MODELADO DE DATOS Y MODELO RELACIONAL

ESTRUCTURA DE MODELO RELACIONAL

Conceptos Básicos

Base de datos:

- Conjunto de información organizada sistemáticamente, no redundante y referida a un tema determinado.
- Se almacenan en uno o varios archivos y pueden contener, además de los datos, distintos tipos de objetos como tablas, vistas, índices, diagramas, procedimientos almacenados, funciones, sinónimos, informes, programas, disparadores, etc.

Relación:

- Es el elemento básico del modelo que se representa como una tabla.
- Todas las relaciones se nombran.

Atributos:

- Son cada una de las propiedades de la relación.
- En una tabla representan a las columnas y también se nombran.

Tuplas:

- En la relación son las filas o registros.
- Contienen los valores que toma cada atributo para cada elemento de la relación.

Grado:

- Es el número de atributos, campos o columnas que tiene la relación.

Cardinalidad:

- Es el número de tuplas, filas o registros de una relación.

Cabecera:

- Define la estructura de la relación y es la primera fila que contiene los atributos.

Cuerpo:

- Es el conjunto de tuplas de la relación.

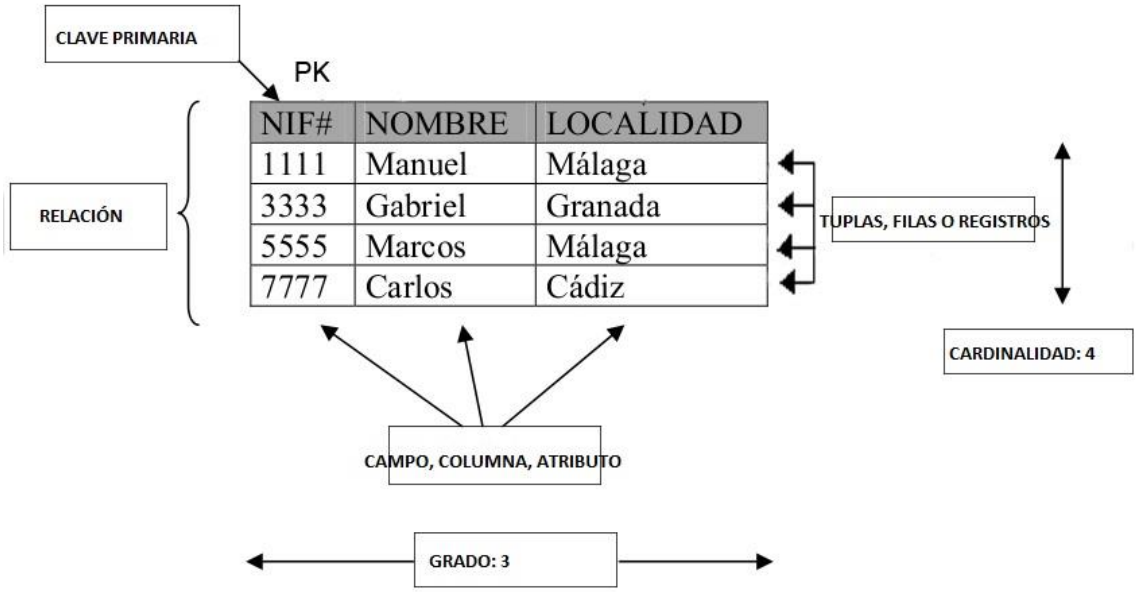
Dominio:

- Es el conjunto de valores homogéneos (del mismo tipo) y atómicos (valores indivisibles) que puede tomar un atributo.

- Los dominios también incluyen las condiciones (restricciones) que deben cumplir los datos.
- Los dominios se nombran y normalmente coinciden con el nombre del campo.
- Hay dominios simples y compuestos, por ejemplo, las fechas se pueden considerar de dos formas, como dominio simple dividido en partes o como un dominio compuesto.

Esquema de relación:

- Está compuesto por el nombre de la relación y su cabecera, pudiendo incluir el dominio.
- Se expresa genéricamente como: R (A1:D1, A2:D2, …) o R (A1, A2, A3), siendo "R" la abreviatura para relación, "A" la de atributo y "D", dominio
- Ejemplo:
    - Morosos (registro, portal, nombre, cantidad, fecha, concepto)



    - El atributo clave principal se subraya.

Estado de la relación u ocurrencia de relación:

- Está constituido por el esquema y el cuerpo de la relación, es decir, por el conjunto de tuplas que puede tener ésta en un momento determinado.

Esquema relacional:

Personas (NIF#, Nombre, Localidad)

TIPOS DE RELACIONES

- **PERSISTENTES.**

- Son aquellas cuyo esquema de relación permanece en la base de datos, deben ser borradas explícitamente.
- Tipos:
  - **Relación base**.
    - Existen por sí mismas, es decir, no son derivadas de otras y tienen contenido propio.
    - No derivada quiere decir que no se ha creado a partir de otra. Se crean especificando su esquema.
  - **Vista.**
    - Son relaciones derivadas.
    - Se definen dando nombre a una expresión de consulta. No almacenan datos, sólo su definición en términos de otras relaciones.
  - **Instantánea.**
    - Son relaciones derivadas, tienen sus propios datos resultado de una consulta o de almacenar otra relación, pero éste contenido es estático, es decir, muestra los datos tal como estaban cuando se creó la instantánea. Si se hace un cambio en la relación base de la que depende (la que las define) la instantánea no se actualizará.
- **TEMPORALES.**
  - Estas relaciones, que tienen contenido, desaparecen en un determinado momento sin la eliminación explícita del usuario.

RESTRICCIONES

- **Concepto**.
  - Estructura u ocurrencia no permitida en una relación.
- **Tipos de restricciones**:
  - **Inherentes.**
    - Son restricciones impuestas por el modelo relacional, como:
      - No puede haber tuplas duplicadas.
      - El orden de las tuplas y de los atributos no es significativo.
      - Cada atributo sólo puede contener un único valor.
      - Toda relación debe estar obligatoriamente normalizada (optimización).
      - **Integridad de entidad.**
        - Ningún atributo que sea la clave primaria puede tomar un valor nulo.
  - **Semánticas o de significado.**
    - Son las que el modelo relacional ofrece a los usuarios para que puedan reflejar fielmente, en el esquema de la relación, el significado del mundo real. Algunas restricciones semánticas son:
      - **Clave primaria (PK).**
        - Es un campo o conjunto de campos que identifican de forma inequívoca cada tupla, registro o fila de la

relación. Cumple dos reglas, una es que no admite duplicados y la otra es que no admite valores nulos.

- **Unicidad (UNIQUE).**
  - Sirve para que los valores de un atributo o conjunto de atributos no puedan repetirse en una relación.
- **Obligatoriedad (NOT NULL).**
  - Hace que uno o varios atributos no contengan valores nulos. Se utiliza en campos en los que la existencia de datos es obligatoria o para campos clave primaria, ya que el campo con PK es obligatoriamente no nulo.
- **Integridad referencial (FK).**
  - Asegura que las tuplas de tablas relacionadas sean válidas, esta restricción se activa cuando se crea una clave foránea e impide que haya datos en la tabla secundaria que no existan en la tabla principal.
- **De rechazo.**
  - Establece las condiciones que se deben cumplir para insertar o utilizar datos.
  - Tipos:
    - **Verificación (CHECK).**
      - Que es cuando la restricción se impone sobre los atributos.
    - **Aserción (ASSERTION).**
      - Son restricciones que afectan a más de un elemento a la vez, como dos tablas relacionadas.
    - **Disparadores o Triggers**
      - Acción que se ejecuta sola al realizarse una operación.

CLAVES

Clave candidata:

- Es un atributo o conjunto de atributos que podría identificar de forma inequívoca a cada tupla de una relación.
- Al menos siempre hay una clave candidata que son todos los atributos (campos).

Clave primaria:

- Es la clave candidata elegida como identificador único de las tuplas.

Clave alternativa:

- Es una clave candidata no elegida como primaria pero que también podría identificar las filas.

Clave simple:

- Es la que está compuesta por un solo atributo.

Clave compuesta:

- Es la que está compuesta por varios atributos.

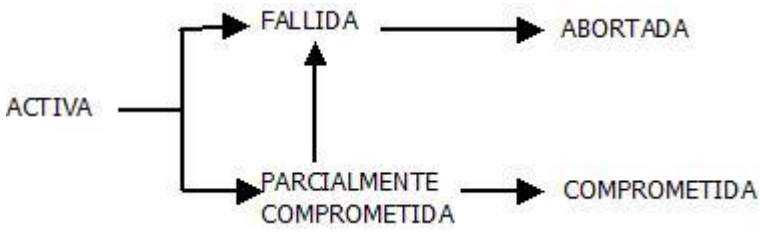
Clave foránea, externa, secundaria o ajena:

- Es un atributo o atributos de una tabla relacionada con otra que tiene la clave principal.

TRANSACCIONES

- **Concepto.**
    - Conjunto de operaciones que realiza un SGBD para llevar a cabo una unidad lógica de trabajo.
- **Características.**
    - Son indivisibles.
    - Se usan en operaciones de modificación de varias tablas, en el acceso simultáneo por parte de varios usuarios a una misma información.
    - En operaciones de concurrencia garantizan la coherencia e integridad de los datos. Los que se modifican quedan bloqueados para otros usuarios hasta que la transacción no finaliza.
    - Hasta que no se termina es posible restaurar los datos a su estado inicial. Los cambios de los datos que se están manipulando serán visibles para otros usuarios al terminar la transacción. Existen transacciones que se validan automáticamente.
- **ACID**.
    - Para que un conjunto de instrucciones sea de verdad una transacción debe cumplir las **características** siguientes:
        - **A: ATOMICIDAD:**
            - Cada transacción es una unidad que se da como un todo, si falla parte de ella, todo falla.
        - **C: CONSISTENCIA:**
            - Una transacción va a llevar la base de datos de un estado valido a otro estado valido.
        - **I: AISLAMIENTO:**
            - Una transacción se ejecuta en aislamiento total e independiente de otra transacción.
        - **D: DURABILIDAD:**
            - Tras validarse una transacción esta es persistente, ya no se puede volver atrás.
- **Estados de una transacción:**
    - **Transacción activa.**

        - Es el estado de una transacción cuando se inicia o durante su ejecución.
    - **Parcialmente comprometida.**
        - Es el estado alcanzado después de ejecutar la última instrucción.
    - **Fallida.**
        - Estado que asume la transacción cuando no se puede continuar la ejecución normal.
    - **Comprometida.**
        - Estado alcanzado cuando una transacción se completa con éxito.
    - **Abortada.**
        - Estado producido al retroceder en la transacción y restablecer la base de datos a un momento anterior al comienzo de la transacción.



FALLOS TÍPICOS EN LAS BASES DE DATOS

- **Transacciones.**
    - **Error lógico**
        - Es una transacción en el sistema gestor de la Base de Datos SGBD. No se ejecutan normalmente debido a errores internos.
        - Datos que no se encuentran.
        - Entradas incorrectas en formato o valor.
        - Desbordamiento de memoria.
    - **Errores del sistema.**
        - El SGBD se encuentra en un estado que impide que las transacciones puedan ejecutarse normalmente.
    - **Interbloqueo.**
        - Bloqueo permanente de un conjunto de procesos que no pueden avanzar, debido a que dependen de recursos o acciones de otros procesos.
- **Caídas del sistema**
    - Errores en el SO.
    - Errores en el Software que gestiona la base de datos.
    - Averías de hardware que provocan la pérdida del contenido de la memoria volátil.
    - Detención del sistema.
- **Técnicas para evitar caídas del sistema.**
    - Replicación de recursos y datos.
    - Balanceo. Repartir cargas.
    - Uso de SAI Sistemas de alimentación ininterrumpidas.

- **Fallos en los soportes de almacenamiento.**
    - Se producen cuando hay un fallo en la transferencia de datos al soporte de almacenamiento, o cuando hay un problema físico en el propio soporte.
    - Como contrapartida tener una buena política de copia de seguridad de datos.

- **Elementos de recuperación ante fallos lógicos que soportan los SGBD.**
    - **Registro Histórico, Registro de transacciones, Archivo de registro o Archivo de log**.
        - Almacena todos los movimientos que se van realizado en la base de datos. Se hacen anotaciones en el Registro Histórico. Es una secuencia de registros que incluye todas las modificaciones realizadas en la Base de Datos. Cada registro describe una única modificación. Suelen contener una serie de atributos:
            - Identificador de la transacción (de tipo numérico)
            - Indicador de elementos de datos
            - Valor anterior (como un punto de recuperación)
            - Valor nuevo.
- **Punto de verificación, comprobación, sincronización o de revisión.**
    - Si hay un fallo se accede al registro histórico hacia atrás hasta encontrar una versión estable de la misma. Si habitualmente vamos guardando puntos de verificación se reduce el número de registros del registro histórico que deben examinarse tras una caída del sistema, ya que al establecer un punto de comprobación se fuerza a que todas las transacciones hasta ese punto se lleven a cabo.

REGLAS DE CODD DEL MODELO RELACIONAL

En 1984 Edgar F. Codd, creador de del Modelo Relacional publicó las **12 Reglas** que un verdadero Sistema Relacional de Bases de Datos debería cumplir. En la práctica algunas de estas reglas son difíciles de implementar, así que un sistema podrá considerarse *más relacional* cuanto más siga estas reglas.

REGLA 0:

- Un SGBD es relacional cuando gestiona las bases de datos usando sólo sus capacidades relacionales.

REGLA 1: REGLA DE LA INFORMACIÓN

- La información o los datos de una base de datos se representan como valores en tablas.
- También se representan en tablas los metadatos (diccionario, catálogo)
- Con un mismo lenguaje (SQL por ejemplo), se debe acceder a ambos.

REGLA 2: REGLA DEL ACCESO GARANTIZADO

- Cualquier dato de una tabla debe ser accesible indicando en qué tabla está, cuál es su columna y cuál es su fila (mediante la clave primaria).

REGLA 3: TRATAMIENTO SISTEMÁTICO DE VALORES NULOS

- Se debe disponer de valores nulos (distintos de la cadena vacía, blancos, 0, etc.) para representar información desconocida o no aplicable de manera sistemática, independientemente del tipo de datos.

REGLA 4: CATÁLOGO DINÁMICO EN LÍNEA BASADO EN EL MODELO RELACIONAL

- Los metadatos se almacenan y se manejan usando el modelo relacional.
- Los metadatos se consultan usando el mismo lenguaje relacional para los datos normales.

REGLA 5: REGLA DEL SUBLENGUAJE DE DATOS COMPLETO

- Debe existir al menos un lenguaje cuyas sentencias sean expresables, mediante una sintaxis bien definida, como cadenas de caracteres y que sea completo, soportando:
    - Definición de datos y estructuras.
    - Manipulación de datos.
    - Restricciones de integridad.
    - Control de datos.
- Estos lenguajes, además de poder tener interfaces más amigables para hacer consultas, etc., deben permitir hacerlo todo de manera textual. Un lenguaje que cumple esto en gran medida es el SQL.

