***Dependencias Funcionales (DF):***

* Una DF es una restricción entre dos conjuntos de atributos de la BD.
* Formalmente explicado 🡪 una DF x🡪y entre dos conjuntos de atributos X e Y que son subconjuntos de atributos (R) de una relación (r), especifica una restricción sobre las posibles tuplas que podrían formar un estado de la relación r en R.
* La restricción nos indica que sí T1 y T2 son dos tuplas cualesquiera en la tabla r y que sí T1[X] = T2[X] entonces debe ocurrir que T1[Y] = T2[Y].
* Esto significa que los valores del componente Y(atributos del subconjunto Y) de una tupla de r dependen de los valores del componente X(atributos del subconjunto X).
* r 🡪 representa una tabla (que tiene sus varios atributos), X e Y representan un subconjunto de atributos de dicha tabla (ósea no todos los atributos de r).
* R 🡪 representa los atributos de esa tabla r.
* X 🡪 Y:
  + El valor del atributo Y está determinado por el valor del atributo X.
    - X 🡪 Y. X entonces Y.
* Si una restricción en R (atributos de la tabla) dice que no puede haber más de una tupla con un valor X en r (donde X se vuelve clave primaria), entonces X 🡪 Y para cualquier Y de R.
  + Pensemos que Y es otro subconjunto de atributos de la misma tabla.
  + Ahora, imagine que tiene una tabla que representa los datos de un alumno, id\_alumno, nombre, apellido, legajo, etc.
  + Sí id\_alumno es clave primaria (ósea X), X🡪 Y para cualquier atributo Y de la tabla. Ya que yo con la clave primaria puedo saber todos los demás datos de mi tabla, conocida la clave primaria yo puedo determinar cualquier otro valor para esa tabla.
* Si x🡪y en R, no se puede afirmar ni negar que Y 🡪 X.
  + La clave primaria puede tener determinación por claves unívocas (cumple Y 🡪 X)
  + Ejemplo donde no ocurre Y 🡪 X:
    - Tengo un id\_alumno 14, que me apunta al apellido García. Como García es un apellido y los apellidos se pueden repetir, no puedo saber el dato id\_alumno.
  + Ejemplo donde sí ocurre Y 🡪 X:
    - Tengo un DNI 412567, el DNI al ser una clave unívoca, que no se repite, puede determinar el id\_alumno (que es la clave primaria).

***Ejemplos de DF:***

Ejemplo1: NroDpto clave primaria, nombre es clave unívoca

* Departamento = (NroDpto, nombre, #empleados)
  + Nrodpto 🡪 nombre? SÍ
  + Nrodpto 🡪 #empleado? SÍ
  + Nombre 🡪 #empleado??
    - Si nombre es clave unívoca, entonces sí cumple.
    - Si nombre NO es clave unívoca, entonces no cumple.

Ejemplo2: NroEmpl es clave primaria

* Empleado = (NroEmpl, nombre, DNI, Sexo)
  + NroEmpl 🡪 nombre? SÍ.
  + NroEmpl 🡪 dni? SÍ.
  + NroEmpl 🡪 sexo? SÍ.
  + DNI 🡪 NroEmpl??
    - Cuando sí? 🡪 cuando es clave unívoca (consideración propia).
    - Que otras dependencias pueden surgir con el DNI?
      * DNI 🡪 nombre.
      * DNI 🡪 NroEmpl.
      * DNI 🡪 Sexo.

Conclusión: tanto las claves primarias como las unívocas pueden determinar a los otros atributos.

Ejemplo3: (nro\_empl, nro\_proy) 🡪 clave primaria compuesta

* Empl\_proyecto = (nro\_empl, nro\_proy, horasTrabajadas, nombre\_empleado, nombre\_proyecto)
  + (nro\_empl, nro\_proy) 🡪 horasTrabajadas?. SÍ
  + Nro\_empl 🡪 nombre\_empleado? SÍ.
  + Nro\_proyecto 🡪 nombre\_proyecto? SÍ.
  + Si seguimos el análisis:
    - (nro\_empl, nro\_proy) 🡪 nombre\_empleado? SÍ
    - (nro\_empl, nro\_proy) 🡪 nombre\_proyecto? SÍ
* Conclusión: el diseño para ésta base de datos es erróneo, ya que permite la repetición de datos.
  + Un empleado que trabaja en 10 proyectos va a estar cargado 10 veces.
  + Un proyecto que tiene muchos empleados, va a estar cargado N veces.
  + Se debe reformular el diseño.

