

# UF1 - Fundamentos conceptuales de BBDD

## ▼ UF1.1 - Introducción a las BBDD

### Que es una BBDD?

Una BBDD es una colección organizada que se almacena y gestiona electrónicamente.

Permite la inserción, consulta, actualización y eliminación de registros.

Se administran gracias a los Sistemas de gestión de BBDD (SGBD)

Ventaja de interacción sin detalles técnicos.

Tipos: SQL y No SQL

### Definición de una BBDD

Una base de datos se compone de una serie de datos organizados y relacionados entre sí que se almacenan y consumen por los sistemas.

- Funciones:
  - Almacenamiento: Deben mantenerse dentro de un sistema informático
  - Organización: Se almacenan de manera lógica
  - Acceso: Deben tener un acceso rápido y eficiente
  - Seguridad: Debe determinarse quien puede acceder a los datos
  - Manipulación: Los datos pueden modificarse o eliminarse
  - Salvaguarda: Realizar backups de forma periódica

### Primera persona en tratar datos automáticamente

Herman Hollerith (Tarjeta perforada)

### 1950

Aparecen las cintas de magnéticas que automatizan información de las nominas.

Leen y transfieren esa información a otras cintas

Desventajas: solo secuencial

### Estandarización de BBDD

- Codasyl(1959):
  - Crecimiento de la industria informática y falta de estandarización.
  - Desarrollo estándares y lenguajes de programación para procesar datos.
- DBTG en Codasyl:
  - Desarrollo sistema ANSI sistema internacional.
- Codasyl crea Cobol:
  - Portabilidad, estandarización, orientado a negocio y alto rendimiento, pero generaron un estándar porque no aceptaron los informes necesarios
  - Codasyl y DBTG sentaron las bases para desarrollar lenguajes de BBDD
- COBOL:
  - Orientado a negocio

- Legible
- Estándar
- DBTG: enfoque en redes
- Codasyl:
  - Falta de estándar:
  - Necesidades diversas
  - Modelo red complejo
  - Modelo relacional evoluciona

## 1970

Sistemas de BDD destacados:

- IBM y Oracle



Edgar Frank Codd (gran revolucion)  
 Modelo relacional  
 12 reglas codd  
 Mejora respecto a modelo de red y jerárquicos  
 Surge Oracle

## 1980-1990



Sequel → SQL  
 Certificado por ANSI  
 1987 estandar ISO  
 1990 evolucion a BBDD orientadas a Objetos

## ▼ UF1.2 - Diseño de BBDD Relacionales

### ▼ Conceptos Fundamentales

#### 🔥 Definición de BBDD SQL


Es un modelo basado en relaciones que permite obtener información de varias entidades.

Normaliza las relaciones para evitar conflictos.

Bases por Edgar Frank Codd en 1970 (Laboratorios IBM)


- Características:
  - Se componen de varias tablas
  - Dos tablas no pueden tener el mismo nombre
  - Cada tabla tiene una serie de registros

- Relaciones entre tabla primaria y secundaria mediante Foreign Key
- La clave primaria es el identificador del registro
- La Foreign Key de la tabla principal es también Primary Key de la secundaria



LA UNIVERSIDAD  
EN INTERNET

FORMACIÓN  
PROFESIONAL



### Relación entre tablas

Ejemplo de relación entre la tabla empleado y la tabla departamento. Con esta relación se indica que un empleado pertenece a un departamento.

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario esporádico de la base de datos. La información puede ser recuperada mediante "consultas" que ofrecen una amplia flexibilidad y eficacia para administrar la información.

El lenguaje más utilizado para gestionar las bases de datos relacionales es SQL (Structured Query Language) o Lenguaje de Consulta Estructurado, un estándar implementado por los principales motores o sistemas de gestión de bases de datos relacionales.

### Base de datos jerárquica

En este tipo de base de datos los datos se organizan utilizando estructuras con forma de árbol plano. Un ÁRBOL es una estructura jerárquica en la que los elementos se suelen denominar NODOS y existen dependencias entre los nodos.

Bajo este modelo es habitual que se produzcan redundancias de datos en inconsistencias ya que varias aplicaciones pueden tratar el mismo dato. Otra característica es que existe una alta dependencia de los datos y las aplicaciones. Modificar la estructura de algún archivo de datos obliga a modificar los programas que los tratan.

Generalmente los datos están repartidos en diferentes archivos. Archivos que pertenecen a una determinada aplicación y sólo es posible acceder a los datos a través de esa aplicación por lo tanto esto provoca aislamiento de datos.

Es un modelo en desuso en la actualidad. Aunque la más extendida fue IMS (Information Management System). Presentado por IBM en 1968.

8

## BBDD Relacional

La manera en la que se estructura la información aporta flexibilidad y permite consultas mas sencillas para el usuario

Lenguaje mas usado SQL.