REGLA 6: REGLA DE ACTUALIZACIÓN DE VISTAS

- Todas las vistas que son teóricamente actualizables se pueden actualizar también por el sistema.
- Una vista es una consulta de una tabla que es mostrada como una tabla virtual y, si se guarda, almacena solo la definición de la consulta, pero no los datos.

REGLA 7: INSERCIÓN, ACTUALIZACIÓN Y BORRADO DE ALTO NIVEL

- Igual que las consultas, los datos pueden ser insertados, actualizados y borrados en filas y/o tablas múltiples.
- Alto nivel quiere decir que, no sólo un único registro se verá afectado por las operaciones anteriores, sino varios.

REGLA 8: INDEPENDENCIA FÍSICA DE DATOS

- El modelo relacional es un modelo lógico que oculta las características de su representación física. En el que la forma de almacenar los datos no debe influir en su manipulación.
- Aunque cambie el esquema físico, el esquema conceptual no debe cambiar.

REGLA 9: INDEPENDENCIA LÓGICA DE DATOS

- Se denomina esquema externo a la visión de los datos por parte de los usuarios finales a través de las aplicaciones.
- Aunque se modifique los datos del esquema conceptual, el esquema externo de debe verse afectado.

REGLA 10: INDEPENDENCIA DE INTEGRIDAD

- Una base de datos relacional debe poder definir limitantes de integridad, que también se almacenan en la base de datos.
- Las restricciones inherentes al modelo relacional son:

    - **Integridad de Entidad**: Toda tabla debe tener una clave primaria para identificar inequívocamente cada registro.
    - **Integridad de Dominio**: Toda columna de una tabla contendrá exclusivamente un conjunto de valores válidos.
    - **Integridad Referencial**: Hay que asegurar que los registros de tablas relacionadas sean válidos, para ello se usan claves externas o foráneas

REGLA 11: INDEPENDENCIA DE DISTRIBUCIÓN

- Las mismas órdenes y programas deben ejecutarse igual en bases de datos centralizada que distribuidas.

REGLA 12: REGLA DE LA NO SUBVERSIÓN

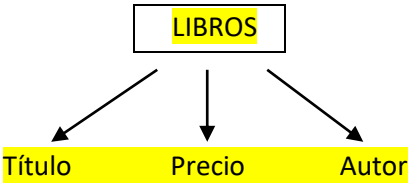
- Un lenguaje de bajo nivel permite sólo gestionar un registro a la vez.
- Si un sistema relacional lo incluye, éste no puede ser usado para saltarse (subvertir) las reglas de integridad y las restricciones expresadas en los lenguajes relacionales de más alto nivel.

DISEÑO DE BASES DE DATOS RELACIONALES

Para crear una base de datos hay que seguir varias fases.

- Análisis de requisitos.
    - Consiste en recabar información sobre el contenido y uso que se piensa dar a la base de datos.
- Diseño conceptual.
    - En esta fase se crea el esquema conceptual de la base de datos independientemente del sistema gestor de base de datos a utilizar. Para el diseño conceptual de base de datos se utiliza el modelo de Entidad Relación E-R.
- Elección de sistema gestor de base de datos.
    - Se elige el modelo de datos y el sistema gestor de base de datos a utilizar.
- Diseño lógico.
    - Se crea el esquema lógico basándose en el esquema conceptual para el modelo de datos del sistema gestor de base de datos elegido.

Esquema conceptual: recopilar información ya estructurada



Esquema lógico:

LIBROS

TITULO	PRECIO	AUTOR
- - -		

- Diseño físico.
    a. Creación de la base de datos usando el lenguaje definición de datos del sistema gestor de base de datos utilizado.
- Fase de uso o utilización.
    a. Se gestionan los datos usando el lenguaje de manipulación de datos.
- Mantenimiento.
    a. Sirve para actualizar, modificar, reparar, etc. las bases de datos.

MODELOS DE DATOS

Según el nivel de abstracción

- Nivel conceptual, nivel lógico y nivel físico.

Según el modelo teórico.

- Red, Jerárquico, Relacional, Orientado a Objetos, Objeto-Relacional, etc.

TIPOS DE B.D.

- Los archivos de texto plano.
- Hojas de cálculo (Excel).
- B.D. jerárquicas.
- B.D. red.

- B.D. relacionales.
- B.D. objeto-relacionales.
- Bases XML, bases que tienen datos semiestructurados.
- Bases No SQL.
- Otras B.D.: B.D. documentales, B.D. temporales, B.D. analíticas, etcétera.

LENGUAJES DE ACCESO A BASE DE DATOS

- **Algebra relacional.**
  - Se basa en la teoría de los conjuntos y es un lenguaje procedimental. El usuario debe indicar la secuencia de operaciones. Del Algebra relacional surge el SQL.
- **Cálculo relacional.**
  - No es procedimental. Hay dos tipos de cálculo, el orientado a tuplas y el orientado a dominios. Del cálculo relacional surge el QBE (QUERY BY EXAMPLE).

ÁLGEBRA RELACIONAL

ALGEBRA RELACIONAL.

- Conjunto de operaciones para obtener un resultado en forma de relación que puede combinarse con otras operaciones para obtener operaciones más complejas y que utiliza un operador distinto para cada una de ellas.

TIPOS DE OPERACIONES.

Operaciones de conjuntos.

- Las operaciones de unión, intersección y diferencia deben cumplir una norma que se denomina compatibilidad respecto a la unión, es decir, las cabeceras deben ser idénticas, con el mismo número de atributos, mismo nombre y pertenecer sus datos al mismo dominio.

Unión U.

- La unión de dos relaciones da otra relación con cabecera idéntica y cuyo cuerpo está formado por todas las tuplas de ambas, en el orden de la unión y sin repeticiones.
- Sintaxis:

R1 U R2

COCHES INYECCIÓN

CODIGO	MARCA	MODELO
002	SEAT	IBIZA GTI
005	SEAT	CÓRDOBA GTI

007	RENAULT	CLIO GTI
009	OPEL	CORSA GTI

COCHES SEAT

CÓDIGO	MARCA	MODELO
001	SEAT	ALHAMBRA GTD
002	SEAT	IBIZA GTI
003	SEAT	LEÓN GTD

COCHES INYECCIÓN U COCHES SEAT

CÓDIGO	MARCA	MODELO
002	SEAT	IBIZA GTI
005	SEAT	CÓRDOBA GTI
007	RENAULT	CLIO GTI
009	OPEL	CORSA GTI
001	SEAT	ALHAMBRA GTD
003	SEAT	LEON GTD

Intersección ∩.

- Cuando se intersecan dos relaciones se obtiene una tercera con la misma cabecera que incluye en su cuerpo, sólo las tuplas comunes a las relaciones.
- Sintaxis:

R1 ∩ R2

COCHES INYECCIÓN ∩ COCHES SEAT

CÓDIGO	MARCA	MODELO
002	SEAT	IBIZA GTI

Diferencia –

- La diferencia entre dos relaciones crea una nueva relación compuesta por todas las tuplas de la primera relación que no aparecen en la segunda. La diferencia no cumple la propiedad conmutativa.
- Sintaxis:

R1 – R2

R2 – R1

COCHES INYECCIÓN – COCHES SEAT

Código	MARCA	MODELO

005	SEAT	CORDOBA GTI
007	RENAULT	CLIO GTI
009	OPEL	CORSA GTI

COCHES SEAT – COCHES INYECCIÓN

CÓDIGO	MARCA	MODELO
001	SEAT	ALHAMBRA
003	SEAT	LEÓN

Producto cartesiano X

- No cumple compatibilidad respecto a la unión, sino respecto al producto. La regla de la compatibilidad respecto al producto, se da entre dos relaciones si los nombres y número de sus atributos no coinciden. El producto de dos relaciones crea una nueva relación con todos los atributos de las relacionadas y cuyo cuerpo contiene la combinación de todas sus tuplas.
- Sintaxis:

R1 X R2

ALUMNOS X ASIGNATURAS

ASIGNATURAS

CODIGO	NOMBRE ASIGNATURA
01	INGLÉS
02	QUÍMICA
03	LITERATURA

ALUMNOS

NOMBRE
ANA
LUIS

Daría como resultado la siguiente tabla: ALUMNOS X ASIGNATURAS

NOMBRE	CÓDIGO	NOMBRE ASIGNATURA
ANA	01	INGLÉS
ANA	02	QUÍMICA
ANA	03	LITERATURA
LUIS	01	INGLÉS
LUIS	02	QUÍMICA
LUIS	03	LITERATURA

Operaciones Relacionales:

Selección σ

- La selección de las tuplas de una relación da como resultado otra relación con la misma cabecera, pero sólo con las tuplas que cumplan la condición.
- Sintaxis:

  σ CONDICIÓN (R)
- La condición consta del atributo del campo, del operador y del valor.

  σ CAMPO OPERADOR DATO (RELACIÓN)

  TIENDA

CÓDIGO	NOMBRE	DEPARTAMENTO	CANTIDAD
001	ANA	INFORMÁTICA	50
002	LUIS	INFORMÁTICA	80
003	EVA	VENTAS	100
004	JUAN	VENTAS	200
005	LUIS	INFORMÁTICA	50

- Ejemplos:
  - Las ventas de Luis:
    - σ NOMBRE='LUIS' (TIENDA)
  - Los que han vendido más de 100 unidades:
    - σ CANTIDAD>=100(TIENDA)
  - Los que han vendido en informática 50 unidades o menos:
    - σ (DPTO='INFORMATICA' (TIENDA)) AND (σ CANTIDAD <=50 (TIENDA))
- **OPERADORES**.
  - =, >, <, >=, <=, <>.
  - BETWEEN……AND.
  - LIKE / NOT LIKE.
  - IN (VALOR1, VALOR2, VALOR3….) / NOT IN (VALOR1, VALOR2, VALOR3….).
  - AND, OR, NOT.
  - IS NULL / IS NOT NULL.

Proyección π

- La proyección de una relación muestra otra relación cuya cabecera sólo contiene los atributos indicados y cuyo cuerpo muestra las tuplas restringidas a dichos atributos.
- Sintaxis:

  π A1, A2, …… AN (RELACIÓN)

  ΠA1, A2, …. (SELECCIÓN DE UNA RELACIÓN)

- Ejemplo:
  - Mostrar código y modelo de los coches que valgan más de 15.000 euros:

COCHES

CÓDIGO	MARCA	MODELO	PRECIO
001	SEAT	IBIZA	15.000
002	OPEL	ASTRA	18.000

Π CÓDIGO, MODELO (σ PRECIO>15.000 (COCHES))

Reunión.

- Simbología:
  - **|X| * Θ**
- Tipos:
  - **Natural**.
    - La unión N**atural** de dos relaciones da como resultado otra relación que contiene las tuplas de ambas con el mismo valor en atributos comunes.
    - Se utiliza mucho en tablas relacionadas mediante clave primaria y clave foránea para buscar registros enlazados.
    - Sintaxis:

R1 * R2

COCHES

CÓDIGO	MODELO
001	IBIZA
002	CÓRDOBA
003	ASTRA

CLIENTES

DNI	NOMBRE	MODELO
111	LUIS	IBIZA
222	ANA	ASTRA
333	EVA	IBIZA
444	ANGEL	FIESTA
555	ANTONIO	IBIZA
666	JUAN	ASTRA

COCHES * CLIENTES

Obtendría una tabla con las cabeceras de ambas pero no repetidos los campos comunes.

CÓDIGO	MODELO	DNI	NOMBRE
001	IBIZA	111	LUIS
001	IBIZA	333	EVA
001	IBIZA	555	ANTONIO

003	ASTRA	222	ANA
003	ASTRA	666	JUAN

- **Reunión Theta**.
  - Sirve para unir relaciones en base a una condición.
  - Sintaxis:

$$R1 \ |x|_{CONDICIÓN\ N}\ R2$$

  - Ejemplo:

$$VENDEDORES \ |x|_{VENDEDORES.CODVEN=PRODUCTOS.CODVEN}\ PRODUCTOS$$

CLIENTES

DNI	NOMBRE	MODELO
111	LUIS	IBIZA
222	ANA	ASTRA
333	EVA	IBIZA
444	ANGEL	FIESTA
555	ANTONIO	IBIZA
666	JUAN	ASTRA

COCHES

CÓDIGO	MODELO
001	IBIZA
002	CÓRDOBA
003	ASTRA

  - Para saber que cliente tiene un Ibiza, primero se deben unir las tablas:
    - COCHES * CLIENTES.
    - Las tablas se unen por los campos comunes.

CODIGO	MODELO	DNI	NOMBRE
001	IBIZA	111	LUIS
001	IBIZA	333	EVA
001	IBIZA	555	ANTONIO
003	ASTRA	222	ANA
003	ASTRA	666	JUAN

$$COCHES \ |x|_{COCHES.MODELO=CLIENTES.MODELO}\ CLIENTES$$

- Con la reunión Theta si se mostraría el campo modelo repetido.

- Cuando llevan un igual se llama EQUIRREUNION.

CLIENTES

DNI	NOMBRE	MODELO	FECHA COMPRA	
111	LUIS	IBIZA	1/2/85	NO
222	ANA	ASTRA	2/10/91	SI
333	EVA	IBIZA	3/8/87	SI
444	ANGEL	FIESTA	10/05/99	--
555	ANTONIO	IBIZA	3/9/88	SI
666	JUAN	ASTRA	3/10/92	NO

COCHES

CÓDIGO	MODELO	FECHA SALIDA AL MERCADO
001	IBIZA	1/2/85
002	CÓRDOBA	2/7/90
003	ASTRA	3/10/92

- Mostrar los coches que se compraron el mismo día que salieron al mercado:
  - (RESULTADO EN ROJO EN LA TABLA CLIENTES):

COCHES |X| COCHES.FECHASALIDA=CLIENTES.FECHACOMPRA CLIENTES

- **Externa**
  - Es una variante de la reunión natural en la que se incluyen las tuplas que no tienen correspondencia en las relaciones a unir. Las tuplas sin correspondencia se rellenan con nulos.
  - Tipos:
    - **Por la izquierda o Left.**
      - Se tienen en cuenta todas las tuplas de la primera relación.
      - Sintaxis:

Sintaxis: R1 ⟕ R2

R1 *LEFT R2

    - **Por la derecha o Right**.
      - Se tienen en cuenta todas las tuplas de la segunda relación.
      - Sintaxis:

Sintaxis: R1 ⟖ R2

R1 *RIGHT R2

    - **Completa o full**.