***Dependencia Funcional Completa:***

>> *Definición:* Sí A y B son atributos de una tabla r, B depende funcionalmente de manera completa de A, sólo si B depende de A pero de ningún subconjunto de A, .

Preguntas:

* (nro\_empl, nro proy) 🡪 nombre\_empleado?
* Nro\_empl 🡪 nombre\_empleado
* ¿Cuál es la completa? Nombre\_empleado cumple la definición ya que depende de nro\_empl, siempre y cuando despedacemos (nro\_empl, nro\_proy).
* (Nro\_empl,nro\_proy) 🡪 nombre\_proyecto
* Nro\_pry 🡪 nombre\_proyecto.
* Idem anterior

***Dependencia Funcional Parcial:***

>> *Definición:* A 🡪 B es una dependencia funcional parcial si existe algún atributo que puede eliminarse de A y la dependencia continúa verificándose.

* (nro\_empl, nro proy) 🡪 nombre\_empleado?
* Nro\_empl 🡪 nombre\_empleado
* La primera es una dependencia parcial, ya que podrías eliminar un atributo de A, y la dependencia seguiría siendo correcta.
* (Nro\_empl,nro\_proy) 🡪 nombre\_proyecto
* Nro\_pry 🡪 nombre\_proyecto.
* Idem anterior

***Dependencia Funcional Transitiva:***

>>*Definición:* es una condición en la cuál A, B y C son atributos de una tabla tales que A 🡪 B y B 🡪 C, entonces C depende transitivamente de A mediante B (A 🡪 C). Dicha dependencia repite información (malo).

* Ejemplo:
  + Nro\_empleado 🡪 nombre,posición, salario, nro\_depto, nombre\_depto (explícitamente, podría no estar y sería implicito que A 🡪 C).
  + Nro\_depto 🡪 nombre\_depto.
    - A = nro\_empleado.
    - B = nro\_depto.
    - C = nombre\_depto.

***Normalización:***

>>*Definición:* técnica de diseño de BD que comienza examinando los nexos que existen entre los atributos (estudia las dependencias funcionales). La normalización identifica el agrupamiento óptimo de estos atributos, con el fin de verificar un conjunto de tablas que soporten adecuadamente los requisitos de datos de la organización, se revisan las tablas para qué todas las dependencias funcionales SEAN completas (óptimo).

>>*Propósito:* producir un conjunto de tablas con una serie de propiedades deseables partiendo desde los requisitos de datos de una organización.

* Esta técnica formal puede usarse en cualquier etapa del diseño de la BD (no es de usarse en la parte del modelo conceptual para no complicar al cliente).
* La redundancia de datos en un modelo es la causa primaria de posibles inconsistencias (que deberemos resolver).
* Primer paso para el proceso de normalización: identificar la CP y las CC de cada tabla del modelo.
* La normalización posee varias formas para resolverse que fueron surgiendo desde 1972: 1NF (Primera Forma Normal), 2NF(Segunda Forma Normal), 3NF(Tercera Forma Normal), BCNF(Boyce Codd Forma Normal) , 4NF (Cuarta Forma Normal), 5NF (Quinta Forma Normal).

Proceso 🡪 Incremental 🡪 Cada vez más restrictivo: el proceso de normalización es incremental y cada vez más restrictivo.

* Comienza con BD de forma NO normal.
* A medida que se avanza en las tablas, tiene un formato cada vez más restringido y son menos vulnerables a anomalías de actualización.
* En general, 1NF es muy restrictiva (siempre se aplica).
* El resto puede ser opcional, de hecho 2NF y 3NF normalmente se aplican siempre.

