4.2.4 Características de un Esquema Conceptual de Base de Datos

En este apartado se da una respuesta a la pregunta: ¿qué es un buen esquema conceptual?

Entre las cualidades deseables de un esquema conceptual están las siguientes:

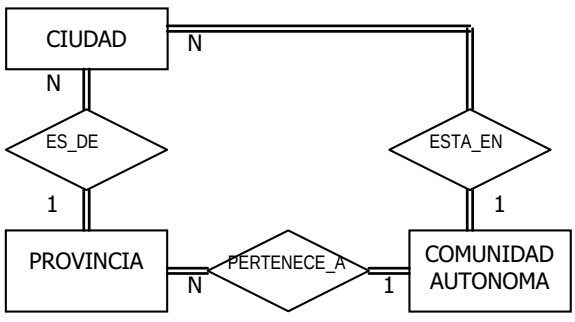
1. **Completitud.** Un esquema es completo cuando representa todas las características propias e importantes del dominio de aplicación y permite la ejecución de toda operación incluida en los requisitos operacionales. La completitud puede, en principio, comprobarse:
   - observando con detalle todos los requisitos del dominio de aplicación y verificando que cada uno de ellos está representado en algún lugar del esquema. Si es así, se dice que *el esquema es completo respecto a los requisitos*, y
   - revisando el esquema, para comprobar que cada concepto es mencionado en los requisitos. Si es así, se dice que *los requisitos son completos respecto al esquema.*
   - Verificando que todos los conceptos involucrados en cada operación pueden ser alcanzados navegando a través del esquema conceptual.

2. **Corrección.** Un esquema es correcto cuando usa adecuadamente los elementos del modelo EER. Un esquema es sintácticamente correcto cuando los conceptos se expresan adecuadamente en el esquema; por ejemplo las generalizaciones se definen entre tipos de entidad, y no entre interrelaciones. Un esquema es semánticamente correcto cuando los elementos (entidades, interrelaciones, etc.) se usan de acuerdo con sus definiciones. La siguiente lista muestra los errores semánticos más frecuentes:
   a. Usar un atributo en lugar de una entidad.
   b. Olvidar una generalización, o la propiedad de herencia de las generalizaciones.
   c. Usar una interrelación con un número erróneo de entidades participantes (por ejemplo, una interrelación binaria en lugar de una ternaria).
   d. Usar una entidad en lugar de una interrelación.
   e. Omitir alguna especificación de cardinalidad mínima o máxima.
   f. Usar una interrelación para expresar que una entidad es especialización de otra.

3. **Minimalidad.** Un esquema es mínimo cuando cada aspecto de los requisitos aparece sólo una vez en el esquema. También se puede decir que un esquema es mínimo si no se puede borrar un concepto del esquema sin perder alguna información. Es decir: si no existe redundancia.

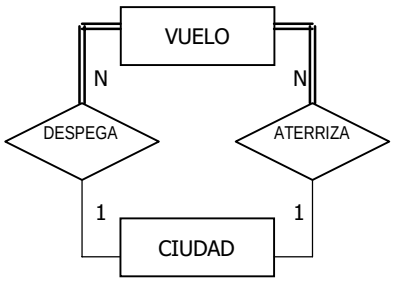
   Un esquema contiene **redundancia** cuando incluye por ejemplo, dos elementos que representan el mismo concepto, y por tanto uno de ellos podría ser eliminado del esquema sin perder información (el esquema no es mínimo). Las siguientes son algunas fuentes de redundancia:

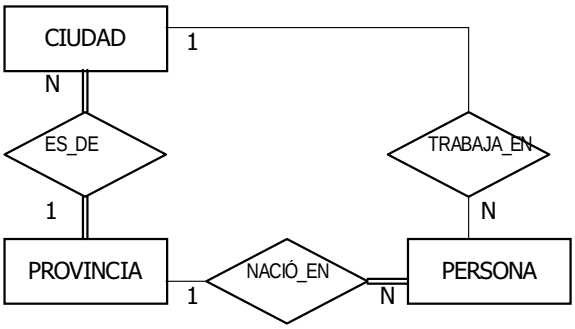
   a) Ciclos de interrelaciones. Cuando una interrelación R1 entre dos entidades posee el mismo contenido de información que una ruta de interrelaciones (R2, R3 ... Rn) que conecta exactamente los mismos pares de ocurrencias de entidades que R1, se tiene redundancia.
   En el ejemplo de la derecha, la interrelación ESTA_EN es redundante, ya que sus ocurrencias se pueden derivar a partir de ES_DE y PERTENE-



