos?
Informáticos	¿Qué procesos son factibles de automatizar? ¿Dónde podemos introducir tecnología de punta? ¿Cómo convenzo a los usuarios de que no tienen la razón?
Contratistas	¿Cómo me hago indispensable? ¿Cómo me aseguro trabajo por un periodo mediano?
Burócratas	¿Cómo aumento el control?

Básicamente, el proceso de resolver un problema consta de cuatro etapas: análisis, diseño, ejecución y control de la solución.



2.1 Análisis del Problema.

El análisis del problema involucra capturar el máximo de información referente a este, obtener la visión del mismo por parte de los involucrados y generar un modelo para cada una de estas visiones. Se deben refinar estos modelos hasta obtener una o varias representaciones (modelos) del problema, los que posibilitan su análisis. Estos modelos que describen las visiones del problema son los que posibilitarán que se vislumbre alguna solución.

Se debe modelar tanto el problema como la situación esperada, de modo de poder diseñar la forma de llegar del estado actual al estado objetivo(problema solucionado).

Lo relevante de este punto es que el análisis es una tarea creativa que requiere de capacidades de abstracción y uso de lenguajes que faciliten la tarea de generación de modelos y análisis de los mismos. En el análisis del problema se deben contestar las siguientes preguntas:

¿Cuál es el problema? (¿Cuál es el estado actual considerado insatisfactorio?)

¿En qué forma se consideraría el problema solucionado? (¿Cuál es el estado deseado?)

¿Qué restricciones existen para llegar a esa solución?

2.2 Diseño de la Solución.

El diseño de la solución es el proceso por el cual se determina cuales son los pasos a seguir para conseguir pasar desde el estado actual (problema) al estado deseado u objetivo. Muchas veces este proceso de diseño involucra a variados componentes, no sólo de índole informática, los cuales no son el objetivo de este curso.
En algunas ocasiones, el modelo del estado objetivo es ya el diseño de la solución, como por ejemplo, en el caso de los sistemas software, el modelo de la situación objetivo se traslapa con la primera versión del diseño.

2.3 Ejecución

En la ejecución, se sigue el plan establecido en el diseño para pasar del estado inicial al estado objetivo (definido en el análisis).
La ejecución del diseño es en muchos casos la construcción en un lenguaje de programación u otra herramienta de implementación computacional de lo diseñado en la etapa previa.

2.4 Control

En esta etapa, se verifica y valida lo ejecutado.
Esto significa que se debe verificar que lo ejecutado corresponde a lo diseñado, y lo diseñado efectivamente permite alcanzar el estado objetivo definido en el análisis.
La validación consiste en responder si esta es LA solución.

3 Dimensiones de un sistema basado en software.

El software es otro modelo de la realidad, definido en lenguajes particulares que incluyen códigos de programa, documentación, manuales de usuario, procedimientos administrativos, bases de datos, entre otros. El software no se produce en el sentido clásico, más bien, se desarrolla.

> "...ESTA BASADO EN UN RECONOCIMIENTO DEL HECHO QUE,..., CUALQUIER PROGRAMA ES UN MODELO DE UN MODELO DENTRO DE UNA TEORÍA DE MODELOS PARA UNA ABSTRACCIÓN DE ALGUNA PORCIÓN DEL MUNDO O DE ALGÚN UNIVERSO DE DISCURSO." *MEIR LEHMANN, PROCEEDING OF THE IEEE VOL. 68 NO. 9 SEPT. 1980.*

Podemos distinguir en un software tres dimensiones, o formas de verlo:

Con respecto al tipo de componente.

Interfaz. La interfaz es aquella componente del software que permite que un usuario interactúe con él. A través de este componente un software recibe las entradas del usuario y le entrega las respuestas.

Lógica de procesamiento. Este componente es aquel que se preocupa de relacionar las entradas que le llegan a través de la interfaz con el repositorio, generando los resultados que el usuario espera.

Repositorio. Este componente alberga a todos aquellos componentes que el software utiliza y almacena incluso después de ejecutarse (el repositorio es persistente en el tiempo). En este componente reside información que es persistente y que puede ser compartida con otros componentes de lógica de procesamiento.

Con respecto al nivel de abstracción:

Conceptual. En este nivel de abstracción nos estamos enfrentando a una representación (modelo) muy cercana a la realidad a modelar, de manera independiente de la plataforma de implementación computacional. De alguna manera, es en este nivel donde deberían encontrarse los modelos que se realicen en la etapa de análisis.

Lógico. Este nivel de abstracción se centra en los aspectos centrales del sistema, pero con una visión más cerca de la implementación en una plataforma definida (puede ser “un tipo de” plataformas). Los modelos generados en la etapa de diseño deberían encontrarse mayoritariamente en este nivel de abstracción.

Físico. En este nivel la abstracción ya es mínima. Estamos ante representaciones directas de la implementación de los sistemas. Modelos de nivel físico deberían documentar la etapa de ejecución.

Con respecto al tipo de modelamiento:

Estático. El componente estático de un sistema software lo constituye la definición de todas sus estructuras, las cuales podrán tomar distintos valores.

Dinámico. El componente dinámico de un software lo constituye su comportamiento, es decir, la definición de respuestas (cambios de estado) a ciertos estímulos (eventos).

Funcional. El componente funcional es aquel que define las transformaciones de las entradas, objetos del repositorio, eventos, etc., en respuestas o salidas del sistema software. Esta componente es aquella que realiza las transformaciones.

Estas tres dimensiones se entrecruzan y conforman una matriz de 3x3x3, en cuyas celdas podemos encontrar distintos lenguajes que apoyan el modelamiento de dicha dimensión.

Componente	Interfaz			Lógica de Procesamiento			Repositorio		
	Nivel de Abstracción			Nivel de Abstracción			Nivel de Abstracción		
Tipo Modelamiento	Conceptual	Lógico	Físico	Conceptual	Lógico	Físico	Conceptual	Lógico	Físico
Estático									
Dinámico									
Funcional									

Para cada una de estas celdas pueden existir uno o más lenguajes adecuados para modelar realidades vistas bajo el prisma de las tres dimensiones consideradas.

4 Características de los Sistemas de Bases de Datos.

> "UNA BASE DE DATOS CONSISTE EN ALGUNA COLECCIÓN DE DATOS PERSISTENTES E INDEPENDIENTES USADOS POR UNA ORGANIZACIÓN DETERMINADA." (DATE, 1995)

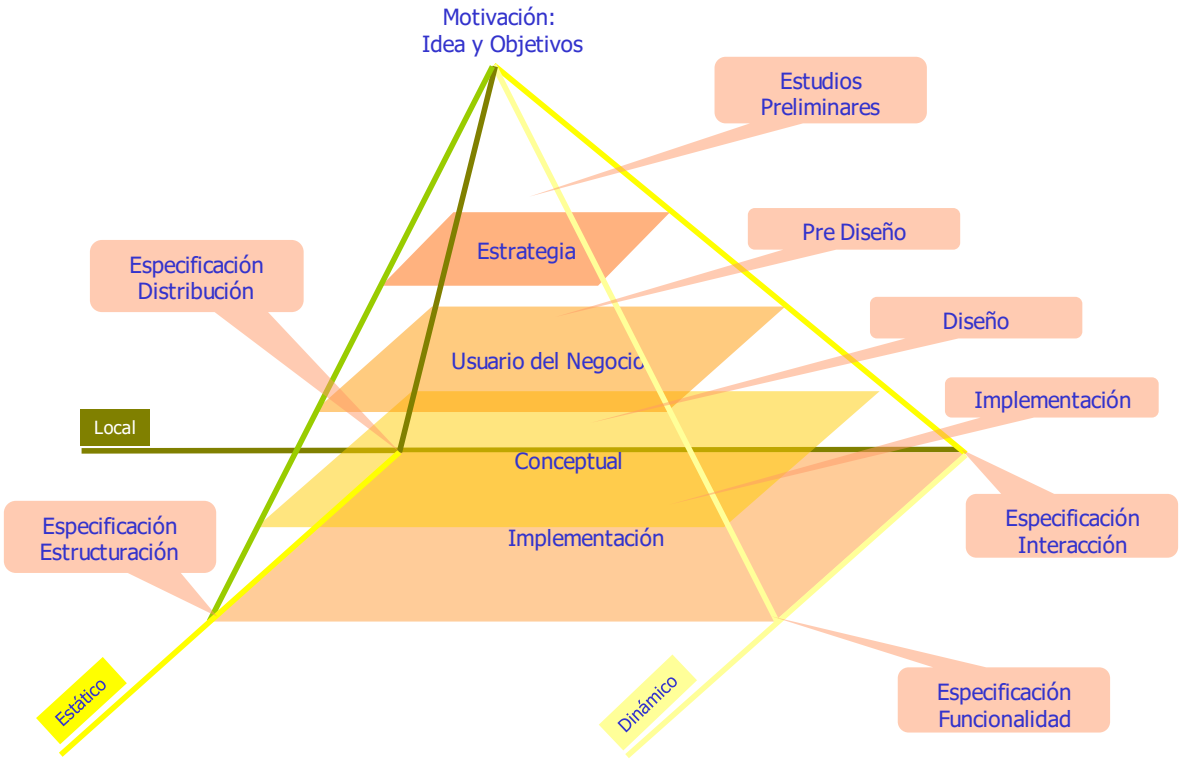
Los sistemas de bases de datos se caracterizan por poseen una fuerte componente de repositorio. Incluso podemos considerar repositorio y base de datos como sinónimos.

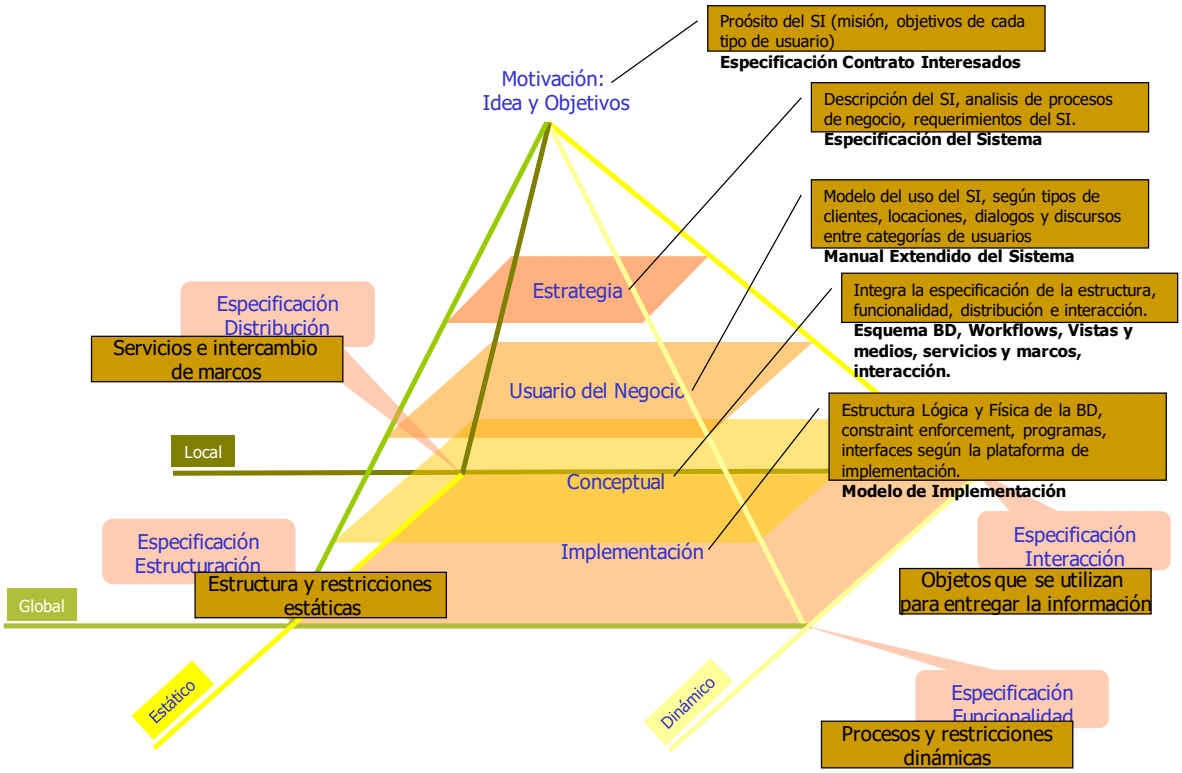
Los sistemas de bases de datos, al ser un repositorio, poseen componente estática, dinámica y funcional, y cada una de estas componentes puede ser vista desde los niveles de abstracción conceptual, lógico o físico.

Podemos precisar un poco más en la composición de este caso particular de sistema basado en software.

Dimensión: Tipo de Modelamiento

Tipo Componente	Componente Sistema Base de Datos	Descripción
Estático	Estructura	Estructuras que constituyen la base de datos.
Estático	Restricciones Estáticas	Reglas que restringen el conjunto de valores (estados) que la base de datos (estructura) puede tomar.
Dinámico	Restricciones Dinámicas	Reglas que restringen las transiciones entre valores (estados) válidos de la base de datos (estructura).
Funcional	Manipulación de los datos.	Definición de los procedimientos por los cuales la base de datos (estructura) cambia de un valor (estado) a otro.


Motivación:
Idea y Objetivos
Estudios
Preliminares
Pre Diseño
Diseño
Implementación
Especificación
Interacción
Especificación
Funcionalidad
Dinámico
Estático
Especificación
Estructuración
Especificación
Distribución
Estrategia
Usuario del Negocio
Conceptual
Implementación
Local


Proósito del SI (misión, objetivos de cada tipo de usuario)
Especificación Contrato Interesados
Motivación:
Idea y Objetivos
Descripción del SI, analisis de procesos de negocio, requerimientos del SI.
Especificación del Sistema
Modelo del uso del SI, según tipos de clientes, locaciones, dialogos y discursos entre categorías de usuarios
Manual Extendido del Sistema
Integra la especificación de la estructura, funcionalidad, distribución e interacción.
Esquema BD, Workflows, Vistas y medios, servicios y marcos, interacción.
Estrategia
Especificación
Distribución
Servicios e intercambio de marcos
Usuario del Negocio
Local
Conceptual
Estructura Lógica y Física de la BD, constraint enforcement, programas, interfaces según la plataforma de implementación.
Modelo de Implementación
Especificación
Estructuración
Implementación
Especificación
Interacción
Objetos que se utilizan para entregar la información
Global
Estructura y restricciones estáticas
Estático
Dinámico
Especificación
Funcionalidad
Procesos y restricciones dinámicas

5 Procesos de Abstracción en el modelamiento.

La abstracción es un proceso mental que se aplica al seleccionar algunas características y propiedades de un conjunto de objetos y excluir otras no pertinentes. En otras palabras, se hace una abstracción al fijar la atención en las propiedades consideradas esenciales de un conjunto de cosas y desechar sus diferencias.

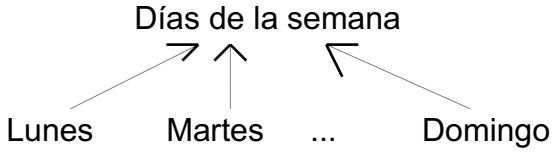
En el modelamiento de datos, se usan tres tipos de abstracciones: clasificación, agregación y generalización.

5.1 Abstracción de clasificación.

Se utiliza para definir un concepto como una clase de objetos de la realidad caracterizados por propiedades comunes.

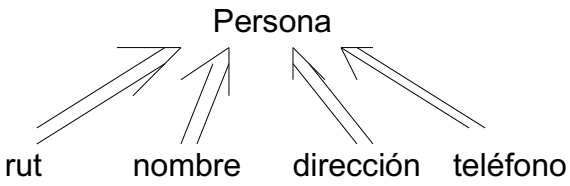
Se representa gráficamente como un árbol de un nivel que tiene como raíz la clase y como hojas los elementos de la clase. Las ramas del árbol se representan por líneas discontinuas. Cada rama del árbol indica que un nodo hoja es un miembro (ES_MIEMBRO_DE) la clase que representa la raíz.

Un mismo objeto real puede clasificarse de varias maneras.



5.2 Abstracción de Agregación.

Define una nueva clase a partir de un conjunto de (otras, no necesariamente distintas) clases que representan sus partes componentes. Se representa por un árbol de un nivel en el cual todos los nodos son clases; la raíz representa la clase creada por agregación de las clases representadas en las hojas. Cada rama del árbol indica que una clase hoja es una parte de (ES_PARTE_DE) la clase representada por la raíz. Para distinguirla de la agregación de clasificación, las ramas dirigidas están representadas por líneas dobles que van de los componentes a los objetos agregados.