Primera Forma Normal (1NF):

* Una tabla que contiene uno o más grupos repetitivos no está en 1NF, o sea una tabla que tenga atributos polivalentes.
* Un modelo estará en 1NF si para toda tabla r del modelo, cada uno de los atributos que la forman son sí y sólo sí monovalentes.
* Ej persona = (dni, nombre, sexo, títulos\* (polivalente))
  + Solución a la polivalencia.
    - Persona = (dni, nombre, sexo)
    - Títulos = (id\_titulo, descripción)
    - Posee = (dni, id\_titulo) 🡪 relación entre Persona y Títulos.

Segunda Forma Normal (2NF):

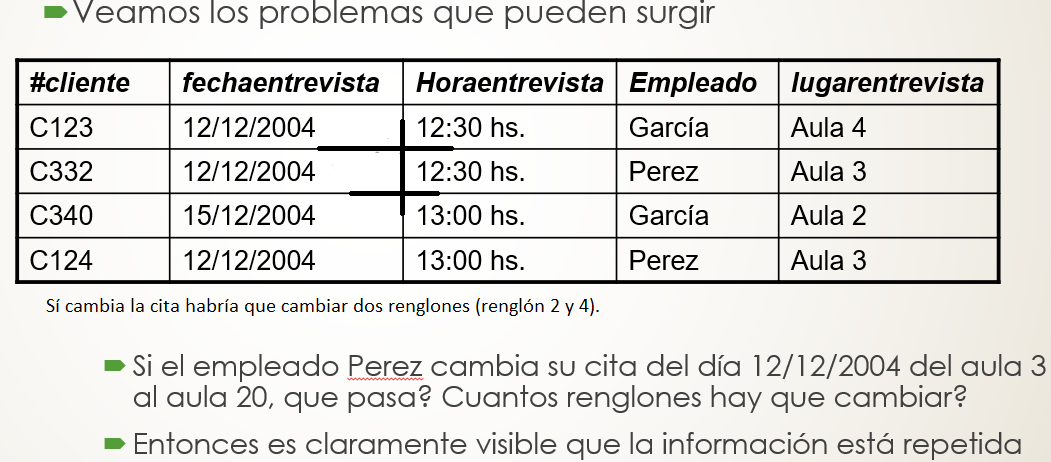
* Una tabla que tenga atributos que dependan parcialmente de otro no está en la 2NF.
* Un modelo está en 2NF sí y sólo sí está en 1NF(incremental) y para toda tabla r del modelo no existen dependencias parciales.
* Ej renta = (#cliente, #propiedad, nombrecliente, nombre propietario, monto renta, fecha inicio, duración)
* Ejemplos
  + Ejemplo1
  + Dependencias:
    - #Cliente, #Propiedad 🡪 nombreCliente, nombrePropietario, monto renta, fecha inicio, duración (DF)
    - #Cliente 🡪 nombreCliente (DP)
    - #Propiedad 🡪 nombrePropietario (DP)
    - Debo solucionar esas dos dependencias.
  + Solución:
    - Cliente = (#Cliente, nombre)
    - Propiedad = (#Propiedad, nombrePropietario)
    - Renta = (#Cliente,#Propiedad, monto renta, fecha inicio, duración)
  + Ejemplo2 🡪 empleadoproyecto = (DNI empleado, #proyecto, horas trabajadas, nombre empleado, nombre proyecto, fecha inicio proyecto, fecha inicio empleado proyecto)
  + Dependencias:
    - DNI empleado, # proyecto 🡪 horas trabajadas, nombre empleado, nombre proyecto, fecha inicio proyecto, fecha inicio empleado proyecto (DF)
    - DNI EMPLEADO 🡪 nombre empleado (DP)
    - #Proyecto 🡪 nombre proyecto (DP)
  + Solución:
    - Empleados = (DNI empleado, nombre empleado)
    - Proyectos = (# proyecto, nombre proyecto, fecha inicio proyecto)
    - Empleado\_Proyecto = (DNI empleado, # proyecto , fecha inicio empleado proyecto, horas trabajadas)