## BBDD Jerárquica

Se organizan a modo de árbol que tiene nodos y estos tienen dependencia unos de otros  
Es habitual encontrar redundancias en este modelo  
Datos repartidos en archivos  
Modelo en desuso

### **BBDD en red**

Es similar a la jerárquica pero permite a un hijo tener relación con varios padres  
Uso en sistemas de mainframe IDMS

### **BBDD orientadas a objetos**

Permite almacenar en la base de datos estado de objetos

Se diseña para trabajar con lenguajes de programación guardando el estado de sus variable y en el momento de la orden

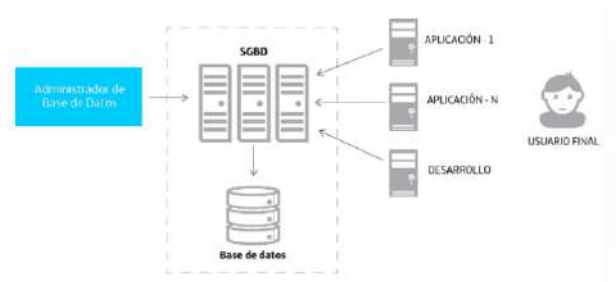
- Características:
  - Encapsulación: Permite ocultar la información al resto d objetos
  - Herencia: Los objetos heredan el comportamiento de otros
  - Polimorfismo: Podemos redefinir el comportamiento de un método
  - Abstracción: Definimos lo que queremos y lo implementamos de varias formas



### **Sistemas gestores de BBDD (SGBD)**

Paquete integral de software, se ejecuta en un servidor, centralizando el acceso y actuando de interfaz para el usuario.

- **Funciones**
  - Creación y mantenimiento de BBDD
  - Control de acceso
  - Manipulación de datos
  - Evitar redundancias
  - Mantener la integridad
  - Optimizar consultas



- **Ventajas:**

- Distribución geográfica
- Disponibilidad y tolerancia a fallos
- Escalabilidad
- Flexibilidad
- Localización de datos
- Gestión de red
- Tolerancia a fallos

**Conceptos:**

- Fragmentación: Horizontal divide por filas, Vertical por columnas
- Replicación: Copia de fragmentos
- Asignación: Donde almacenar cada fragmento

**Retos**

- Coherencia de datos
- Latencia y rendimiento
- Seguridad
- Complejidad de gestión
- Tolerancia a fallos

**Tipos de arquitecturas**

- Arquitectura cliente servidor
- Arquitectura de pares (P2P)
- SGBDD

**Asegura la consistencia en SGBDD**

- Control de concurrencia distribuida
- Protocolos de consenso
- Replicación sincrónica y asíncrona

Two phase commit en un protocolo de BBDD distribuidas para garantizar que los nodos involucrados en la Se utiliza para asegurar la consistencia y atomicidad de los datos

Tiene 2 fases:

- Fase de preparación: Se envía una solicitud a todos los nodos preguntando si están listos para realizar
- Fase de compromiso: Si todos los nodos responden que sí el coordinador envía un mensaje de confirmación

## Relational Database Management System (RDBMS)

**unir** LA UNIVERSIDAD EN INTERNET **FORMACIÓN PROFESIONAL** **PROEDUCA**

**Sistema de gestión de la memoria**  
Encargado de decidir que parte de la memoria se dedica a cada tarea del RDBMS. Su función es que haya suficiente memoria para que el RDBMS funcione eficazmente y a la vez no desperdiciar memoria de la que necesita el Sistema Operativo para que la máquina funcione.

**Gestión de entrada y Salida**  
Consigue que los accesos a los datos sean adecuados.

**Procesador de lenguaje**  
Interpreta las instrucciones SQL (y de otros lenguajes varios) que los usuarios lanzan a la base de datos.

**Control de procesos**  
Gestiona los programas en ejecución necesarios para el funcionamiento de la base de datos.

**Control de la red**  
Controla las conexiones a la base de datos desde la red y evita problemas en caso de desconexión.

**Control de transacciones**  
Permite gestionar las transacciones (conjunto de operaciones de manipulación de datos que se pueden validar o anular).

11

Posee varios subsistemas que se encargan de gestionar cada servicio

Gestión de memoria: Decide que partes de la memoria se dedican a que tarea

Gestión de entrada y salida: Garantiza el acceso adecuado a los datos

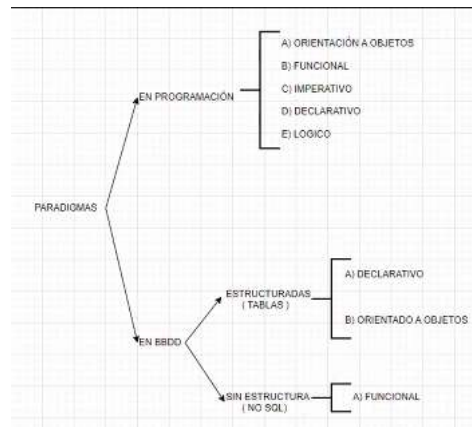
Procesador de lenguajes: Interpreta las instrucciones SQL

Control de procesos: Gestiona los programas necesarios para que la BBDD funcione

Control de la red: Controla las conexiones a BBDD

Control de transacciones: Permite gestionar las transacciones (manipulaciones de datos en BBDD).

