MODELOS Y SISTEMAS DER 7 Año

Profesor: Omar Alejandro Bazar

UNIDAD 02

MODELO ENTIDAD RELACIÓN

El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real consistente en objetos básicos llamados entidades y de relaciones entre estos objetos.

Se desarrolló para facilitar el diseño de bases de datos permitiendo la especificación de un esquema de la empresa que representa la estructura lógica completa de una base de datos.

El modelo de datos E-R es uno de los diferentes modelos de datos semánticos; el aspecto semántico del modelo yace en la representación del significado de los datos.

El modelo E-R es extremadamente útil para hacer corresponder los significados e interacciones de las empresas del mundo real con un esquema conceptual. El modelo de datos entidad-relación (E-R) está basado en una percepción del mundo real consistente en objetos básicos llamados entidades y de relaciones entre estos objetos.

Se desarrolló para facilitar el diseño de bases de datos permitiendo la especificación de un esquema de la empresa que representa la estructura lógica completa de una base de datos.

El modelo de datos E-R es uno de los diferentes modelos de datos semánticos; el aspecto semántico del modelo yace en la representación del significado de los datos.

El modelo E-R es extremadamente útil para hacer corresponder los significados e interacciones de las empresas del mundo real con un esquema conceptual.

OBJETOS BÁSICOS DEL MODELO E-R

Los conceptos básicos previstos por el modelo ER son entidades, relaciones y atributos.

ENTIDADES Y CONJUNTO DE ENTIDADES

Una ***entidad*** es un objeto que existe y puede distinguirse de otros objetos. La entidad puede ser concreta, por ejemplo: una persona o un libro; o abstracta, por ejemplo, un día festivo o un concepto.

Un ***conjunto de entidades*** es un grupo de entidades del mismo tipo. El conjunto de todas las personas que tienen una cuenta en el banco, por ejemplo, puede definirse como el conjunto de entidades clientes.

Una entidad está representada por un conjunto de ***atributos.*** Los posibles atributos del conjunto de entidades clientes son nombre, documento, calle y ciudad. Para cada atributo existe un rango de valores permitidos, llamado ***dominio*** del atributo. El dominio del atributo nombre podría ser el conjunto de todos los nombres de personas de cierta longitud.

RELACIONES Y CONJUNTO DE RELACIONES

Una ***relación*** es una asociación entre varias entidades. Por ejemplo, es posible definir una relación que asocia al cliente Gutiérrez con la cuenta 401.

Un ***conjunto de relaciones*** es un grupo de relaciones del mismo tipo. Se definirá el conjunto de relaciones *clientecuenta* para denotar la asociación entre los clientes y las cuentas bancarias que tienen.

La relación *clientecuenta* es un ejemplo de una ***relación binaria***, es decir, una que implica a dos conjuntos de entidades.

Existen conjuntos de relaciones que incluyen a n-conjuntos de entidades, ***relaciones n-arias***, por ejemplo, la relación ternaria *cliecuentasuc* que especifica que el cliente Gutiérrez tienen la cuenta 401 en la sucursal Córdoba.

Las ***relaciones recursivas (Reflexivas)*** son relaciones binarias que conectan una entidad consigo misma.

Una relación también puede tener ***atributos descriptivos o rótulos***. Por ejemplo, fecha podría ser un atributo del conjunto de relaciones *clientecuenta*. Esto especifica la última fecha en que el cliente tuvo acceso a su cuenta.

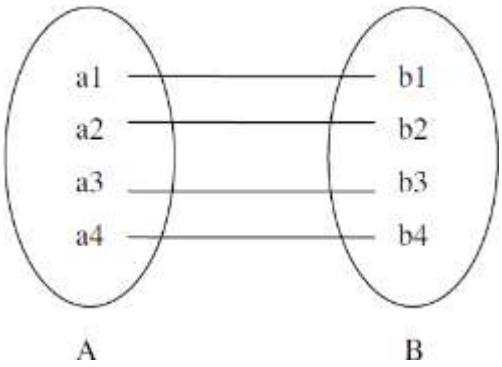
CARDINALIDADES DE MAPEO

Un esquema ER empresarial puede definir ciertas limitantes con las que deben cumplir los datos contenidos en la base de datos. Una limitante importante es la de las ***cardinalidades de mapeo*** que expresan el número de entidades con las que puede asociarse otra entidad mediante una relación.

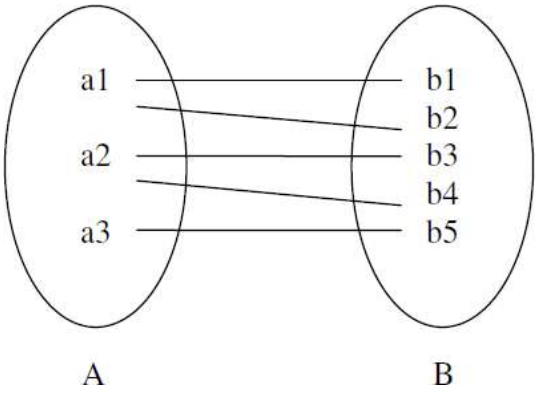
Las cardinalidades de mapeo son más útiles al describir conjuntos binarios de relaciones, aunque también son aplicables a conjuntos n-arios de relaciones.

Para un conjunto binario de relaciones R entre los conjuntos de entidades A y B, la cardinalidad de mapeo puede ser:

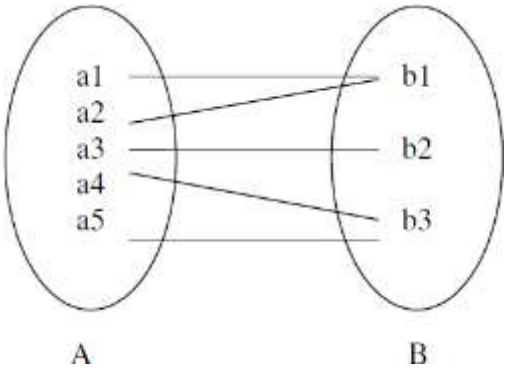
Una a una: una entidad de A está asociada únicamente con una entidad de B y una entidad de B está asociada solo con una entidad de A.