La clasificación y la agregación son las dos abstracciones básicas utilizadas para construir estructuras de datos dentro de la base de datos y dentro de los lenguajes convencionales de programación. La clasificación es el procedimiento utilizado cuando, partiendo de elementos individuales de información, se identifican tipos de campos o atributos. La agregación es el procedimiento mediante el cual se reúnen tipos de campos relacionados en grupos como por ejemplo tipos de registros.

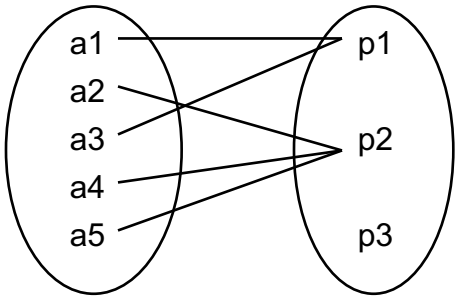
5.3 Abstracción de generalización.

Define una relación de subconjunto entre elementos de dos o mas clases. Cada generalización se representa con un árbol de un nivel, en el que todos los nodos son clases, con la clase genérica como raíz y las clases subconjunto como hojas; cada rama del árbol expresa que una clase hoja es un (ES_UN) subconjunto de la clase raíz. Para distinguir la generalización de otras abstracciones, se usa una flecha sencilla apuntando hacia la raíz. Esta abstracción, a pesar de ser muy común e intuitiva, no se usa en muchos modelos de datos. Sin embargo es muy útil por su cualidad fundamental de herencia: en una generalización, todas las abstracciones definidas para la clase genérica son heredadas por las clases subconjunto.

6 Propiedades de las correspondencias entre las clases.

6.1 Agregación Binaria.

Es una correspondencia que se establece entre dos clases. Se puede representar una agregación binaria entre dos clases mediante la descripción de éstas como conjuntos y el trazado de una línea entre un elemento de cada conjunto para representar que están agregados.



Representación para la agregación "POSEE".

Al observar la figura, se puede decir que la persona p1 posee los autos a1 y a2, la persona p2 posee los autos a2,a4 y a5, mientras que la persona p3 no posee autos. De esto último se puede observar que no es obligatorio que todas las personas posean autos, pero al parecer todos los autos deben tener un dueño. Esta última característica es propia de cada agregación, y se refieren a la cardinalidad de correspondencia entre las clases.

6.1.1 Cardinalidad mínima.

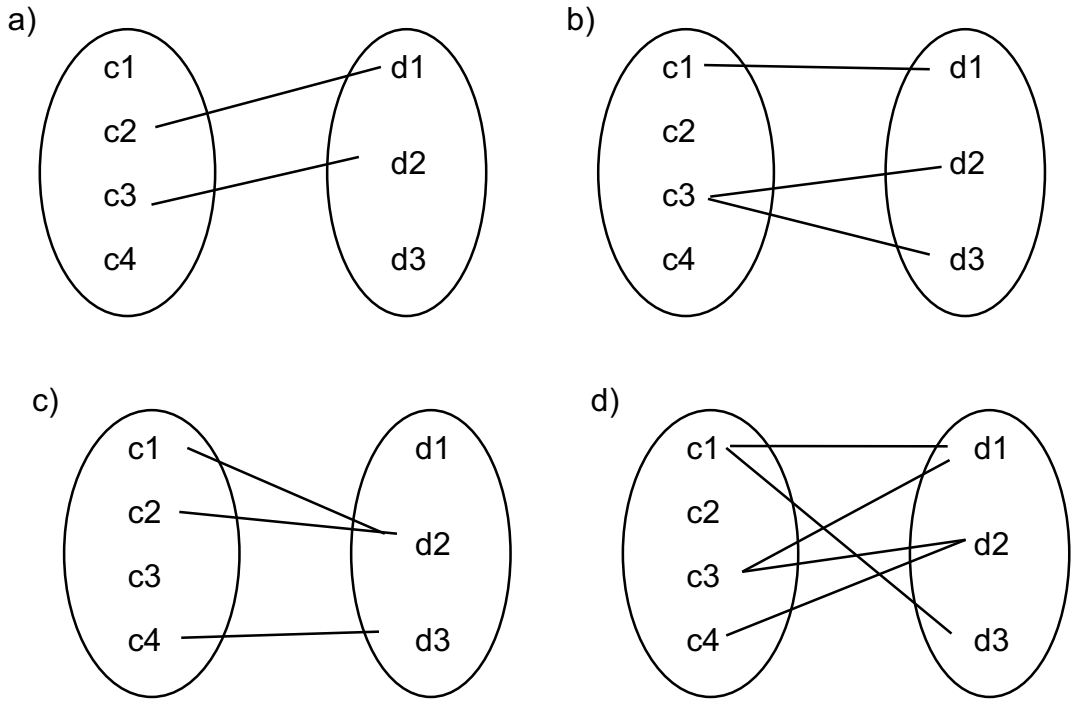
Consideremos una agregación A entre las clases C y D. La cardinalidad mínima de C en A, denotada por card-min(C,A), es el menor número de correspondencias en las que cada elemento de C puede tomar parte. Análogamente se define card-min(D,A).

Si card-min(A,B)=0, entonces se dice que la clase A tiene una participación opcional en la agregación B. Si card-min(A,B)>0, entonces se dice que la clase A tiene una participación obligatoria en la agregación B.

6.1.2 Cardinalidad máxima.

Consideremos una agregación A entre las clases C y D. La cardinalidad máxima de C en A, denotada por card-máx(C,A), es el mayor número de correspondencias en las que cada elemento de C puede tomar parte. Análogamente se define card-máx(D,A).

Si card-max(C,A)=1 y card-max(D,A)=1, se dice que la agregación es de uno a uno. Si card-max(C,A)=n y card-max(D,A)=1, se dice que la agregación es de uno a muchos. Si card-max(C,A)=1 y card-max(D,A)=n, se dice que la agregación es de muchos a uno. Si card-max(C,A)=n y card-max(D,A)=m, se dice que la agregación es de muchos a muchos. Nótese que n y m representan valores mayores que 1.



a) Agregación Uno a Uno. Card(C,A)=(x,1) y Card(D,A)=(x,1), con x en {0,1}.
b) Agregación Uno a Muchos. Card(C,A)=(x,n) y Card(D,A)=(y,1), con x en {0,1,...,n} e y en {0,1}.
c) Agregación Muchos a Uno. Card(C,A)=(x,1) y Card(D,A)=(y,n), con x en {0,1} e y en {0,1,...,n}.
d) Agregación Muchos a Muchos. Card(C,A)=(x,n) y Card(D,A)=(y,m), con x e y en {0,1,...,n), n y m valores indefinidos mayores que uno.

6.2 Agregación n-aria.

Es una correspondencia establecida entre tres o más clases. Se mantiene las definiciones de cardinalidades máxima y mínima.

Cardinalidad Mínima. Consideremos una agregación A entre las clases C1,C2,..., Cn. La cardinalidad mínima de Ci en A, denotada por card-min(Ci,A), es el menor número de correspondencias en las que cada elemento de Ci puede tomar parte.

Cardinalidad Máxima. Consideremos una agregación A entre las clases C1,C2,..., Cn. La cardinalidad máxima de Ci en A, denotada por card-max(Ci,A), es el mayor número de correspondencias en las que cada elemento de Ci puede tomar parte.

6.3 Generalizaciones.

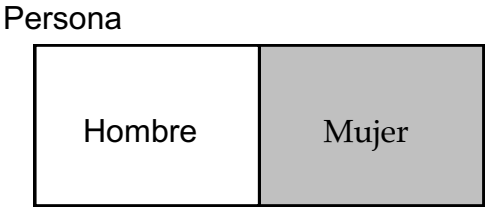
Una abstracción de generalización establece una correspondencia entre la clase genérica (raíz) y las clases subconjunto. Considérese la clase Persona como una generalización de las clases Mujer y Hombre; casa elemento de éstas corresponde exactamente a un elemento de la clase Persona. En esta generalización, cada elemento de la clase Persona corresponde a un elemento de la clase Mujer o la clase Hombre, pero nunca a ambas. Esto es una característica de cada generalización y se denomina cobertura.

6.3.1 Cobertura total o parcial.

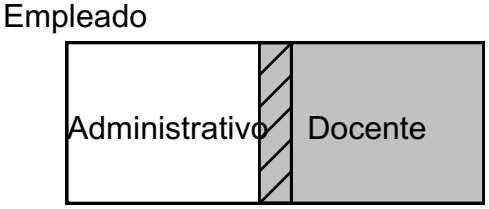
La cobertura de una generalización es total (t) si cada elemento de la clase genérica corresponde al menos a un elemento de las clases subconjunto; es parcial (p) si existe algún elemento de la clase genérica que no corresponde a ningún elemento de las clases subconjunto.

6.3.2 Cobertura exclusiva o superpuesta.

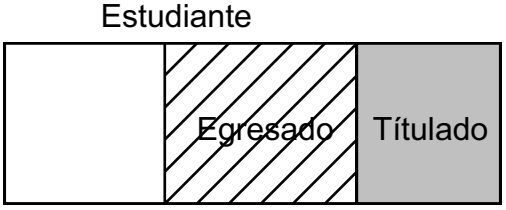
La cobertura de una generalización es exclusiva (e) si cada elemento de la clase genérica corresponde, a lo más a un elemento de las clases subconjunto; es superpuesta (s) si existe algún elemento de la clase genérica que corresponde a elementos de dos o más clases subconjunto diferentes.



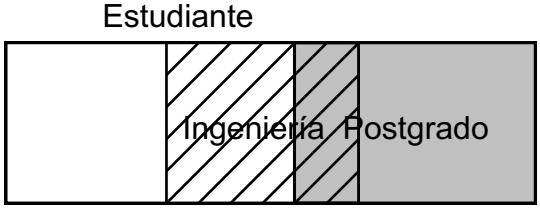
a) total, exclusiva. Todas las personas son Hombres o Mujeres, pero no ambos.



b) total, superpuesta. Todos los empleados son Administrativos o Docentes, pudiendo haber empleados desempeñando ambas funciones.



c) parcial, exclusivo. Algunos estudiantes son egresados, mientras que otros están titulados, pero no hay ningún estudiante en ambas situaciones.



d) parcial, superpuesta. Algunos estudiantes son de Ingeniería y otros son de postgrado, y hay algunos estudiantes que son de ingeniería y también participan en postgrado.

7 Concepto de Dato y Modelo de Dato.

7.1 El significado de dato.

La percepción del mundo puede ser descrita como una sucesión de fenómenos. Desde el comienzo de los tiempos el hombre ha tratado de descubrirlos, ya sea que los entienda completamente o no.

La descripción de estos fenómenos es llamada DATO. Los datos corresponden al registro discreto (no continuo) de hechos acerca de un fenómeno, con lo cuál ganamos información acerca del mundo que nos rodea (Información: incremento del conocimiento que puede ser inferido de los datos).

Usualmente el dato y su significado son registrados juntos, ya que el lenguaje natural es lo suficientemente poderoso para hacerlo. Por ejemplo "el kilo de pan cuesta $460" registra el valor (460) y su significado o semántica (valor del kilo de pan en pesos).

En ciertos casos los datos están separados de su semántica. Por ejemplo, una planilla de notas es una tabla de datos. Su interpretación está implícita y se supone que quien la lee conoce su significado.

El uso del computador para procesar datos ha traído consigo una mayor separación entre los datos y su interpretación. Mucha de la interpretación de los datos está explícita. Consideremos por ejemplo un programa que calcula integrales definidas, este programa recibe valores de entrada y genera valores como salida. Sin embargo, el programa en si no tiene conocimiento si el problema resuelto es de termodinámica o electromagnetismo.

Han habido dos razones para separar los datos de su significado:

- los computadores no manejan (bien) el lenguaje natural, que es la mejor forma de dar interpretación y significado a un dato.
- el almacenamiento del significado de los datos ocupa espacio, e inicialmente este era escaso y costoso.

Así, tradicionalmente la interpretación de los datos se deja al usuario y al sistema manual externo al computador.

En muchos sistemas la interpretación de datos se encuentra en los programas que hacen uso de ellos, de modo que los datos pasan a ser una simple colección de valores.

Por otra parte, supongamos que algo de la semántica de los datos se codifica junto con ellos . Así los datos no son solo valores, sino que también tienen una semántica, y los datos están más cerca de la interpretación del mundo. Ellos forman una "vista" del mundo, la que no es exacta ni concreta, sino que usualmente es bastante abstracta.

Los datos no son estáticos, y corresponden a un mundo que está en constante cambio. La flexibilidad en la interpretación de los datos permite capturar los aspectos dinámicos del mundo y al mismo tiempo, proveer una estructura estable para los datos. Esta flexibilidad se puede tener de dos formas:

- El sistema puede permitir que los mismos datos sean vistos de diferente forma. Por ejemplo, diferentes aplicaciones puedan usar los mismos datos y dar su propia semántica.

- Diferentes datos pueden ser vistos de la misma forma. Por ejemplo, se quiere ver a los gerentes, secretarias y empleados sólo como trabajadores de una organización, no importando su cargo. Aquí la interpretación debe ser lo suficientemente abstracta para que diferentes vistas del mundo se vean de la misma forma.

7.2 Modelamiento de Datos.

Es aparente que una interpretación del mundo es necesaria, la que debe ser suficientemente abstracta para que no sea afectada por la dinámica del mundo (los pequeños cambios), y debe ser suficientemente robusta para poder representar como los datos y el mundo se relacionan. Una herramienta como esta es llamada *modelo de datos*, el cual permite representar en forma más o menos razonable alguna realidad. El modelo de datos permite realizar abstracciones del mundo, permitiendo centrarse en los aspectos macros, sin preocuparse de las particularidades; así nuestra preocupación se centra en generar un esquema de representación, y no en los valores de los datos.

Los modelos de datos nos permiten capturar parcialmente el mundo, ya que es improbable generar un modelo que lo capture totalmente.

Sin embargo, se puede tener un conocimiento relativamente completo de la parte del mundo que nos interesa. Así, un modelo captura la cantidad de conocimiento tal que cumpla con los requerimientos que nos hemos impuesto previamente.

DATO: la siguiente tupla: <nombre del objeto, propiedad del objeto, valor, tiempo>

Esta definición es correcta, ya que cada vez que se describe un fenómeno, éste se refiere a un objeto (nombre del objeto) y ciertas características (propiedades del objeto) el cual tiene un valor en un momento determinado (tiempo).

⍁Ejemplo. El *precio* del *pan* es $*980* .

nombre: precio del pan
propiedades : (unidad, $), entero no negativo.
valor: 980
tiempo: hoy

En general, el modelar un objeto no se considera el tiempo, sino que éste se considera implícito en la semántica de él.

⍁Ejemplo. Consideremos el caso de una matriz:

nombre: matriz_coeficiente

propiedades: +, -, *, a[i,j] $\in$ R

valor : [1 2]
[3 4]

ESQUEMA
Para una aplicación particular de un modelo de datos, el modelamiento de la realidad se llama esquema.

Un esquema es una definición genérica que identifica categorías (ejemplo: libro, autor, etc.), sus propiedades (nombre, título) y sus relaciones (escrito).

Por ejemplo, un modelo de datos simple es una archivo (tabla). Aplicando este modelo a una situación particular se puede tener el siguiente esquema:

Persona (Nombre, Edad, Dirección), donde Persona es el nombre genérico de una entidad, y Nombre, Edad y Dirección son nombres genéricos para los atributos.

7.2.1 Modelo de Datos.

Un modelo de datos define las reglas por las cuales los datos son estructurados. Esta estructuración, sin embargo, no da una interpretación completa acerca del significado de los datos y la forma en que serán usados. Las operaciones que se permiten efectuar a los datos deben ser definidos.

⍁Ejemplo. Una lista puede ser tratada como pila o fila, dependiendo de las operaciones que se permitan sobre ella.

Generalmente las operaciones están relacionadas con la estructura de los datos y tienen validez en el contexto en que fueron definidos.

BASE DE DATOS
Una colección de datos estructurados en una forma particular, según un esquema particular, se denomina base de datos.

Todo modelo de datos debe poder capturar las propiedades estáticas y dinámicas de una realidad. Las propiedades estáticas corresponden a lo que es relativamente invariante en el tiempo, son siempre verdadero y no cambia en el tiempo.
⍁Ejemplo. Que los precios se midan en $ es relativamente invariante.

Las propiedades dinámicas corresponden a la naturaleza evolutiva del mundo. Por esto, para todo modelo debe ser posible capturar los dos tipos de propiedades.