🔥 Identifica los paradigmas de Programación



- **Paradigmas de programación:**

Paradigma: Un Enfoque o estilo de programación enfocado a resolver un tipo de problema concreto

🔥 El paradigma de programación orientado a las BBDD es el **DECLARATIVO**

- Paradigma Funcional
- Paradigma declarativo
- Paradigma Imperativo
  - Paradigma procedural
- Paradigma de programación orientada a objetos

## ▼ Fases del Diseño

- Diseño Conceptual: Estudio de recursos de la información
- Diseño Lógico: Transforma el modelo conceptual al modelo de datos
- Diseño físico: Modelo lo más eficiente posible adaptado al gestor de bases de datos

## Propuesta de diseño

### Elmasri y Navathe

- Fase 1: Recopilación de datos y análisis de requerimientos  
Se analizan las necesidades del cliente, grupos de usuarios y áreas de aplicación
- Fase 2
  - A) Diseño del esquema conceptual: Identifica entidades y atributos
  - B) Diseño de transiciones: Identifica transiciones y procesos entre los conceptos
- Fase 3
  - Elección de un SGBD:  
Se valoran los factores técnicos económicos, de servicio, organizativos y de rendimiento
- Fase 4  
Transformación del modelo de datos al modelo lógico
- Fase 5

Diseño de BBDD física: Se define como se almacenarán los datos y como se va a acceder a ellos

- Fase 6

Implementación de sistema de base de datos

Creación del sistema de BBDD,: Tablas y relaciones definidas en el modelo conceptual

Creación de archivos de BBDD

Implementación de transacciones

## **Tipos de Usuarios**

- Finales
- Programadores de BBDD
- Administrador de BBDD



**unir** LA UNIVERSIDAD EN INTERNET | FORMACIÓN PROFESIONAL | **PROEDUCA**

### Manipulación


Después de agregar la información a la base de datos, puede ser **modificada y/o eliminada**. El software de la base de datos gestiona las reglas de manipulación de los datos.

### Seguridad

Las bases de datos deben mantener la información de una forma segura y que se encuentre disponible cuando ocurra un fallo como una pérdida de un disco. Las copias de seguridad (backups) y recuperación (recovery) son los procesos más importantes para **asegurar la disponibilidad de la información**.

### Base de datos relacional

Es un tipo de base de datos que cumple con el modelo relacional, siendo ésta técnica la más utilizada en la actualidad para implementar bases de datos. Este modelo está basado en conceptos muy sencillos, permitiendo establecer relaciones (interconexiones) entre los datos (que están guardados en tablas), a través de otras conexiones, obtener información de varias entidades.



El diagrama ilustra un modelo de base de datos relacional. En el centro, hay dos cilindros, uno azul y uno verde, que representan bases de datos o servidores. Arriba de ellos, hay una red de líneas que conectan diferentes bloques de texto, representando las relaciones entre las tablas de la base de datos. Los bloques de texto contienen información sobre clientes, productos y proveedores, con campos como nombre, dirección, teléfono y correo electrónico.

6

## ▼ UF1.3 - Modelo Conceptual

- **Definimos:**

- Entidad : Objeto tangible de la vida real. Ej: Empresas, Personas, Lugares, Componentes

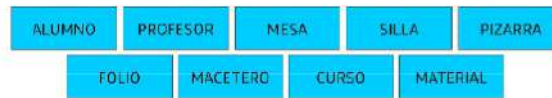
Se representan con rectángulos y en mayúsculas

- **Tipos:**

- **Fuertes:** No dependen de ninguna otra para existir
- **Débiles:** Depende de otra para existir. Notas es débil de Alumnos

### Ejemplo

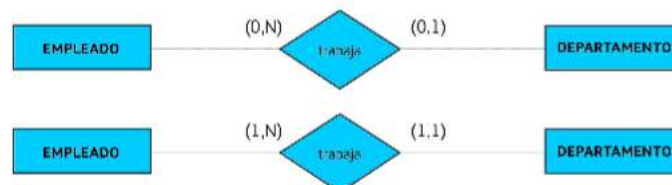
Tenemos una empresa de formación y queremos hacer una base de datos con la información recogida en el aula. Viendo lo que hay en el aula, identificamos objetos de los que se puede aglutinar información:



- Interrelaciones: Asociación entre 2 entidades (se representa en minúsculas y con un rombo)

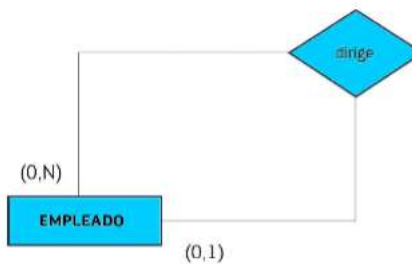


- Cardinalidad: Denota el máximo y mínimo de ocurrencias en la relación de las 2 entidades



- Grado de relación:
  - Binaria: Asocia la entidad con otra
  - Unaria: Asocia la entidad consigo mismo