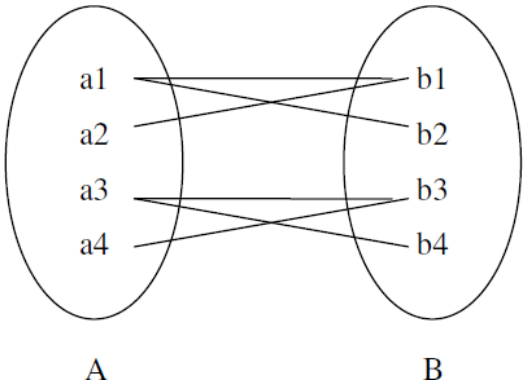
Una a muchas: una entidad en A está asociada con varias entidades de B, pero una entidad de B puede asociarse únicamente con una entidad de A.



Muchas a una: una entidad de A está asociada únicamente con una entidad en B, pero una entidad de B está relacionada con varias entidades de A.



Muchas a muchas: una entidad en A está asociada con varias entidades de B y una entidad en B está vinculada con varias entidades de A.



Para ilustrar lo anterior, considérese el conjunto de relaciones *clientecuenta*. Si en un banco dado una cuenta puede pertenecer únicamente a un cliente y un cliente puede tener varias cuentas, entonces el conjunto de relaciones *clientecuenta* es una a muchas, de cliente a cuenta. Si una cuenta puede pertenecer a varios clientes, entonces el conjunto de relaciones *clientecuenta* es una a muchas, de cuenta a cliente, entonces en definitiva el conjunto de relaciones *clientecuenta* es muchas a muchas.

Las ***dependencias de existencia*** constituyen otra clase importante de limitantes. Si la existencia de la entidad x depende de la existencia de la entidad y, entonces se dice que x es dependiente por existencia de y.

Funcionalmente esto quiere decir que, si se elimina y, también se eliminará x. Se dice que la entidad y en una entidad dominante y que x es una entidad subordinada. Por ejemplo, supongamos que tenemos los conjuntos de entidades cuenta y transacción. Se forma la relación *cuentatransac* entre estos dos conjuntos es decir que para una cuenta determinada pueden existir varias transacciones. Esta relación es una a muchas de cuenta a transacción. Cada entidad transacción debe estar relacionada con una entidad cuenta. Si se elimina una entidad cuenta, entonces deben eliminarse también todas las entidades transacción vinculada con esa cuenta. Por lo contrario, pueden eliminarse entidades transacción de la base de datos sin afectar ninguna cuenta. Por lo tanto, el conjunto de entidades cuenta es dominante y transacción es subordinada en la relación *cuentatransac*.

CLAVES PRIMARIAS

Una tarea muy importante dentro de la modelación de bases de datos consiste en especificar cómo se van a distinguir las entidades y las relaciones. Conceptualmente, las entidades individuales y las relaciones son distintas entre sí, pero desde el punto de vista de una base de datos la diferencia entre ellas debe expresarse en términos de sus atributos. Para hacer estas distinciones, se asigna una ***clave primaria*** a cada conjunto de entidades, ésta es un conjunto de uno o más atributos que, juntos, permiten identificar en forma única a una entidad dentro del conjunto de entidades. Por ejemplo: el atributo documento del conjunto entidades cliente es suficiente para distinguir a una entidad cliente de otra, por lo tanto, puede ser la clave primara de ese conjunto de entidades.

Es posible que un conjunto de entidades no tenga suficientes atributos para formas una clave primaria. Por ejemplo: el conjunto entidades transacción tiene tres atributos: numtransac, fecha e importe. Aunque cada entidad transacción es distinta, dos transacciones hechas en cuentas diferentes pueden tener el mismo número de transacción, entonces el conjunto entidades transacción no tienen una clave primaria. Una entidad de un conjunto de este tipo se denomina ***entidad débil*** y una entidad que puede tener una clave primaria recibe el nombre de **entidad fuerte**. El concepto de entidades fuertes y débiles está relacionado con el de “dependencia por existencia”.