- Tiene en cuenta las tuplas de ambas relaciones.
  - Sintaxis:

Sintaxis: R1 ⊠ R2

R1 *FULL R2

División ÷

- Se emplean dos relaciones que se denominan dividendo(R1) y divisor (R2), el número de atributos de la relación dividendo debe ser mayor que los de la relación divisor, los atributos de la relación divisor deben estar incluidos en la relación dividendo.
- La división de dos relaciones da otra relación cuya cabecera contiene los atributos de dividendo no incluidos en divisor y cuyo cuerpo contiene los valores de los atributos de dividendo correspondiente con divisor.
- Sintaxis:

R1÷R2

COCHES

CÓDIGO	MODELO
001	IBIZA
002	CÓRDOBA
003	ASTRA

COCHES SEAT

MODELO
IBIZA
TOLEDO

COCHES ÷ COCHES SEAT

CÓDIGO
001

- El campo común, modelo, no sale en el resultado

Proyección generalizada π

- Permite el uso de operaciones aritméticas dentro de operaciones de proyección. Es una mezcla de proyección y ampliación.
- Sintaxis:

π A1, A2, A3, … EXPRESIÓN DE CÁLCULO AS CAMPO NUEVO (r)

- Ejemplo:

Π NOMBRE, PRECIO/KG*1,10 AS INCREMENTO (ϬNOMBRE= 'NUBES' (CHUCHERIAS))

Operaciones Adicionales:

Ampliación α

- Sirven para crear una relación que incluirá un campo o atributo nuevo cuyos valores proceden de una expresión de cálculo.
- Sintaxis:

R α EXPRESIÓN DE CÁLCULO (NOMBRE ATRIBUTO NUEVO)

- Ejemplos:

VENTAS

CÓDIGO	CANTIDAD	PRECIO
001	20	80
002	30	20

VENTAS α CANTIDAD*PRECIO (TOTAL)

CÓDIGO	CANTIDAD	PRECIO	TOTAL
001	20	80	1600
002	30	20	600

- Aumentar un 10% al precio:
  - **VENTAS α total*1,10(INCREMENTO)**
  - Incremento es una nueva columna para el resultado del cálculo.

Resumen Ω

- Equivalen a las consultas de totales.
- Crean una relación cuya cabecera contiene los atributos especificados más uno nuevo cuyos valores son resultado de una función de agregado.
- Funciones de agregado:
- Suma, media, máximo, mínimo, contar, total.
- Sintaxis:

R (LISTA ATRIBUTOS) Ω FUNCIÓN (ATRIBUTO CÁLCULO) (NOMBRE NUEVA COLUMNA)

- Ejemplos:

MOROSOS

NOMBRE	CANTIDAD
LUIS	10
LUIS	20
ANA	10
JUAN	80
LUIS	10
ANA	30
JUAN	20

- Total debido por persona.

MOROSOS (NOMBRE) Ω SUM (CANTIDAD) (TOTAL)

NOMBRE	TOTAL
LUIS	40
ANA	40
JUAN	100

- Total sin agrupar por nombres: el campo de agrupamiento se deja vacío pero hay que poner los paréntesis.

MOROSOS () Ω SUM (CANTIDAD) (TOTAL)

TOTAL
180

División generalizada ÷

- Elimina la restricción de la división normal de que el grado de dividendo debía ser mayor que el del divisor.
- El grado es el número de campos.
- Con la división generalizada se obtiene una nueva relación cuya cabecera contiene los atributos no comunes de ambas relaciones y cuyos valores tienen correspondencia a través de atributos comunes.

COCHES (DIVIDENDO)

Código	Modelo
10	IBIZA
20	CORSA

CLIENTES (DIVISOR)

CÓDIGO	NOMBRE
10	LUIS

RESULTADO DIVISIÓN GENERALIZADA

MODELO	NOMBRE
IBIZA	LUIS

Cambio de nombre:

Renombrar ρ

- Lo que hace es crear otra relación en la cual se cambian los nombres de uno, alguno o todos los atributos.
- Sintaxis:

  R ρ ATRIBUTOS NUEVOS (ATRIBUTOS ORIGINALES)
- Ejemplos:
  - Cambiar de El nombre al atributo DNI por el nuevo de NIF:

    CLIENTES ρ NIF (DNI)
  - Si queremos cambiar el nombre a varios atributos se separan éstos con comas.

    TIENDAS ρ cod, nombretienda (codchuche, nombre)

Asignación:

Asignación ←

- Permite asignar el valor o resultado de una expresión algebraica a una variable, relación, atributo, etc.
- Sintaxis:

  ETIQUETA ←EXPRESIÓN
- Ejemplo:

  COCHESECONOMICOS ← σ PRECIO < 5000 (COCHES)

Modificación BD:

Inserción.

- Añade nuevos registros a la relación.

- Se expresa con una operación de unión y otra de asignación, hay que especificar todos los datos que se van a insertar en el mismo orden al de los atributos y del mismo tipo.
- Sintaxis:

R←R U {(DATO 1, DATO 2, …DATO N)}

- Ejemplo:

CHUCHERIAS←CHUCHERIAS U {(2,'REGALIZ',80)}

Eliminación.

- Elimina tuplas enteras. Se expresa como una consulta, una diferencia y una asignación, por tanto, es la mezcla de tres operaciones.
- Sintaxis:

R←R – σ CONDICIÓN (R)

- Ejemplo:

Asignación diferencia selección

Chucherías ← chucherías – σ NOMBRE = 'NUBES' (CHUCHERIAS)

Actualización.

- Permite modificar los valores de uno o más campos.
- Para ello se utilizan operaciones de proyección normal y generalizada y operaciones de asignación.
- No se generan campos nuevos.
- Sintaxis:

R ← π A1, A2, A3, ...AN MODIFICADO (R)
R ← π A1, A2, A3, ...AN MODIFICADO (σ CONDICION (R))

- Ejemplos:
  - La primera sintaxis cambia todos los valores que pasan a llamarse todos esponjitas.

CHUCHERIAS← π COD, NOMBRE ='ESPONJITAS', PRECIO/KG (CHUCHERIAS)

  - La segunda sintaxis sólo cambiaría las nubes por esponjitas.

CHUCHERIAS← π COD, NOMBRE ='ESPONJITAS', PRECIO/KG (σNOMBRE='NUBES' (CHUCHERIAS))

Operaciones complementarias:

Ordenación.

- Permite ordenar una relación en base a uno o varios campos criterios.
- Sintaxis:

  T A1, A2…. (R)

- Ejemplo:

  T NOMBRE (CHUCHERIAS)

Eliminar duplicados.

- Sirve para eliminar registros repetidos en las relaciones.
- Sintaxis:

  δ (RELACIÓN)

TIPOS DE OPERACIONES	
Operadores de conjuntos Unión ∪ Intersección ∩ Diferencia **-** Producto cartesiano **X**	**Relacionales** Selección **σ** Proyección **Π** Reunión: natural *, theta **|x|**, externa. División ÷ Proyección generalizada **Π**
Adicionales Ampliación **α** Resumen **Ω** División generalizada ÷	**Modificación B.D.** Inserción Eliminación Actualizacion
Asignación Asignación ←	**Cambio de nombre** Renombrar ***ρ***

NORMALIZACIÓN

CONCEPTO.

- Es un proceso mediante el cual se transforman datos complejos en otros más simples, estables y fáciles de manejar y mantener.

OBJETIVOS.

- Eliminar o reducir redundancias de datos
- Eliminar anomalías de actualización (inserción, borrado y modificación).
- Eliminar ambigüedades, eliminando o simplificando dependencias entre atributos (campos).
- En definitiva, se persigue la integridad y consistencia de los datos.

VENTAJAS.

- Una base de datos normalizada ocupa menos espacio.
- La simplificación de algo complejo mejora su comprensión.
- Facilidad de consulta.
- Rapidez de búsqueda de la información.

INCONVENIENTES:

- Se crean múltiples tablas que pueden complicar el manejo de la base de datos o hacer que se pierdan dependencias entre los campos.

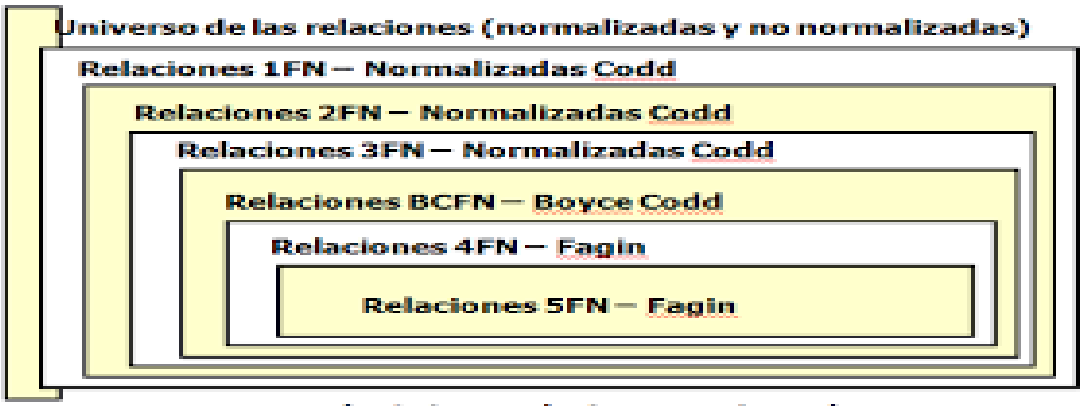
ANOMALÍAS POR EL MAL DISEÑO DE UNA BASE DE DATOS

- **Redundancia.**
    - Repetición de datos que hace que una B.D. ocupe más espacio y sea más dificultosa la realización de búsquedas en ella.
    - Las redundancias producen también **anomalías de actualización**, que son tres:
        - **Modificación.**
            - Si se modifica un valor, éste hay que cambiarlo en varias tuplas por estar éstas repetidas. Con ello se corre el riesgo de que queden tuplas sin modificar con lo que los datos ya no serían consistentes.
        - **Borrado.**
            - Si se elimina una tupla y ésta tiene datos repetidos en otras habrá que eliminar varias por repetición y podría ocurrir que alguna no se borrase. Como efecto secundario pueden borrarse datos importantes que no se puedan luego recuperar.
        - **Inserción**.
            - Uno de los problemas que da es que puede resultar imposible añadir datos nuevos debido a la ausencia de otros.

COMO SE SOLUCIONAN LAS ANOMALÍAS.

- Para solucionar las anomalías se dividen las tablas en otras menores, es decir, se proyectan las relaciones. Para descomponer o normalizar se usan formas normales y el proceso termina cuando no se puede normalizar más o se ha llegado a la forma normal objetivo.
- Las formas normales se abrevian con las letras FN y las principales son:

- 1FN, 2 FN y 3 FN, fueron diseñadas por Codd.
- FNBC o Forma Normal Boyce-Codd, que es una variante de la 3 FN, pero más estricta.
- 4FN y 5FN, diseñadas por FAGIN.
- 6FN.
- Forma normal Clave/Dominio.



Visión de la Teoría de Normalización

FORMAS NORMALES

1 FN – PRIMERA FORMA NORMAL.

- Los valores de los atributos no pueden ser un conjunto de valores o un grupo repetitivo y para eliminar redundancias los campos deben contener valores atómicos (indivisibles). Para que una tabla esté en primera forma normal **1 FN** se crean campos nuevos para incluir sólo valores únicos o se separan valores en distintas tuplas.

DNI	NOMBRE	DIRECCIÓN	TELÉFONO
778	ANA RUIZ	ASTURIAS, 1	444,888
321	EVA LÓPEZ	LEÓN, 2 – TOLEDO, 3	721

- Primera opción: Se ha creado campo apellido.

DNI	NOMBRE	APELLIDO	DIRECCIÓN	TELÉFONO
778	ANA	RUIZ	ASTURIAS, 1	444
778	ANA	RUIZ	ASTURIAS, 1	888
321	EVA	LÓPEZ	LEÓN, 2	721
321	EVA	LÓPEZ	TOLEDO, 3	721

- Segunda opción:

DNI	NOMBRE	APELLIDO	DIRECC1	DIRECC2	T1	T2
778	ANA	RUIZ	ASTURIAS 1		444	888
321	EVA	LÓPEZ	LEÓN, 2	TOLEDO, 3	721	

DEPENDENCIAS:

- Es una conexión entre uno o más atributos, una tabla se compone de atributos y dependencias.
- **Simbología**:

  ATRIBUTO A→ATRIBUTO B

  DETERMINANTE IMPLICADO

- Que A determine a B o que B dependa de A quiere decir que cada vez que A tome un valor, B siempre tendrá el mismo valor.
- **Tipos:**

Dependencia funcional mutua, Interdependencia o Dependencia reflexiva.

- Se da cuando ocurren dependencias funcionales A->B y B->A simultáneamente.
- **Simbología**:
  - A<->B
- Ejemplo:
  - Libro(Código_libro, ISBN, Titulo, Páginas, Editorial)
  - Código_libro -> ISBN
  - ISBN -> Código_libro
  - Dependencia reflexiva o Interdependencia:
    - Código_libro <-> ISBN
    - Código_libro e ISBN son claves candidatas, ya que cada una por separado identifica unívocamente a un libro, por lo que existe interdependencia o dependencia funcional mutua entre ambas.

Dependencia transitiva.

- Si un atributo B depende de un atributo A y un atributo C depende de B, pero no de A, se dice que existe una dependencia transitiva de C respecto de A.
- **Simbología**:

  ATRIBUTO A →ATRIBUTO B
  ATRIBUTO B→ATRIBUTO C

  Entonces:

  ATRIBUTO A→ATRIBUTO C
  ATRIBUTO A -- →ATRIBUTO C

  Ejemplo:

  FECHA NACIMIENTO →EDAD
  EDAD→VOTAR
  FECHA NACIMIENTO →VOTAR

DEPENDENCIA TRANSITIVA

CÓDIGO TRABAJADOR	EMPLEO	PRIMA
1235	MECÁNICO	400
1780	SOLDADOR	200
2090	MECÁNICO	400

CÓDIGO TRABAJADOR →EMPLEO
EMPLEO→PRIMA
CÓDIGO TRABAJADOR→PRIMA

Dependencia completa.

- Si el atributo A es realidad un conjunto de atributos, hay dependencia completa si se depende por completo de él y no de un subconjunto suyo.

Dependencia parcial.

- Una dependencia parcial se da cuando un atributo no depende por completo del determinante sino de una parte de él.