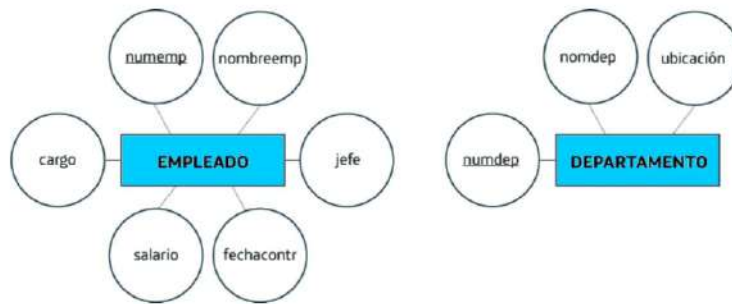
- Binarias: son las relaciones más normales o habituales y asocian dos tipos de entidad distintas. Los ejemplos que hemos visto hasta ahora son de ese tipo.
- Unarias: son las que asocian un solo tipo de entidad consigo mismo, denominándose en este caso, tipo de interrelación reflexiva o recursiva.



- Correspondencia Relación: Según las cardinalidades máximas tendremos los siguientes tipos

1:1	De uno a uno
1:N	De uno a muchos
N:N	De muchos a muchos

- Atributos: Definen cuales son las características de cada entidad

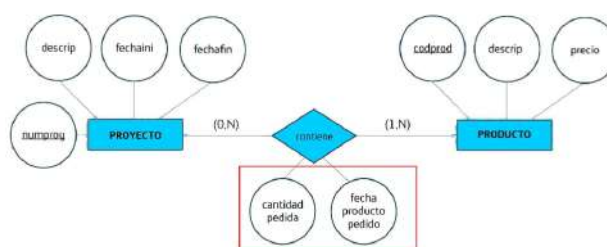


■ Representación Atributos:



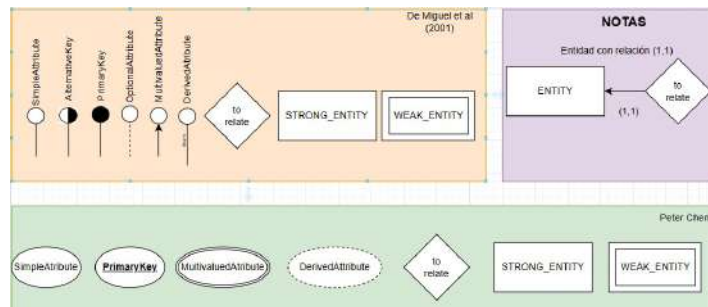
**Tipos de atributos:**

- Multivaluado: Pueden tener más de un valor para la misma entidad.
- Derivado: Valor calculado a partir de otra información ya existente
- Atributo clave: Atributo con valor distinto para cada instancia de un tipo de entidad. Una clave identifica de forma única cada entidad concreta
- Atributos opcionales: Pueden indicarse o no (en la relación)



- Dominios: Valores asignados a cada tipo de atributo, texto, alfanumérico, Entero...
  - Continuo: Comprendido entre máximo mínimo DNI, salario
  - General: valores determinados fecha, sexo
  - Semánticos: Significado del texto
  - Sintácticos: Valores sin significado

**Nomenclatura en Peter Chen y Miguel Et. Al.**



## Paso a seguir para el diseño de una BBDD relacional

1. Definir los requisitos específicos del sistema
2. Identificar las Entidades
  - a. Fuertes
  - b. Débiles
3. Relación entre entidades
4. Asignación de atributos a entidades y relaciones
  - a. Especificar dominio

## ▼ UF1.4 - Restricciones de modelo relacional

### 12 Reglas Codd

#### 1. Contexto histórico de las 12 reglas

##### 1.1 Quien fue Edgar F. Codd y que importancia tuvo en el desarrollo de las BBDD relacionales?

Edgar F. Codd fue un científico que desarrolló el **modelo relacional de bases de datos**, introduciendo el uso de **tablas** para organizar la información de manera más eficiente. Este modelo dio origen a los **Sistemas de Gestión de Bases de Datos Relacionales** y al lenguaje **SQL**. En 1985, Codd estableció las **12 reglas** para definir qué es una base de datos relacional, transformando la forma en que se gestionan los datos en la tecnología moderna.

##### 1.2 Porque formulo esas 12 reglas?Cuál era su objetivo con ellas?

Formulo estas 12 reglas para que todos los sistemas de bases de datos que quisieran denominarse relacionales tuvieran unas bases conjuntas que permitiesen desarrollar un estándar, con el objetivo de unificar el diseño y el desarrollo y este tipo de sistemas a nivel internacional.

#### 🌐 1. Regla Fundamental

- Todo sistema debe tener tablas para gestionar los datos

#### 📄 2. Regla de la Información

- Toda la información en una base de datos relacional se presenta en tablas.

#### 🔑 3. Regla del Acceso Garantizado

- Cada dato tiene que ser accesible mediante su nombre de tabla, clave primaria y nombre de columna.