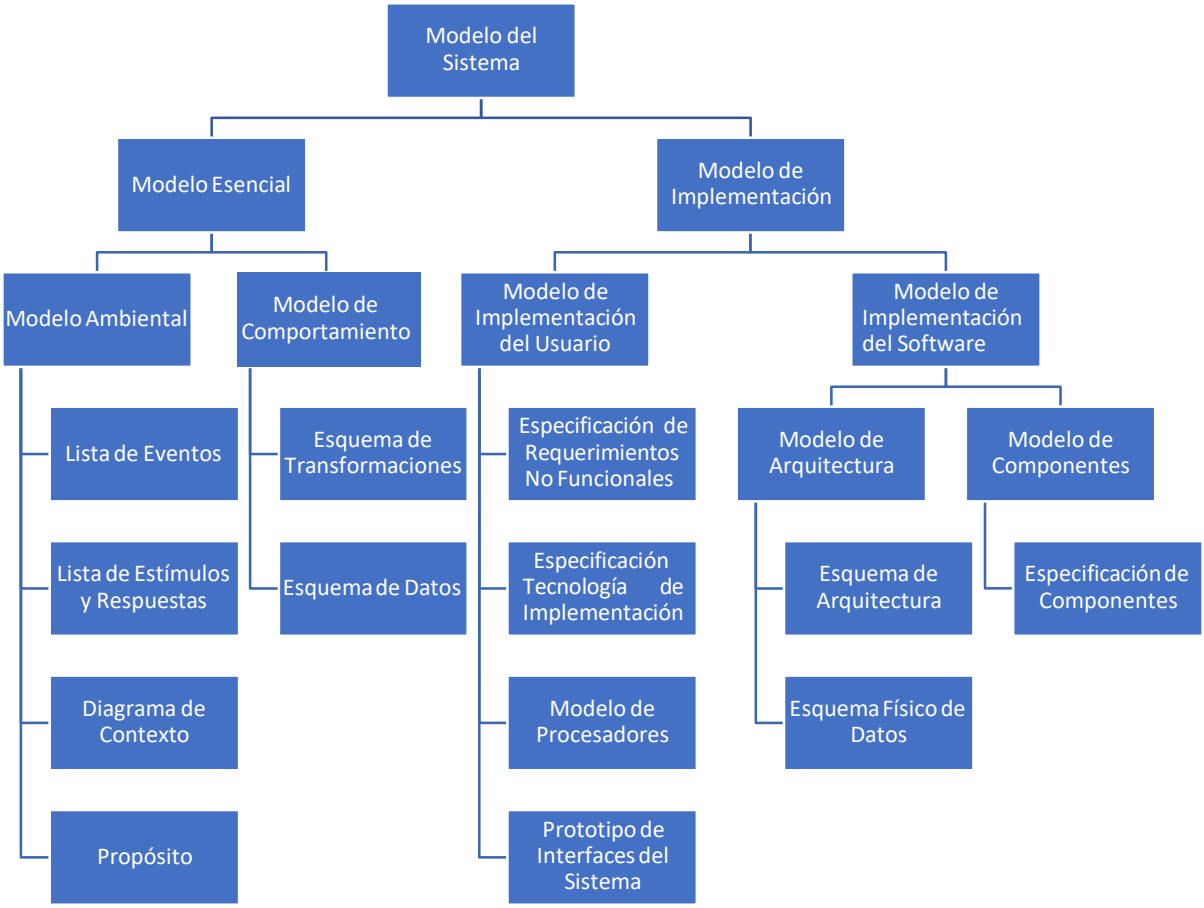
Un conjunto de entidades débiles no tiene una clave primaria sin embargo es preciso tener una forma de distinguir entre todas las entidades del conjunto, aquella que depende de una entidad fuerte de otro conjunto relacionado. El **discriminador** de un conjunto de entidades débiles es un conjunto de atributos que permite hacer esta distinción. Por lo tanto, para nuestro ejemplo el discriminador es numtransac ya que para cada cuenta estos números identifican en forma única cada una de las transacciones.

La clave primaria de un conjunto de entidades débiles está formada por la clave primaria de la entidad fuerte de la que depende su existencia y su discriminador. En el caso del conjunto de entidades transacción, su clave primaria es (numcuenta, numtransac), donde numcuenta identifica a la entidad dominante de una transacción y numtransac distingue a las entidades transacción dentro de la misma cuenta.

Los conjuntos de relaciones también tienen claves primarias. Sus claves primarias se forman tomando todos los atributos que constituyen las claves primarias de los conjuntos de entidades que definen el conjunto de relaciones. Por ejemplo: documento es la clave primaria de cliente y numcuenta es la clave primaria de cuenta. Por lo tanto, la clave primaria del conjunto de relaciones clientecuenta es (documento, numcuenta).

METODOLOGÍA ESTRUCTURADA

Dentro de la metodología estructurada, el Modelo Entidad Relación se encuadra en el Esquema de Datos, y el Esquema de Datos es parte del Modelo de Comportamiento, que a su vez es parte del Modelo Esencial, como podemos ver en el siguiente cuadro.



ESQUEMA DE DATOS

OBJETIVO:

Descripción de los datos que el sistema debe conocer (recordar) para poder responder a los estímulos. Es una visión pasiva del sistema y necesita mostrar:

- Las relaciones entre los datos, que no pueden mostrarse en los almacenamientos del DFD, pues generan requerimientos falsos. Las especificaciones de procesos del DFD muestran todas las relaciones entre los objetos, mediante los accesos esenciales; esos accesos **pueden eliminarse** de la especificación, ya que serán expresados en este esquema.
- Detalle de los datos: Identificadores y atributos descriptivos.

HERRAMIENTAS:

Las herramientas usadas son:

- Diagrama de Entidad Relación (conocido como DER), con extensiones de súper/sub tipo y tipos de objetos asociativos. A lo largo del texto hablaremos indistintamente de objetos y tipos de objetos, que serán equivalentes a las entidades del DER; de la misma manera, asumiremos que atributo y elemento de dato tienen el mismo significado.

- Diccionario de datos, con la siguiente composición:

  DD = {Objeto} + {Objeto_Asociativo} + {Objeto_Débil} + {Super_Tipo} + {Relación} + ({Estructura_de_Datos}) + {Atributo}

TÉCNICAS:

CONSTRUCCIÓN DEL ESQUEMA DE DATOS

Esta técnica es el resultado de aplicar criterios vertidos por variados autores. Los resultados que se obtienen son mucho más "naturales" que los obtenidos con otras técnicas.

El aspecto datos es más estable que el aspecto funcional, en la mayoría de los sistemas; también es mucho más difícil *pensar* el esquema de datos. La mayor dificultad se presenta en establecer la estructura de los objetos, y las relaciones entre los mismos; muchas veces decimos que es posible afirmar que un DER está mal, pero no puede asegurarse que un DER esté completamente bien.

1) Identificar Objetos: Un objeto es la representación de una cosa de existencia real o artificial, que interesa al sistema. Puede ser algo tangible, un rol desempeñado por una persona u organización, un incidente o una interacción. Es importante destacar que no siempre habrá una correspondencia uno a uno entre los objetos del Esquema de Datos y los objetos del mundo real (a los que hemos anteriormente llamado “cosas”). Muchos objetos reales son complejos -es decir, poseen una estructura- y están formados por otros objetos. Pensemos, por ejemplo, en un auto con sus diferentes partes. Si nos interesa registrar información de las distintas componentes del automóvil, no quedará otra alternativa que representarlos como distintos objetos en el DER.

2) Individualizar identificadores únicos de objetos: Un identificador es un atributo que confirma la existencia de un objeto dado, y la identidad de las distintas instancias del mismo. Si no puede encontrarse un identificador, o el objeto posee una sola instancia, este elemento **NO ES UN OBJETO**. Un objeto será **débil** si no es identificable por sus propios atributos y que para poder serlo requiere de uno o más atributos externos.