MODELO DE DATOS
Se define el modelo de datos M consistente de dos partes:
G: un conjunto de reglas que lo generan.
O: un conjunto de operaciones.

El conjunto de reglas G expresan las propiedades estáticas de un modelo de datos y corresponden a lo que se denomina generalmente Data Definition Language (DDL). Este define las estructuras permitidas para el modelo de datos M.

El conjunto G se puede dividir en dos:
- Gs: las estructuras permitidas.
- Gc: las restricciones del modelo.

Así, Gs genera las categorías y estructuras para un modelo, y Gc las restricciones.

Utilizando esta última notación, un esquema S consiste de dos partes: una estructura Ss y restricciones Sc, donde Sc es una lista explícita de restricciones que no deben ser violadas.

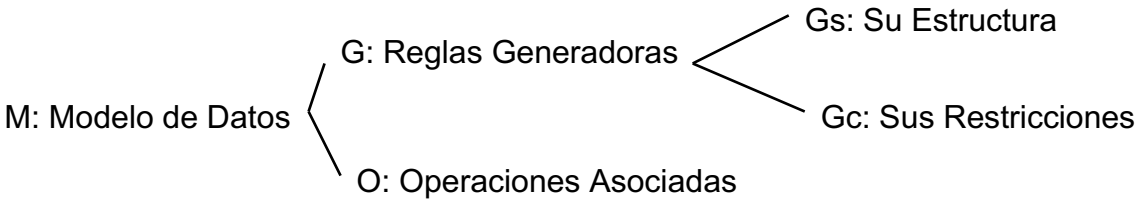
Por ejemplo, en la definición de la entidad persona, un atributo particular CI (Cédula de Identidad) puede ser declarado como clave, esto es, en un instante dado no puede haber dos personas con el mismo valor para CI. Ss no prohibe dos ocurrencias, pero Sc sí.

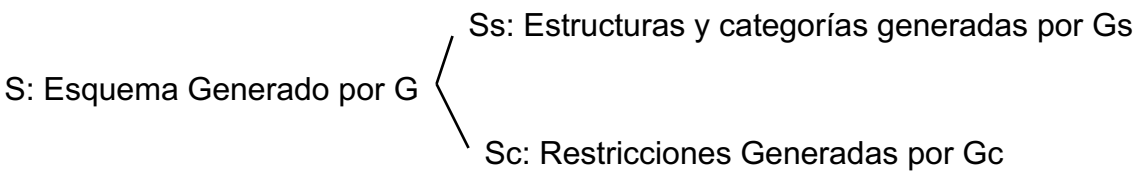
Un modelo de datos también puede tener restricciones que son inherentes a él, las que generalmente se incorporan en Ss (la estructura).

⇗Ejemplo, sólo se permite relaciones entre objetos mediante una estructura de árbol.

Las reglas de generación G son generadoras de un conjunto de esquemas S, en el que cada uno de ellos define estructuras y restricciones particulares para los datos. Hoy muchas bases de datos D en términos de la ocurrencia del esquema S, pero todos tienen la misma estructura genérica y obedecen a las mismas restricciones definidas en S.

En resumen:





D: Base de Datos —— Ocurrencia del esquema S. Las restricciones Sc deben ser siempre válidas, para toda ocurrencia D del esquema S.

7.2.2 Operaciones.

Las propiedades dinámicas de un modelo de datos son expresadas por un conjunto de operaciones O, las que generalmente son llamadas Data Manipulation Language (DML). Estas propiedades definen las acciones permitidas para una base de datos, tal que transforme la ocurrencia Di en la ocurrencia Dj.

No todas las operaciones definidas en O causan cambios en la base de datos, pero si causan un cambio en el estado de ella. El estado de una base de datos no es un objeto de ella, pero está asociado a la base de datos, y cambia como resultado de una operación.

⍁Ejemplo. Consideremos el modelo de datos tipo Tabla; este modelo tiene como una de sus operaciones la operación "read", que permite recorrer la tabla en forma secuencial. Esta operación no cambia el contenido de la Base, pero si su estado (cambia el registro actual, que ahora es el siguiente).

Parte II. Lenguajes.

8 Modelos Conceptuales, Lógicos y Físicos.

Hay tres tipos de diseños en el modelamiento, los cuales tienen directa relación con los modelos que ocupan: modelos conceptuales, lógicos y físicos.

En la Figura se puede apreciar el proceso de diseño de bases de datos. Los requisitos de datos constituyen un componente de los requisitos de un producto y son una entrada al diseño conceptual.

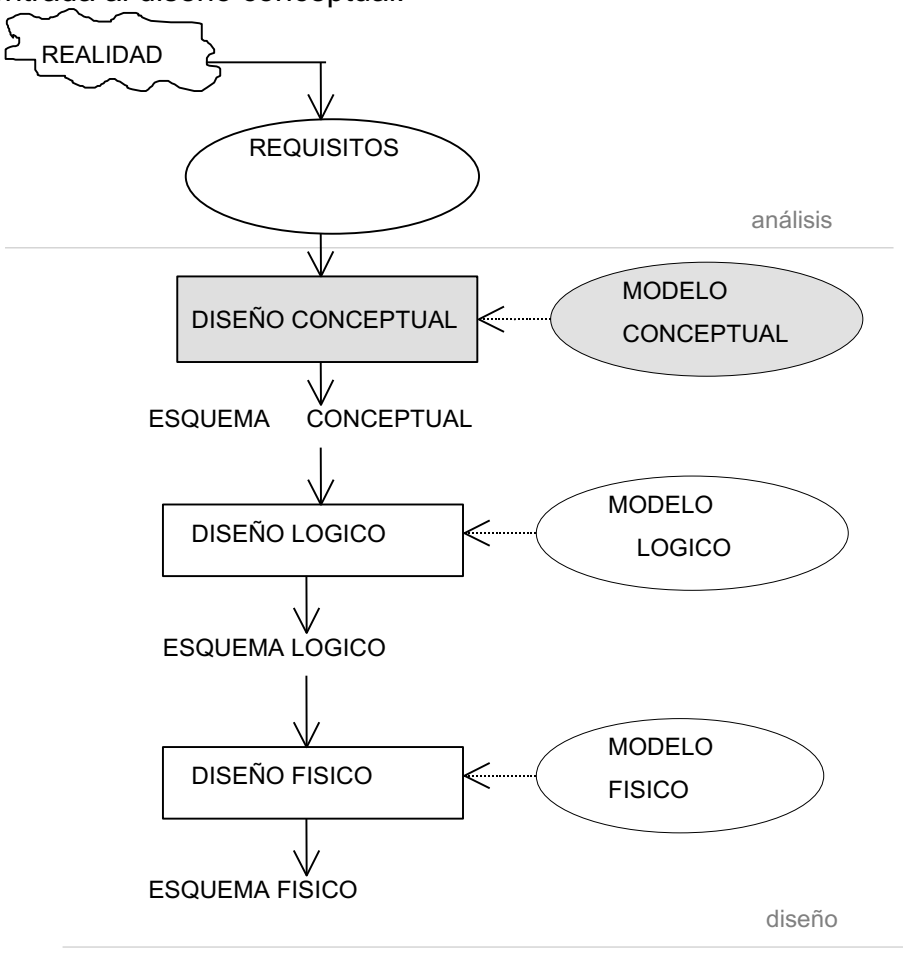


Figura 2-1

Diseño Conceptual.

Recibe como entrada la especificación de requerimientos y su resultado es el esquema conceptual de la base de datos, que es una descripción de alto nivel de la estructura de la base de datos, independiente del software que se use para manipularla.

Modelos Conceptuales: MER, CCER, HERM, Modelos OO, Formalismo Individual, Redes Semánticas, Redes de Transición de Estados.

Diseño Lógico.

Recibe como entrada el esquema conceptual y da como resultado un esquema lógico, que es una descripción de la estructura de la base de datos que puede procesar el software DBMS.

Modelos Lógicos: Relacional, de Redes, Jerárquico, Redes Semánticas, Redes de Transición de Estados, Modelos OO.

Diseño Físico.

Recibe como entrada el esquema lógico y da como resultado un esquema físico, que es una descripción de la implementación de una base de datos en la memoria secundaria, describe las estructuras de almacenamiento y los métodos usados para tener un acceso efectivo a los datos.

Modelos Físicos: Modelo Unificador, Memoria de Elementos.

9 Modelo Entidad Relación (MER, Entity Relationship Model)

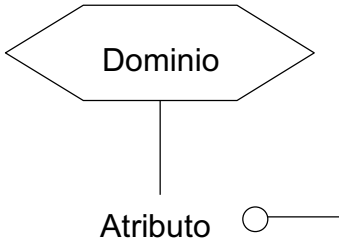
En 1976, Peter Chen publicó el modelo entidad relación, el cual tuvo gran aceptación principalmente por su expresividad gráfica. Sobre esta primera versión han trabajado numerosos autores, generando distintas extensiones de mayor o menor utilidad y de aceptación variable en el medio académico y profesional. Muchas de estas extensiones son muy utiles, pero poco difundidas debido principalmente a la ausencia de herramientas automatizadas que apoyen su uso.
A continuación se definen los elementos del modelo entidad relación básico.

9.1 Dominio.

Conjunto de valores de un mismo tipo. Se define como un conjutno, ya sea por extensión o comprensión.

9.2 Atributo.

Elemento de un Dominio. Aporta mediante su rótulo, la semántica de los valores del Dominio al que está asociado.



9.3 Entidad.

Consideremos un número de conjuntos cada uno orientado a un tipo particular de objetos. Estos conjuntos están relacionados con dominios y atributos.

Si consideramos la relación dada por el producto cartesiano de estos dominios, una interpretación que se le da a cada una de estas tuplas es que cada una corresponde a una *entidad* particular.

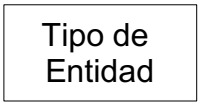
⇗Ejemplo.

(Juan , 70, 80, 50)

(Pedro , 90, 50, 70)

9.4 Tipo de Entidad.

Los Tipos de Entidad representan clases de objetos de la realidad. Además se componen de atributos, los cuales representan las características de un tipo de entidad.



En términos de abstracción, un "tipo de entidad" corresponde a la agregación de atributos.

⇗Ejemplo. El tipo de entidad Persona, corresponde a la agregación de los atributos (Rut, Nombre, Dirección, Fecha Nacimiento)

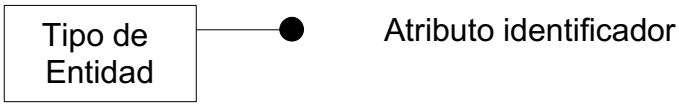
Así las entidades son una ocurrencia de un Tipo de Entidad.

9.5 Identificador de un tipo de entidad.

Un atributo I, posiblemente compuesto, de un tipo de entidad TE, es un Identificador de TE si y sólo si satisface las siguientes 2 propiedades independientes del tiempo.

a. Unicidad. En cualquier momento dado, no existen dos elementos en TE con el mismo valor de I.

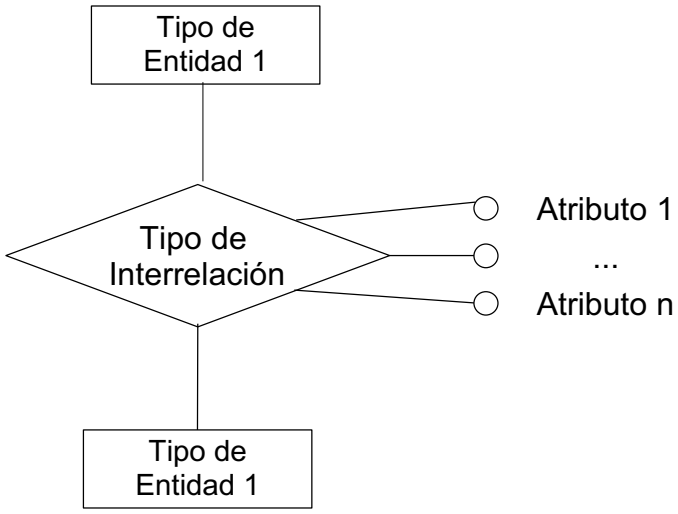
b. Minimalidad. Si I es compuesto, no será posible eliminar ningún atributo componente de I sin destruir la propiedad de unicidad.



9.6 Tipo de Interrelación.

Los Tipos de interrelación representan agregaciones de dos o más entidades (interrelaciones binarias o n-arias) no necesariamente diferentes.

El Identificador de un Tipo de Interrelación, se forma a partir de los identificadores de los tipos de entidad que relaciona.



⇗Ejemplo. la relación dueño-de puede ser interpretada como un tipo de interrelación entre dos tipos de entidades Persona y Auto.

9.7 Interrrelación.

Es la ocurrencia de un tipo de interrelación.

⇗Ejemplo. (Juan, LK-2346), considerando el tipo de interrelación dueño-de.

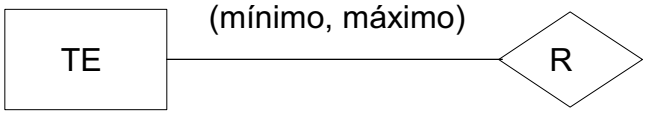
9.8 Cardinalidad de tipo de entidad con respecto a un tipo de interrelación.

Para los tipos de interrelación la cardinalidad máxima (mínima) establece el mayor (menor) número de correspondencias en cada una de los tipos de entidad involucradas en la interrelación.

Se define la Cardinalidad del Tipo de Entidad TE con respecto al tipo de interrelación R como:

$$Card(TE,R) = (mínimo, máximo), \text{ con } mínimo, máximo \in \{0,...,n\} \text{ y}$$
$$mínimo \leq máximo.$$

donde toda ocurrencia de TE debe participar al menos *mínimo* veces, y a lo más *máximo* veces en R.



✍Ejemplo

- Tipos de entidades (y atributos)

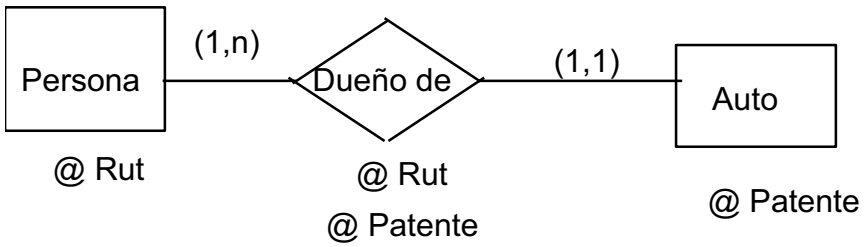
Auto : Patente (clave)
Año Fabricación
Color

Persona: Rut (clave)
Nombre
Dirección

- Tipos de Interrelaciones

Dueño_de: Rut (clave)
Patente (clave)
Fecha (clave)

- Diagrama MER



(1,n) : Una persona puede tener uno o más autos
(1,1): Un auto puede tener sólo un dueño .

✍Ejercicio.

Modelar un sistema de biblioteca que permita saber :

- autor de un libro
- libros de un autor
- préstamos de un alumno.
- materia de un libro
- editorial de un libro

¸ Desarrollo.