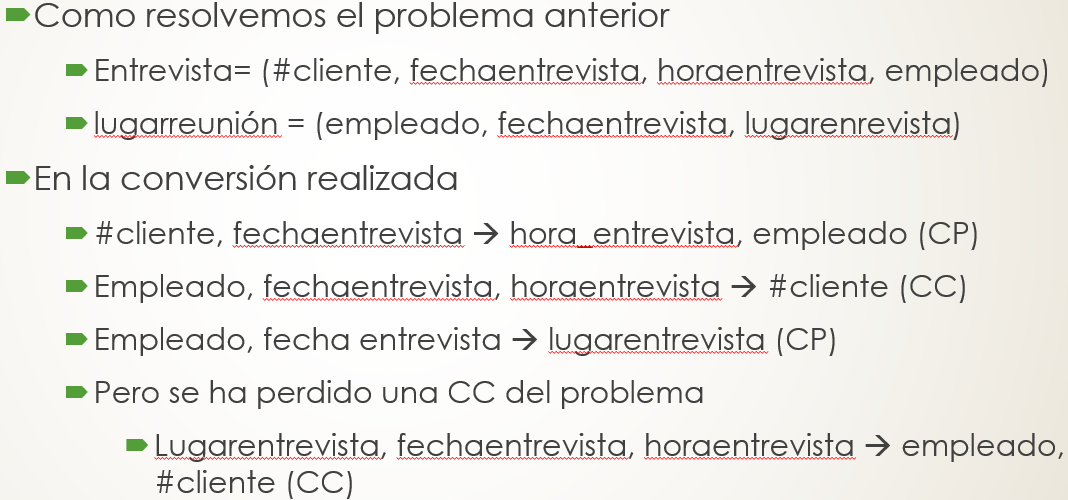
Tercera Forma Normal (3NF):

* Una tabla que tenga atributos que dependan transitivamente de otro no está en 3NF.
* Un modelo está en 3NF sí y sólo sí está en 2NF(incremental) y para toda tabla r del modelo no existen dependencias transitivas.
* Ejemplos:
  + Ejemplo1: empleado = (DNI empleado, nombre empleado, #depto, nombre depto)
  + Dependencias:
    - DNI empleado 🡪 nombre empleado, # depto, nombre depto (DF)
    - # depto 🡪 nombre depto (DT)
  + Solución:
    - Empleados = (DNI empleado, nombre empleado, #Depto)
    - Deptos = (# depto , nombre depto)
  + Ejemplo2: parcelas = (#propiedad, municipio, numero parcela, área, precio, tasa fiscal)
  + Dependencias:
    - #propiedad 🡪 municipio, numero parcela, área, precio, tasa fiscal. (DF)
    - Municipio 🡪 Tasa fiscal. (DT)
    - Area 🡪 precio. (DT)
  + Solución:
    - Parcela = (#propiedad, municipio, numero parcela, area)
    - Municipio = (municipio, tasa fiscal)
    - Áreas = (área, precio)

Boyce Codd Forma Normal (BCNF):

* Una tabla que tenga atributos que dependan de acuerdo a la definición de Boyce Codd de otro no está en BCNF.
* Un modelo está en BCNF sí y sólo sí está en 3NF (incremental) y para toda tabla r del modelo no existen dependencias de Boyce Codd.
* Dependencia de Boyce Codd: es una dependencia poco común, puede generar grandes repeticiones de información
* Algunos comentarios
  + Fue propuesta como una suavización de 3NF.
  + Pero resultó ser más restrictiva.
* Otra acepción de Boyce Codd-
  + Una tabla está en BCNF si y sólo sí todo determinante (La X de X🡪Y) es una clave candidata o primaria.
* Ejemplos:
  + Ejemplo entrevista = (#cliente, fecha entrevista, hora entrevista, empleado, lugar entrevista)
  + Dependencias:
    - #cliente, fecha entrevista 🡪 hora entrevista, empleado, lugar entrevista. (CP)
    - Empleado, fecha entrevista, hora entrevista 🡪 #cliente (CC)
    - Lugar entrevista, fecha entrevista, hora entrevista 🡪 empleado, #cliente (CC)
    - Empleado, fecha entrevista 🡪 lugar entrevista.
  + Analizando la cuarta DF:
    - El determinante no es CC o CP (solo puedo sacar el lugar de entrevista, no puedo saber ni el cliente ni la hora de entrevista, también un empleado en una misma fecha puede repetirse) 🡪 no está en BCNF.