3) Identificar relaciones entre objetos: Una relación es una asociación esencial de la memoria para permitir los accesos esenciales y mejorar la descripción de los objetos. Generalmente las relaciones se implementan a través de atributos referenciales (Si, por ejemplo, fuésemos a implementar este esquema en una base de datos relacional, las relaciones se implementarían a través de claves foráneas). Ya que aquí estamos en la esencia, NO DEBEN utilizarse tales atributos para representar asociaciones entre objetos (por ejemplo, no debería incluirse el atributo Código de Cliente en el objeto Factura - siempre y cuando exista el objeto Cliente-). Las asociaciones se representan únicamente a través de relaciones.

4) Clasificar relaciones: Según los conceptos de:

    a. **Grado:** es la cantidad de objetos que intervienen en la relación (unaria, binaria, etc.);

    b. **Conectividad:** mapa de la asociación entre objetos (1:1, 1:N, M:N).

    c. C**ondicionalidad:** indica si la participación de cada uno de los objetos en la relación es obligatoria u opcional, es decir, si existen o no instancias de los objetos intervinientes en la relación.

5) Identificar atributos (elementos de datos que el sistema maneja).

6) Asignar atributos a objetos y relaciones, materializando así la relación intra-objeto, según ese atributo describa una característica del objeto, y/o dependa funcional y no transitivamente del identificador del objeto/relación.

7) Identificar objetos asociativos. Un objeto asociativo es un elemento que se comporta como relación y como objeto al mismo tiempo. Para que exista una instancia del mismo, **deben existir instancias de todos los objetos que relaciona.**

8) Identificar súper/sub-tipos, agrupando objetos que posean atributos comunes y alguna condición de diferenciación.

9) Dibujar el DER, según la notación de la herramienta.

10) Eliminar elementos redundantes o fuera del alcance del sistema.

11) Generar una entrada en el Diccionario de Datos por cada elemento del DER.

12) Validar aplicando:

    a. **Normalización**, con las observaciones indicadas más adelante.

    b. Técnica de preguntas y respuestas.

13) Revisar el esquema con el resto del modelo.

DIAGRAMA DE ENTIDAD RELACIÓN (DER)

INTRODUCCIÓN

Un sistema puede ser descripto de distintas formas. El sistema no cambia; lo que estamos haciendo es verlo desde distintos puntos de vista, priorizando un aspecto diferente del sistema cada vez. Mediante el DER priorizamos el ***aspecto datos***.

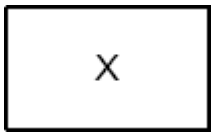
En el DFD hay caminos; éstos muestran el fluir de los datos, es el sistema en funcionamiento. Podemos ver que es lo que está pasando. En el DER también hay caminos, pero éstos representan las **relaciones** entre los distintos tipos de información que maneja el sistema; aquí no fluyen datos como en un DFD.

El DER representa los **datos almacenados** en un sistema presentado como una red de almacenamientos conectados por relaciones; es una vista estática, se ve al sistema como una entidad pasiva.

El DER de un sistema es más resistente al cambio que un DFD. Por ejemplo, en una organización, las políticas de crédito a un cliente pueden cambiar (funciones), pero la entidad *Cliente* sigue existiendo como tal. El DER depende del ambiente, y, por lo tanto, es menos factible que cambie.

CONVENCIONES

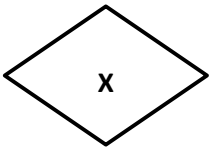
TIPO DE OBJETO (ENTIDAD):



Rectángulo

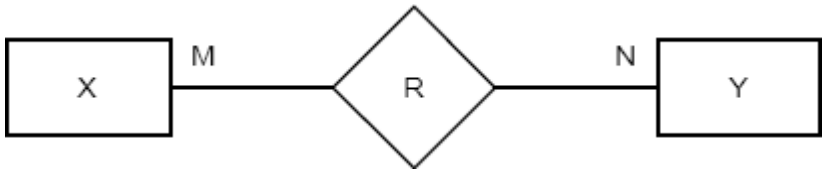
Es representado mediante un rectángulo con nombre. Agrupa bajo un mismo nombre un conjunto de cosas del mismo tipo pertenecientes al mundo real. Este objeto juega un rol significativo en el sistema a ser descripto, por ejemplo, *Clientes*, *Artículos*, etc.

RELACIÓN:



Rombo

Es representada mediante un rombo con nombre. Es el resultado de algún proceso del mundo real que vincula tipos de objetos que participan en ese proceso. Una relación es una abstracción porque ésta no describe el proceso, únicamente describe la forma en que las entidades se combinan. Sólo se define una relación entre dos o más tipos de objetos si dicha relación está basada en políticas, reglas o leyes que interesen al sistema que se modela.

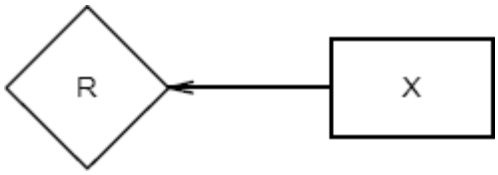


Las relaciones son típicamente multidireccionales, es decir, X está relacionado con Y y viceversa.

Se dice que X está relacionado con Y cuando desde una ocurrencia (**instancia**) de X puedo obtener la/s ocurrencia/s de Y que le corresponden.

El **orden** de la relación es **M:N**. Por ejemplo, si X es *Departamento*, Y es *Empleado*, y sabemos que un empleado trabaja en un solo departamento, y que en un departamento trabajan muchos empleados, el orden de la relación R (*Trabaja*) es **1:N**.

TIPO DE OBJETO ASOCIATIVO:

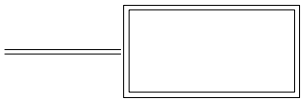


Rectángulo unido a un Rombo

Es representado mediante un rectángulo unido a un rombo. Surge cuando una relación contiene además información (Agregación / Asociación).