- Tipos de entidad

Autor

@	Código Autor	(string)
	Nombre	(string)
	Fecha Nacimiento	(fecha)
	Nacionalidad	(string)

Libro

@	Código Libro	(string)
	Título	(string)
	Año Publicación	(año)

Alumno

@	Matrícula	(numérico)
	Nombre	(string)

Ejemplar

@	Código Ejemplar	(numérico)

Materia

@	Código Materia	(numérico)
	Materia	

Carrera

@	Código Carrera	(numérico)

Editorial

@	Código Editorial	(numérico)
	Editorial	

- Tipos de Interrelaciones

Autor_de :

@ Código Libro

@ Código Autor

Estudia :
@ Matrícula
@ Código Carrera

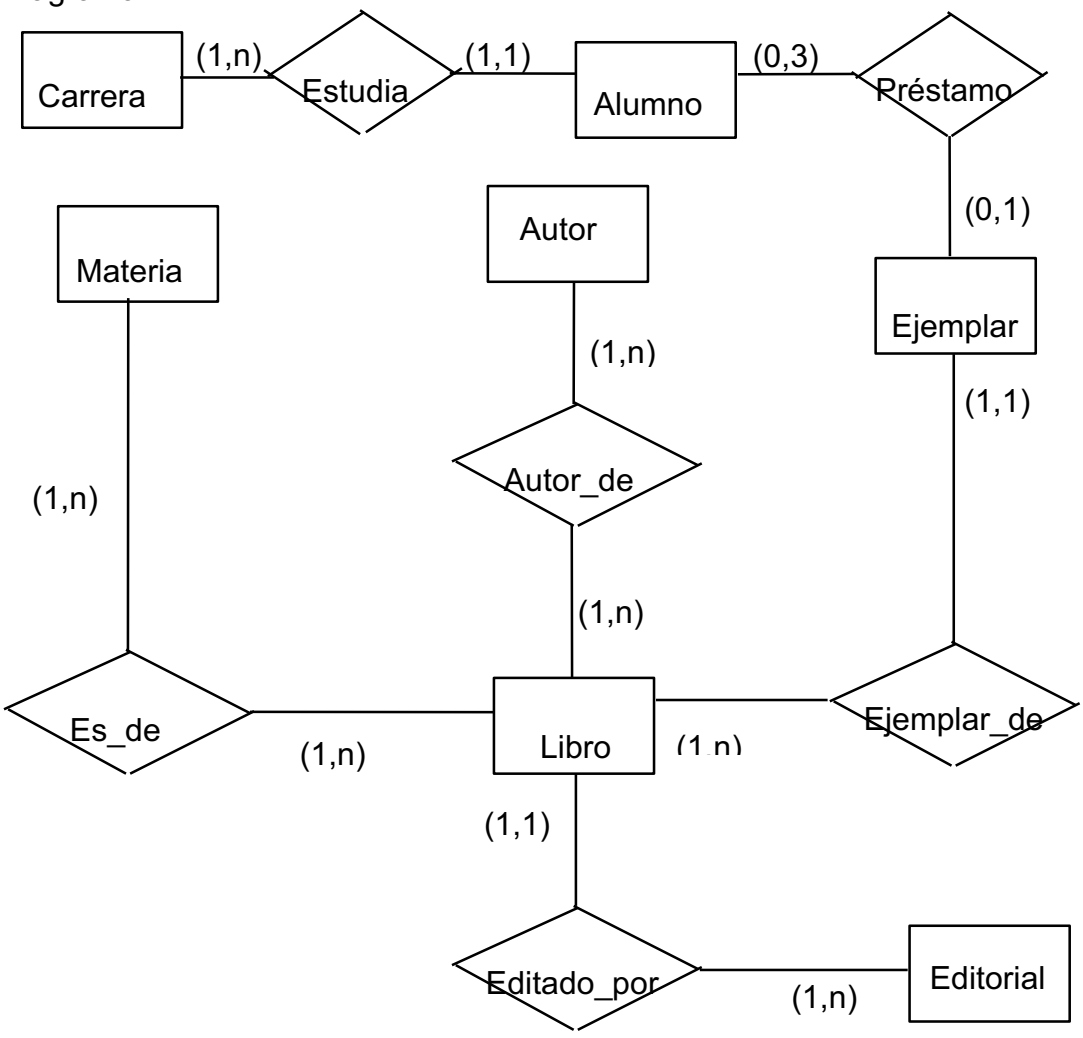
Prestamo :
@ Código Ejemplar
@ Matrícula
Fecha_préstamo (fecha)
Fecha_devolución (fecha)

Editado_por :
@ Código Editorial
@ Código Libro

Ejemplar_de :
@ Código Ejemplar
@ Código Libro

Es_de :
@ Código libro
@ Código Materia

- Diagrama MER



Lectura de las cardinalidades:

- Un Alumno estudia una y sólo una Carrera.
- Una Carrera es estudiada por uno o muchos Alumnos.
- Un Alumno puede tener en préstamo ninguno o a lo más tres Ejemplares.
- Un Ejemplar puede no estar en préstamo o estar en Préstamo a lo más una vez.
- Un Ejemplar corresponde a uno y sólo un Libro.
- Un Libro tiene uno o muchos Ejemplares.
- Un Autor es autor de uno o muchos Libros.
- Un Libro fue escrito por uno o muchos Autores.
- Un Libro es acerca de una o muchas Materias.
- Una Materia es abordada por uno o muchos Libros.
- Una Libro es editado por una y sólo una Editorial.

- Una Editorial ha editado uno o muchos Libros.

9.9 Cómo Modelar en MER

Para modelar en MER se sigue generalmente el siguiente orden:

a. Identificar los tipos de entidades.
b. Identificar los tipos de interrelaciones.
c. Encontrar las cardinalidades.
d. Identificar los atributos de cada tipo de entidad.
e. Identificar las claves de cada tipo de entidad.

La regla básica es distinguir tipos de entidades e interrelaciones de atributos. Así, los atributos deben ser atómicos y característicos del tipo de entidad o interrelación que describan.

También los atributos deben pertenecer al tipo de entidad o interrelación que describen y no a otro tipo.

Otra diferencia entre tipo de entidad y atributo es que, por ejemplo, se puede tener el tipo de entidad Empleado, que tiene como atributo el departamento al que pertenece. En forma alternativa se pueden tener los tipos de entidades Empleado y Departamento, y el tipo de interrelación Trabaja_en, que relaciona un empleado con el departamento donde trabaja.

Esta segunda alternativa es mejor desde el punto de vista del modelamiento conceptual y presenta una clara diferencia entre atributo y tipos de entidad.

9.9.1 Reglas para elegir identificadores.

i. No deben existir dos entidades con el mismo valor del identificador (en los tipos de entidad).
ii. En los tipos de interrelación, la clave es la composición de las claves de los tipos de entidad involucrados, en caso que no se pueda utilizar la clave de un subconjunto de ellos.

9.9.2 Ejercicios Propuestos.

1. Construir un esquema MER para una secretaría de universidad. La secretaría matiene datos sobre cada asignatura, incluyendo el profesor, lista de alumnos y la hora y el lugar de las clases. Para cada par estudiante-asignatura se registra su nota.

2. Construir un esquema MER para una compañía de seguros de autos con un conjunto de clientes, cada uno de los cuales es propietario de un número de autos. Cada auto tiene asociado el número de accidentes registrados.

3. Construir un esquema MER para un hospital con un conjunto de pacientes y un conjunto de médicos. A cada paciente se le asocia un registro de los análisis realizados.

4. Construir un esquema MER para modelar la documentación requerida para un esquema conceptual E-R.

5. Diseñar un esquema MER que recoja la organización de un sistema de información en el que se quiere tener información sobre municipios, viviendas y personas. Cada persona sólo puede habitar en una vivienda, pero puede ser propietaria de más de una. Nos interesa también la interrelación de las personas con su cabeza de familia. (Haga los supuestos que estime convenientes para justificar sus decisiones de diseño).

6. Diseñar un esquema MER que recoja la organización de las carreteras de todo el país. Se sabe que las carreteras se encuentran divididas en tramos, un tramo siempre pertenece a una única carretera y no puede cambiar de carretera, existen una serie de áreas en las que se agrupan los tramos, cada uno de los cuales no puede pertenecer a más de un área y un tramo puede pasar por varios términos municipales, siendo un dato de interés el km del tramo por el que entra en dicho término municipal y el km por el que sale.

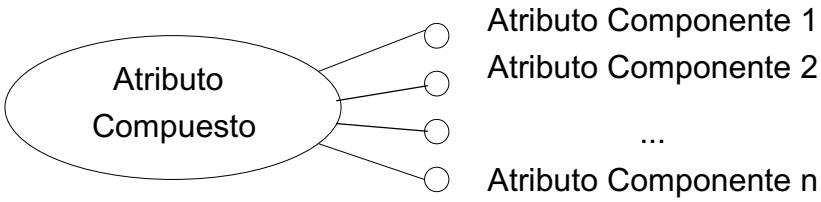
10 Modelo Entidad Relación Extendido

El modelo entidad relación ha sido mejorado por varios autores, incorporándole elementos que aumentan su expresividad y apoyan completitud de la especificación de la base de datos o realidad modelada.

A continuación se presentan las extensiones más usadas, que enriquesen lo expuesto en el capítulo anterior.

10.1 Atributo Compuesto.

Corresponde a grupos de atributos que tienen afinidad en cuanto a su significado o a su uso.



10.1 Cardinalidad.

Caracteriza a los atributos de un tipo de entidad y a los tipos de interrelación.

(Las definición aquí utilizada corresponde a la realizada por Tardieu).

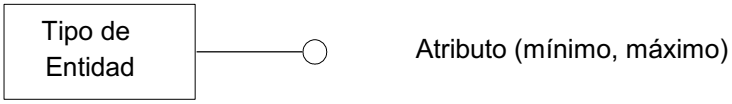
10.1.1 Cardinalidad de atributo con respecto a un tipo de entidad.

Para los atributos, la cardinalidad mínima indica el número mínimo de valores de un atributo asociado con cada caso (ocurrencia) de una entidad o interrelación. La cardinalidad máxima indica el número máximo de valores para un atributo asociado a cada caso de una entidad o interrelación.

Se define la Cardinalidad del Atributo A con respecto al tipo de entidad TE como:

$$Card(A,TE)=(\text{mínimo}, \text{máximo}), \text{ con } \text{mínimo}, \text{máximo} \in \{0,...,n\} \text{ y}$$
$$\text{mínimo} \leq \text{máximo}.$$

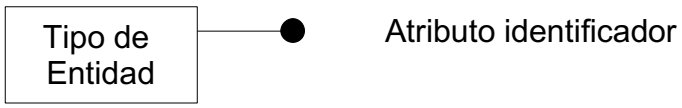
donde un elemento de A debe participar al menos *mínimo* veces, y a lo más *máximo* veces en cada ocurrencia de TE.



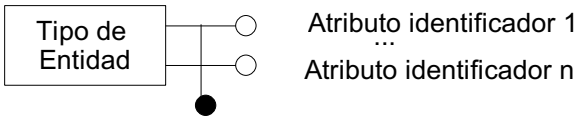
10.2 Identificador de un tipo de entidad.

Sea TE un tipo de entidad, sean A_1, A_2..., A_n atributos monovalentes obligatorios de TE, sean TE_1, TE_2..., TE_m otros tipos de entidad vinculados a TE por R_1, R_2..., R_m, tipos de interrelación (binarias) obligatorias. Considérese un posible identificador $I = \{ a_1, a_2..., a_n, TE_1, TE_2..., TE_m\}$, $n >= 0$, $m >= 0$, $n + m >= 1$. El valor del identificador para un caso particular te del tipo de entidad TE se define como el conjunto de todos los valores de los atributos a_i (i = 1,2, ..., n) y todos los casos de los tipos de entidad TE_j (j = 1,2, ..., m) vinculadas con te.

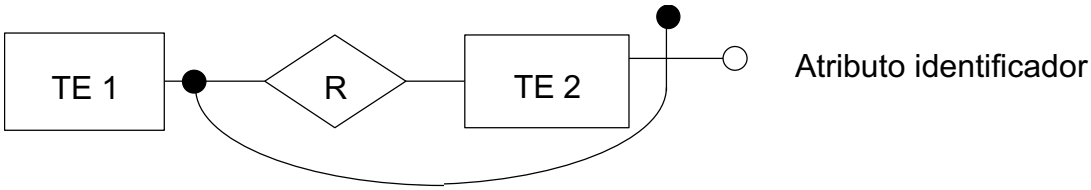
Cada entidad puede tener múltiples identificadores alternativos.



Identificador simple e interno



Identificador Compuesto e Interno



Identificador compuesto, mixto y externo

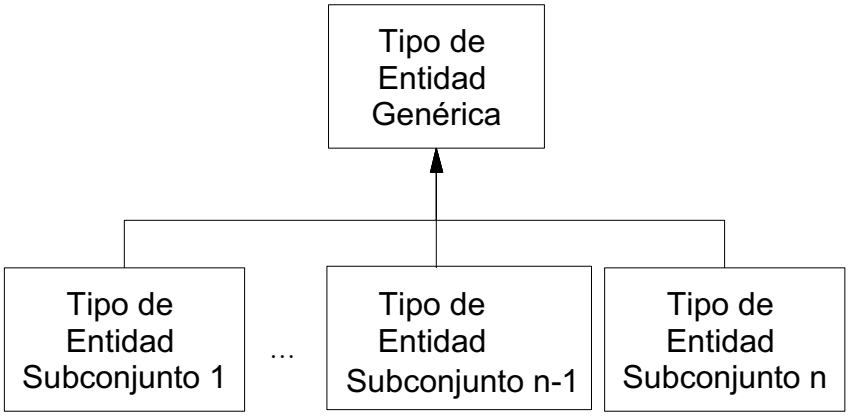
10.2.1 Clasificación de los tipos de entidad según sus identificadores.

- Tipo de Entidad Fuerte: Tipo de entidad con identificador interno.
- Tipo de Entidad Débil: Tipo de entidad con identificador externo o mixto.

10.3 Estructura de Generalización.

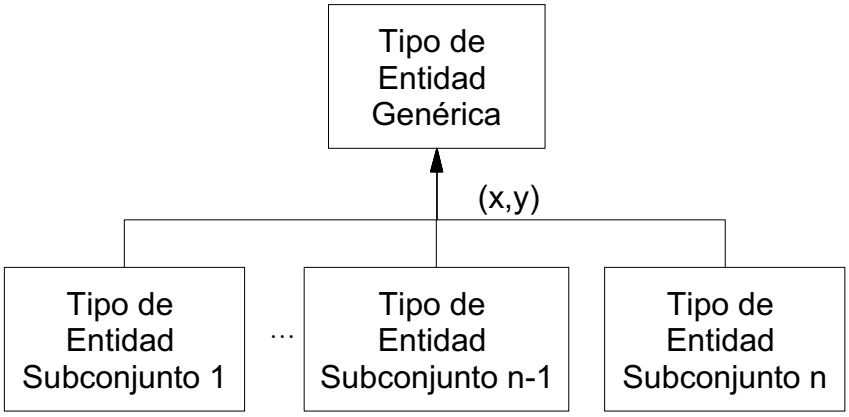
Un tipo de entidad TE (tipo de entidad genérica) es una generalización de un grupo de tipos de entidades STE_1 , STE_2 , ..., STE_n (tipos de entidad subconjunto) si cada entidad de los tipos de entidad STE_1 , STE_2 , ..., STE_n es también una entidad del tipo de entidad TE. (Lo opuesto a la generalización se denomina especialización.)

Además cada atributo, interrelación o generalización definida para un tipo de entidad genérica, será heredado por todas las entidades subconjunto de la generalización.



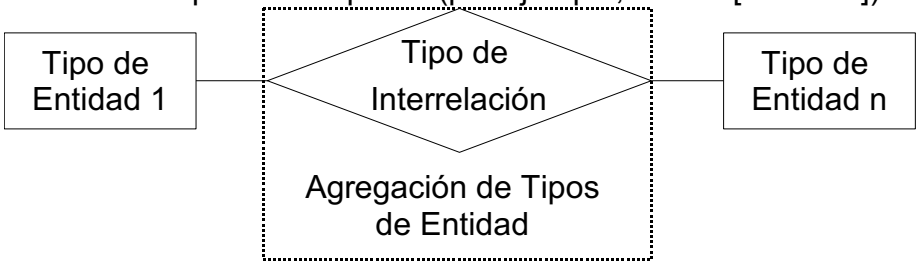
10.4 Cobertura.

Las jerarquías de generalización presentan la propiedad de cobertura. La cobertura puede ser parcial o total y exclusiva o superpuesta. La cobertura parcial o total permite especificar una restricción entre el tipo de entidad genérica y sus tipos de entidad subconjunto, donde todos los elementos del tipo de entidad genérico deben pertenecer a alguno de sus tipos de entidad subconjunto (si es total), o no (si es parcial). La cobertura exclusiva o superpuesta permite especificar una restricción entre los tipos de entidad subconjunto, donde los elementos que pertenecen a un tipo de entidad subconjunto pueden pertenecer también a otro tipo de entidad subconjunto (si es superpuesto) o no (si es exclusiva).



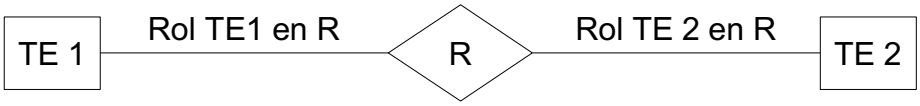
10.5 Agregación de Tipos de Entidad.

Un tipo de interrelación y los tipos de entidad que relaciona, puede ser manejado como un tipo de entidad en un nivel de abstracción mayor, lo que posibilita que se pueda interrelacionar con otros tipos de entidad. Este mecanismo es conocido como Estructura de Agregación o Agregación de Tipos de Entidad, en aquellas extensiones del MER que la incorporan (por ejemplo, CCER [Varas98]).