#### ❓ 4. Tratamiento Sistemático de Valores Nulos

- Los sistemas relacionales deben maneja valores nulos como tal, Null tiene que ser null!!!

## 5. Catálogo Activo Relacional

- La estructura de la base de datos debe tener un diccionario que pueda consultarse a través de SQL.

## 6. Sublenguaje de Datos

- Debe existir un lenguaje de consulta que soporte operaciones como definición de datos, creación de vistas y manipulación de datos.

## 7. Actualización de Vistas

- Si una vista puede ser teóricamente actualizada, el sistema debe ser capaz de hacerlo.

## 8. Inserción, Actualización y Borrado de Alto Nivel (Transacción)

- Las operaciones de inserción, actualización y borrado deben poder realizarse en grupos de datos.

## 9. Independencia Física de los Datos

- Los cambios en cómo se guardan los datos no deben afectar cómo se accede a ellos.

## 10. Independencia Lógica de los Datos

- Los cambios en las tablas no deben afectar las aplicaciones que usan esos datos.

## 11. Independencia de la Integridad

- Las reglas para mantener los datos correctos se definen y almacenan en la base de datos, no en las aplicaciones.

## 12. Independencia de la Distribución

- El usuario no debe notar si los datos están distribuidos en diferentes ubicaciones.

## 13. La Regla de la No Subversión

- No se debe poder usar un nivel bajo del sistema para violar reglas y restricciones de un nivel más alto.

### 1.3 Porque cada una de estas reglas es importante en un sistema de BBDD relacional?

Estas reglas garantizan que el modelo relacional cumpla con los estándares del modelo relacional asegurando la , coherencia, la flexibilidad y escalabilidad de la misma.

## 2. Impacto de las reglas en la evolución de las BBDD relacionales

### 2.1 Como crees que las 12 reglas influyeron en el desarrollo de las BBDD relacionales modernas?

El modelo relacional mejoró la eficiencia de las bases de datos, asegurando la independencia, manipulación lógica y consistencia de datos. Esto permitió que el modelo que popularizara y se desarrollasen tecnologías como SQL y los SGBDR.

- Estandarización
- Eficiencia
- Flexibilidad
- Escalabilidad

### 2.2 Cuales son las principales limitaciones de los sistemas desarrollados en base a las 12 reglas Codd frente a los sistemas de BBDD no relacionales?

- **Escalabilidad:** Menos eficiente en escalabilidad horizontal.
- **Flexibilidad:** Requiere un esquema rígido; NoSQL permite esquemas flexibles.
- **Datos no estructurados:** NoSQL maneja mejor datos no estructurados.
- **Consultas:** Relacionales son fuertes en consultas complejas; NoSQL prioriza acceso rápido.
  - Eficiencia
  - Manipulación

- Lenguaje SQL
- Restricciones principales del modelo Relacional
  - Restricción de PRIMARY\_KEY:
    - La clave primaria asegura que cada fila en una tabla sea única y no se repitan los valores
  - Restricción UNIQUE
    - Impide que los valores se repitan en diferentes filas de la misma tabla pero si pueden estar vacíos
  - Restricción NOT NULL
    - Obliga a que ciertos atributos deben ser siempre llenados, evitando valores nulos.
  - Restricción FOREIGN\_KEY
    - Utilizamos la clave primaria de la tabla con la que vamos a enlazar nuestra entidad débil
- Más restricciones del modelo relacional:
  - Default:
    - Se le asigna un valor por defecto que adoptará en caso de que no se defina
  - Assertion:
    - La condición se establece sobre elementos de distintas relaciones
  - Check:
    - Especifica una condición que debe cumplir ciertos valores
  - Triggers:
    - Acción específica que se lanza cuando una acción concreta sucede

## ▼ UF1.5 - Operaciones de manipulación

- Conceptos
  1. Generar y/o capturar
  2. Evaluar
  3. Almacenar
  4. Destruir
  5. Recuperar
  6. Clasificar
  7. Analizar
  8. Manipular
  9. Sintetizar
  10. Usar como información