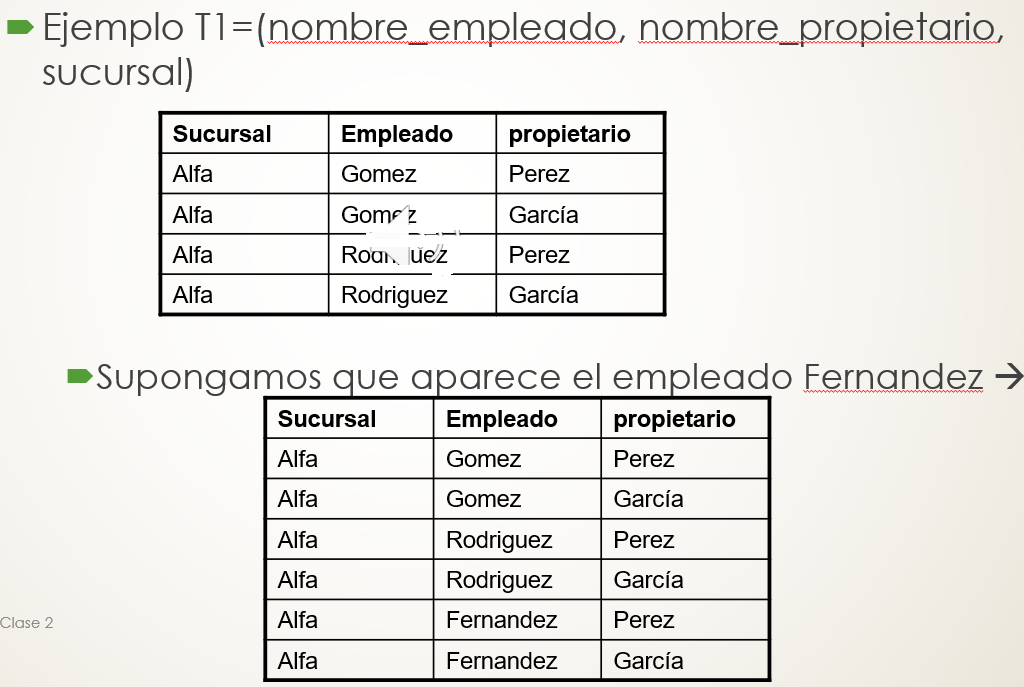


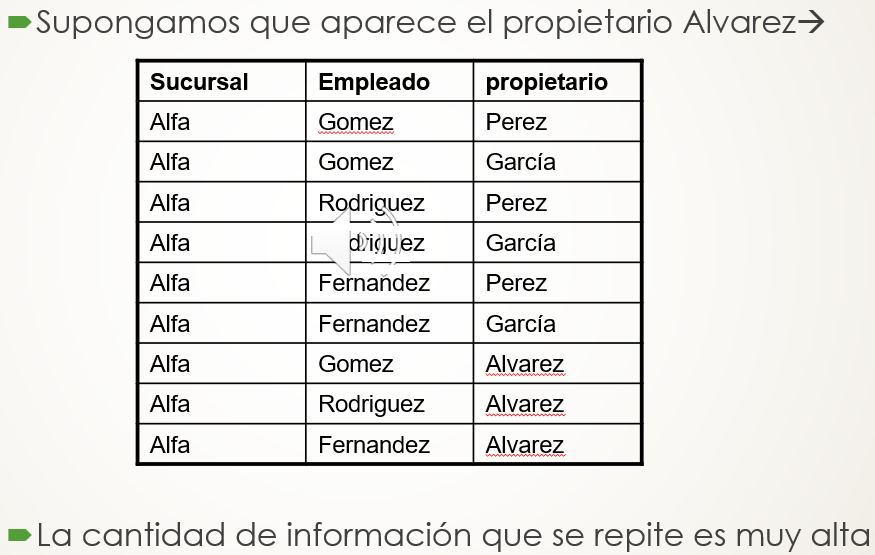
* ¿Entonces? ¿Qué se hace?
  + La decisión de si es mejor detener el proceso en 3NF o llegar a BCNF depende de dos factores:
    - La cantidad de redundancias que resulten de la presencia de una DF de Boyce Codd.
    - De la posibilidad de perder una CC con la cuál se podrían realizar muchos más controles sobre los datos.
    - ¿Qué prefiero? Redundancia y mayor control sobre los datos, o no tener redundancia.
    - Se dice que Boyce Codd es más duro ya que tengo que sacrificar algún control que tenía por una clave.
    - Es una decisión enteramente del diseñador.