DNI	CODIGO PROYECTO	NOMBREALUMNO	HORAS
333	A	LUIS	40
333	B	LUIS	50
333	C	LUIS	30
444	C	ANA	40
555	B	JUAN	50
555	C	JUAN	30

DNI, CÓDIGO PROYECTO →HORAS (DEPENDE COMPLETAMENTE DE CÓDIGO PROYECTO)
DNI→NOMBRE (ES PARCIAL PORQUE SÓLO DEPENDE DEL DNI)

DNI	CODIGO PROYECTO	NOMBREALUMNO	HORAS	NOMBRE PROYECTO
333	A	LUIS	40	ARQUITECTURA
333	B	LUIS	50	INGENIERÍA
333	C	LUIS	30	INFORMÁTICA

444	C	ANA	40	INFORMÁTICA
555	B	JUAN	50	INGENIERÍA
555	C	JUAN	30	INFORMÁTICA

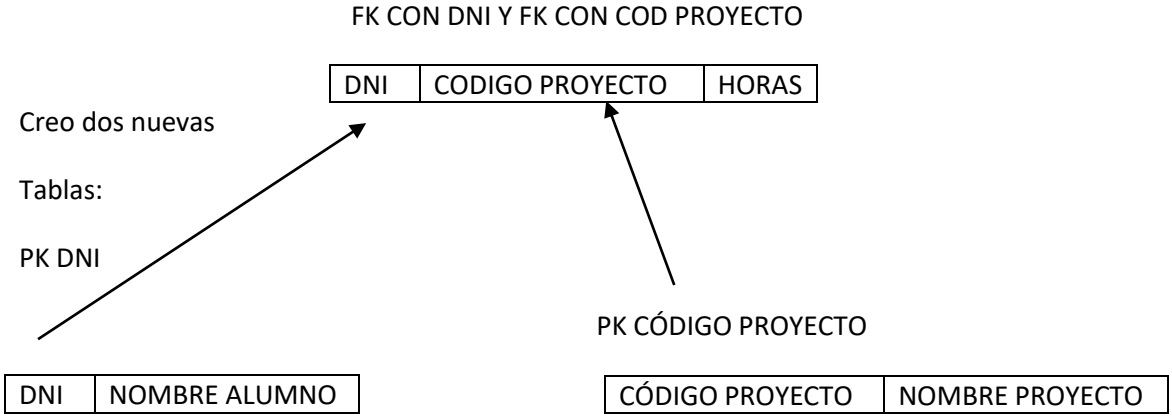
DNI, CÓDIGO PROYECTO→HORAS
DNI→NOMBRE
CÓDIGO PROYECTO→NOMBRE PROYECTO

2 FN - SEGUNDA FORMA NORMAL

- Una relación está en segunda forma normal:
  - Si está en primera forma normal, es decir la forma previa.
  - Todos los atributos que no sean clave dependan por completo de la clave primaria, es decir, que no existan dependencias parciales.
- **Eliminar dependencias parciales**:
  - Determinar que atributos no clave no dependen por completo de la clave primaria.
  - Eliminar esos atributos de la tabla base y crear con ellos otra tabla, incluyendo los atributos de la clave primaria de la cual dependen
  - Incluir esos atributos de la clave primaria de la cual dependen en la relación base para poder enlazar todas las relaciones.
- Ejemplo anterior:

DNI, CÓDIGO PROYECTO→HORAS
DNI→NOMBRE
CÓDIGO PROYECTO→NOMBRE PROYECTO

La tabla base se queda:



3 FN - TERCERA FORMA NORMAL

- Una relación está en tercera forma normal si está en segunda forma normal y no existen dependencias transitivas.
- En una dependencia transitiva los atributos no clave dependen de otros atributos no clave.
- Ejemplo:
    - En el caso de añadir una columna nueva sobre el uso o no del ordenador para el proyecto, la tabla anterior quedaría:

C DEPENDE DE A Y DE B QUE NO ES CLAVE, B DEPENDE DE A:

2 FN

CÓDIGO PROYECTO	NOMBRE PROYECTO	USO DEL ORDENADOR
B	INGENIERÍA	SI
A	ARQUITECTURA	SI
C	INFORMÁTICA	SI
D	GASTRONOMÍA	NO

- **Eliminar dependencias transitivas:**
    - Crear una tabla o relación con los atributos dependientes del atributo no clave e incluir éste también
    - Mantener el atributo no clave que provoca la dependencia transitiva en la tabla base.
    - Por tanto, "Nombre de proyecto" que es el campo que provoca la dependencia, y "Uso del ordenador", se van a otra tabla:

NOMBRE PROYECTO	USO DEL ORDENADOR

CÓDIGO DE PROYECTO	NOMBRE PROYECTO

FNBC - FORMA NORMAL BOYCE-CODD.

- Una relación está en FNBC si está en tercera forma normal y no hay dependencias redundantes y los únicos determinantes son claves candidatas no compuestas. Realmente es una ampliación o forma más estricta de la tercera forma normal.
- Una tabla no está en FNBC si las claves candidatas son compuestas, si las claves candidatas se encubren y tienen al menos un atributo en común (se solapan), si hay redundancias entre dependencias.
- Transformar a FNBC una relación:
    - Separar las claves candidatas compuestas
    - Separar redundancias
- Ejemplos:

ASISTIR (CODIGOCURSO, NOMBRECURSO, CODIGOESTUDIANTE, NOTA)
CODIGO CURSO ←→NOMBRE CURSO EXISTE REDUNDANCIA
CODIGO CURSO, CODIGO ESTUDIANTE→NOTA

CODIGO CURSO	NOMBRE CURSO

CODIGO CURSO	CODIGO ESTUDIANTE	NOTA

- Ya están en 3FN Y EN FNBC.

DEPENDENCIA MULTIVALOR

- Existe una dependencia multivalor cuando para cada valor de un primer atributo, existen múltiples de un segundo atributo.
- Sintaxis:
  - A→→B
  - El atributo A multidetermina al atributo B o el atributo B es multidependiente del atributo A.

4 FN – CUARTA FORMA NORMAL.

- Una relación está en cuarta forma normal si está en tercera forma normal o en FNBC y si no existen dependencias multivalor entre varios atributos.
- A→→B
- A→→C
- Una relación no está en cuarta formal normal cuando existen dependencias multivalor entre atributos clave o no clave.
- La solución es crear varias relaciones manteniendo en cada una dependencia múltiple.
- Ejemplos:

DNI	TELEFONO	EMAIL
8787A	78787878	A@LLLL.ES
7444 B	55556565	

- La relación está en 3fn
- DNI→→TELEFONO
- DNI→→EMAIL
- Se crean 2 tablas para pasarlas a 4FN.

DNI	TELEFONO

DNI	EMAIL

CONDUCTOR	**VEHÍCULO**	**CARGA**
JUAN	FURGONETA	MUEBLES
LUIS	FURGONETA	MUEBLES
PEPE	FURGONETA	MUEBLES
JUAN	CAMIÓN	ESCOMBROS
LUIS	CAMIÓN	ESCOMBROS
PEPE	GRUA	COCHE
JUAN	TRAILER	CONTENEDOR
LUIS	TRAILER	CONTENEDOR

- CONDUCTOR →→ VEHÍCULO
- VEHÍCULO →CARGA
- Al pasarlas a 4FN salen 2 tablas:

VEHÍCULO	CARGA
FURGONETA	MUEBLES
CAMIÓN	ESCOMBROS
GRÚA	COCHE
TRAILER	CONTENEDOR

CONDUCTOR	**VEHÍCULO**
JUAN	FURGONETA
LUIS	FURGONETA
PEPE	FURGONETA
JUAN	CAMIÓN
LUIS	CAMIÓN
PEPE	GRUA
JUAN	TRAILER
LUIS	TRAILER

5 FN - QUINTA FORMA NORMAL o FNPU - FORMA NORMAL PROYECCIÓN-UNIÓN.

- Una relación no está en quinta forma normal porque existen muchas tuplas y pocos atributos que producen muchas redundancias y dependencias multivalor. Los inconvenientes son que dificulta el diseño, genera múltiples tablas y es muy rara y polémica la quinta forma.
- Una relación está en quinta forma normal cuando:
  - Está en cuarta forma normal
  - Cuando las proyecciones de las que depende combinadas forman la tabla original.

VENDEDOR	MARCA	PRODUCTO
LUIS	FORD	COCHE
LUIS	FORD	CAMIÓN
LUIS	MERCEDES	COCHE
LUIS	MERCEDES	CAMIÓN
JUAN	FORD	COCHE

La tabla está en 4FN.

Proyecciones.

VENDEDOR

MARCA

PRODUCTO

VENDEDOR	MARCA

TABLA A

VENDEDOR	PRODUCTO

TABLA B

MARCA	PRODUCTO

TABLA C

COMBINANDO A CON B A*B

VENDEDOR	MARCA	PRODUCTO
LUIS	FORD	COCHE
LUIS	FORD	CAMIÓN
JUAN	FORD	COCHE
LUIS	MERCEDES	COCHE
LUIS	MERCEDES	CAMIÓN
JUAN	FORD	COCHE

COMBINANDO B CON C B*C

VENDEDOR	PRODUCTO	MARCA
LUIS	COCHE	FORD
LUIS	COCHE	MERCEDES
LUIS	CAMIÓN	FORD
LUIS	CAMIÓN	MERCEDES
JUAN	COCHE	FORD
JUAN	MERCEDES	COCHE

- Aparece una **tupla espuria** que no aparecía en la tabla original pero que al combinar sí aparece.
- Se divide la tabla original en tantas tablas como se necesiten teniendo cada una en común los campos clave o los campos que permitan enlazar entre sí las distintas tablas.

FN DOMINIO-CLAVE (DK/FN)

- Una restricción del dominio indica que valores están para un atributo.
- Una restricción clave indica que atributos identifican una sola fila en una tabla.

MODELO ENTIDAD – RELACIÓN

- Se basa en el mundo real, como su nombre indica es un modelo que está compuesto por objetos y relaciones entre ellos. Se realiza a partir del análisis de requisitos.
- Existen dos modelos, el clásico o básico (ER) y el extendido (ERE).
- Conceptos del modelo básico:

Entidad:

- Es un objeto del mundo real del que se quiere almacenar información. Se les da nombre normalmente con sustantivos.
- Un **ejemplar** es cada una de las posibles ocurrencias de una determinada entidad.
- Una entidad hace referencia a:
    - **Sujetos**. Son personas u organizaciones que originan transacciones. Cliente, alumno, profesor, tienda, escuela, etc.
    - **Objetos**. Son entes tangibles.
    - **Eventos**. Son transacciones originadas por los sujetos que afectan a los objetos.
    - **Lugares**. Son la ubicación de los sujetos y los objetos. Dirección, calle, provincia, localidad, etc.
    - **Abstracciones**. Son los conceptos que califican, clasifican o miden otras entidades.
- Tipos de Entidad:
    - **Entidad fuerte**.
        - Es una entidad con existencia propia.
        - Ej. Cliente y cuenta corriente. Cliente es la fuerte y cuenta es la débil ya que necesita el cliente para existir.
    - **Entidad débil**.
        - Es una entidad que:
            - Depende de otra entidad para existir o es una entidad que requiere para su identificación de los atributos identificadores de otra entidad.
            - Ej. Vendedor y Producto. Vendedor fuerte y producto débil, ya que para identificar el producto vendido necesitas identificar al vendedor.

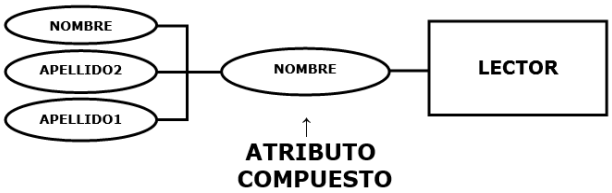
Atributo:

- Son las propiedades de una entidad o los datos que la definen o identifican.
- Tipos de atributos:
    - **Atributo identificador.**
        - Distingue de manera única cada ocurrencia de una entidad. Pueden ser principales o alternativos.
    - **Atributo descriptor.**
        - Describen cada ocurrencia de una entidad.
- Subtipos de atributos:
    - **Obligatorio.**
        - Es aquel que toma valores obligatoriamente.
    - **Opcional.**
        - No toman valores obligatoriamente.
    - **Monovaluado o univaluado.**
        - Toman un solo valor.
    - **Multivaluado.**

        - Toman multitud de valores.
    - **Derivado.**
        - Es aquel cuyo valor se obtiene a partir de otros atributos.
    - **Simple.**
        - Es aquel único y no divisible.
    - **Compuesto.**
        - Es aquel que es un agregado de varios atributos.

- Dominio:
    - Es una característica de los atributos, en concreto es el conjunto de valores y restricciones sobre las que se define un atributo.

Relación:

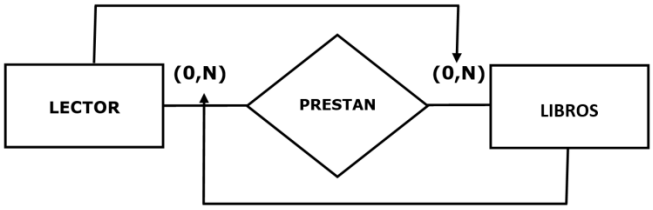


- Es una asociación entre una o varias entidades.
- Refleja las interrelaciones entre entidades y sus dependencias.
- Las relaciones pueden tener atributos, pero sólo descriptores.
- Se suelen expresar con tiempos verbales.
- Ej. Alumno-asignatura → Aprobar. Si no se encuentra el verbo, temporalmente se puede coger parte de las palabras de la relación ALASIG, por ej. (Alumno-Asignatura)
- Características de las relaciones:

    - **Grado.**
        - Es el número de entidades que participan en la relación. Ej. Alumno-Asignatura es de grado dos porque tiene dos entidades.
    - **Cardinalidad.**
        - Es el número máximo de ejemplares u ocurrencias de una entidad asociados a los de otra entidad.

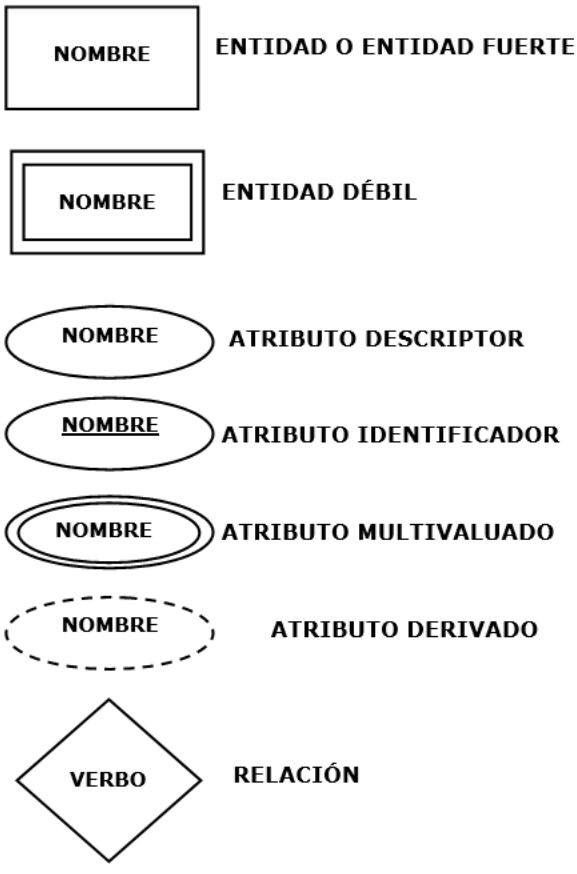
- Tipos de relaciones:
    - **Uno a Uno (1 , 1).**
        - Un ejemplar de la entidad A se relaciona con un solo ejemplar de la entidad B, ej. coche-matrícula.
    - **Uno a muchos (1 , N).**
        - Un ejemplar de la entidad A se relaciona con varios ejemplares de la entidad B, por ejemplo, un lector varios libros prestados, un cliente varias cuentas en el Banco.
    - **Varios a Varios (N , M).**
        - Varios ejemplares de la entidad A se relacionan con varios ejemplares de la entidad, por ejemplo, varios actores de cine que han trabajado con varios directores de cine.