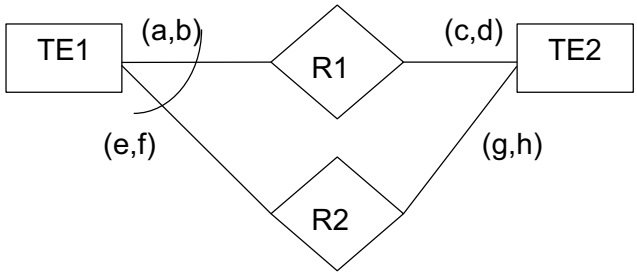
10.6 Roles de Tipos de Entidad en Tipos de Interrelación.

Un Rol de un Tipo de Entidad en un Tipo de Interrelación es la función que aquel cumple dentro de ésta. La definición de roles permite atribuirle a un tipo de entidad su semántica dentro de la agregación, aportándole mayor expresividad al esquema y permitiendo disminuir ambigüedades en la definición de cardinalidades (esto cobra mayor importancia en aquellos tipos de interrelación que involucran a un mismo tipo de entidad más de una vez).



10.7 Tipos de Interrelaciones Exclusivas con respecto a un Tipo de Entidad.

(Esta definición se obtuvo en base a aquella en [deMiguel93]).
Sea TE un tipo de entidad y sea un conjunto de tipos de interrelación RE= $\{R_1,...,R_n\}$ tales que TE $\in R_i$, i en {1,...,n}, RE se dice exclusivo, si cada ocurrencia de TE sólo puede estar presente a lo más en un R_i, i en {1,...,n}.
Observación: En este caso la cardinalidad mínima de TE con respecto a R_i, con i en {1,...,n} debe ser 0.



10.8 Restricciones en MER extendido.

Las restricciones estáticas especifican los estados posibles de la base de datos modelada en un esquema dado. En un esquema MER la principal restricción estática está dada por la estructura (pertenencia de un atributo a un tipo de entidad o interrelación, tipos de entidad que relaciona un tipo de interrelación), y también se pueden especificar las siguientes.

- Dominio.
- Cardinalidad de atributo con respecto a un tipo de entidad.
- Cardinalidad de un tipo de entidad con respecto a un tipo de interrelación.
- Identificadores.
- Cobertura.
- Tipos de Interrelación Exclusivas con respecto a un Tipo de Entidad.

Las restricciones dinámicas son aquellas que restringen los cambios de estado en la base de datos. Estos aspectos, no son soportados por el modelo entidad relación.

10.9 Estrategia para modelar con MER.

Se debe hacer uso de los conceptos de abstracción básicos: clasificación, agregación y generalización. Para ello se pueden seguir los procesos siguientes.

1. Identificar Tipos de Entidad y las relaciones que existen entre ellos.
2. Descomponer un tipo de entidad en dos o más tipos de entidad, relacionados o no, o participando en una estructura de generalización.
3. Descomponer un tipo de interrelación en varias.
4. Identificar atributos para cada elemento.
5. Definir identificadores para los tipos de entidad.
6. Definir restricciones de cardinalidad y cobertura.
7. Verificar que el esquema resultante es correcto con respecto a la especificación (representa toda la realidad descrita).
8. Verificar que el esquema es correcto con respecto al buen uso del modelo.
9. Analizar modificaciones al esquema.

10.10 Esquema MER y Documentación.

El esquema conceptual de una base de datos en el modelo entidad relación no es sólo el diagrama que se genera al utilizar las reglas generadoras del modelo, sino también la documentación textual asociada.

En este último punto, cobran mayor importancia aquellos aspectos que no quedan explícitamente especificados en el esquema gráfico, ya sea por un criterio estético o por falta de expresividad del modelo.

Comunmente, los dominios no se incorporan en el esquema gráfico, y su definición ni siquiera tiene representación, por lo que su documentación fuera del esquema es obligatoria. También es necesario hacer énfasis en restricciones estáticas que no fueron modeladas, y, en caso de existir restricciones dinámicas, estas deben especificarse fuera del esquema, dado que el modelo entidad relación no las soporta.

Para efectos de documentación, se propone anexar al esquema MER (gráfico), las tablas siguientes con la informacón que corresponda.

10.10.1 Tipos de Entidad.

Tipo de Entidad		
Descripción		
Atributo	Dominio	Cardinalidad

Notación para los atributos que son identificadores: Atributo@

10.10.2 Tipos de Interrelación

Tipo de Interrelación		
Descripción		
Tipos de Entidad Relacionados	Rol	Cardinalidad
Atributo	Dominio	Cardinalidad

10.10.3 Atributos Compuestos.

Atributo	
Descripción	
Presente en	

Notación para Descripción:

Atributo Componente 1+ Atributo Componente 2+ … + Atributo Componente n : El atributo se compone de Atributo Componente 1, Atributo Componente 2, … , Atributo Componente n. Cada uno de los atributos Atributo Componente i debe documentarse separadamente.

10.10.4 Atributos.

Atributo	
Descripción	
Dominio	
Presente en	

Notación para presente en:

Nombre1(TE) : El objeto donde se usa el atributo se denomina Nombre1 y es un Tipo de Entidad.

Nombre2(TI): El objeto donde se usa el atributo se denomina Nombre2 y es un Tipo de Interrelación.

Nombre3(@TE): El objeto donde se usa el atributo se denomina Nombre3 y es un Tipo de Entidad, siendo este atributo (parte de) el identificador.

Nombre4(A): El objeto donde se usa el atributo se denomina Nombre4 y es un atributo compuesto.

10.10.5 Dominios.

Dominio	
Definición	
Atributos	

10.10.6 Estructuras de Generalización.

Generalización	Tipo de Entidad Genérica	
Cobertura		
Tipos de Entidad Subconjunto		

10.10.7 Agregación de Tipos de Entidad

Agregación	Tipo de Interrelación	
Nombre Agregación		

10.10.8 Restricciones Estáticas no modeladas.

Restricciones Estáticas		
Id Restricción	Objetos Involucrados	Restricción

10.10.9 Restricciones Dinámicas.

Restricciones Dinámicas		
Id Restricción	Objetos Involucrados	Restricción

10.11 Cualidades de un esquema de datos.

10.11.1 Completitud

Un esquema es completo cuando representa todas las características pertinentes al dominio de la aplicación. Se puede comprobar en principio mirado en detalle todos los requerimientos del dominio de la aplicación y verificando que cada uno de ellos esté representado en algún lugar del esquema (el esquema es completo respecto a los requerimientos) y también se puede revisar el esquema para verificar que cada concepto esté mencionado en los requerimientos (los requerimientos están completos respecto al esquema).

10.11.2 Corrección.

Un esquema es correcto cuando usa con propiedad los conceptos del modelo (MER en este caso).

Un esquema es sintácticamente correcto cuando los conceptos se definen con propiedad en el esquema; por ejemplo, los subconjuntos y las generalizaciones se definen entre entidades pero no entre interrelaciones. Un esquema es semánticamente correcto cuando los conceptos (entidades, interrelaciones, etc.) se usan de acuerdo con sus definiciones. Por ejemplo, es un error semántico usar un atributo para representar los productos de un empresa manufacturera cuando se necesita representar varias propiedades de los productos (por ejemplo, código del producto, precio, partes, etc.), porque un atributo es una propiedad elemental.

Errores semánticos más frecuentes:

1. Usar un atributo en lugar de una entidad.
2. Olvidar una generalización (o un subconjunto).

3. Olvidar una propiedad de herencia de las generalizaciones.
4. Usar una interrelación con un número erróneo de entidades (por ejemplo, una interrelación binaria en vez de una ternaria).
5. Usar una entidad en lugar de una interrelación.
6. Olvidar algún identificador de una entidad.
7. Omitir alguna especificación de cardinalidad mínima o máxima.

10.11.3 Minimalidad.

Un esquema es mínimo cuando cada aspecto de los requerimientos aparece sólo una vez en el esquema. También se puede decir que un esquema es mínimo si no se puede borrar del esquema un concepto sin perder alguna información. Cabe señalar que algunas veces es aconsejable permitir alguna redundancia en el esquema; sin embargo, esta redundancia debe documentarse. Esto se logra, por lo regular, añadiendo al esquema conceptual una tabla que indica cómo se calculan los datos derivados a partir de otros datos.

10.11.4 Expresividad.

Un esquema es expresivo cuando representa los requerimientos de una forma natural y se puede entender con facilidad a través del significado de las construcciones del esquema, sin necesidad de explicaciones adicionales.

10.11.5 Legibilidad.

Esta es una propiedad del diagrama que representa gráficamente al esquema. Un diagrama tiene buena legibilidad cuando respeta ciertos criterios estéticos, tales como evitar los cruces de lineas, trazar los cuadros (tipos de entidades) y los rombos (tipos de interrelaciones) de un tamaño similar, que las conexiones sean trazos verticales u horizontales, dejar los niveles jerárquicos superiores sobre los inferiores y minimizar el número de 'esquinas' en el diagrama.

10.11.6 Autoexplicación.

Un esquema se explica a sí mismo cuando puede representar un gran número de propiedades usando el modelo conceptual por si mismo, sin otros formalismos (por ejemplo, anotaciones en lenguaje natural).

10.11.7 Extensibilidad.

Un esquema se adapta fácilmente a requerimientos cambiantes cuando puede descomponerse en partes (módulos o vistas), a fin de aplicar los cambios dentro de cada parte.

10.11.8 Normalidad.

El concepto de normalidad viene de la teoría de la normalización, asociada al modelo relacional. Las formas normales (primera, segunda, tercera, Boyce/Codd, cuarta y quinta), pretenden mantener la estructura lógica de los datos en una forma normal purificada, mitigando los problemas de las anomalías de inserción, borrado y actualización que ocasionan trabajo innecesario porque deben aplicarse los mismos cambios a varios casos de datos, así como el problema de pérdida accidental de datos o la dificultad de representación de determinados hechos.

11 Formalismo Individual

El Formalismo Individual es una herramienta de modelación, o modelo de datos, que permite generar esquemas para cierta realidad. El formalismo individual es un método eminentemente semántico.

Este modelo es utilizado como lenguaje de alto nivel para otros modelos.

Los componentes básicos del F.I. son:

- Individuo
- Relación
- Propiedad

Individuo

Es una familia de objetos que se caracterizan por tener las mismas características. Se les debe definir sin hacer referencia a otros individuos y cada ocurrencia de un individuo debe ser distinguible de otro.

Las características principales de un individuo son:

- la elección de cada individuo es una elección de quien realiza el esquema.
- un individuo debe tener existencia propia, y se le debe poder describir por sí solo, es decir, sin hacer referencia a otros componentes
- cada ocurrencia de un individuo debe ser distinguible de otras
- entre todas las propiedades de un individuo, al menos una o un grupo de ellas nos debe permitir identificar en forma única una ocurrencia de él; a esta propiedad o grupo de propiedades se le llama *Identificador*.

Un individuo queda completamente definido cuando se conoce su *nombre*, *identificador*, y lista de *propiedades* (también llamada entidad o rubrica).

Ocurrencia de Individuo.

Es la ocurrencia o instancia de un individuo, esto es, corresponde a un objeto concreto de la realidad.

Relación

Es una asociación que se establece entre individuos. En general una relación es consecuencia de asociaciones que existen en la realidad.

Una relación se define por cuatro proposiciones:

- su elección depende del interés de la persona que está modelando.
- no tiene existencia propia, materializa una asociación entre dos o más individuos.
- cada ocurrencia de una relación debe ser distinguible de otras
- las propiedades de una relación son comunes a todas sus ocurrencias. Esta lista de propiedades puede estar vacía, en este caso, la relación no tiene ninguna propiedad propia y no hace más que representar un enlace semántico (generalmente de pertenencia).

Para definir completamente una relación se utilizan los siguientes conceptos:

- Colección: es la lista de individuos que componen una relación. Una relación puede tener una colección de dos o más individuos.
- Identificador: es la concatenación de los identificadores de los individuos que componen la relación.
- Rúbrica: está constituida por la lista de identificadores y propiedades propias.

Ocurrencia de Relación.

Es la ocurrencia de la relación entre dos individuos.
⍁Ejemplo. Juan es hijo de Pedro.

Propiedad

Es una característica o atributo que permite describir a los individuos y relaciones.
⍁Ejemplo. nombre.

La ocurrencia de una propiedad es un valor. Los valores que toman las propiedades pertenecen a un dominio dado.

Restricciones.

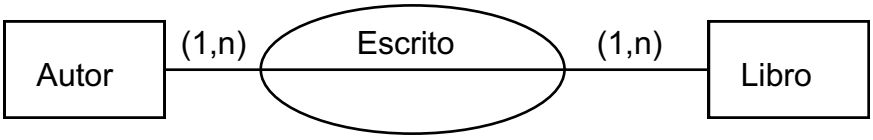
Cardinalidad.

Se refiere a la cantidad de veces, máximo y mínimo, que una ocurrencia de un individuo puede participar en una relación.

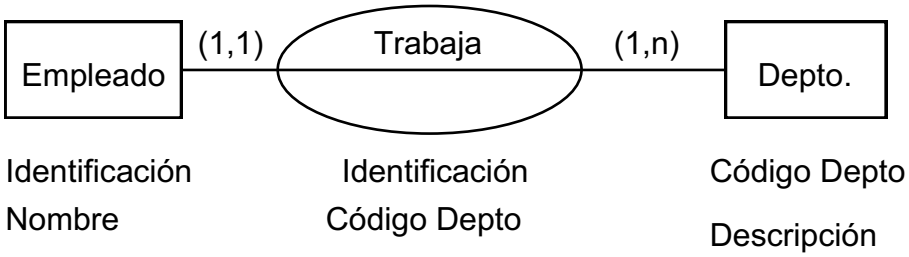
Relación Implícita.

Cuando dos individuos A y B están formando una relación, y la cardinalidad de la relación no es muchos es a muchos, entonces la relación se puede omitir, perteneciendo a uno de los individuos.

⇗Ejemplo.

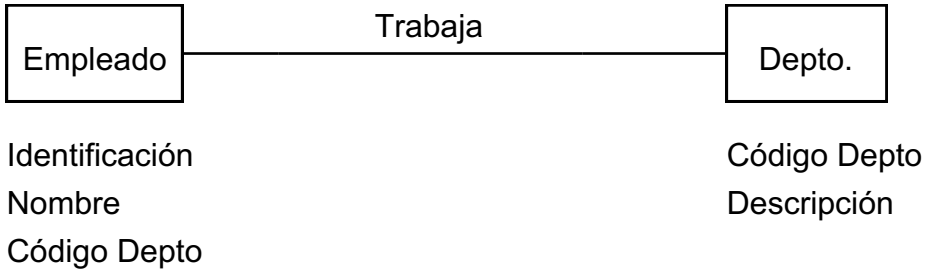


Esta es una relación del tipo muchos es a muchos, donde una ocurrencia de Autor puede participar una o muchas veces en la relación Escrito, y una ocurrencia de Libro también puede aparecer una o muchas veces en la relación Escrito. Esto significa que un Autor debe haber escrito al menos un libro para que se le considere como tal, que un Autor puede escribir más de un libro, que un Libro puede ser escrito por más de un Autor y que todo Libro debe tener a lo menos un Autor.

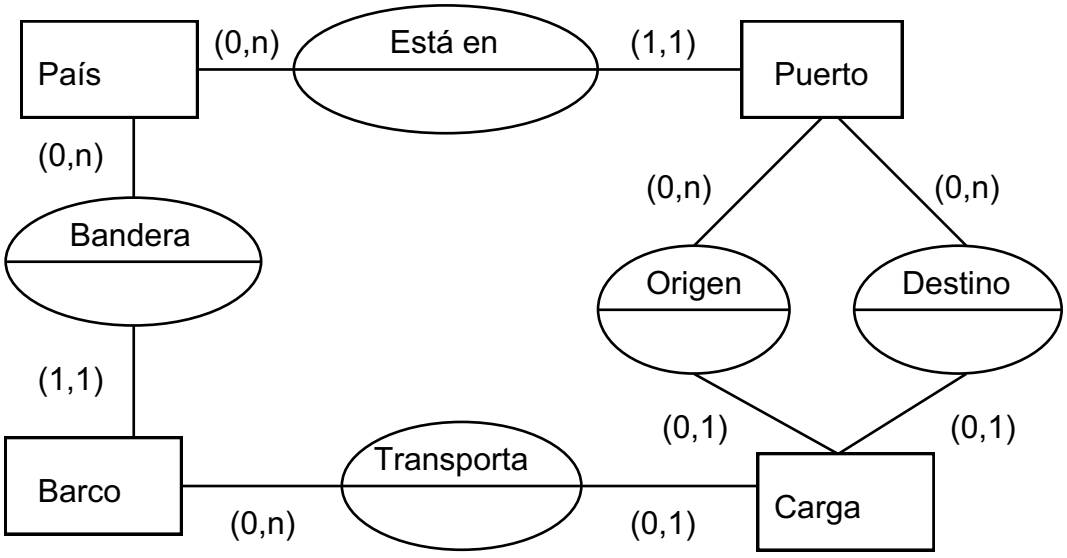


Esta relación no es muchos a muchos, donde una ocurrencia de Empleado debe participar una y sólo una vez en la relación Trabaja, mientras que una ocurrencia de Depto puede participar una o muchas veces en esta relación. Esto significa que un Empleado debe trabajar sólo en un Departamento, y no puede no trabajar en ninguno, un Departamento debe tener al menos un empleado, pero puede tener más de uno.