CE_A (una ciudad está en la comunidad autónoma a la que pertenece su provincia), por ello debería ser eliminada del esquema.
Cuando un ciclo contiene varias interrelaciones mutuamente redundantes, es posible eliminar cualquiera de ellas.

No todos los ciclos de interrelaciones son fuentes de redundancia: lo serán o no dependiendo de su significado.
En el ejemplo de la derecha, existen dos interrelaciones sobre las mismas entidades pero con significados completamente diferentes: no forman un ciclo, por no poseer el mismo contenido de información.



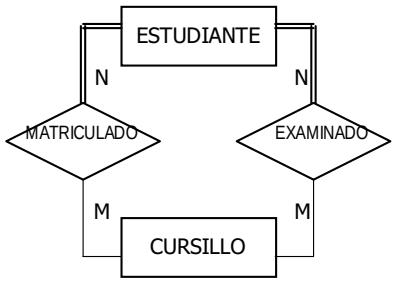


En el ejemplo de la izquierda, aunque existe un ciclo no existe redundancia, pues una persona no tiene por qué haber nacido en la misma provincia en la que está la ciudad donde trabaja.

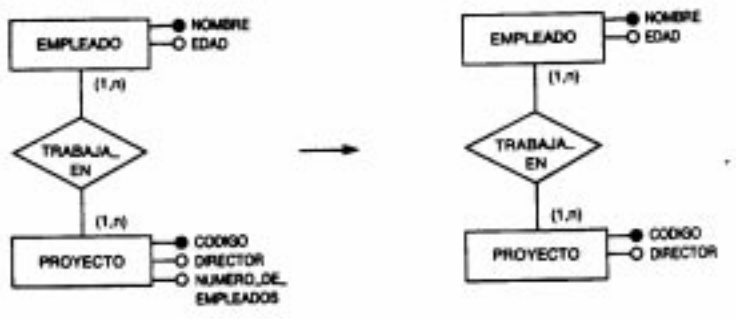
A pesar de que la redundancia en un ciclo depende del significado (semántica), es posible realizar comprobaciones sintácticas para detectar si existe redundancia en un ciclo de interrelaciones.

Por un lado, puede estudiarse la equivalencia entre las (dos) ramas del ciclo. Por ejemplo, si la interrelación que se obtiene al combinar dos o más interrelaciones 1:1 es también una interrelación 1:1, nunca puede ser equivalente a una interrelación 1:N, N:1 o N:M. De manera similar, la interrelación obtenida al combinar dos o más interrelaciones 1:N es también 1:N (es el caso del ejemplo de las ciudades, provincias y comunidades), de modo que no puede equivaler a una interrelación 1:1, N:1 o N:M. En todos los demás casos, no se puede deducir la cardinalidad de la interrelación combinada; por ejemplo, la combinación de una interrelación 1:N con una interrelación N:1 puede producir cualquier tipo de interrelación (puede ser 1:1,1:N, N:1 o N:M).

Por otro lado, los ciclos pueden ocultar algún tipo de *restricciones de contención*, que también es síntoma de redundancia. En el ejemplo, ESTUDIANTE y CURSILLO están relacionados por las interrelaciones binarias MATRICULADO y EXAMINADO; a pesar de que las interrelaciones no son semánticamente equivalentes (y por tanto el ciclo no es redundante), se observa que existe la restricción de que un estudiante no puede ser examinado en un curso en el que no está matriculado. Esto significa que la interrelación EXAMINADO está contenida en la interrelación MATRICULADO. Este tipo particular de redundancia puede evitarse, incluyendo un atributo booleano (*examinado*) en la interrelación MATRICULADO y eliminando EXAMINADO.