### Operaciones de manipulación de BBDD

- Actualización
  - Inserción
  - Borrado
  - Modificación
- Consulta

- Consulta de datos
- Representación
  - Inserción


--	--	--	--	--	--


- Borrado


- Modificación








# UF2 - Modelo lógico

## ▼ UF2.1 - Modelo Lógico

- Operaciones que podemos realizar con nuestras tablas

✓ Editar	EST2000		EST2001			Asignatura			
	Nombre	Apellidos	ID	Nombre	Apellidos	IdA	NomAsig	DurSemanal	
	1	Ana	García López	3	Beatriz	Dominquez Galindo	1	Matematicas	180
	2	José Carlos	Román Herrera	2	José Carlos	Román Herrera	3	Lengua	50
						2	Fisica	200	
Usuarios									
IdU	Nombre	Cod							
1	Unknown17	1001							
2	Ferchu33	1000							
3	Merko2	1001							
UNION_EST2000yEST2001									
ID	Nombre	Apellidos							
1	Ana	García López							
2	José Carlos	Román Herrera							
3	Beatriz	Dominquez Galindo							
INTERSECCION_EST2000yEST2001									
ID	Nombre	Apellidos							
2	José Carlos	Román Herrera							
DIFERENCIA_EST2000yEST2001									
ID	Nombre	Apellidos							
1	Ana	García López							
SELECCION_Asignatura( DurSema > 75 )									
IdA	NomAsig	DurSemanal							
1	Matematicas	180							
2	Fisica	200							
PROYECCION_Asignatura ( IdA , NomAsig ) ( DurSema > 75 y IdA >=2)									
IdA	NomAsig								
2	Fisica								
JOIN_UsuariosYvideojuegos( Cod )									
IdU	Nombre	Cod	NomVideo						
1	Unknown17	1001	Fifa						
2	Ferchu33	1000	Counter						
3	Merko2	1001	Fifa						

### ○ UNION

Tabla formada por el total de filas de otras "n" tablas

### ○ INTERSECCIÓN

Filas coincidentes entre "n" tablas

### ○ DIFERENCIA

Filas no coincidentes entre "n" tablas

- **SELECCIÓN**

Tabla formada por parte de las de las filas de otras tablas

- **PROYECCIÓN**

Extrae un subconjunto de columnas

- **JOIN**

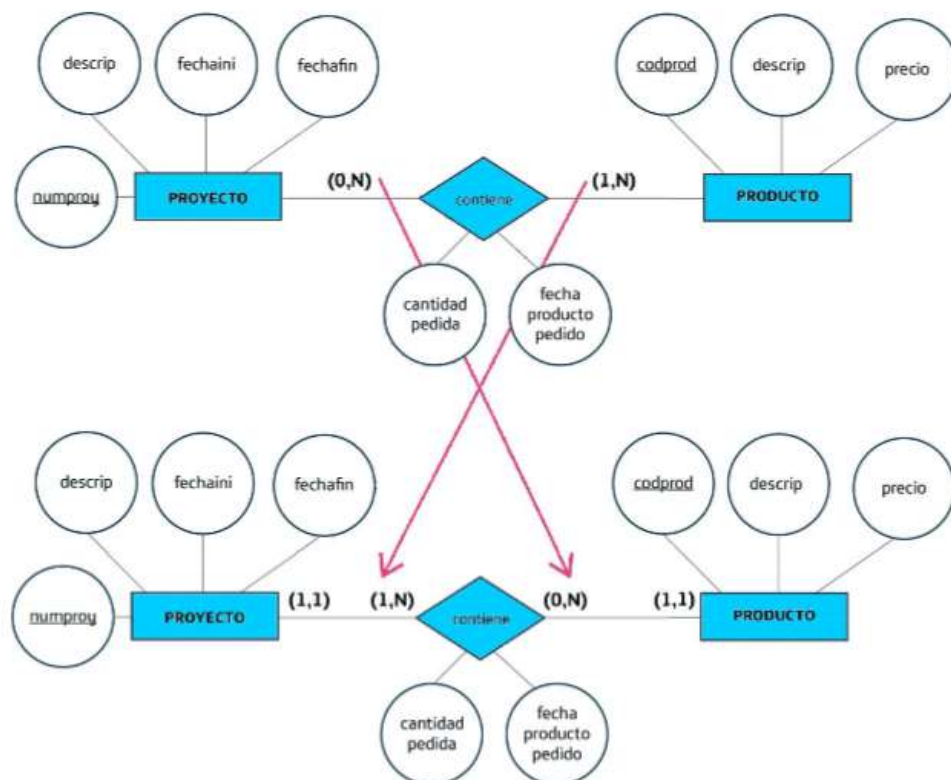
Combinar filas de una tabla con filas de otras

## Metodología

### 1. Transformar entidades y/o relaciones en tablas

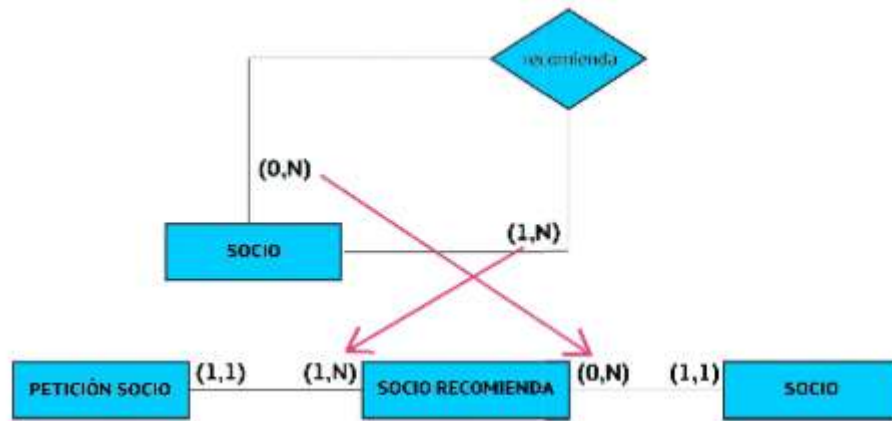
- 1.1 - Revisar las relaciones una a una
- 1.2 - Transformar las entidades N:N