- Características de las cardinalidades:
  - **Cardinalidad máxima.**
    - Es el número de ejemplares de una entidad que pueden relacionarse con uno de otra u otras.
  - **Cardinalidad mínima**.
    - Es el número mínimo de ejemplares de una entidad que pueden relacionarse con uno de otra u otras.
  - **Representación**:
    - Se suelen representar con dos valores entre paréntesis (mínimo, máximo): (0, 1), (1, 1), (0, 1), (1, N), (M, N).
    - Ejemplo:
      - Cardinalidad de lector con respecto a libros:

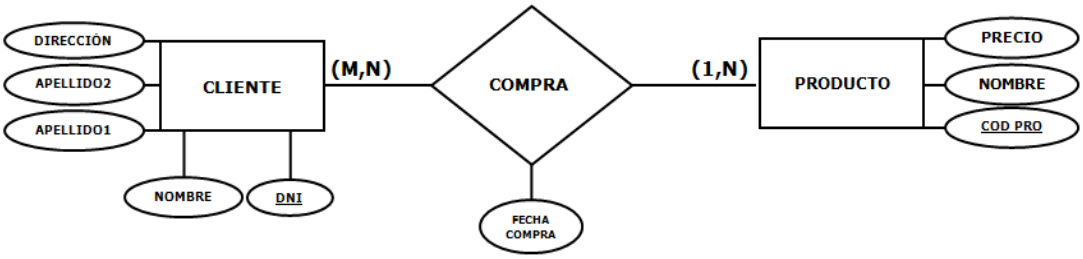


- Lector (0, N) Libros → el lector puede llevarse de 0 a varios libros.

SIMBOLOGÍA DEL MODELO ENTIDAD-RELACIÓN



- Un cliente puede adquirir uno o varios productos, modelar incluyendo tres o más atributos en la relación, es posible que la relación también tenga atributos.



Esquema relacional

CLIENTES (DNI, NOMBRE, APELLIDO1, APELLIDO2, DIRECCIÓN)
PK

PRODUCTOS (COD_PRO, NOMBRE, PRECIO)
PK

COMPRA (DNI, COD_PRO, FECHA COMPRA)
FK **FK**

REDUCCIÓN DEL DIAGRAMA ENTIDAD-RELACIÓN A TABLAS

Reglas:

- Para cada conjunto de entidades fuertes crear una tabla con el nombre de cada entidad, incluyendo sus atributos y el identificador como clave primaria.
- Para cada conjunto de entidades débiles crear tablas con el nombre de cada una, incluyendo sus atributos propios más la clave primaria de la entidad fuerte a la que está subordinada.
- Para cada relación o conjunto de relaciones crear una tabla con los atributos propios más las claves primarias de todas las entidades que formen parte de la relación.

ENTIDADES FUERTES Y DÉBILES – DEPENDENCIAS

- **De existencia**.
    - La existencia de una ocurrencia para una entidad depende de la existencia de otra, aunque tenga identificador.
- **De identificación**.
    - En este caso, la entidad débil no tiene identificador, de modo que necesita un identificador de otra entidad con la que se relaciona.
- De existencia y de identificación al mismo tiempo.

RELACIONES UNARIAS, BINARIAS Y N-ARIAS

- Relación unaria- Aquella en la que participa una sola entidad.
- Relación binaria- Aquella en la que participan dos entidades.
- Relación ternaria- Aquella en la que participan tres entidades.
- Relación N-Arias – Aquella en la que participan múltiples entidades.

- **Relaciones Reflexivas: consiste en relacionar una entidad consigo misma.**

Figura 1. Relación Reflexiva

- **Relaciones binarias: consiste en unir dos entidades mediante una relación.**

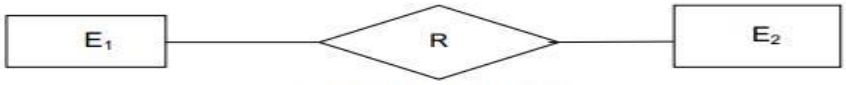


Figura 2. Relación binaria.

- **Relaciones ternarias: Son aquellas que unen tres entidades mediante una relación.**

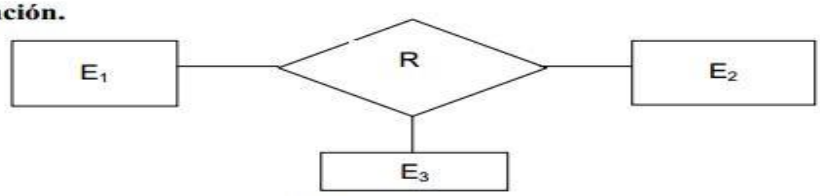
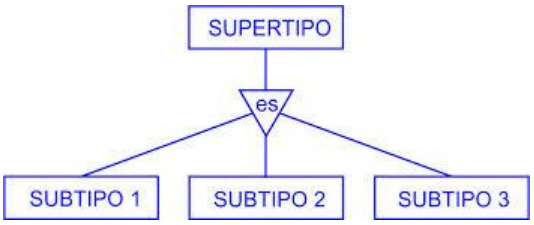


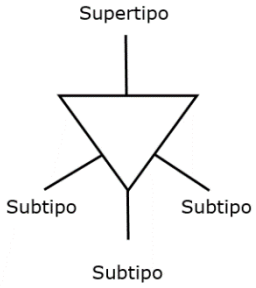
Figura 3. Relación Ternaria

MODELO ENTIDAD RELACIÓN EXTENDIDO (ERE)

1. **Generalización - Especialización**
    - Existen entidades que comparten un determinado número de atributos, para clarificar el esquema ER interesa crear otra entidad de nivel superior que contenga los atributos comunes. A la entidad de nivel superior también se le denomina supertipo, superclase o clase general. A la entidad de nivel inferior se le denomina también subtipo, subclase o clase especializada.
    - **Generalización:** Es un mecanismo de abstracción que permite especializar una entidad en subtipos.

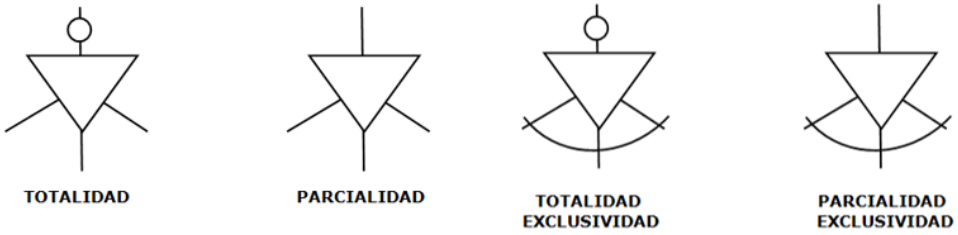
- **Especialización:** Es un mecanismo de abstracción que permite diferenciar entidades que comparten atributos comunes.





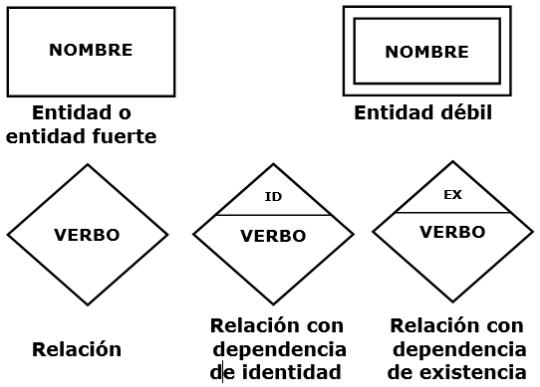
RESTRICCIONES DE GENERALIZACIÓN-ESPECIALIZACIÓN

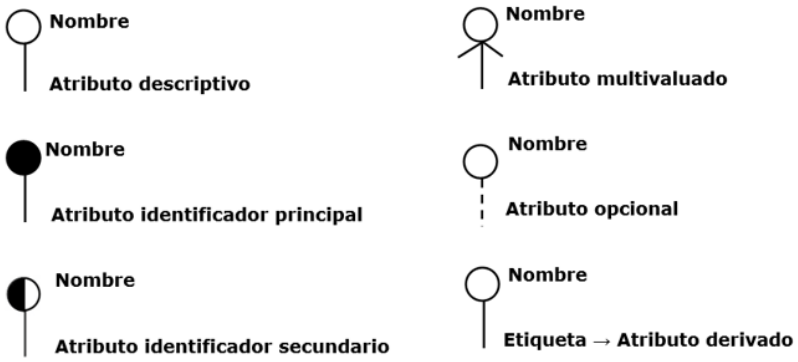
- **Totalidad.**
  - Las ocurrencias de los subtipos de una generalización cubren al supertipo, es decir, no hay ocurrencias en el supertipo que no pertenezcan a ningún subtipo.
- **Parcialidad**.
  - Se representa parte de todos los posibles subtipos que un supertipo puede tener.
- **Exclusividad.**
  - No hay ocurrencias que pertenezcan a más de un subtipo, a estos se les denomina subtipos disjuntos.
- **Solapamiento.**
  - Hay ocurrencias que pertenecen a más de un subtipo a la vez.



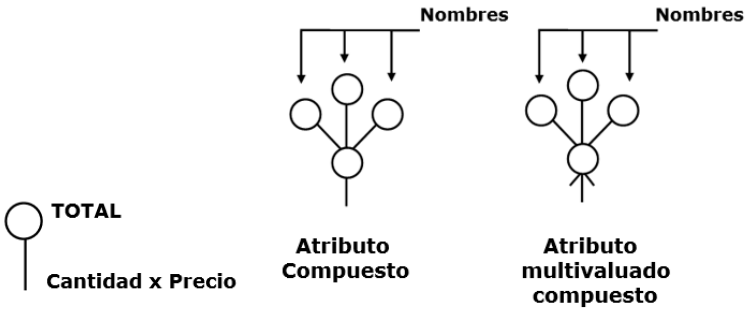
Nota: Falta Solapamiento en las 2 primeras imágenes.

SIMBOLOGÍA





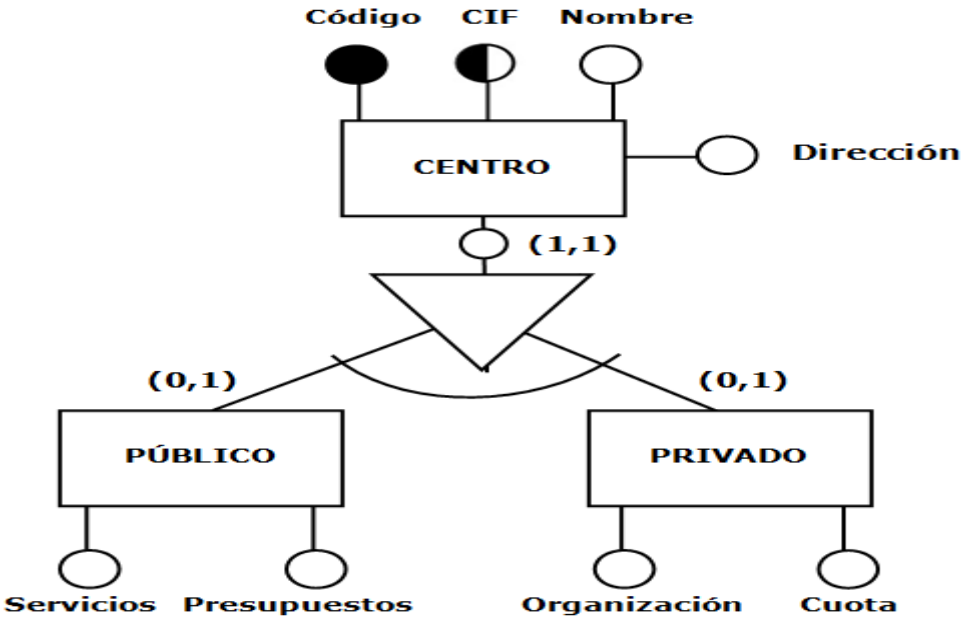
El atributo derivado es el que sirve para hacer operaciones de cálculo. Ejemplo:



CONVERTIR EN TABLAS UNA GENERALIZACIÓN-ESPECIALIZACIÓN DEL MODELO ERE.

Opción A:

- En ésta se crean tablas sólo para las entidades especializadas incluyendo en ellas sus atributos específicos más los heredados del supertipo.



PÚBLICO (CÓDIGO, CIF, NOMBRE, DIRECCIÓN, SERVICIOS, PRESUPUESTOS)

PRIVADO (CÓDIGO, CIF, NOMBRE, DIRECCIÓN, ORGANIZACIÓN, CUOTA)

Opción B:

- Se crea una tabla para la entidad de nivel superior con los atributos comunes y otra u otras para las entidades especializadas con sus atributos exclusivos más la clave primaria de la entidad superior, que en las tablas específicas actuará como clave foránea.

CENTRO (CÓDIGO, CIF, NOMBRE, DIRECCIÓN)
PK

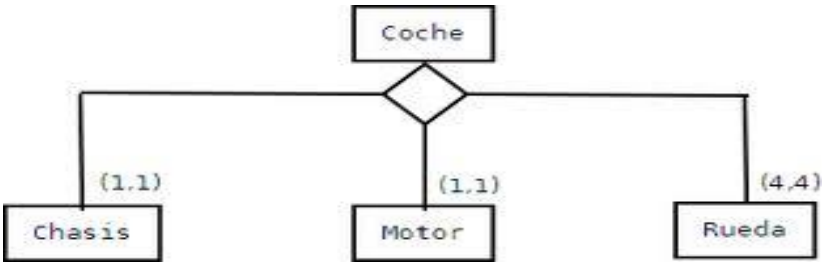
PÚBLICO (CÓDIGO, SERVICIOS, PRESUPUESTO)
FK

PRIVADO (CÓDIGO, ORGANIZACIÓN, CUOTA)
FK

AGREGACIÓN

- Un agregado es un todo compuesto de varias partes.
- Tipos:
  - **a) Compuesto-Componente**

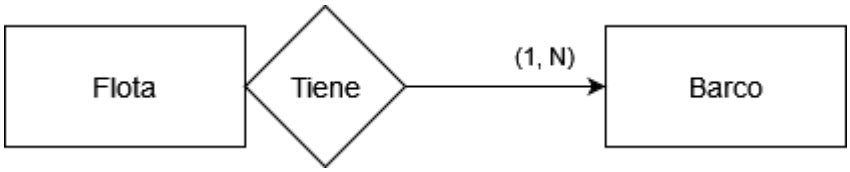
    En este tipo de agregación un todo o agregado se obtiene por la unión de diversas partes o componentes.



b) Miembro-Colección.