Este tipo de objeto es una relación que además posee atributos propios. Este tipo de objeto sólo existe mientras existan los objetos que relaciona.

OBJETOS O ENTIDADES DÉBILES:

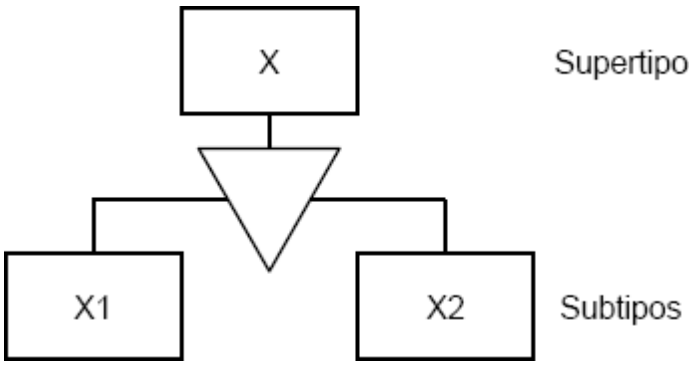


Rectángulo con línea doble

Es representado mediante un rectángulo con línea doble. La línea de relación hacia el objeto fuerte también se dibuja doble.

Un objeto que posee identificadores internos que determinan unívocamente a cada una de sus instancias es un OBJETO FUERTE. Un objeto que deriva su existencia a partir del conjunto de atributos identificadores de otro u otros objetos es un OBJETO DÉBIL. Dichos atributos también se conocen como atributos externos.

SUPERTIPO Y SUBTIPOS



El **Supertipo** es representado como un rectángulo vinculado a sus subtipos por medio de un triángulo. Es el resultado de tratar una clase de objetos similares como un nuevo tipo de objeto (Generalización / Clasificación). (Ej.: *Vehículo* es un supertipo compuesto de *Auto* y *Camión*).

El **Subtipo** es una entidad. Es el resultado de tratar un subconjunto de algún tipo de objeto como un nuevo tipo de objeto (Diferenciación funcional).

Consideraciones Prácticas:

Para determinar si el modelo es completo, una buena forma de hacerlo es escribir todas las preguntas a las que debe responder el sistema y verificar si las relaciones del modelo permiten obtener las respuestas. Así como un DFD es erróneo cuando no produce una salida a partir de una entrada, un DER es erróneo si no puede responder a una pregunta de interés del sistema.

Hay que tener mucho cuidado con el nivel de detalle bajo el cual se agrupan los datos. No debe ser ni muy detallado ni muy general.

Lo importante es representar las interconexiones entre las lógicas principales dentro del área de interés del sistema.

- De ser muy detallado, nos damos cuenta cuando dos tipos de objetos están descriptos por los mismos atributos y tienen las mismas relaciones (Ej.: *Vendedores_Internos* y *Vendedores_Externos* que son tratados de la misma forma).
- De ser muy general nos damos cuenta cuando debemos identificar bajo qué condiciones son válidos determinados subconjuntos de atributos de un tipo de objeto. (Ej.: tener definido como entidad *Cajas_de_Ahorro* que incluye como subtipos *Caja_Ahorro_Común* y *Caja_Ahorro_Especial*, y cada una de éstas debe tener atributos específicos distintos).

Reglas de Conexión:

- Un tipo de objeto puede o no estar conectado y si lo está, puede ser a una o más relaciones.
- Una relación debe conectarse a uno o más Objetos.
- Un objeto no puede estar conectado directamente a otro.
- Una relación no puede estar conectada directamente a otra.

Reglas de Consistencia Interna:

- No puede haber tipos de objetos con el mismo nombre.
- Se debe tener un identificador único para cada tipo de objeto, relación, instancia de objeto e instancia de relación.
- No incluir relaciones irrelevantes para el sistema.
- Eliminar relaciones que no puedan existir en el mundo real.
- Eliminar relaciones que son redundantes.
- Un DER es consistente si puede proveer todos los datos requeridos en la aplicación de la técnica pregunta-respuesta.

Reglas de Claridad Semántica:

- No incluir atributos irrelevantes para el sistema.
- Todos los atributos de una entidad que son relevantes para el sistema deben ser incluidos.

- Si un objeto sólo tiene su identificación como atributo, quizás sea conveniente eliminarlo e incluir la información en otra Entidad.
- Agrupar en súper / subtipos los objetos que dependen de relaciones idénticas.

MISCELÁNEAS

Es necesario incluir en el DD la información de todos los elementos del DER del sistema. A continuación, detallamos las convenciones a utilizar (que es un grupo de los tantos posibles).

Agregaremos un ***"@"*** a los atributos que sirvan como identificación al tipo de objeto (*Clave*), y ***"ref"*** a dichos atributos cuando figuran en una relación.

TIPO_DE_OBJETO	=	Significado + @Identificador + {Atributos}
TIPO_DE_OBJETO_ASOCIATIVO	=	Significado + @Identificador-Ref-N + @Identificador-Ref-M + {Atributos}
SUPERTIPO	=	Significado + Atributos_Comunes + {Subtipos}
SUBTIPO	=	* Idem Tipo de Objeto *
RELACIÓN	=	Significado + [@Identificador-Ref-1 + @Identificador-Ref-1 * Caso 1:1 * | @Identificador-Ref-1 + @Identificador-Ref-N * Caso 1:N * | @Identificador-Ref-M + @Identificador-Ref-N * Caso M:N *]

CONVENCIONES

Notación para describir la composición de los elementos del DD:

=	... está compuesto por ...
+	... y ...
[|]	o exclusivo.
{ }	repetición (en la llave izquierda desde; en la derecha, hasta).
( )	opcional.
* *	comentario.

NORMALIZACIÓN EN LA CONSTRUCCIÓN DEL ESQUEMA DE DATOS.