Tenemos que generar una entidad intermedia (entidad débil)



- 1.3 - Eliminar las relaciones recursivas

Sustituir cada una de ellas por una entidad débil



- 1.4 - Eliminar las relaciones entre tres o más entidades

Sustituir cada una de ellas por una entidad débil

- 1.5 - Eliminar las relaciones redundantes

Una relación es redundante cuando se puede obtener esa misma información mediante otras relaciones

## 2. Asignar atributos permanentes en las tablas

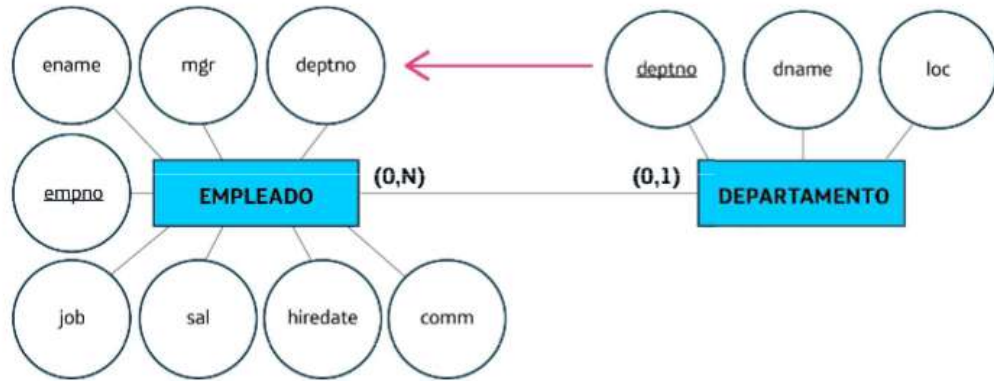
- Definir atributos:
  - Asignar cada atributo a un nombre
  - Especificar tipo de datos y dominio
  - Si es un código indicar su tipo y validación
  - Definir claves candidatas y primarias

## 3. Expandir las claves

- 3.1 - Interrelaciones 1:1

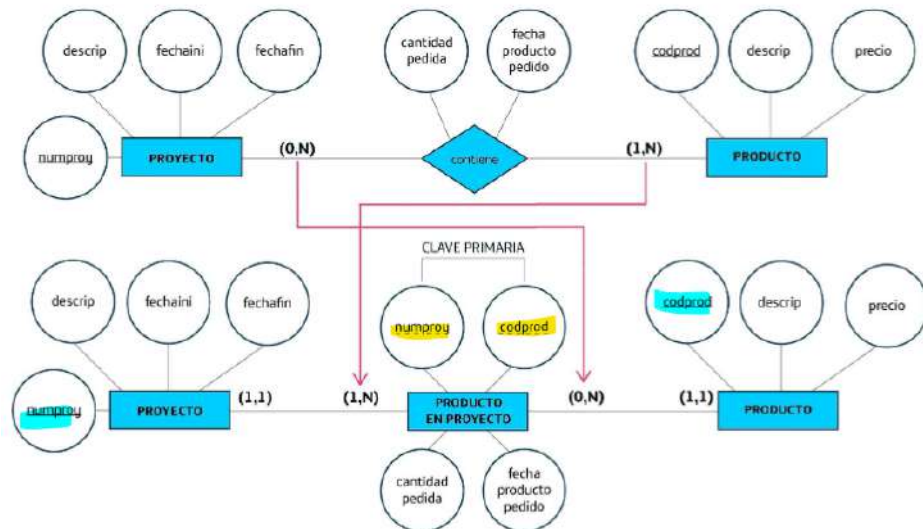
Se asigna a la entidad débil la clave primaria de la entidad más importante

- 3.2 - Interrelaciones 1:N



- 3.3 - Interrelaciones N:N

- Se migra la clave primaria (numproyecto) de la entidad PROYECTO, como clave ajena en la entidad renacida.
- Se migra la clave primaria (codproducto) de la entidad PRODUCTO, como clave ajena en la entidad renacida.
- Estas dos claves ajenas, forman la clave primaria, con las siguientes consideraciones:
- Si la suma de las dos claves ajenas da valores únicos, éstas dos forman la clave principal.
  - Si no dan valores únicos:
    - o Añadir una columna a la clave que de valor único.
    - o Crear una columna denominada "artefacto" (inventada) que proporcione unicidad.



## 2. Tratamiento de opcionalidad

**0 = opcionalidad**

**1 = obligatoriedad (al menos 1)**

## 3. Repasar las relaciones de integridad

- Integridad de entidades:

Asegurarse de que tengan clave primaria

- Integridad referencial

Tienen el mismo valor que la clave primaria.

Si la participación de la entidad hijo en la relación es total, entonces la clave ajena no admite nulos; si es parcial, la clave ajena debe aceptar nulos.