En este caso, se puede hacer uso de una relación implícita (por la cardinalidad uno es a uno de Empleado), quedando el esquema siguiente:



✍Ejercicio.
Describir la realidad modelada por el siguiente esquema



ˌ Desarrollo.

- Un Puerto está en un País.

- Un País puede tener muchos o ningún Puerto, así como muchos o ningún Barco con su Bandera.
- Un Barco debe tener la bandera de un sólo país.
- Un Barco puede ir vacío (sin Carga) o con muchas Cargas. Las Cargas tienen a lo más un Puerto de origen y un Puerto de Destino.
- Los Puertos pueden ser Origen y Destino para muchas o ninguna Carga.

12 Modelo Relacional.

Este modelo fue propuesto pro Codd en 1970 y se divide en tres partes, las cuales separan la estructura, la integridad y la manipulación de los datos.

12.1 Estructura de Datos Relacional.

La estructura de datos relacional tiene como elemento fundamental la relación. Aquí no existe diferencia entre entidades y relaciones o entre individuos y relaciones.

Una *relación* constituye lo que podríamos llamar una tabla. Una *Tupla* corresponde a una fila de esta tabla y un *atributo* a una columna. El número de tuplas de una relación se denomina *cardinalidad* y el número de atributos se denomina *grado*.

La *clave primaria* es un identificador único para la tabla, es decir, un atributo o combinación de atributos tal que nunca existen dos tuplas de la relación con el mismo valor en ese atributo o combinación de atributos.

Por último, pero no por eso menos importante, un *dominio* es una colección de valores, de los cuales uno o más atributos (columnas) obtienen sus valores reales.

Para efecto de modelación, interesa reconocer relaciones, atributos, dominios y claves primarias. La cardinalidad de una relación se considera a un nivel de implementación.

12.2 Propiedades de las relaciones.

- No existen tuplas repetidas.
- Las tuplas no están ordenadas (de arriba hacia abajo).
- Los atributos no están ordenados (de izquierda a derecha).
- Todos los valores de los atributos son atómicos.

12.3 Reglas de Integridad Relacional

12.3.1 Claves primarias.

Una clave candidata para una relación R es un atributo K posiblemente compuesto, tal que satisface las siguientes dos propiedades independientes del tiempo:

- Unicidad. En cualquier momento dado, no existen dos tuplas en R con el mismo valor de K.

- Minimalidad. Si K es compuesto, no será posible eliminar ningún componente de K sin destruir la propiedad de unicidad.

De entre las claves candidatas se elige la clave primaria.

Ningún componente de la clave primaria de una relación puede en algún momento no tener valor (aceptar nulos).

Esto significa que no tiene sentido modelar una entidad que no podemos identificar ni distinguir unas de otras.

12.3.2 Claves Foráneas.

En el modelo relacional se denominan *claves ajenas o claves foráneas* a una referencia de una relación a otra, mediante su clave. Este concepto lo conocemos en el formalismo individual como una relación implícita.

Una Relación (R1) puede poseer como uno de sus atributos (A) una clave primaria de otra relación (R2). Este atributo (A) constituye una clave foránea en R1 y referencia a R2.

En este caso las claves foráneas responden al mismo patrón de las relaciones implícitas del formalismo individual, es decir, existen cuando la cardinalidad de la relación es uno es a muchos.

La regla de integridad referencial nos indica que ningún atributo A que constituye una clave foránea en una relación R1 y referencia a la clave primaria de una relación R2 (no necesariamente distinta) puede tomar un valor que no esté presente en la relación referenciada R2. Esto significa, que la base de datos no debe contener valores de clave ajena sin concordancia.

12.4 Álgebra Relacional.

Consiste de un conjunto de operadores de alto nivel que operan sobre relaciones. Cada uno de estos operadores toma una o dos relaciones como entrada y produce una nueva relación como salida (propiedad de clausura).

Codd definió un conjunto de 8 operadores, los que se describen a continuación.

1. Restricción. Extrae las tuplas especificadas de una relación dada (o sea, restringe la relación sólo a las tuplas que satisfagan una condición especificada).

2. Proyección. Extrae los atributos especificados de una relación dada.

3. Producto. A partir de dos relaciones especificadas, construye una relación que contiene todas las combinaciones posibles de tuplas, una de cada una de las dos relaciones.

4. Unión. Construye una relación formada por todas las tuplas que aparecen en cualquiera de las dos relaciones especificadas.

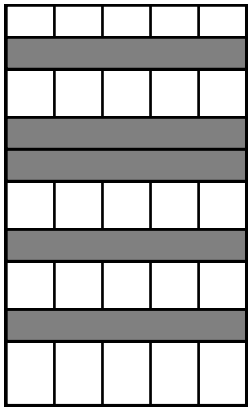
5. Intersección. Construye una relación formada por todas aquellas tuplas que aparecen en las dos relaciones especificadas.

6. Diferencia. Construye una relación formada por todas las tuplas de la primera relación que no aparezcan en la segunda de las dos relaciones especificadas.

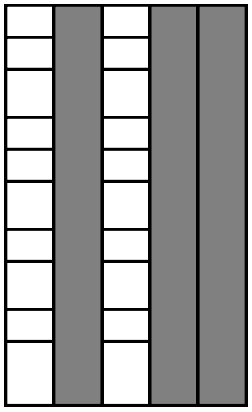
7. Reunión. A partir de dos relaciones especificadas, construye una relación que contiene todas las posibles combinaciones de tuplas, una de cada una de las dos relaciones, tales que las dos tuplas participantes en una combinación dada satisfagan alguna condición especificada.

8. División. Toma dos relaciones, una binaria y otra unaria, y construye una relación formada por todos los valores de un atributo de la relación binaria que concuerdan (en el otro atributo) con todos los valores en la relación unaria.

Restricción



Proyección



Producto

a		z		a	z
b	x	y	=	a	y
c				b	z
				b	y
				c	z
				c	y

División

a	x			z	
a	y			y	=
a	z	div			
b	x				
c	z				
c	y				

Reunión

a1	b1		b1	c1		a1	b1	c1
a2	b2	JOIN	b2	c2	=	a2	b2	c2
a3	b2		b3	c3		a3	b2	c2

12.5 Normalización.

El diseño de esquemas para generar bases de datos relacionales debe considerar el objetivo de almacenar información sin redundancia innecesaria, pero que a la vez nos permitan recuperar información fácilmente. Una técnica consiste en diseñar esquemas que tengan una forma normal adecuada.

Las propiedades indeseables que trae un mal diseño son básicamente

- repetición de información
- incapacidad para representar cierta información
- pérdida de información.

Las formas normales, definidas en la teoría relacional, nos permiten evitar que estas propiedades indeseables aparezcan en una base de datos basada en un esquema mal diseñado. Un esquema debe estar a lo menos en tercera forma normal, para que sea aceptable.

Hay que considerar que las reglas de normalización están dirigidas a la prevención de anomalías de actualización e inconsistencias en los datos. Ellas no reflejan ninguna consideración de rendimiento. En cierta forma pueden ser visualizados como orientadas por el supuesto de que todos los atributos no clave serán actualizados frecuentemente.

12.5.1 Formas Normales.

PRIMERA FORMA NORMAL
Una relación está en primera forma normal (1FN) si y sólo si todos los dominios simples subyacentes contienen sólo valores atómicos.

Otra forma de expresar la primera forma normal es decir que todas las ocurrencias de un tipo de registro deben contener el mismo número de campos.

⇗ Ejemplo.

Consideremos el caso de agentes que representan compañías que fabrican productos. Una relación sin normalizar que indique los productos que venden los representantes es:

AGENTE	COMPAÑÍA	PRODUCTO1	PRODUCTO2	... [1]
Caro [2]	Ford	auto	camión	
	GM	auto	camión	
Jeria	Ford	auto		
Bravo	Ford			

Una relación que representa la misma situación y no transgrede la primera forma normal sería:

AGENTE	COMPAÑÍA	PRODUCTO
Caro	Ford	auto
Caro	Ford	camión
Caro	GM	auto
Caro	GM	camión
Jeria	Ford	auto
Bravo	Ford	

SEGUNDA FORMA NORMAL

Una relación está en segunda forma normal (2FN) si y sólo si está en 1FN y todos los atributos no clave dependen por completo de la clave primaria.

La segunda forma normal es transgredida cuando un campo no clave es un dato sobre un subconjunto de una clave (compuesta).

⇗ Ejemplo.

Consideremos el siguiente esquema propuesto para un registro de inventario.

[1] Repetición variable de atributos, n productos.

[2] Se forma un grupo.

ARTÍCULO	**BODEGA**	CANTIDAD	DIRECCIÓN-BODEGA

Aquí, la clave está formada por (ARTÍCULO,BODEGA).

Se puede observar fácilmente que DIRECCIÓN-BODEGA es un dato acerca de BODEGA y no de ARTICULO, por lo que no se estaría cumpliendo con la segunda forma normal.

Los problemas básicos de diseño son:

- La dirección de la bodega se repite para cada artículo que se almacena en esa bodega (redundancia).
- Si la dirección de bodega cambia, cada registro que se refiera a un artículo almacenado en esa bodega debe ser actualizado. Debido a la redundancia, los datos pueden llegar a ser inconsistentes, con diferentes registros indicando diferentes direcciones para la misma bodega (integridad).
- Si en algún momento no hubiera partes almacenadas en alguna bodega, no habría un registro para anotar la dirección de la bodega (anomalía).

Para satisfacer la segunda forma normal, el esquema anterior debe ser reemplazado por el siguiente:

ARTÍCULO	**BODEGA**	CANTIDAD

BODEGA	DIRECCIÓN

TERCERA FORMA NORMAL

Una relación está en tercera forma normal (3FN) si y sólo si está en 2FN y todos atributos no clave dependen de manera no transitiva de la clave primaria.

La tercera forma normal es transgredida cuando una propiedad no identificada (no clave) es un dato acerca de otro campo no clave.

Ejemplo. El esquema siguiente no está en 3FN.

EMPLEADO	PADRE	DIRECCIÓN-PADRE

Ahora, el siguiente esquema no transgrede la 3FN.

EMPLEADO	PADRE

PADRE	DIRECCIÓN-PADRE

Estas son las tres formas normales básicas, existen además la forma normal de Boyce/Codd, la cuarta forma normal, quinta forma normal.

⇗ Ejercicio.

Un hospital mantiene un sistema de control de drogas en el cual las siguientes características aparecen como las relevantes:

- Las drogas están mantenidas en estantes especiales.
- Las drogas son provistas por distintos proveedores
- Existe un archivo que incorpora datos para permitir la ubicación de los proveedores usuales o alternativos de las drogas.
- Siempre que una droga es usada para una intervención y/o tratamiento, los registros del archivo indicado anteriormente es actualizado.
- Cuando la cantidad de la droga en stock cae bajo un cierto nivel, es puesta en una lista de re-orden. Se revisan los fabricantes de la droga y se ubican el proveedor usual o alternativo para ella y se emite una orden de compra para ella.
- Ocasionalmente pedidos urgentes son hechos por teléfono.
- Las drogas recibidas traen adjunto un recibo el cual es chequeado con los detalles de la droga. El registro de la droga es actualizado y la droga es ubicada en el estante correspondiente.

⸢ Desarrollo.
Supuestos de Diseño.

Los principales supuestos que soportan la normalización del sistema son los que se indican a continuación.

1. Existen Ubicaciones (por ejemplo casilleros) en donde se almacenan todas las versiones de una droga.
2. Sólo se almacena a lo más una droga (en todas sus versiones) en una ubicación.
3. Una droga y sus versiones es almacenada en una y sólo una ubicación.
4. Una droga tiene una o más versiones, las cuales se identifican por un código (versión).
5. Una versión es única, y pertenece sólo a una droga.
6. No existen dos versiones con el mismo nombre y código para la misma droga.

7. Un laboratorio puede producir una o varias versiones de drogas.
8. Un laboratorio cuenta con uno o más proveedores.
9. Un proveedor representa a uno o más laboratorios.
10. Un proveedor distribuye todas las drogas que produce un laboratorio al cual representa.
11. Una droga tiene sólo un proveedor usual.
12. Todos los proveedores que proveen una droga y no están catalogados como su proveedor usual constituyen sus proveedores alternativos.
13. Un proveedor puede ser proveedor usual de ninguna, una o muchas drogas.
14. Dos proveedores pueden tener la misma dirección o teléfono.

Relaciones, Atributos y Dominios.

Se constituye el sistema de las siguientes relaciones:

- Ubicación (ubicación, estado)

Objetivo: Contener todas las ubicaciones destinadas para el almacenamiento de las drogas.

ubicación: numérico de largo 4. Varía de 1 a 9999. Único.
estado: caracter de largo 3. Toma valores 'ocu' o 'dis', para indicar ocupado y disponible respectivamente.

Claves candidatas: ubicación.
Clave primaria: ubicación.
Claves foráneas: no tiene.

- Proveedor (proveedor, nombreproveedor, fono, dirección)

Objetivo: contener la información de los proveedores de drogas del hospital.

proveedor: numérico de largo 4. Varía de 1 a 9999. Único.
nombreproveedor: caracter de largo 35. Nombre de los proveedores. Único.
fono: numérico de largo 7. Varía de 1 a 9999999.
dirección: caracter de largo 50. Dirección de los proveedores.

Claves Candidatas: proveedor, nombreproveedor.
Clave primaria: proveedor.
Claves foráneas: no tiene.

- Laboratorio (laboratorio, nombrelaboratorio)

Objetivo: contener la información de los laboratorios que producen drogas que se utilizan en el hospital.

laboratorio: numérico de largo 4. Varía de 1 a 9999. Único.
nombrelaboratorio: caracter de largo 15. Nombre de los laboratorios. Único.

Claves candidatas: laboratorio, nombrelaboratorio.
Clave primaria: laboratorio.
Claves foráneas: no tiene.

- Droga (droga, nombredroga, stock, stockmin, ubicación, proveedor)

Objetivo: contener la información de las drogas que se utilizan y mantienen en el hospital.

droga: numérico de largo 4. Varía de 1 a 9999. Único.
nombredroga: caracter de largo 10. Nombre de las drogas. Único.
stock: numérico de largo 4. Mayor que 0. Stock actual de la droga.
stockmin: numérico de largo 4. Mayor que 0. Stock mínimo de la droga.
ubicación: numérico de largo 4. Ubicación de la droga. Único.
proveedor: numérico de largo 4. Proveedor usual de la droga. varía de 1 a 9999.

Claves candidatas: droga, nombredroga, ubicación.
Clave primaria: droga
Claves foráneas: ubicación, referencia a ubicación en la relación Ubicación.
proveedor, referencia a proveedor en la relación Proveedor.

- Versión (droga, versión, nombreversion, laboratorio)

Objetivo: contener la información de las distintas versiones que existen para cada droga que se utiliza en el hospital.

droga: numérico de largo 4. Varía de 1 a 9999.
versión: numérico de largo 4. Varía de 1 a 9999. .
nombreversión: caracter de largo 35. Nombre de las versiones. Único.
laboratorio: numérico de largo 4. Varía de 1 a 9999.

Claves candidatas: (droga, versión), nombreversion.
Clave primaria: (droga, versión)
Claves foráneas: laboratorio, referencia a laboratorio en la relación Laboratorio.

- ProvLab (proveedor, laboratorio)

Objetivo: contener la información acerca de cuales son los proveedores de un laboratorio.

proveedor: numérico de largo 4. Varía de 1 a 9999.
laboratorio: numérico de largo 4. Varía de 1 a 9999.

Claves candidatas: (proveedor, laboratorio)
Clave primaria: (proveedor, laboratrorio)
Claves foráneas: proveedor, referencia a proveedor en la relación Proveedor
laboratorio, referencia a laboratorio en la relación Laboratorio.