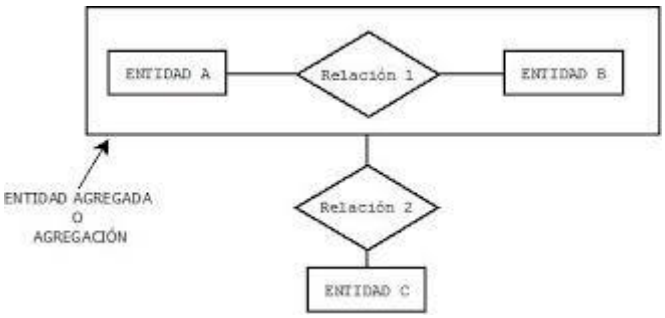
Un todo agregado es una colección de miembros de un mismo tipo de entidad.

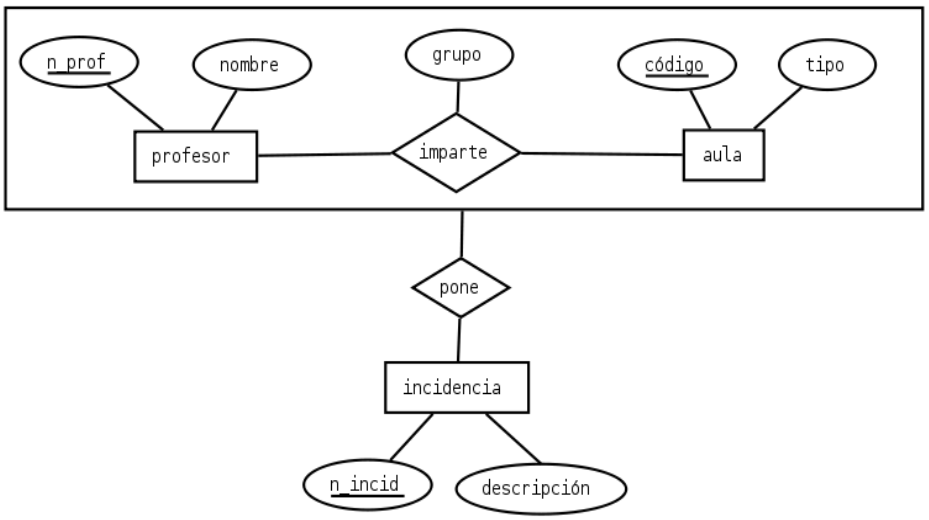
Flota (Colección) – Barco (Miembro)



c) Interrelación- Interrelación.

Esta última agregación permite crear una abstracción a través de la cual las relaciones se tratan como entidades de un nivel superior.





MODELO DE BASE DE DATOS ORIENTADO A OBJETOS

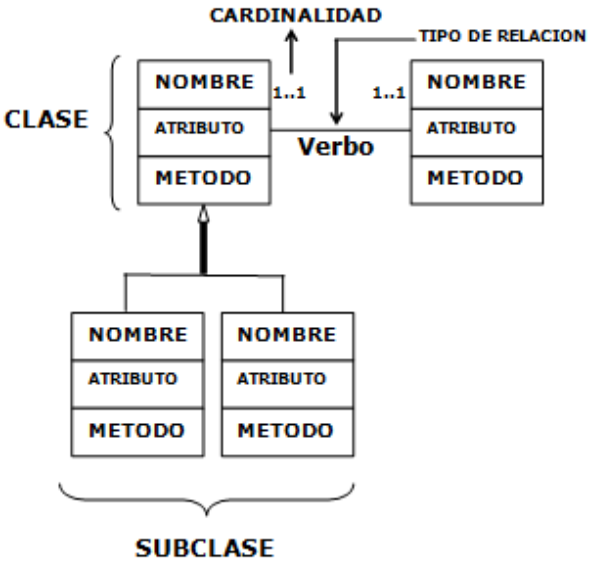
Conceptos Básicos:

- **Clase.**
  - Abstracción conceptual que permite definir un conjunto de objetos de un mismo tipo.
- **Objeto.**
  - Representación de un ejemplar de una clase.
- **Instanciación.**
  - Mecanismo que permite crear objetos de una clase determinada.
  - Ejemplo:
    - Clase Persona ➔Objeto Luis.
  - Instancia es sinónimo de objeto.

DIAGRAMAS DE CLASES (SIMBOLOGÍA):

- **Clasificador.**
  - Es donde se especifican las clases.
  - Su símbolo es un rectángulo dividido en tres partes en las que se incluye el nombre de la clase, sus atributos y sus métodos, que son las operaciones o funciones que se pueden realizar con los atributos.
- **Relaciones.**
  - Muestran las interacciones entre las clases y las hay de varios **tipos**:
    - Asociación:
      - Relación básica de conexión entre clases. Se representa con una línea de unión o como una flecha con una o dos puntas.
    - Dependencia:
      - Se da cuando una clase depende para su existencia de otra. Se representa con una flecha de trazo discontinuo cuya punta se dirige a la clase de la cual se depende.
    - Generalización:
      - Permite conectar una clase general o superclase con otra más específica o subclase. La simbología es una flecha con la punta hueca que apunta a la superclase.

- Al número de ejemplares u objetos de una clase que se relacionan contra se le llama multiplicidad o cardinalidad. Es similar a la cardinalidad del modelo E-R, sin embargo, las cardinalidades mínima y máxima se separan con 2 puntos, así, por ejemplo, (1, N) pasa a ser (1...N).



ADMINISTRADOR DE LA BASE DE DATOS (ABD – DBA)

- Persona encargada de definir y controlar las bases de datos.
- Proporciona asesoría a los desarrolladores, usuarios y ejecutivos que la requieran.
- Puede ser una única persona o un equipo de personas responsables del control y manejo del SGBD.
- Tiene(n) experiencia en:
  - SGBD.
  - Diseño de bases de datos.
  - Sistemas operativos.
  - Comunicación de datos.
  - Hardware.
  - Programación.
- Aptitudes:
  - Además de las técnicas, es deseable que tenga otras como habilidades sociales, cierto grado de diplomacia, manejo del personal, etc.
  - Conocimiento profundo de las políticas y normas de la empresa, así como sus criterios de aplicación.

Responsabilidades del Administrador de Bases de Datos

Administrar la estructura de la Base de Datos:

- Participa en el diseño de la base de datos o supervisa el trabajo del personal de diseño de la BD.
- Selecciona el SGBD más adecuado a utilizar.

- Tras el diseño de la base de datos construye ésta con el SGBD seleccionado, incluyendo los procedimientos y controles necesarios para asegurar su integridad.

Administración de la Actividad de Datos:

- El DBA no es usuario del sistema, protege los datos, no los introduce o procesa.
- Garantiza la concurrencia estableciendo los estándares, procedimientos y documentación para que los usuarios trabajen de forma cooperativa y complementaria, al ser la base de datos un recurso compartido.

Administrar el Sistema Manejador de Base de Datos:

- El DBA debe estandarizar todos los procesos de operación como, por ejemplo:
  - Como se introducen los datos (tipos, formatos.).
  - Como se procesan los datos.
  - Como se muestran los datos.
  - Como se accede a un archivo.
  - Como se determinan los índices.
  - Como se accede a los datos de forma concurrente.

Establecer el Diccionario de Datos:

- La estructura de la base de datos debe registrarse para que todos los participantes en el diseño puedan acceder a él.
- El diccionario debe incluir con claridad los tipos de datos a utilizar, sus ámbitos de influencia y sus restricciones de seguridad.

Asegurar la Confiabilidad de la Base de Datos:

- Mantener la integridad de los datos.
- Hay que diseñar un sistema de bases de datos resistente y capaz de recuperarse de fallos o usos inadecuados.
- Uso de herramientas de reparación de los errores que puedan sufrir los datos tras un fallo inesperado (Corte de suministro eléctrico, por ejemplo).

Confirmar la Seguridad de la Base de Datos:

- Protección de la base de datos de usos malintencionados o no autorizados.
- Creación de usuarios, permisos y privilegios.
- Monitorizar la actividad de los usuarios.
- Ajustar los derechos de acceso a datos compartidos para evitar que 2 o más usuarios o grupos de usuarios accedan simultáneamente a los mismos datos quedando comprometida la integridad de estos.
- Utilización de técnicas de recuperación para:
- Anticipar fallos y la respuesta ante ellos.
- Indicar a los usuarios qué hacer ante caídas del sistema y que es lo primero que debe realizarse tras la puesta en marcha.
- Cómo iniciar el proceso de recuperación.
- Que copias de seguridad utilizar.
- Programar la re-ejecución del tiempo perdido y de las tareas pendientes.
- Es importante establecer un calendario para llevar a cabo estas actividades sin afectar a otros sistemas dentro de la organización que hagan uso de los mismos recursos de hardware y software.

Funciones Básicas del Administrador de la Bases de Datos:

- **Definición del Esquema Conceptual:**

- Decidir qué datos almacenar y tras ello construir el esquema conceptual utilizando el Lenguaje de Definición de Datos (DDL).
- Es el esquema de la base de datos se crea escribiendo un conjunto de definiciones que son traducidas por el compilador de DDL a un conjunto de tablas que son almacenadas permanentemente en el diccionario de datos.

- **Definir el Esquema Interno:**
  - Definición de la estructura de almacenamiento o esquema interno correspondiente usando el Lenguaje de Definición de Datos (DDL), así como la correspondencia entre los esquemas interno y conceptual.
- **Creación de Bases de Datos y Tablas.**
- **Especificación de las Restricciones de Integridad de los Datos:**
  - Privilegios en bases de datos, tablas, campos, …
  - Integridad Referencial, integridad de entidad, …
  - Utilización de vistas.
- **Administración de la Concurrencia:**
  - Que datos son consultados y actualizados en un ambiente multiusuario.
  - Tipos de control de la concurrencia:
    - Concurrencia de lectura (SELECT).
    - Concurrencia de actualización (INSERT, DELETE y UPDATE).
- **Optimización del Acceso a Datos:**
  - Utilización de Índices, estadísticas de actualización y distribución de datos.
- **Vincularse con los Usuarios:**
  - El DBA debe comunicarse con los usuarios y ayudarles en lo que precisen.
- **Utilización de procedimientos de Respaldo y Recuperación:**
  - Copias de seguridad.
  - Uso de técnicas de recuperación ante fallos del sistema.
- **Supervisión:**
  - Supervisar el desempeño para que éste se ajuste a las necesidades de la empresa.
  - Realizar los ajustes apropiados cuando cambien los requerimientos.
- **Concesión de Autorización para el Acceso a los Datos.**
  - Concesión de diferentes tipos de autorización, permite al administrador de la base de datos regular que partes de la base de datos van a poder ser accedidas por varios usuarios.
- **Administrar la actividad de datos y la estructura de la base de datos:**
  - El DBA debe administrar el SGBD mismo. Deberá compilar y analizar estadísticas relativas al rendimiento del sistema e identificar áreas potencialmente problemáticas.
  - Ya que la base de datos es utilizada por múltiples de usuarios, el DBA requiere investigar todas las quejas sobre el tiempo de respuesta del sistema, la precisión de los datos y la facilidad de uso. Si se requieren cambios el DBA deberá planearlos y ponerlos en práctica.
- **Monitorización de la actividad de los usuarios:**

- El DBA deberá vigilar periódica y continuamente las actividades de los usuarios en la base de datos.
- Debe realizar informes estadísticos que indiquen cuáles fueron los usuarios activos, que archivos y que elementos de datos han sido utilizados, e incluso el método de acceso que han aplicado, así como los tipos de errores producidos.
- El análisis de estos informes servirá para determinar si se necesita una modificación en el diseño de la base de datos para mejorar su rendimiento o para facilitar las tareas de los usuarios.

- **Cambios en el Software o el Hardware:**
  - El DBA debe sopesar la necesidad de implantar versiones nuevas del SGBD e incluso el cambio de SGBD actual por otro distinto. En este caso, no sólo debe comunicarlo a los usuarios, sino que éstos deben ser formados en su uso.
  - Además, en caso de cambio también deberá administrar y controlar la migración tanto de las estructuras, como de los datos y las aplicaciones al nuevo SGBD.
  - Así mismo, cambios en el hardware pueden implicar también modificaciones que produzcan cambios en la configuración o en algunos parámetros de operación del SGBD.

RECURSOS HUMANOS ASOCIADOS A LAS BASES DE DATOS:

- Profesionales que definen, preparan, gestionan y mantienen las bases de datos.
- Tipos:
  - **Directivos/as.**
    - Organizadores y coordinadores del proyecto a desarrollar.
    - Encargados de decidir los recursos a utilizar.
    - Planificar el tiempo y las tareas.
    - Atención al cliente.
    - Dirigir las entrevistas y reuniones pertinentes.
  - **Analistas.**
    - Definen y analizan los requerimientos del usuario.
    - Describen la funcionalidad del sistema.
    - Encargados de controlar el desarrollo de las bases de datos aprobadas por la dirección.
    - Diseñan la base de datos (esquema interno y conceptual).
    - Coordinan la programación de la base de datos.
  - **Administradores/as de Bases de Datos.**
    - Personas que diseñan la base de datos.
    - Definen los datos, el acceso a estos y sus restricciones.
    - Definen permisos y roles de los demás usuarios de la base de datos.
    - **Funciones del administrador de bases de datos:**

- Definir el esquema de la BD a nivel lógico (definir la estructura de la BD) y a nivel físico (como se va a almacenar, donde lo hará y el acceso) lo hace con el Lenguaje de Definición LDD).
- Preservar la privacidad; se encarga de crear los usuarios, permisos y privilegios.
- Evitar el acceso no autorizado a la BD.
- Cumplir la LOPD con la BD.
- Mantenimiento, modificación y monitorización del esquema. Preservar la integridad de los datos.
- Diseñar un protocolo de actuación para solventar problemas comunes o fallos graves del sistema.
- Hacer copias de seguridad.
- Supervisar el rendimiento de la BD.

  - **Administrador de red.**
    - Mantienen, configuran e instalan el hardware y el software de red.
    - Se encargan del diseño de la red y de su seguridad.
  - **Desarrolladores/Programadores:**
    - Realizan las aplicaciones de usuario final de BD.
    - Para ello utilizan distintos lenguajes de programación.
    - Hay SGDB que incorporan lenguajes de cuarta generación, que permiten crear interfaces de usuario de forma rápida a través de formularios.
  - **Equipo de Mantenimiento:**
    - Dan soporte a los usuarios en su trabajo diario.

USUARIOS FINALES.

- Personan que trabajan directamente con la información almacenada o que acceden de forma ocasional a una base de datos con un acceso mínimo (privilegios y permisos).
- Tipos:
  - **Expertos/as o Avanzados:**
    - Pueden utilizar las funciones **CRUD** porque tienen permisos y conocimientos.
    - Funciones CRUD:
      - CREATE.
      - READ.
      - UPDATE.
      - DELETE.