b) Atributos derivados. La redundancia puede deberse a la existencia de un atributo cuyo valor puede calcularse a partir de los valores de otros atributos del esquema; de ahí que los atributos derivados puedan omitirse en un esquema EER mínimo. Los atributos derivados pueden ser extremadamente útiles para mejorar la eficiencia de una base de datos; este criterio decidirá, en última instancia, la conveniencia de almacenar datos derivados durante un diseño lógico (siguiente etapa en el proceso de diseño de una BD). Se recomienda incluir los datos derivados redundantes en el esquema conceptual, pero indicando con claridad las reglas pertinentes para su cálculo.

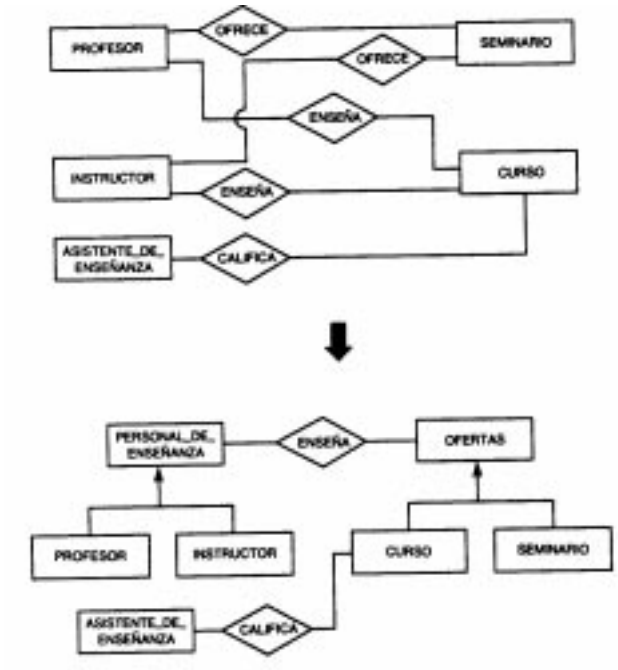


En resumen, es tarea del diseñador decidir si acepta la redundancia en el esquema conceptual o la elimina. En cualquier caso, la redundancia puede ser fuente de anomalías en los datos (se verá en temas posteriores); por ello, debe estar claramente indicada en el esquema.

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

Para mejorar la expresividad es posible realizar las siguientes acciones:

a) Utilizar nombres adecuados, que reflejen el significado de los elementos del esquema (nada de denominar dos interrelaciones con TIENE1 y TIENE2).



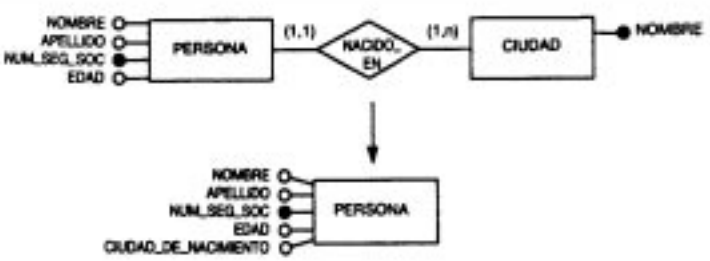
b) Crear generalizaciones. Esta transformación se aplica cuando se descubre que dos entidades distintas con propiedades similares, pertenecen a una misma jerarquía de generalización / especialización. La adición de una generalización proporciona sencillez y concisión al esquema resultante, mediante el uso de la propiedad de herencia.

(aplicación de una primitiva ascendente)

c) Eliminar subtipos de entidad "colgantes" en las jerarquías de generalización. Puede suceder que durante el proceso de elaboración del EC, el diseñador haya creado una jerarquía de generalización. Si al final del proceso de diseño, y debido a los diferentes refinamientos y reestructuraciones del esquema, los subtipos no se distinguen por ninguna propiedad específica, pueden reducirse al supertipo. La diferencia entre los distintos tipos de entidad se expresa mediante un atributo.