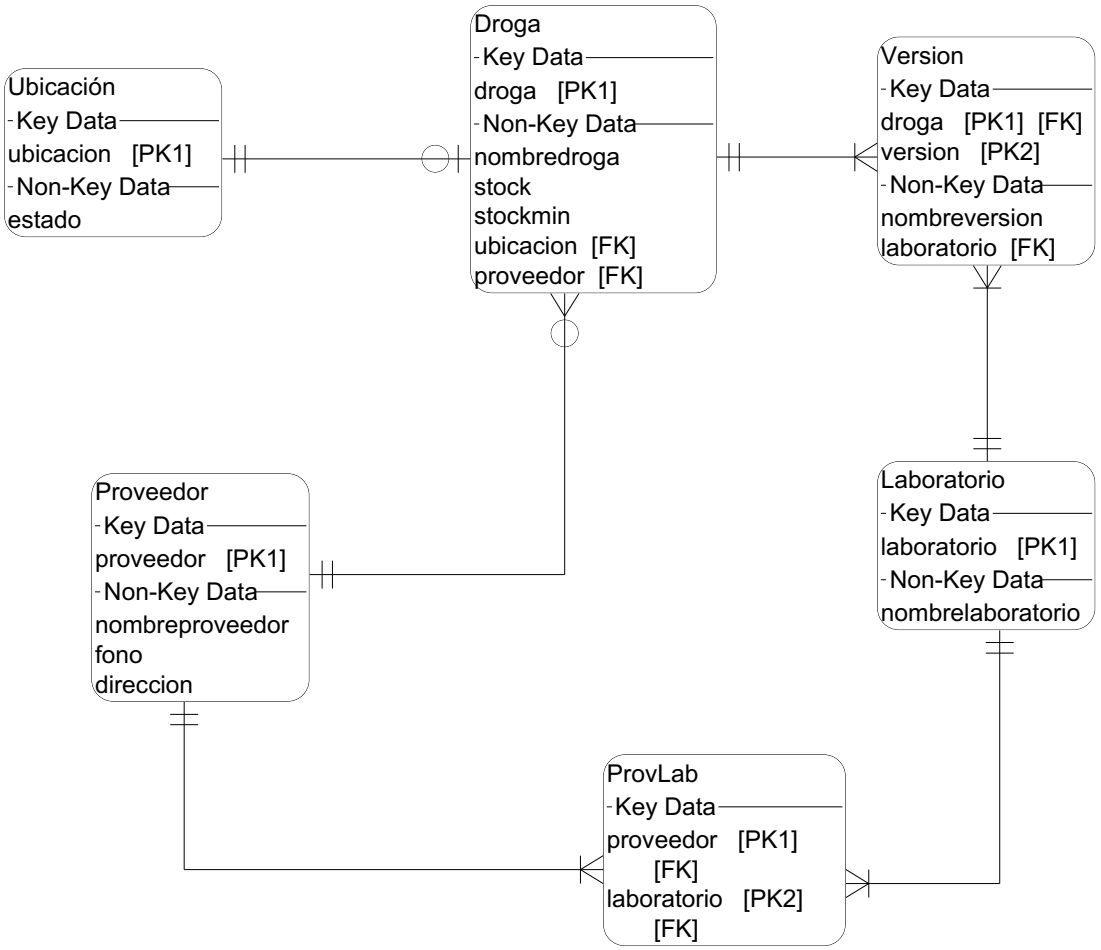
Restricciones de Integridad.

Además de las restricciones de integridad de las entidades (claves primarias no nulas), las de integridad referencial para las claves foráneas y las dadas por el dominio de los atributos, se tienen las que se declaran a continuación.

Si un proveedor es proveedor (usual) para una droga, este proveedor debe representar a un laboratorio que produzca una versión de esa droga.

Si una droga tiene una ubicación, entonces el estado de esa ubicación debe ser "ocupado".

Esquema.



☞Observación.

El formalismo gráfico utilizado explícita la implementación de interrelaciones (del MER) entre relaciones(del modelo Relacional) a través de claves foráneas. Las cardinalidades se representan por la notación pie de gallo, donde || se utiliza para la cardinalidad (1,1) o uno es a uno , ○| para la cardinalidad (0,1) o cero o uno, ○< para la cardinalidad (0,n) o cero es a muchos y >| para la cardinalidad (n,1) o muchos es a uno.

13 Redes Semánticas.

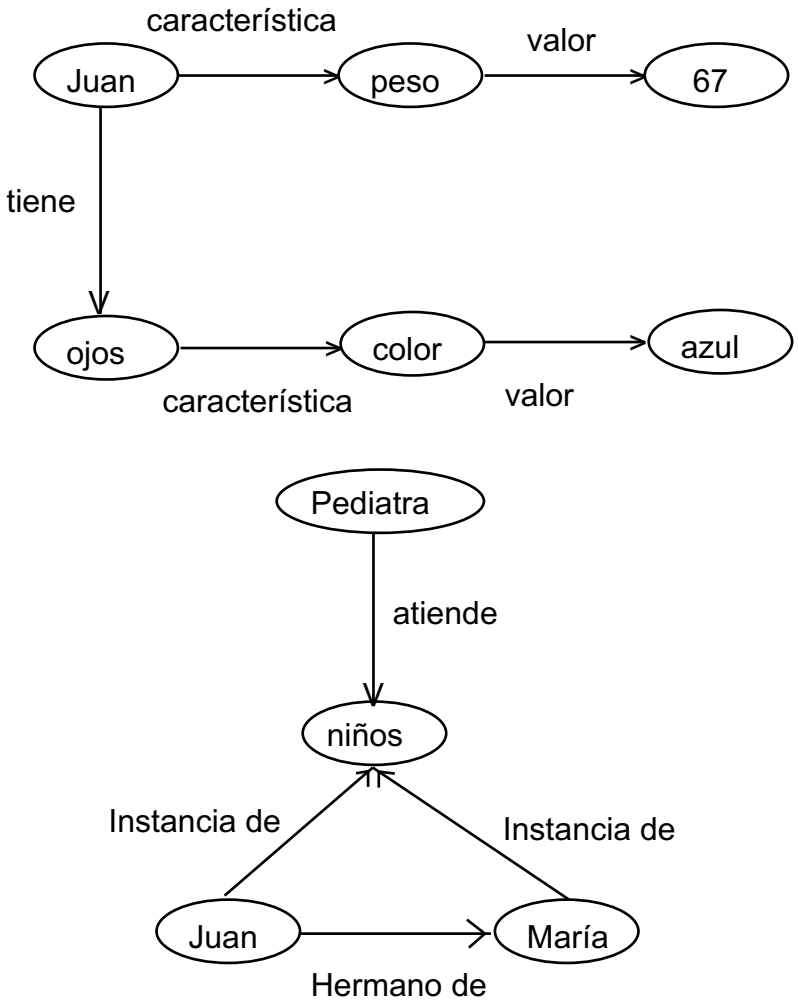
Las redes semánticas fueron desarrolladas por quienes trabajan en el área de la inteligencia artificial. Las estructuras básicas de este modelo consisten en nodo y arcos formando una red (un grafo). El objetivo de estas redes es la organización y representación del conocimiento general acerca del mundo.

El objetivo inicial para el desarrollo de las redes semánticas fue el entender el lenguaje natural, más que la clasificación de datos. Otra característica de las redes semánticas es que existen tantas como las necesidades que han tenido diferentes investigadores en diferentes proyectos.

Así, resulta difícil decidir a qué se le llama modelo de datos Red Semántica. Esto se debe a que se han tenido diferentes modelos de datos red semántica que son buenos al representar una realidad específica. Se puede decir entonces que cualquier grafo en el cual los nodos se conecten por medio de arcos se le puede llamar red semántica, siempre que nodos y arcos estén etiquetados.

Para tener semántica en un grafo, se necesita definir cuidadosamente el significado de nodos y arcos, y cómo son usados.

Las primeras redes semánticas usaban diferentes nodos y arcos para representar las asociaciones presentes en la memoria humana. Estas primeras redes fueron poco uniformadas en su estructura, no distinguiendo adecuadamente entre diferentes tipos de nodos y arcos. Por ejemplo, objetos individuales (instancias) y clases de objetos (entidades) coexistían en la misma red semántica. No se tenía una clara diferencia entre los nodos que denotaban instancias y los que denotaban clases, por ejemplo, las siguientes redes semánticas.

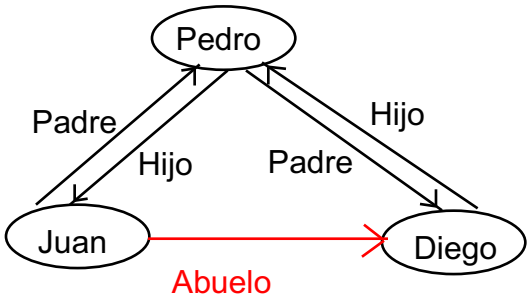


13.1 Características de las Redes Semánticas.

a. Diferencian entre tipos de objetos de las instancias. Así, se llama *clasificación* al proceso de ir de instancias de objetos a tipos de objetos.

b. Se introduce el concepto de *distancia semántica*, cantidad de arcos que separan un nodo de otro. En otros modelos de datos esta distancia sólo tiene implicancias en la performance, y generalmente no se considera ni tiene ninguna connotación semántica.

En las redes semánticas la distancia puede ser importante, y es usada para localizar objetos poco o muy relacionados, dependiendo de la distancia. En algunos casos se puede disminuir la distancia agregando arcos con ese propósito.

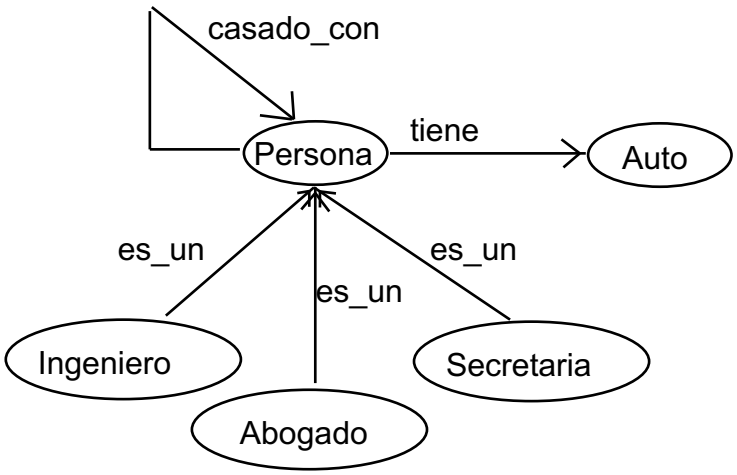


c. En las redes semánticas también se tiene la idea de *partición*: es el contexto de una red, en el sentido de tener una *subred*, así para una tarea o trabajo específico sólo una parte de la red está disponible.

Esta facilidad resulta útil en el momento de realizar búsquedas, ya que se limita el espacio de búsqueda.

d. También se tiene la jerarquía de tipo (u objeto). Los tipos de jerarquías que se tienen en una red semántica son PARTE-DE y ES-UN. La existencia de una jerarquía implica que se permite la *herencia* donde un objeto que pertenece a una clase hereda todas las propiedades de la clase.

⇗Ejemplo.

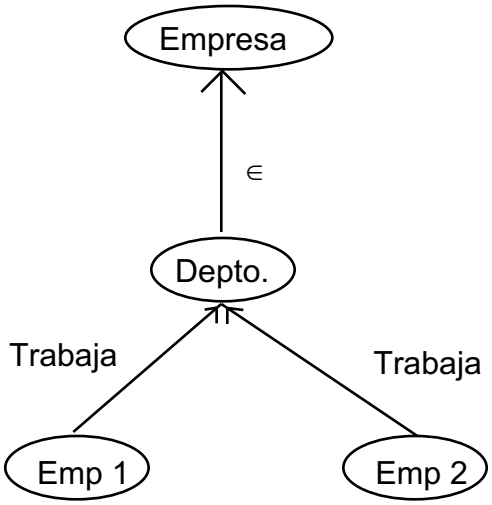


La herencia no se refiere a la herencia de atributos y sus valores, sino que también se heredan los tipos de relaciones permitidas para esa clase, esto es, su comportamiento.

⇗Ejemplo. Si un empleado es una persona, y la relación "casado con" es válida para persona, entonces también es válida para Ingeniero, Abogado y Secretaria.

⇗ Ejemplo. La representación de que un Empleado trabaja en un Departamento, vista en un modelo ER y Red Semántica.

Modelo E/R



Modelo Red Semántica

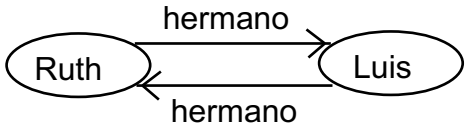
13.2 Descripción Formal de Red Semántica.

13.2.1 Estructuras.

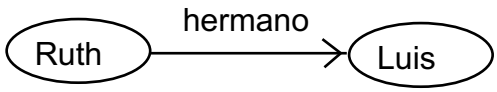
Las estructuras de cualquier red semántica consiste de un grafo (Red). La forma en que se distingue entre las diferentes redes y arcos determina el tipo de red semántica, y así mismo el modelamiento de datos para el cual fue creada.

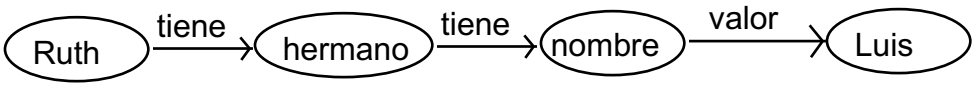
⇗Ejemplo. Se puede tener un modelo de datos red semántica en que los nodos representan cosas (instancias o valores de entidades) y los arcos pueden representar relaciones entre los nodos.

⇗Ejemplo.



En lo antes dicho se tiene que una unidad de información puede ser considerada ya sea como una cosa (nodo) o hecho (arco). Aquí se procura cierta libertad de interpretación, respecto a qué son arcos y qué son nodos. Una forma de decidir si una unidad es cosa (nodo) o hecho (arco) es preguntarse si más adelante tiene que relacionarse con otra cosa (nodo). Si es así, entonces es nodo.





Los nodos pueden ser caracterizados de acuerdo a las cosas que representan. Una de estas caracterizaciones (no la única) es la que establece que los nodos representan *conceptos, eventos, características* y *valores*.

a. Conceptos.
Son constantes para la realidad que se desea representar, y son utilizados para especificar valores (Ejemplo. Juan, la persona Juan)

b. Eventos.
Corresponden a acciones que ocurren en la realidad que se modela. Así, "Juan golpea a Pedro" se puede representar mediante un nodo "golpear" y dos nodos conceptos "Juan" y "Pedro".
Los arcos que conectan nodos eventos y nodos concepto corresponden a los roles que los conceptos juegan en el evento. En el ejemplo, "Juan" es el agente (el que produce el golpe) y "Pedro" el objetivo (el que recibe el golpe).

c. Características.

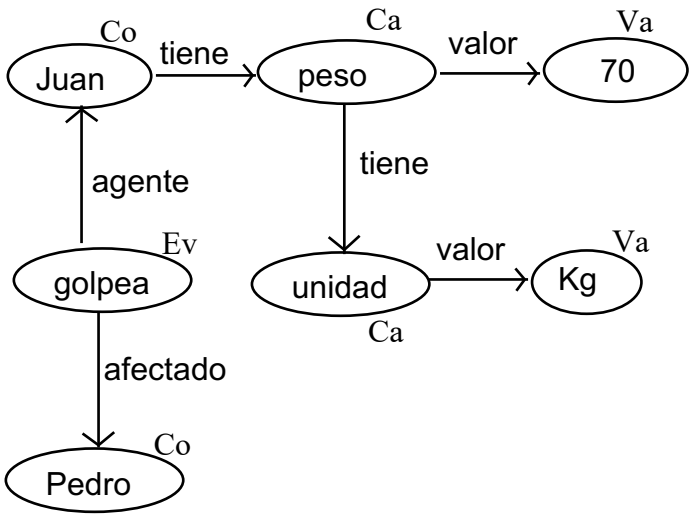
Son nodos que describen propiedades de un concepto. Corresponden a los atributos de un concepto. Ejemplo: el "peso" de Juan.

d. Valores.
Son nodos correspondientes a dominios de valores. Ejemplo: el peso de Juan es "48 Kg.". Corresponden a los valores que pueden tomar las características.

Ejercicios.
a. Juan golpea a Pedro.
b. Juan pesa 70 Kg.
Genere los esquemas y caracterice cada nodo, según lo visto.

Desarrollo.

Conceptos: Juan, Pedro
Eventos: golpea
Características: peso
Valores: 70 Kg.