Para detectar objetos y para eliminar redundancias, podemos valernos de alguna de las técnicas de **normalización**. El problema consiste en que la mayoría de dichas técnicas tienen como objetivo construir un **esquema procesable**, introduciendo factores tecnológicos que, a este nivel, **están prohibidos**. Uno de estos factores es el generado por la primera forma normal, que impide la existencia de grupos repetitivos.

Para poder solucionar estos problemas, se sugiere lo siguiente:

- Utilizar una técnica de normalización que permita trabajar con relaciones anidadas (o sea, elementos repetitivos).
- Utilizar una técnica de normalización tradicional y luego volver a desnormalizar las relaciones anidadas.

En ambos casos se deberá testear con el usuario si los nuevos objetos o relaciones son válidos para el dominio de información del sistema. Además, se aplique o no una técnica de normalización, el concepto de **dependencia funcional** es muy útil para la construcción del Esquema.

EJEMPLO

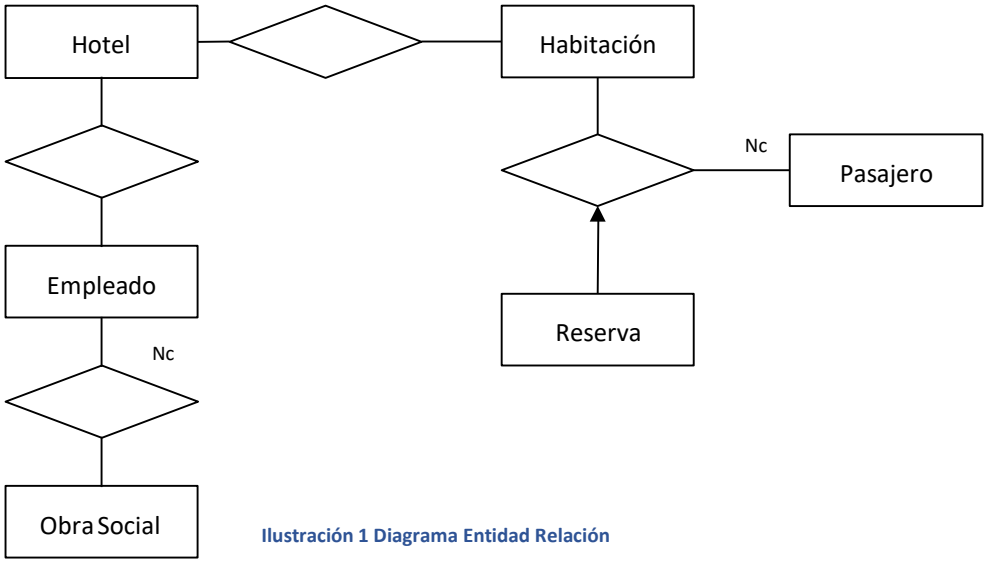


Ilustración 1 Diagrama Entidad Relación

Tabla 1 Ejemplo de Diccionario de Datos

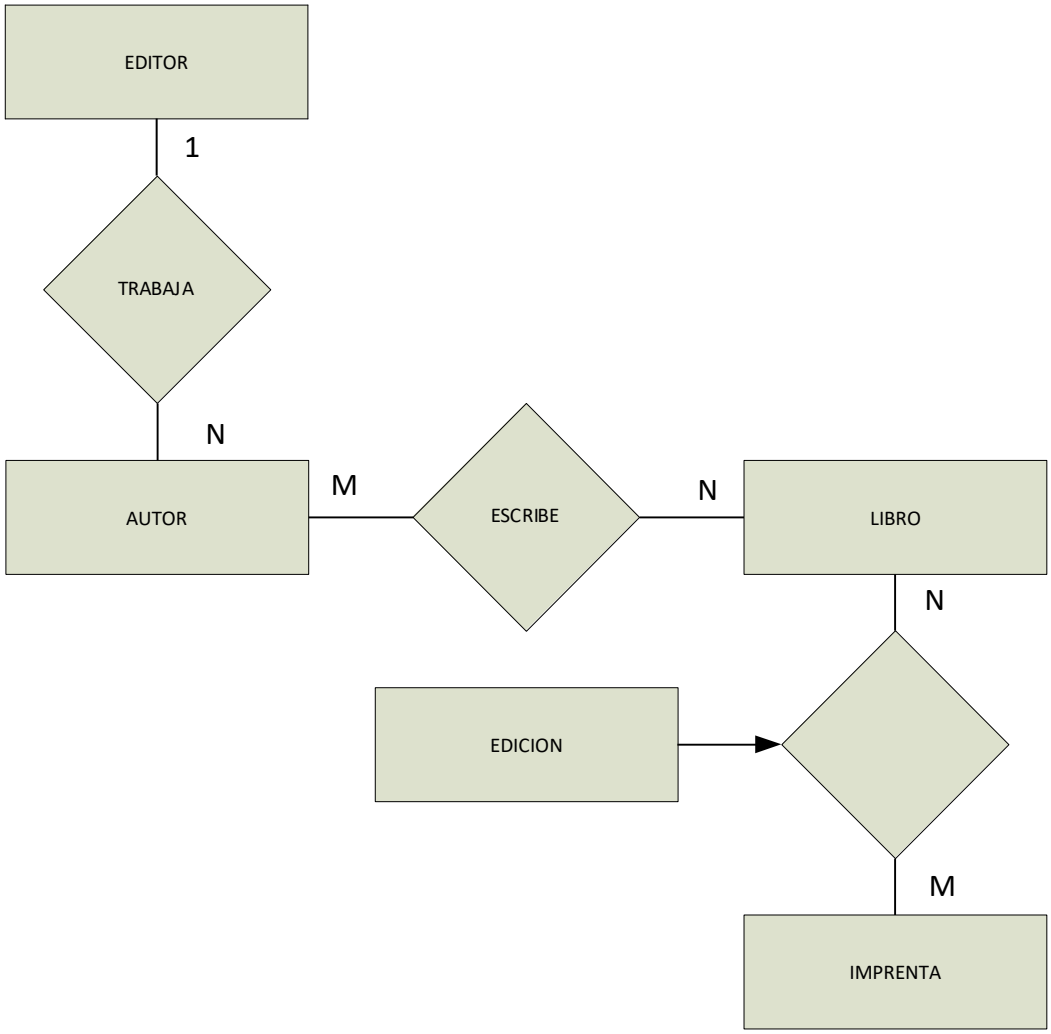
Cuenta	=	Empleado_ref_Nc + Obra_Social_ref_1
Empleado	=	@Legajo + Nombre + Sueldo
Habitación	=	@Identificador + Número + Ubicación + Capacidad + Categoría
Hotel	=	@Nombre + @Ciudad
Obra_Social	=	@Nombre + ImporteDesde + ImporteHasta
Pasajero	=	@ID + TipoDoc + NroDoc + Nombre + NroTarjetaCredito
Reserva	=	@ID_Reserva + FechaDesde + FechaHasta + NroPersonas + Pasajero_ref_Nc + Habitación_ref_M
Tiene	=	Hotel_ref_1 + Habitación_ref_N
Trabaja	=	Hotel_ref_1 + Empleado_ref_Nc