***Dependencias Multivaluadas (DM):***

* La posible existencia de DM en una tabla se debe a 1NF, que impide que una tupla tenga un conjunto de valores diferentes.
* Así, si una tabla tiene dos atributos multivaluados, es necesario repetir cada valor de uno de los atributos con cada uno de los valores del otro. Así se garantiza la coherencia en la BD.
* En general, una DM se da entre atributos A,B y C en una tabla de modo que para cada valor de A hay un conjunto de valores de B y un conjunto de valores de C, sin embargo los conjuntos B y C no tienen

nada entre sí.





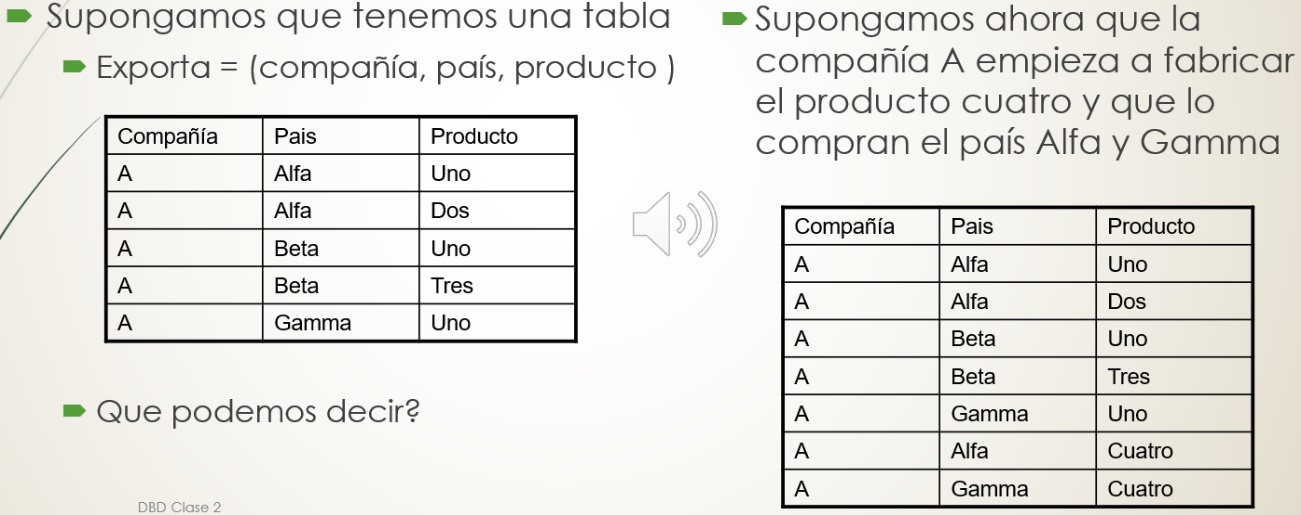
* Se dice A 🡪🡪B y que A🡪🡪C
  + Se lee “A multidetermina B” y “A multidetermina C”
* Cuando yo tengo un valor para el atributo A puedo obtener múltiples valores de B.
  + Ej, una fecha de nacimiento puede multideterminar a muchas personas que nacieron el mismo día.
* No es un problema el hecho que un atributo esté multideterminado.
  + El problema que puede surgir es que perdamos el control sobre la multideterminación.