        - Usan el lenguaje de manipulación de datos (LMD), para acceder a la base de datos desde comandos.
        - Saben manejar SQL.
        - Usan la base de datos para gestión avanzada y toma de decisiones.
        - Su uso puede ser habitual u ocasional.
    - **Habituales:**
        - Usan las aplicaciones creadas por los desarrolladores para consultar y actualizar datos.
        - Trabajan en la empresa a diario con estas herramientas.
    - **Ocasionales:**
        - Usan un acceso mínimo a la base de datos a través de una aplicación que permite usar ciertos datos (por ejemplo, consultar horarios de ferrocarriles vía Internet).

SISTEMA DE GESTIÓN DE BASES DE DATOS (SGBD) - DATA BASE MANAGEMENT SYSTEM (DBMS).

HISTORIA Y EVOLUCIÓN DE LOS SGBD.

- El uso de sistemas de bases de datos automatizados se desarrolló a partir de la necesidad de almacenar grandes cantidades de datos, para su posterior consulta, producidas por las nuevas industrias que creaban gran cantidad de información.
- Herman Hollerit (1860-1929) fue denominado el primer ingeniero estadístico de la historia, ya que invento una computadora llamada “Máquina Automática Perforadora de Tarjetas" Para hacer el censo de Estados Unidos en 1880 se tardaron 7 años para obtener resultados, pero Herman Hollerit en 1884 creó la máquina perforadora, con la cual, en el censo de 1890 dio resultados en 2 años y medio, donde se podía obtener datos importantes como número de nacimientos, población infantil y número de familias. La máquina uso sistemas mecánicos para procesar la información de las tarjetas y para tabular los resultados.
- Las primeras bases de datos manejaban ficheros que eran almacenados en tarjetas o soportes magnéticos. Cuando los ordenadores evolucionan, aparecen las cintas y los discos, a la vez que las maquinas son dotadas de mucha más potencia y facilidad de manipulación, es por tanto en ese momento cuando las bases de datos comienzan a ser realmente útiles.

Antes de 1960:

- **1945**: Invención de las cintas magnéticas (primer medio que permite búsquedas)
- **1957**: Instalación del primer computador comercial.

- **1959**: McGee propone el concepto de acceso generalizado a datos almacenados electrónicamente.
- **1959**: IBM presenta el sistema Ramac.

Los años 60:

- **1961**: Charles Bachman diseña el primer SGBD generalizado, el almacén de datos integrados (Integrated Data Store, IDS) para General Electric. Trabajando junto con Weyerhaeuser Lumber, desarrolló el primer acceso concurrente a la base de datos IDS. Más tarde, todavía en General Electric, desarrolló el producto "dataBasic", que ofrecía servicios de bases de datos a los usuarios del lenguaje Basic de tiempo compartido; Bachman popularizó los diagramas de estructuras de datos o "diagramas de Bachman".

Entre 1965-1970:

- Muchos proveedores crean sistemas generalizados de manejo de archivos.
- IBM desarrolla su Sistema de Gestión de Información (IMS - Information Management System,).
- El sistema IMS DB/DC (Base de Datos/Comunicación de Datos), fue el primer sistema de comunicación de datos a escala. IBM y American Airlines crean SABRE.
- **Sistemas de navegación de 1960:**
    - Acceso a datos mediante navegación de registro en registro.
- **Modelo de Bases de Datos Jerárquico:**
    - Se dice que comienza a mediados de los 60 con el proyecto Apolo (hombre a la luna), para gestionar la inmensa cantidad de datos que requería el mismo.
    - La encargada, la NAA (North American Aviation), desarrolló el software GUAM (General Update Access Method) basado en el concepto de árbol, para poder unir varias piezas pequeñas en una pieza más grande, y así sucesivamente hasta ensamblar el producto final.
    - A mediados de los 60, NAA se une con IBM para perfeccionar GUAM con el software IMS (Information Management System).
    - Almacena datos enlazando (con direcciones físicas) los registros (nodos) en forma de árbol donde un nodo padre de datos puede tener varios nodos hijos.
- **Modelo de Bases de Datos en Red:**
    - A mediados de los 60, General Electric desarrolló IDS (Integrated Data Store), dirigido por uno de los pioneros en los sistemas de bases de datos, Charles Bachmann, para satisfacer la necesidad de representar relaciones entre datos, más complejas de las que se podían modelar con los sistemas jerárquicos y en parte, para imponer un estándar de bases de datos.

- **CODASYL** (Conference on Data System Languages) es el nombre de una conferencia de finales de los años 60 en la que participaron organismos privados y públicos del gobierno de Estados Unidos con la finalidad de definir estándares (Codasyl definió el lenguaje COBOL) para la informática de gestión. De ahí salió DBTG (Data Base Task Group - Grupo de Tareas para Bases de Datos), grupo que definió las especificaciones estándar que permitieran la creación de bases de datos y el manejo de los mismos, dando lugar al modelo de red de bases de datos.
- En 1971 se presentó el informe final, aunque éste no fue formalmente aceptado por ANSI (American National Standards Institute).
- Almacena datos enlazando (con direcciones físicas) los registros (nodos) en forma de árbol donde un nodo hijo puede tener varios nodos padres.

Los años 70: La tecnología de bases de datos experimenta un rápido crecimiento.

- **1970**: Ted Codd, investigador asociado de IBM, desarrolla el modelo relacional.
- **1971**: Informe del grupo de trabajo sobre bases de datos (DBTG) de CODASYL.
- **1975**: El Special Interest Group on Management of Data (Grupo de Interés Especial de la A C M dedicado a gestión de datos), organiza la primera conferencia internacional SIGMOD.
- **1975**: La Very Large Data Base Foundation (Fundación para Bases de Datos Muy Grandes), organizó la primera conferencia internacional sobre bases de datos muy grandes (VLDB).
- **1976**: Peter Pin-Shan Chen introduce el modelo de Entidad-Relación (Modelo E-R).
- **Modelo de Bases de Datos Relacional:**
  - Desarrollado a principios de los 70 por Edgar F. Codd para IBM, atrajo rápidamente la atención de los medios académicos. Basado en la teoría matemática de conjuntos, consta de las siguientes características:
  - Almacenamiento de datos en tablas.
  - Simple y fácil de implementar.
  - Normalización u optimización.
  - Conjunto de restricciones para evitar redundancias en los datos.
  - Manipulación de datos basada en el Álgebra y el Cálculo Relacional.
  - Una parte de la información se usa como clave, identificando de manera unívoca un registro concreto.

- IBM desarrolla el prototipo de SGBD System R, basado en el modelo relacional de Codd, que con el tiempo se convertirá en Database 2 (DB2).

Los años 80:

- Desarrollo de SGBD para ordenadores personales (dBase, Paradox, etc.).
- **Principios 80:** En Suecia, el artículo de Codd generó la base de datos Mimer SQL en la universidad de Uppsala. Mimer introdujo la gestión de transacciones

(distintas operaciones que se ejecutan a la vez como una sola), para dar robustez a las aplicaciones, una idea que fue recogida en muchos otros SGBD.

- **1983:** Un estudio de ANSI/X3/SPARC revela que se habían implementado más de 100 sistemas relaciónales a principios de los años ochenta.
- **1985:** Se publica la norma preliminar de SQL. Influencia de los "lenguajes de cuarta generación" en el mundo de los negocios. ANSI propone un lenguaje de definición de redes NDL (Network Definition Language).
- Durante la década de 1980 el auge de la programación orientada a objetos influyó en el modo de manejar la información de las bases de datos. Programadores y diseñadores comenzaron a tratar los datos en las bases de datos como objetos.
- Otro gran foco de atención durante la década fue el incremento de velocidad y fiabilidad en el acceso. Replicando la información más importante y solicitada en una base de datos temporal de pequeño tamaño con enlaces a la base de datos principal, se podía buscar mucho más rápido en la base de datos pequeña que en la grande. Esta mejora de prestaciones introdujo la indización, que acabarían incorporado la totalidad de los SGBD.
- **Tendencias futuras en los años 80:**
    - Sistemas expertos de base de datos.
    - SGBD orientados a objetos.
    - Arquitectura cliente-servidor para bases de datos distribuidas.

Los años 90:

- Demanda para extender las capacidades de los SGBD para nuevas aplicaciones.
- Aparición de SGBD comerciales orientados a objetos.
- Demanda de aplicaciones que utilicen datos de diversas fuentes.
- Demanda para aprovechar procesadores paralelos masivos (MPP).
- Aparición de SGBD relacionales comerciales (DB2, Oracle Database, Sysbase, Informix, SQL Server, MySQL, etc.).
- Surgen nuevas aplicaciones de las bases de datos, trabajo con redes y gestión de datos distribuidos.
- Nuevas características de los SGBD para manejar datos espaciales, temporales y multimedia, incorporando capacidades activas y deductivas.
- Aparición de normas para consulta e intercambio de datos (SQL2, PDES, STEP).

Años 2000:

- **Sistemas NoSQL:**
    - El siglo XXI trajo una nueva tendencia en las bases de datos: el NoSQL, que introducía una línea no relacional significativamente diferente de las clásicas, que se caracterizada porque:
        - No requieren por lo general esquemas fijos.
        - Evitan las operaciones *join* almacenando datos desnormalizados.
        - Están diseñadas para escalar horizontalmente.

        - La mayor parte de ellas pueden clasificarse como almacenes clave-valor o bases de datos orientadas a documentos.
    - Aplicaciones más populares:
        - MongoDB, MemcacheDB, Redis, CouchDB, Hazelcast, Apache Cassandra y HBase, todas ellas de código abierto.

- **Sistemas XML:**
    - Las Bases de Datos XML forman un subconjunto de las Bases de Datos NoSQL. Todas ellas usan el formato de almacenamiento XML, que está abierto, legible por humanos y máquinas y ampliamente usado para interoperabilidad.
    - Aplicaciones más populares: BaseX, eXist, MarkLogic Server, MonetDB/XQuery, Sedna.

ARQUITECTURA DE BASE DE DATOS ANSI/X3/SPARC.

- Las arquitecturas de bases de datos han evolucionado mucho desde sus comienzos, aunque la considerada estándar hoy en día es la descrita por el comité ANSI/X3/SPARC, que data de finales de los años setenta.
    - **ANSI** (American National Science Institute), es un organismo científico de Estados Unidos que ha definido diversos estándares en el campo de las bases de datos.
    - **X3** es la parte de ANSI encargada de los estándares en el mundo de la electrónica.
    - **SPARC** (Standard Planning and Requirements Committee), comité de planificación de sistemas y reparaciones es una subsección de X3 encargada de los estándares en Sistemas Informáticos en especial del campo de las bases de datos. Su logro fundamental ha sido definir un modelo de referencia para las bases de datos.

- **Modelo ANSI/X3/SPARC:**
    - El grupo ANSI ha marcado la referencia para la construcción de SGBD. Basándose en estudios anteriores en los que se indicaban tres niveles de abstracción de la base de datos, ANSI profundiza más en esta idea y define cómo debe ser el proceso de creación y utilización de estos niveles.
    - En el modelo ANSI se indica que hay tres **niveles** que permiten crear esquemas o diseños de base de datos:
        - El nivel físico, o de máquina.
        - El nivel externo, o de usuario.
        - El nivel conceptual o de esquema.
    - Así mismo, describió las interacciones entre estos tres niveles y todos los elementos que conforman cada uno de ellos, siendo el objetivo principal de esta arquitectura separar los programas de aplicación de la base de datos física.
    - Características importantes inherentes a los sistemas de bases de datos:
        - La separación entre los programas de aplicación y los datos.
        - El manejo de múltiples vistas por parte de los usuarios.

        - El uso de un catálogo o diccionario para almacenar el esquema de la base de datos.
    - En 1975, el comité ANSI-SPARC propuso la arquitectura de tres niveles para los sistemas de bases de datos, al resultar muy útil a la hora de conseguir estas tres características.

SISTEMA DE GESTIÓN DE BASES DE DATOS (SGBD) - DATA BASE MANAGEMENT SYSTEM (DBMS).

- **Concepto.**
    - Conjunto de programas que sirven para definir, construir y manipular una BD.
    - Software que permite a los usuarios procesar, describir, administrar y recuperar los datos almacenados en una BD.
- **Definición o creación de una BD.**
    - Consiste en especificar las estructuras, tipo de datos y restricciones para los datos que se almacenarán.
- **Manipulación de una BD.**
    - Operación es de inserción, borrado, modificación y análisis o consulta de los datos.
- **Funciones de un SGBD.**
    - Acceso a la información usando herramientas de consulta.
    - Proporciona métodos para mantener la integridad de los datos.
    - Administración de usuarios y permisos. Restricción de accesos no autorizados.
    - Generación de informes.
    - Control de concurrencia (acceso simultáneo a datos).
    - Copias de seguridad y recuperación de datos.
    - Proporcionar entornos visuales para facilitar trabajo de usuarios.
    - Descripción unificada de los datos e independiente de las aplicaciones.
    - Independencia de las aplicaciones respecto a la representación física de los datos.
    - Definición de vistas parciales de los datos para distintos usuarios.
- **Herramientas de un SGBD:**
    - Para la creación y especificación de los datos, así como la estructura de la BD.
    - Para la creación y administración de la estructura física requerida en las unidades de almacenamiento.
    - Herramientas para la manipulación de datos: Insertar, borrar, consultar y modificar datos.
    - Herramientas de recuperación en caso de desastre.

- Herramientas para creación de copias de seguridad.
- Herramientas de gestión de la comunicación de la BD.
- Herramientas creación de aplicaciones que utilicen los esquemas de datos. (Esquema externo: Visión de los datos que tienen los usuarios finales).
- Herramientas de instalación de la BD.
- Herramientas para la importación y exportación de datos.

- **Lenguajes proporcionados por un SGBD:**
  - **Lenguaje de Definición de Datos (LDD - DDL):**
    - Permite la descripción o definición de las estructuras que almacenarán los datos.
    - Instrucciones:
      - CREATE, ALTER, DROP.
  - **Lenguaje de Manipulación de Datos (LMD-DML):**
    - Permite tareas de actualización y consulta de los datos.
    - Instrucciones:
      - INSERT, SELECT, DELETE, UPDATE.
  - **Lenguaje de Control de Datos (LCD-DCL):**
    - Permite el control del acceso a los datos.
    - Instrucciones:
      - COMMIT, ROLLBACK, REVOKE, GRANT, DENY.

REPRESENTACIÓN DE LOS DATOS EN SISTEMAS DE BASES DE DATOS.