ANEXO A: TÉCNICA DE PREGUNTAS Y RESPUESTAS

La técnica de preguntas y respuestas consiste en verificar si las respuestas a una lista de preguntas propuestas por el usuario se encuentran contempladas en el esquema de datos construido.

EJEMPLO:

La editorial ABC trabaja con varios y diferentes autores quienes escriben los libros que serán publicados. Algunos autores escriben solo un libro, mientras los otros han escrito varios. La editorial también trabaja con varias imprentas. Distintas ediciones de un libro pueden ser realizadas por distintas imprentas. Se desea guardar la fecha de cada edición y la cantidad de ejemplares editados en la misma. Un editor de esta compañía trabaja con varios autores a la vez, editando y produciendo los libros; es el trabajo del editor preparar la última copia revisada pasada a máquina y lista para ser impresa.



DICCIONARIO DE DATOS

AUTOR	=	@id_autor + nombre + dirección + tel. + ...
EDICION	=	LIBRO-ref-N + IMPRENTA-ref-M + fecha edición + cant.ejemplares

EDITOR	=	@id_editor + nombre + …
IMPRENTA	=	@id_imprenta + dirección + tel + ...
LIBRO	=	@ISBN + título + género

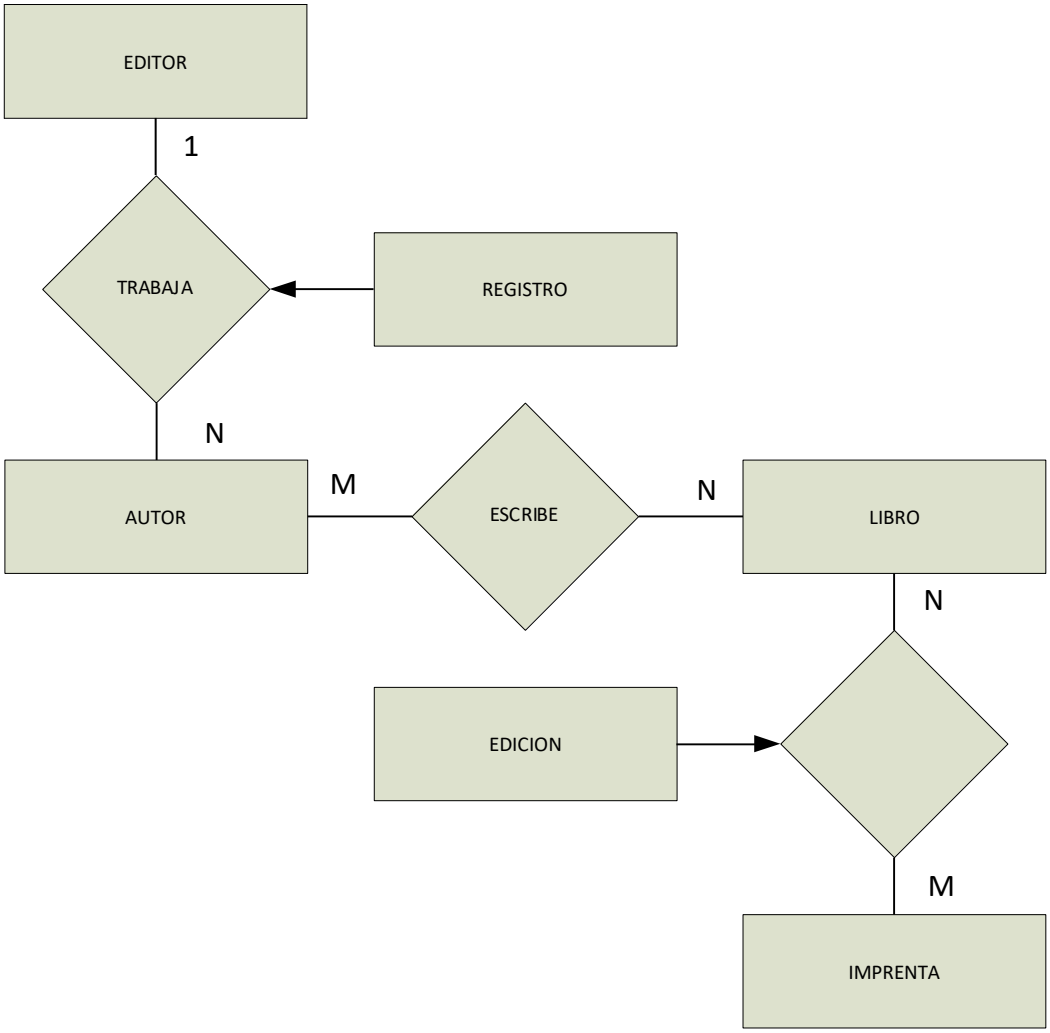
Ahora se aplica la técnica de preguntas y respuestas. El usuario desea saber:

1)Qué libros editó cada editor en un determinado momento. 2)
   a. Que libros escribió cierto autor,
   b. en un determinado periodo
3)
   a. Cuál es el autor de un determinado libro,
   b. Dónde y cuándo nació dicho autor.
4)En qué imprenta se imprimió cierto libro. 5)
   a. Cuántos ejemplares de un cierto libro se imprimieron en una imprenta.
   b. en una cierta fecha o período.
6)Con qué editor trabaja cada autor en este momento. 7)
   a. Con qué editor trabajó cierto autor en una determinada fecha.
   b. Cuántas veces trabajó con dicho autor y por cuánto tiempo.
8) Con qué imprentas se relaciona un editor, por los libros con los que está trabajando.
9) Cuál es el género en el que se especializa ahora cada editor.