- Integridad de dominio

Si es NOT NULL o no si lo es tiene que tener dato

Restricción de dominios: Conjunto de valores que un dato puede tomar

Reglas de negocio: Operaciones realizadas sobre los datos

## 4. Normalización

## Primera Forma Normal (1FN)

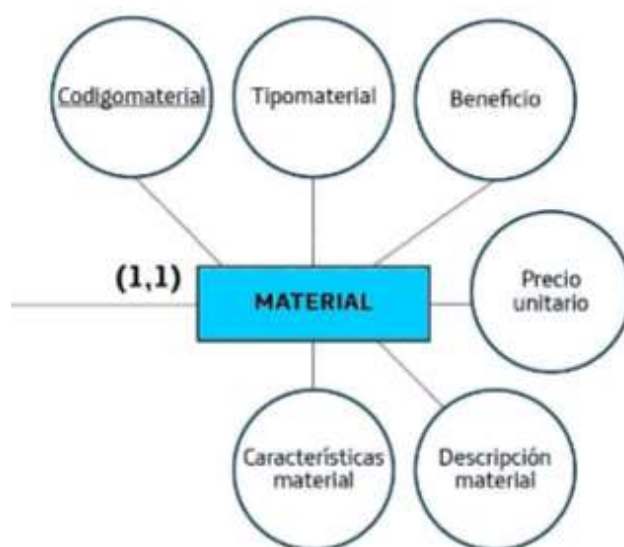
- Tiene un identificador único principal
- No existen grupos repetidos de datos
- Cada atributo corresponde a una columna

## Segunda Forma Normal (2FN)

- Se aplica generalmente a tablas relacionadas
- La PRIMARY\_KEY de estas tablas está formada por la unión de 2 FOREIGN\_KEY que hacen referencia a la PRIMARY\_KEY de las tablas a las que está asociada esa entidad débil

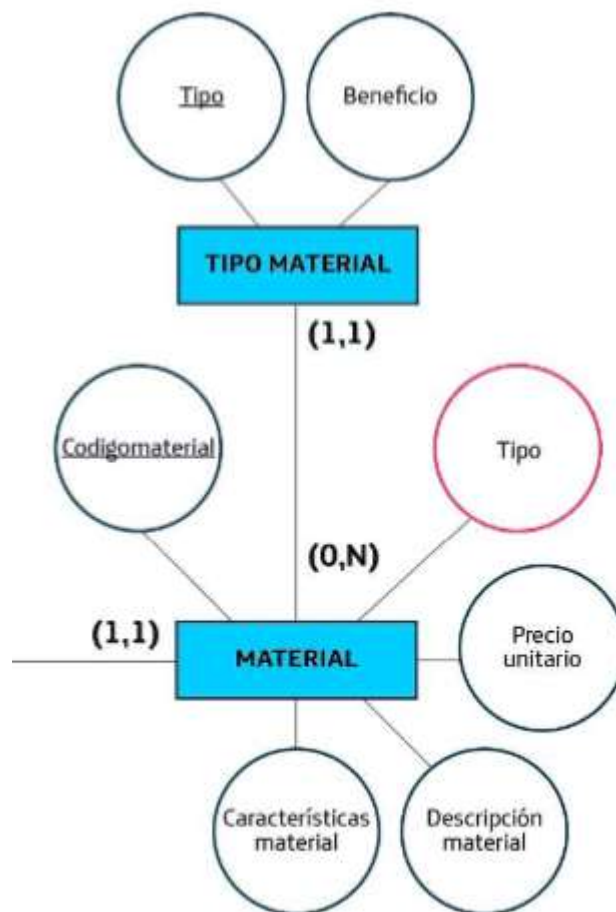
## Tercera Forma Normal (3FN)

- Todos los campos que no son claves son independientes.



- En este paso el atributo beneficio de la entidad material corresponde a la suma del tipo del material más el beneficio que obtenemos del,

por lo que en la fase de normalización elevaremos esta relación a una tabla que contendrá como FK el tipo.



### Ejemplo de Diccionario de datos del modelo expuesto

- CLIENTE
  - Codigocliente (CP, NOT NULL, numérico).
  - NombreCliente (NOT NULL, alfanumérico).
  - Descripción (alfanumérico).
- PEDIDO
  - Numeropedido (CP, NOT NULL, numérico).
  - Codigocliente (ü, NOT NULL)-
  - Fechapedido (fecha).
- MATERIAL
  - Codigomaterial (CP, NOT NULL, alfanumérico).

descripcion (alfanumérico)-

- precio (numérico).
- características (alfanumérico).

Tipo (CAJ NOT NULL, numérico).

- TIPOMATERIAL

Tipo (CP, NOT NULL, numérico).

- Beneficio (numérico).

- LINEA DE MATERIAL

- Numeropedido (CA, NOT NULL, numérico),
- Codigomaterial (CAJ NOT NULL, alfanumérico),
- cantidadpedida (numérico).
- Numeropedido + Codigomaterial (CP).