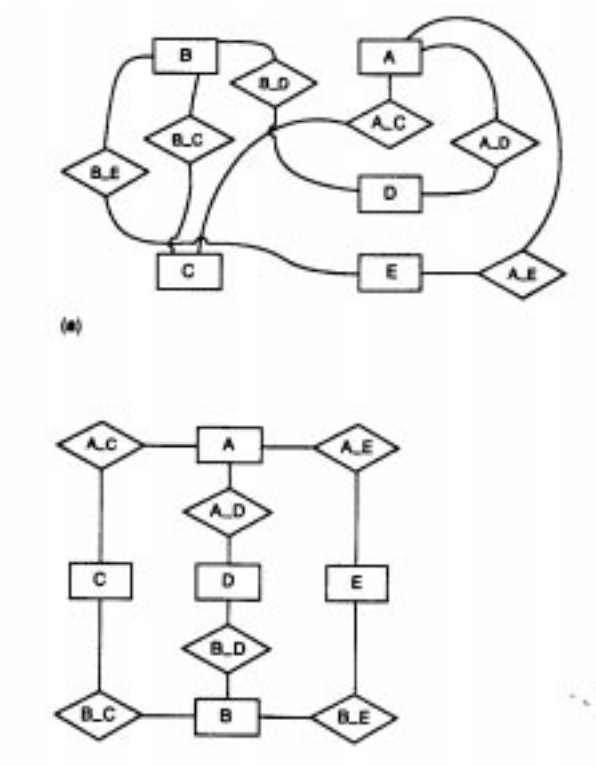
Esto se puede aplicar también cuando los subtipos tienen sólo unas cuantas propiedades distintivas; en este caso, el supertipo adquiere todas las propiedades de los subtipos,

d) Eliminar de entidades colgantes. Una entidad E es *colgante* si posee pocos atributos, posiblemente sólo uno *A*, y una conexión con otra entidad (llamada *principal)* a través de una interrelación R; en este caso, es conveniente simplificar el esquema eliminando la entidad colgante y la interrelación, pasando los atributos A de la entidad colgante a la principal,



como se muestra en el ejemplo de la derecha. Las cardinalidades mínima y máxima del nuevo atributo *A* en la entidad principal pueden calcularse como combinación de las cardinalidades de la entidad colgante E en la interrelación R y del atributo original *A* dentro de la entidad E.

Las dos últimas acciones producen esquemas más simples y generan una típica situación de conflicto entre la expresividad y la autoexplicación (ver a continuación) del esquema resultante: un esquema compacto es a veces más comprensible que uno más grande; sin embargo, al fusionar entidades se puede perder precisión en la descripción de sus propiedades.



5. Legibilidad. Esta es una propiedad del diagrama que representa gráficamente al esquema. Un diagrama es legible cuando respeta ciertos criterios estéticos que hacen el diagrama elegante. Los principales criterios son los siguientes:

a. Los diagramas deben trazarse en hojas cuadriculadas, para que los cuadros que representan entidades y los rombos que indican interrelaciones tengan aproximadamente el mismo tamaño y las conexiones sean trazos horizontales y verticales (sin curvas).

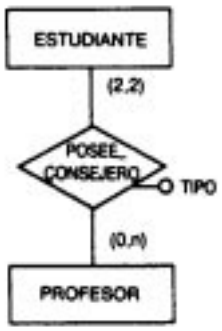
b. Se debe tender a construir estructuras simétricas.

c. Se minimiza el número global de cruces de líneas (los cruces frecuentes disminuyen el "ancho de banda de percepción" del lector).

d. Debe minimizarse el número global de esquinas a lo largo de las líneas de conexión.

e. En las jerarquías de generalización / especialización, el tipo de entidad padre debe situarse por encima de los subtipos (hijos), y los subtipos deben situarse simétricamente respecto al supertipo.

6. Autoexplicación. Un esquema se explica a sí mismo cuando representa un gran número de propiedades usando las construcciones del modelo conceptual en sí, sin usar otros formalismos (por ejemplo, anotaciones en lenguaje natural).