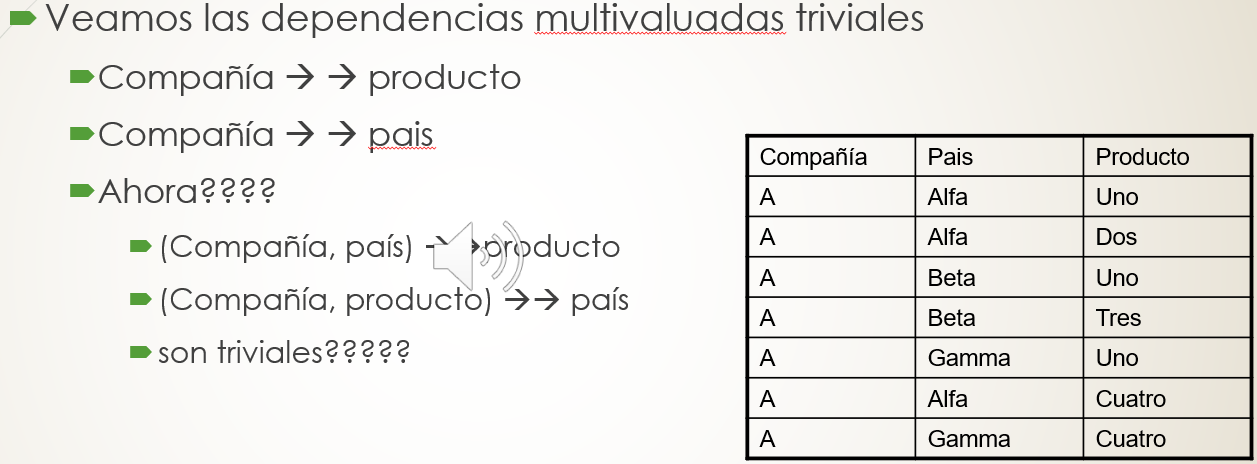
Cuarta Forma Normal (4NF):

* Un modelo está en 4NF sí y sólo sí está en BCNF y para toda tabla r del modelo sólo existen dependencias multivaluadas triviales.
  + ¿Cuáles son triviales? 🡪
    - A🡪🡪B
    - A🡪🡪C
  + Y…
    - (A, B) 🡪🡪C
    - (A, C) 🡪🡪C
  + La dependencia multivaluada no trivial genera una mayor repetición de información.
* Del ejemplo anterior:
  + (Sucursal, empleado) 🡪🡪propietario.
  + (Sucursal, propietario) 🡪🡪 empleado.
* Pero yo también podría…
  + Sucursal 🡪🡪Propietario.
  + Sucursal 🡪🡪Empleado.
* Entonces, como el par marcado en verde puede ser reducido a un solo atributo que me obtiene la MISMA información (celeste). Decimos que el par es una dependencia multivaluada NO trivial.
* Solución:
  + T1 = (sucursal, empleado)
  + T2 = (sucursal, propietario)

Quinta Forma Normal (5NF):

* Un modelo está en 5NF si está en 4NF y no existen relaciones con dependencias de combinación.
* Una dependencia de combinación es una propiedad de la descomposición que garantiza que no se generen tuplas espurias al volver a combinar las relaciones mediante una operación del álgebra relacional.
  + En otras palabras (español please):





* Se repite información
* NO caemos en 4NF.
* ¿Por qué no caemos en 4NF?
  + No existen dependencias multivaluadas NO triviales.
  + (Compañía,país) 🡪🡪 producto: me va a señalar TODOS los productos que compró un país determinado.
  + (Compañía,producto) 🡪🡪 país: me va a señalar TODOS los paises que compraron un producto determinado.
  + Esa información recién explicada no es conseguible si sacamos un atributo del par. Por lo que son pares dependientes multivaluados triviales.
* Como se repite información por la naturaleza del problema se generan dependencias de combinación.
  + Como solución al ejemplo anterior:
    - Exporta = (compañía, país) 🡪 a que países exporta una compañía.
    - Fabrica = (compañía, producto) 🡪 que productos fabrica cada compañía.
    - Compra = (país, productos) 🡪 que compra cada país
* La 5NF es MUY poco probable.
* Existen más formas normales, no las vamos a ver.