En particular, si tuviéramos por ejemplo autos y personas, ambos corresponderían a un tipo de nodo CONCEPTO, y no hay diferencia entre el tipo de nodo Auto y el tipo de nodo Persona.

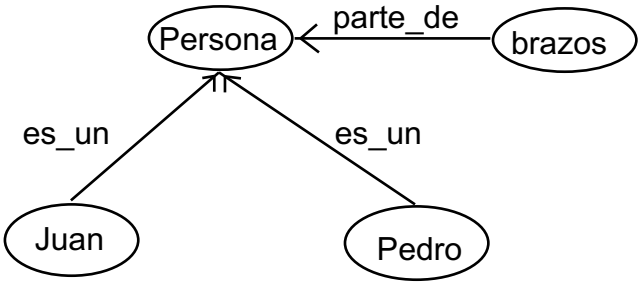
Complementariamente se agrega la clasificación por CLASES. Así, Juan y Pedro son conceptos que pertenecen a la clase Persona, que es a su vez otro concepto. Para lograr esta pertenencia a clases se usa el arco ES_UN.

Además se tiene el arco PARTE_DE, que permite generar estructuras de composición.

Ejemplo.
a. Pedro y Juan son personas.
b. Las personas tienen brazos.

Desarrollo.

Conceptos: Persona, Pedro, Juan, brazos.



Ejercicio.

- Las personas tienen dos manos.
- Las manos tienen cinco dedos.
- Pedro y Juan son personas.
- Pedro golpea a Juan.
- Las personas usan ropa.
- Las personas comen conejos.
- Los conejos son mamíferos.
- Las personas son mamíferos.
- Juan tiene una mano.

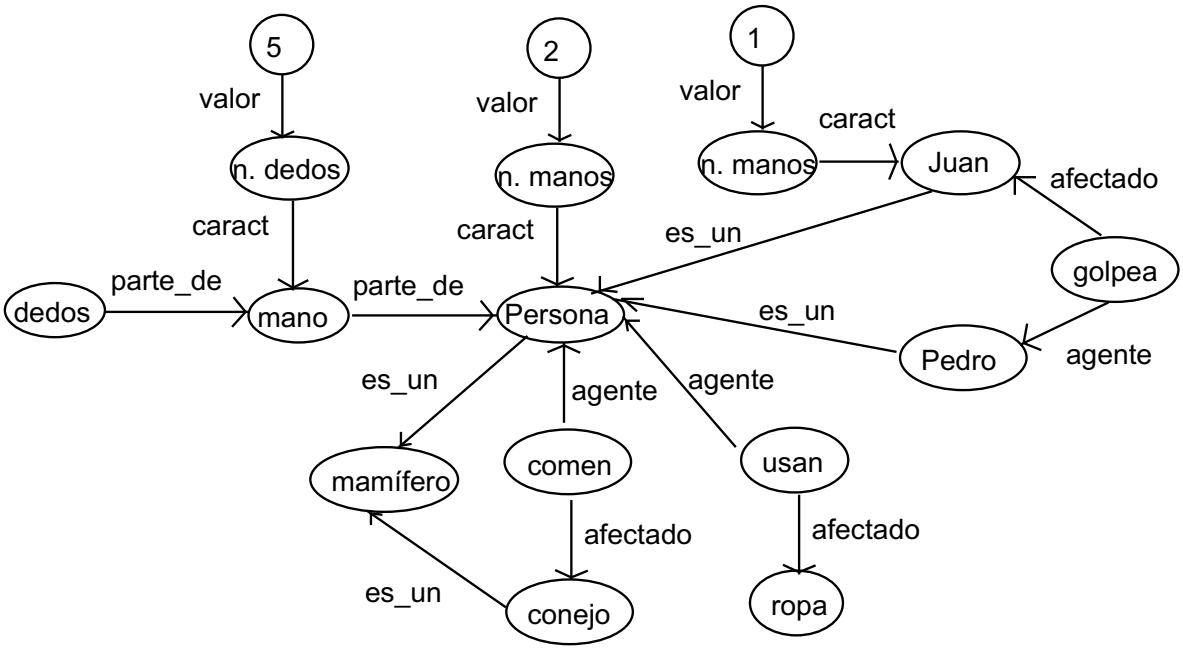
Desarrollo.

Eventos: golpea, usan, comen.

Conceptos: personas, manos, dedos, Pedro, Juan, conejos, mamíferos.

Características: número de manos, número de dedos.

Valores: 2,5,1.



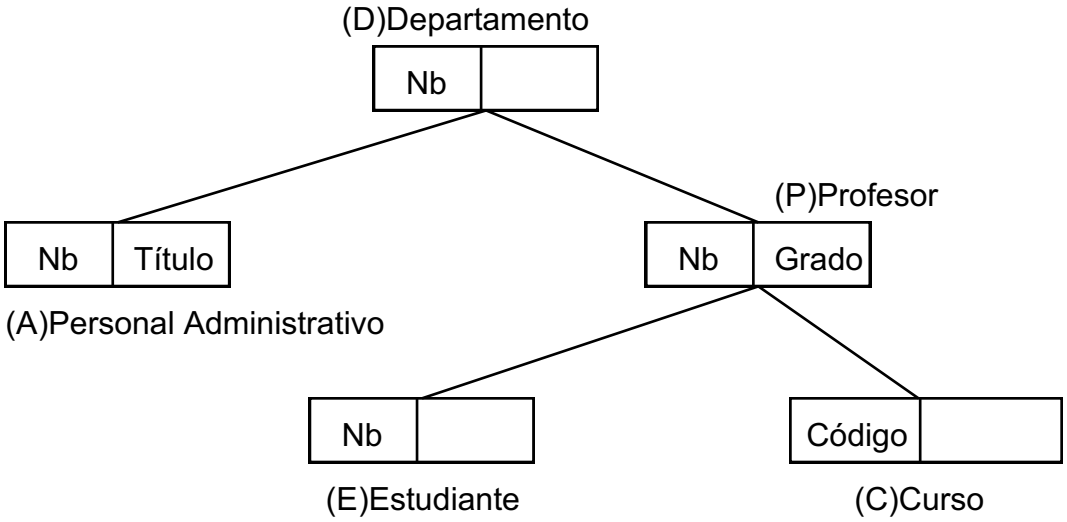
☞Observación.

En este caso, persona tiene dos manos, pero Juan (que es persona) tiene una, lo que se denota "ocultando" la característica "n. manos" de persona mediante la misma característica de Juan.

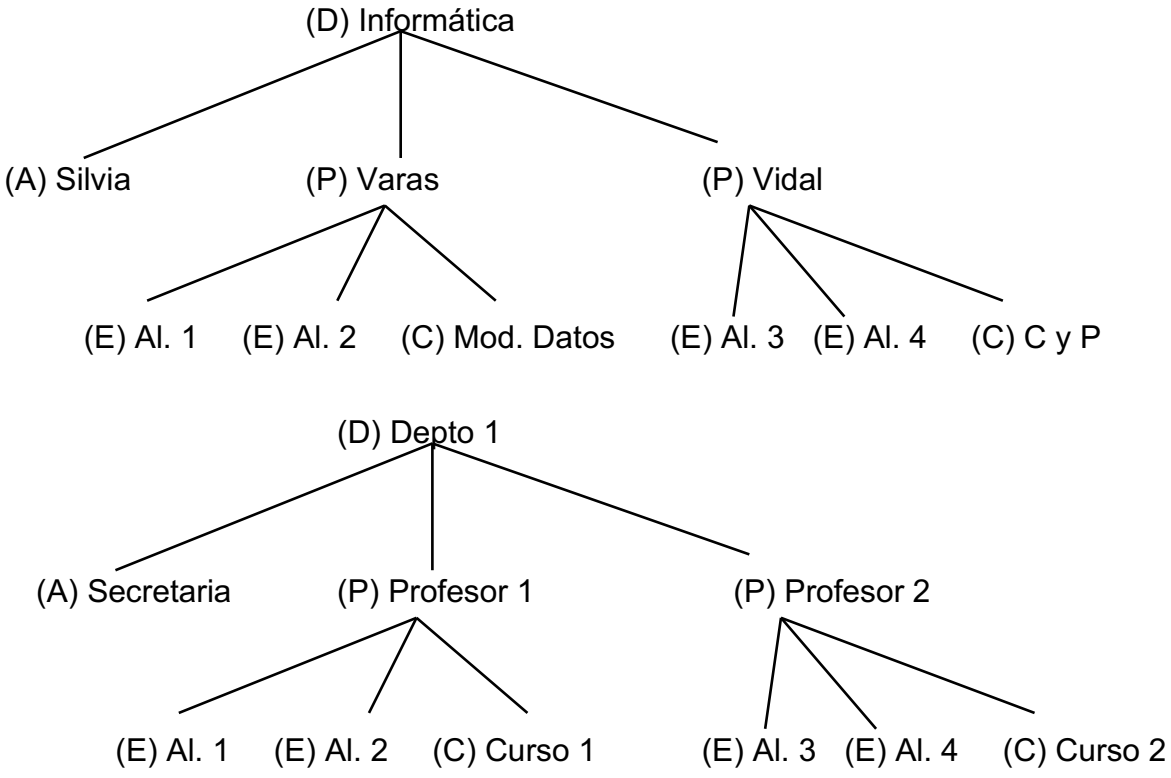
14 Modelo de Datos Jerárquico.

La estructura básica de este modelo de datos es el tipo de interrelación padre-hijo entre pares de tipos de registros. En la figura se define una base de datos con cinco tipos de registros (Departamento., Personal Administrativo, Profesor, Estudiante y Curso).

⇗Esquema Jerárquico o Árbol de Definición:



⇗Base de Datos:



En el modelo Jerárquico se tiene la conexión padre-hijo, y en consecuencia no puede haber un hijo sin un padre (no puede estar desconectado).

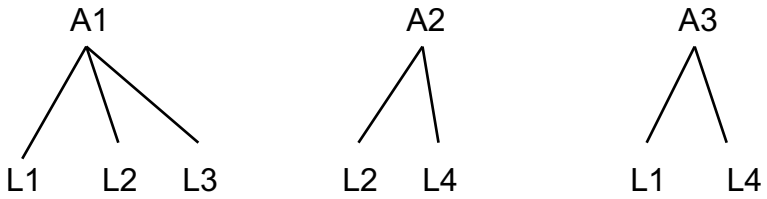
Una jerarquía debe obedece la estructura de un árbol. Así, el único nodo sin padre es el nodo raíz. Esto trae otra consecuencia, si se borra el padre también desaparecerán los hijos en la base de datos.

Además hay problemas con las relaciones muchos a muchos: un hijo no puede tener muchos padres, por lo que se debe repetir (duplicación de datos).

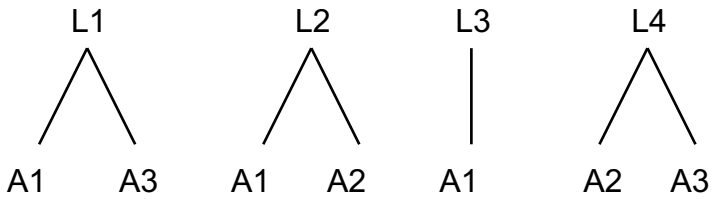
Ejemplo. Un autor tiene muchos libros escritos, los que a su vez pudieron ser escritos por muchos autores.

Existen dos alternativas:

y la base de datos quedaría según el esquema i:



y según el esquema ii:



¿Cuál es mejor? Cada uno sirve para obtener diferente información con mayor facilidad, por lo que la elección será condicionada a las necesidades del problema.

15 Modelo Orientado a Objetos.

El Modelo Orientado a Objetos se basa en el paradigma de programación orientada a objetos. Este paradigma ha tenido gran aceptación debido a que es de gran naturalidad buscar objetos en la realidad a modelar.

15.1 Estructura de Objetos.

El modelo orientado a objetos se basa en encapsular código y datos en una única unidad, llamada *objeto.* La interfaz entre un objeto y el resto del sistema se define mediante un conjunto de *mensajes.*

El motivo de este enfoque puede ilustrarse considerando una base de datos de documentos en la que los documentos se preparan usando uno entre varios paquetes software con formateador de texto. Para imprimir un documento debe ejecutarse el formateador correcto en el documento. Bajo un enfoque orientado a objetos, cada documento es un objeto que contiene el texto de un documento y el código que opera sobre el objeto. Todos los objetos del tipo documento responden al mensaje *Imprimir* pero lo hacen de forma diferente. Cada documento responde ejecutando el código formateador adecuado. Encapsulando dentro del objeto documento la información acerca de cómo imprimirlo, podemos tener todos los documentos con la misma interfaz externa al usuario (aplicación).

En general, un objeto tiene asociado:

- Un conjunto de *atributos* que contienen datos acerca del objeto. A su vez, cada valor de un atributo es un objeto.
- Un conjunto de *mensajes* a los que el objeto responde.
- Un conjunto (puede ser unitario) de *métodos*, que es un procedimiento o trozo de código para implementar la respuesta a cada mensaje. Un método devuelve un valor (otro objeto) como respuesta al mensaje.

Puesto que la única interfaz externa de un objeto es el conjunto de mensajes al que responde, es posible modificar la definición de métodos y atributos sin afectar a otros objetos.

También es posible sustituir un atributo por un método que calcule un valor.

⇙ Ejemplo. Un objeto documento puede contener un atributo de *tamaño* que contenga el número de bytes de texto en el documento, o bien un método de *tamaño* que calcule el tamaño del documento leyéndolo y contando el número de bytes.

La capacidad de modificar la definición de un objeto sin afectar al resto del sistema está considerada como una de las mayores ventajas del modelo de programación orientada a objetos.

15.2 Jerarquía de Clases.

Normalmente en la realidad a modelar existen muchos objetos similares. Por similar se quiere decir que tienen una naturaleza parecida, por lo que son candidatos a poseer atributos, métodos y responder a mensajes comunes en un contexto orientado a objetos.

Para el caso de los objetos similares, sería un trabajo inútil definir cada uno de ellos por separado. Por tanto, se agrupan para que conformen una *clase*. A cada uno de estos objetos se le llama *instancia* de su clase. Todos los objetos de una clase comparten una definición común , aunque difieran en los valores asignados a los atributos.

⇙ Ejemplo. En un banco, son objetos los clientes, las cuentas y los préstamos.

Una clase incluye:

- Un atributo con valores en un conjunto cuyo valor es el conjunto de todos los objetos que son instancias de la clase.
- Implementación de un método para el mensaje *nuevo*, el cual crea una nueva instancia de la clase.

Un esquema orientado a los objetos, normalmente requiere un gran número de clases. Sin embargo, a menudo se da el caso que varias clases son similares. Para permitir la representación directa de similitudes entre clases, se utiliza una jerarquía de especialización, como aquella utilizada por el modelo MER extendido.

⇗ Ejemplo. En un banco, es de esperarse que los *Empleados* y los *Clientes* tengan ciertos atributos comunes, como rut, nombre, dirección y teléfono, sin embargo los *Empleados* podrían tener un atributo exclusivo, como salario, mientras que *Clientes* tendrán uno como tasa_crédito. Aquí los empleados y clientes pueden ser representarse por especializaciones de una clase *Persona*. Los atributos y los métodos específicos para Empleado se asocian a la clase *Empleado*, en forma análoga, se actúa para los clientes. Los atributos y métodos comunes para empleados y clientes se asocian a la clase *Persona*.

15.2.1 Herencia.

Una propiedad interesante para efectos de modelación es la herencia. Ésta se manifiesta cuando se tiene una clase que es una especialización de otra que se encuentra en un nivel superior en la jerarquía de clases. La clase subordinada (en un nivel inferior) hereda los métodos, mensajes y atributos de la clase superior.

Cuando una clase que hereda de otra necesita incluir un atributo extra, éste se le asocia, y no afecta a la clase superior. Por otro lado, es común que en una clase que constituye una especialización, se desee modificar alguno de los método heredados para adecuar la respuesta a algún mensaje. En este caso, se modifica el comportamiento de esta clase, y el método nuevo se le asocia.

15.3 Identidad de Objeto

En este paradigma no necesitamos detectar los atributos que compondrán el identificador de un objeto. Aquí cada objeto conserva su *identidad* independientemente de su estado.

El *estado* de un objeto está dado por los valores que en un momento tengan sus atributos. En el caso de las otras técnicas, identificábamos un elemento de la realidad en base al valor de algún conjunto de sus atributos, por ejemplo el rut de una persona o la patente de un auto. Aquí podemos utilizar el mismo criterio para fines de búsqueda de información, pero a diferencia de los sistemas como los relacionales, aunque modifiquemos el rut de un objeto persona, éste seguirá siendo el mismo objeto.