(a)

(b)

Como ilustración de un esquema que no es autoexplicativo, considérese la representación de estudiantes y sus tutores (consejeros) de máster y de doctorado.

Cada estudiante posee como máximo un tutor de máster y uno de doctorado, y el mismo estudiante puede (en momentos distintos) ser tanto estudiante de máster como de doctorado.

Esta restricción no puede representarse por completo en el esquema, porque ningún concepto del modelo permite declarar que «si un objeto estudiante pertenece a dos ocurrencias de la interrelación POSEE_CONSEJERO, el atributo *tipo* debe adoptar dos valores distintos (figura (a)).

Si en cambio, se usan dos interrelaciones diferentes entre ESTUDIANTE y PROFESOR, se podrá imponer la restricción, definiendo los valores adecuados de las cardinalidades mínima y máxima de las interrelaciones (figura (b)).

7. Extensibilidad. Un esquema es extensible si se adapta fácilmente a los cambios en los requisitos, y esto se consigue si se puede descomponer en partes (módulos o vistas), a fin de aplicar los cambios dentro de cada parte.

4.3 DISEÑO DE TRANSACCIONES

Esta etapa debe realizarse en paralelo, y de forma coordinada, con la fase de diseño del esquema conceptual, hasta lograr un diseño estable de EC y el modelo de procesos (es decir, que ambos diagramas sean mutuamente consistentes y completos).

El objetivo es diseñar las características de los procesos o aplicaciones[1] conocidos que se realizarán sobre los datos. Es importante especificar las características funcionales de estas **transacciones** lo antes posible dentro del proceso de diseño. De esta forma, el esquema de la BD seguro que incluirá toda la información requerida por dichas transacciones[2].
Normalmente sólo se conocen algunas de las transacciones en el momento del diseño, extraídas de los requisitos de los usuarios, pero suelen ser las más importantes.

Las transacciones pueden agruparse en tres categorías:
Transacciones de obtención, para obtener datos; por ejemplo, si estamos diseñando un sistema de reservas de vuelos, una transacción de este tipo sería obtener un listado de todos los vuelos nocturnos entre dos ciudades en un día determinado.
Transacciones de actualización, para introducir datos nuevos, modificar los existentes o eliminarlos; por ejemplo la reserva de una plaza en un vuelo determinado.
Transacciones mixtas, para obtener y actualizar datos en una misma transacción; por ejemplo buscar las reservas de ciertos clientes en un vuelo, para después, anular dichas reservas.

Usualmente, se utilizan técnicas para especificar los **procesos** del sistema (y obtener el *modelo de procesos* del sistema). Sea cual sea la técnica usada, una especificación conceptual de transacciones (independiente del sistema) incluirá la identificación, para cada transacción, de a) datos de entrada y de salida, y b) su comportamiento funcional (flujo de control interno).

BIBLIOGRAFÍA BÁSICA

[ACPT99] Atzeni, P., Ceri, S., Paraboschi, S, Torlone, R.: **Database Systems: Concepts, Languages and Architectures.** McGraw-Hill, 1999.

[EN97] Elmasri, R.; Navathe, S.B.: **Sistemas de bases de datos. Conceptos fundamentales**. Addison-Wesley Iberoamericana. 2ª Edición, 1997.

[dMP93] De Miguel, A.; Piattini, M.: **Concepción y diseño de bases de datos**: **del Modelo E/R al Modelo Relacional**. RA-MA, 1993.

---

[1] Algunos autores, como ocurre en [EN97], utilizan el término ***transacciones***. Podemos considerar este concepto como *englobado* dentro del concepto más general de *aplicación*, *proceso*, *función*, ... cuando se refieren a operaciones complejas que se realizarán sobre la información almacenada en una base de datos, una vez implementado el sistema. De hecho, un proceso es una operación compleja que puede consistir en una o varias transacciones.

[2] Además, conocer la importancia de las transacciones, la frecuencia con la que se realizarán, el rendimiento con el que se espera que se ejecuten, etc. influye de forma importante en las decisiones de diseño físico de la base de datos (se verá más adelante).