- **ESQUEMA FÍSICO O INTERNO:**
  - Forma en que están almacenados los datos.
  - Visión requerida sólo por el administrador para gestionar más eficientemente la BBDD.
  - En este esquema aparecen las unidades de disco, archivos y carpetas de sistemas.
- **ESQUEMA CONCEPTUAL:**
  - Esquema teórico de los datos.
  - Datos organizados por estructuras reconocibles del mundo real y sus relaciones.
  - El esquema es desarrollado por el diseñador de la BD.
- **ESQUEMA EXTERNO:**
  - Visión de los datos que tienen los usuarios finales obtenida a través de aplicaciones que abstraen la realidad conceptual, y que el usuario no conoce.
  - Cada aplicación produce un esquema externo o vista de usuario distinta.
  - El conjunto de todas las vistas se denomina **esquema externo global.**

- **Independencia de Esquemas.**
  - Los esquemas deben funcionar de forma independientemente.

- Tipos de independencia:
  - **Independencia física de los datos.**
    - Aunque cambie el esquema físico (por ejemplo, la B.D. se almacena en otro disco), el esquema conceptual no debe verse afectado.
  - **Independencia Lógica de los datos.**
    - Aunque se modifiquen los datos del esquema conceptual, los esquemas externos no deben verse afectados.

ARQUITECTURA ANSI/X3/SPARC.

- Una Base de Datos debe estar estructurada en 3 niveles diferentes:
  - Externo.
  - Conceptual.
  - Interno.

CLASIFICACIÓN DE LOS SGBD.

- **Según ámbito de aplicación:**
  - **Propósito general.**
    - Son SGDB adaptables a cualquier ámbito.
    - Son genéricos para poder ofrecer servicios a la mayoría de aplicaciones del mercado.
    - Ejemplos:
      - MS Access, Oracle Database, MySQL, SQL Server, MariaDB.
  - **Propósito específico.**
    - Están diseñados para dar soporte a una aplicación determinada.
    - No son reutilizables con otras aplicaciones.
    - Su rendimiento es máximo, pero tienen un coste de desarrollo elevado.
    - Se utilizan en entornos muy concretos y específicos (aerolíneas, entidades bancarias, …).
- **Según número de usuarios a los que da servicio.**
  - **Monousuario.**
    - Suelen estar destinados al ámbito doméstico y solo proporcionan acceso a un usuario a la vez.
    - Ejemplos:
      - MS Access, Libre Office Base, Open Office Base.
  - **Multiusuario.**
    - Permiten el acceso concurrente a múltiples usuarios a la vez.
    - Ejemplos:

        - MySQL, Oracle Database, SQL Server.
- **Según coste o tipo de licencia:**
    - **Libre**.
        - La empresa que lo desarrolla no cobra por su uso.
        - Ejemplos:
            - Libre Office Base, Open Office Base, SQLite, MariaDB.
        - También los hay de libre uso, pero cobran servicios adicionales.
    - **De pago**.
        - Cobran por la licencia de uso. Oracle Database, SQL Server.
    - **Doble Licencia**.
        - Versiones libres y de pago.
        - Hay privativos que ofrecen versiones Express gratuitas y limitadas, o completamente funcionales por un tiempo determinado, con fines de prueba o aprendizaje.
- **Según Modelo de Datos.**
    - Clasifican según un modelo lógico de datos.
    - El modelo define como se almacena la información y como se interactúa con ella.
    - **Modelos**.
        - **Jerárquico.**
            - Almacena la información en una estructura jerárquica y ramificada.
            - No gestiona bien la redundancia.
            - Ejemplos:
                - IMS de IBM, SYSTEM 2000 DE INTEL.
        - **Red o CODASYL DBTG.**
            - Es una variante del jerárquico.
            - Resuelve el problema de la redundancia del anterior permitiendo que los nodos hijos tengan más de un padre.
            - Ejemplos:
                - DMS 1100 de UNIVAC, TOTAL de CINCOM, EDMS de XEROX, PHOLAS de PHILIPS, DBOMP de IBM, IDS de HoneyWell.
        - **Relacional (SGBDR).**
            - Desplazó a los anteriores y es el que más éxito ha tenido.
            - Se basa en establecer relaciones entre las distintas entidades de la BD.
            - Ejemplos:
                - SYSTEM R y QBE de IBM, Oracle Database, MySQL, MariaDB, Informix, MS Access y SQL

Server de Microsoft, Visual FoxPro, SQLite, PostgreSQL, ...

- **Orientado a objetos (SGBDO).**
  - Se basan en la Programación Orientada a Objetos (POO).
  - La información se representa igual que en los lenguajes orientados a objetos.
  - Se manejan objetos y clases que es donde se almacenan los datos y las operaciones que se pueden hacer con ellos.
  - Las bases de datos se diseñan para trabajar en conjunción con lenguajes como JAVA, .NET, C++.
  - Ejemplos:
    - Objectivity, Ontos, O2, Gemstone, Versant, Jasmine, Itasca, Fast Object, Poet.
  -
- **Objeto-Relacional (SGBDOR).**
  - Son una combinación de las dos anteriores.
  - También se denominado Modelo Relacional Extendido.
  - Están basados en el estándar SQL99.
  - Ejemplos:
    - PostgreSQL, Oracle Database, Informix, SQL Server, Universal Server.
- **Transaccionales (OLTP Online Transaction Processing).**
  - Se combinan generalmente con las bases de datos relacionales, para poder realizar operaciones atómicas que se dan o no se dan todas juntas en caso de algún problema.
- **Multidimensional (OLAP Procesamiento Analítico en Línea).**
    - Son parecidas al modelo relacional pero mucho más complejas porque están optimizadas para los DATA WAREHOUSE y aplicaciones OLAP.
  - **Data WareHouse.**
    - Almacenes de datos.
    - Bases de datos corporativas que tienen información de distintas fuentes, y que pueden ser analizada desde distintas perspectivas.
  - **OLAP.**
    - En este tipo de bases, los campos o atributos pueden representar dos cosas:

        - Las dimensiones de una entidad o métricas que se van a estudiar asociadas a esos datos.
        - Las tablas aquí se asimilan a cubos o hipercubos.

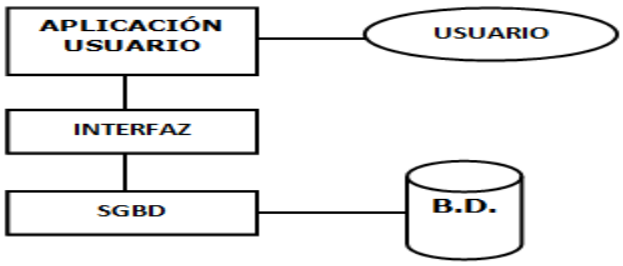
- **SGBD NoSQL o Not Only SQL.**
    - Estos sistemas difieren del modelo clásico de base de datos porque, o no usan SQL, o lo pueden soportar, pero no lo usan o lo utilizan poco.
    - Es el número mayor de SGBD que existe (http://nosql-database.org/).
    - **Características:**
        - Los datos almacenados no requieren estructuras fijas, como las que se requieren en el modelo relacional.
        - No soportan determinados tipos de relación, como las uniones.
        - No garantizan completamente las características ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).
    - **Tipos:**
        - **Orientadas a documentos.**
        - Gestionan datos semiestructurados que se almacenan en un formato XML, JSON, BSON.
        - Ejemplos:
            - MongoDB, Azure Document DB, …
        - **Orientados a columnas.**
            - Organizan los datos por columnas en lugar de por tuplas.
            - Ejemplos:
                - Apache Flink, Cassandra, …
        - **Orientados a clave-valor.**
            - Guardan tuplas que contienen una clave y su valor o conjunto de valores asociados.
            - Un dato se recupera buscando su clave.
            - Ejemplos:
                - Redis, DynamoDB, …
        - **Orientadas a grafos.**
            - Representan la información como nodos de un grafo.

        - Una BD de este tipo debe estar normalizada.
        - Ejemplos:
            - Neo4J, Infinity Graph, ...
    - **Deductivas/lógicas.**
        - Permiten hacer deducciones en base a inferencias.
        - Se basan en reglas y hechos que son almacenados.
        - Se fundamentan en materias como la lógica matemática y el cálculo relacional
        - Utilizan lenguajes declarativos como Prolog o Datalog.
- **Según su distribución o ubicación de la Base de Datos.**
    - **Centralizado.**
        - Tanto el SGDB como los datos están centralizados en un único equipo.
        - Base de datos y SGBD almacenados en un solo lugar, un solo equipo.
    - **Distribuido.**
        - El SGBD y los datos pueden estar separados, y no solo en máquinas distintas sino en ámbitos geográficos diferentes.
        - Base de datos y SGBD pueden estar ubicados en múltiples equipos conectados en red.
        - Tipos:
            - **Homogéneos.**
                - Sistemas distribuidos que tienen todos el mismo SGBD.
            - **Heterogéneos.**
                - Se usan distintos SGBD en cada sitio donde esté instalado el sistema distribuido.
                - Se les suele llamar sistemas federados o sistemas multibase.
- **Según lugar de almacenamiento**.
    - **Nube.**
        - Las bases de datos se almacenan en los servidores de un proveedor, pudiéndose acceder a ellas a través de Internet.
        - DBaaS (Database as a Service), contratación de proveedores de servicios para el almacenamiento de los datos.
        - Ventajas:
            - Ahorro de espacio físico.
            - Disminución de costes.
            - Acceso a datos desde cualquier dispositivo o lugar con conexión a Internet.
        - Inconvenientes:

        - El principal, problemas de acceso por fallos en la conexión a Internet.
    - Ejemplos:
        - Oracle Database Cloud Service, MySQL, MariaDB, SQL Server, Google Cloud SQL, SAP HANA, ….
- **Local.**
    - En este grupo entran bases como:
        - Las se almacenan en un equipo local sin conexión a otros equipos.
        - Las que una organización almacena y gestiona a través de una red local (LAN).
        - A estas últimas solo pueden acceder los equipos que estén conectados a dicha red local.

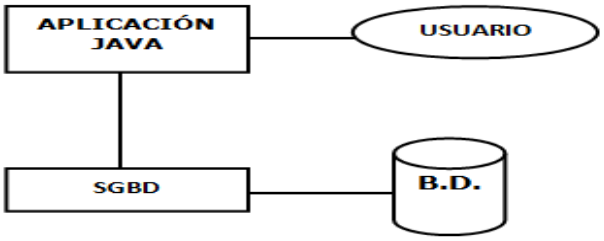
CONECTIVIDAD DE BASE DE DATOS.

- Una aplicación para acceder a una base de datos escrita para un SGBD, es posible que no funcione si no se modifica para acceder a bases de datos pertenecientes a otro SGBD.
- Para solucionar el problema se necesitan herramientas de acceso que garanticen la conectividad.

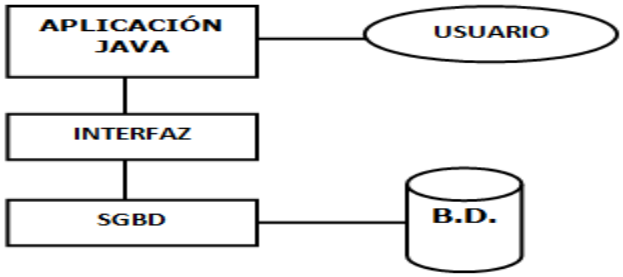


- **Herramientas de conectividad:**
    - **ODBC (Open Database Conectivity).**
        - Conectividad Abierta de Bases de Datos.
        - Proporciona una interfaz de universal y abierta que permite acceder a una base de datos desde cualquier aplicación independientemente del SGBD utilizado.
        - ODBC crea un interfaz de cliente SQL entre el SGBD y la aplicación, es decir, añade una capa intermedia, por lo que el modelo se habla de modelo de tres capas.
        - Para que funcione esta conectividad se necesita tener instalado un driver ODBC.
        - **Instalación origen de datos ODBC.**
            - Inicio / Panel de Control / Herramientas Administrativas/ Origen de datos ODBC.
    - **JDBC (Java Database Conectivity).**
        - Conectividad De Base De Datos Con Java.

- Permite a los programadores en Java ejecutar instrucciones en SQL y a cualquier programa escrito en Java acceder a cualquier SGBD.
- Necesita un driver JDBC para poder funcionar.
- Modelos de JDBC:
  - **Dos capas:**
    - La aplicación de JAVA se conecta directamente a la base de datos. Es una configuración típica cliente-servidor.



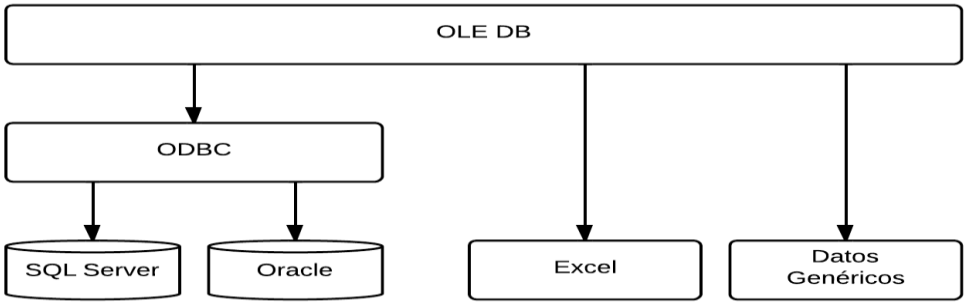
  - **Tres capas:**
    - La aplicación de JAVA transfiere las instrucciones a una capa intermedia que se encarga de enviar las instrucciones SQL a la base de datos.



- **ADO Y ADO.NET (ActiveX Data Objects).**
  - Herramienta de Microsoft creada para la conectividad de base de datos.
  - ADO.NET sustituye a ADO.
  - ADO.NET está pensado para los entornos de programación .NET de Microsoft, para que así los programadores puedan crear aplicaciones para acceder a base de datos.
  - Se puede utilizar con otros Sistemas Operativos (Linux, MAC OS X y otros).

- Proporciona también acceso a datos relacionales XML y a otras aplicaciones.

- **OLE DB (Object Linking and Embedding Database).**
  - Incrustación y vinculación de objetos de base de datos.
  - Tecnología de Microsoft para acceder a base de datos que van desde MS Access, Hojas de cálculo de Excel o



SGBD más potentes como SQL Server u Oracle.

- **MDAC (Microsoft Data Access Components).**
  - Componentes De Microsoft Para Acceso A Datos.
  - Conjunto de componentes de acceso a datos que permiten la creación de aplicaciones que puedan conectarse a base de datos.
  - Incluye ADO, OLE DB y ODBC.
- **PDO (PHP Data Objects).**
  - Objetos de Datos de PHP.
  - Extensión PHP para acceder a bases de datos.
  - El acceso a una base de datos a través de PDO se efectúa mediante un controlador que presenta las características de la base de datos.
  - Numerosas bases de datos, como MySQL, Oracle, Microsoft SQL Server y SQLite, disponen de un controlador PDO.

- **DRIVERS PUENTE.**
  - Drivers de conectividad que usan la tecnología de otro para poderse conectar a una base de datos.
  - Ejemplos:
  - ODBC to JDBC, JDBL to ODBC, OLE DB to ODBC, ADO.NET to ODBC.