En la forma en que está construido el esquema de datos, hay preguntas que no se pueden responder, por ejemplo:

- Para contestar la pregunta 2b, debería agregarse en el diccionario de datos de LIBRO, la fecha en la cual el mismo se realizó.
- Para contestar la pregunta 3b, debería agregarse en el diccionario de datos de AUTOR, lugar y fecha de nacimiento.
- Para contestar las preguntas 6, 7a, 7b, y 8, debería agregarse en un objeto asociativo entre EDITOR y AUTOR que contenga el identificador de cada uno de ellos más un conjunto repetitivo que incluye la fecha de comienzo del trabajo más la duración.
- Para contestar la pregunta 9, debería agregarse en el diccionario de datos de EDITOR, el o los géneros en que se especializa.

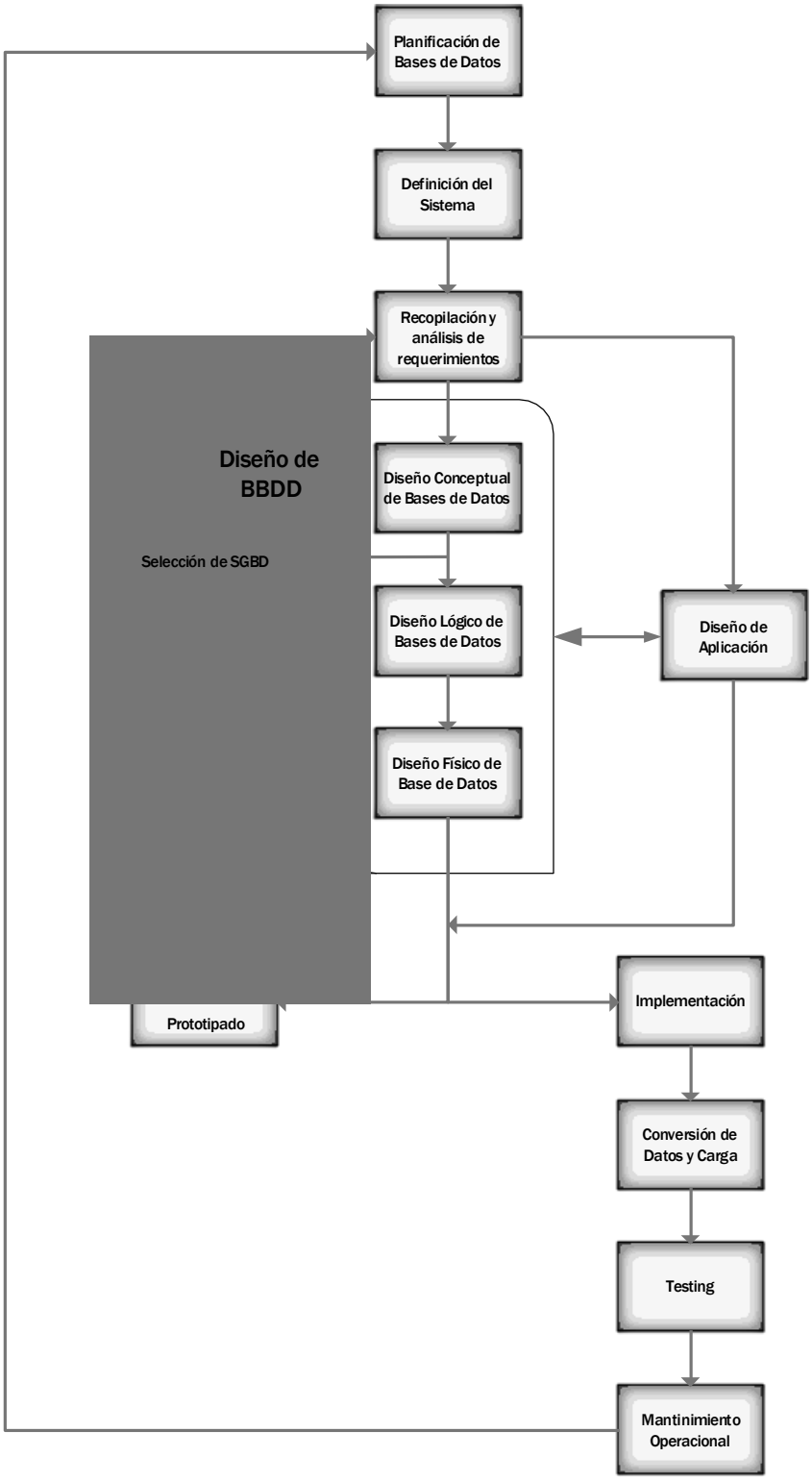
Luego de estas modificaciones se obtendría:



DICCIONARIO DE DATOS

AUTOR	=	@id_autor + nombre + dirección + tel. + fecha_nac +lugar_nac
EDICION	=	LIBRO-ref-N + IMPRENTA-ref-M + fecha edición +cant.ejemplares
EDITOR	=	@id_editor+ nombre + {género}
IMPRENTA	=	@id_imprenta + dirección + tel + ...
LIBRO	=	@ISBN + título + género + fecha_escritura
REGISTRO	=	AUTOR-ref-M + EDITOR-ref-1 + fecha_comienzo_trabajo + duración

ANEXO B: CICLO DE VIDA CENTRADO EN LOS DATOS[17]



[17] Database Systems - A Practical Approach to Design, Implementation, and Management – Chapter 10