Normalización

versión: 2021.3   13 de mayo de 2021

Índice

1. ¿Qué es la normalización?

- La traducción del esquema conceptual al lógico no es única. No todas las alternativas posibles son igual de buenas.
- Es útil contar con una medida de la calidad de la agrupación de los atributos en relaciones.
- Las formas normales son un indicador de esta calidad.
- La falta de calidad de un diseño deficiente provoca problemas:
  - Redundancias de Datos.
  - Anomalías de actualización.
  - Filas incorrectas.
  - Exceso de espacio ocupado.
  - Un diseño relacional sin redundancias es menos vulnerable a inconsistencias y anomalías de actualización.
- La Forma Normal (FN) satisfecha por un esquema relacional determina:
  - Su grado de calidad respecto a esos problemas.
  - Cuanto más alta es la FN en la que está: mejor calidad.
  - Una FN se define con unas normas que debe cumplir el esquema basadas en Dependencias Funcionales (DFs) y Multivaloradas (DMs).
  - La Normalización mejora esos problemas descomponiendo el esquema relacional en otros que cumplan FN más exigentes (aquellas con numeración más alta).

2. Redundancias y anomalías

- Consideremos las siguientes tablas:

Empleados

Id-empleado	NombreE	DirecciónE	Puesto	Salario	Centro
123A	Ana Almansa	c/Argentales	Profesor	20.000	Informática
456B	Bernardo Botín	c/Barcelona	Administrativo	15.000	Matemáticas
789C	Carlos Crespo	c/Cruz	Catedrático	30.000	CC.Empresariales
012D	David Díaz	c/Daroca	Ayudante	10.000	Informática

Centros

NombreC	DirecciónC	Teléfono
Informática	Complutense	123
Matemáticas	Complutense	456
CC.Empresariales	Somosaguas	789

Empleados-Centros

Id-empleado	NombreP	DirecciónP	Puesto	Salario	Centro	DirecciónC	Teléfono
123A	Ana Almansa	c/Argentales	Profesor	20.000	Informática	Complutense	123
456B	Bernardo Botín	c/Barcelona	Administrativo	15.000	Matemáticas	Complutense	456
789C	Carlos Crespo	c/Cruz	Catedrático	30.000	CC.Empresariales	Somosaguas	789
012D	David Díaz	c/Daroca	Ayudante	10.000	Informática	Complutense	123

- Vemos que en la tabla *Empleados-Centros* se guarda mucha información. En concreto, se guarda información sobre empleados y sobre centros un poco ”mezclada”.

- Vemos que el teléfono, que va asociado al centro y no al empleado, aparece repetido para todos los empleados del mismo centro (primera y cuarta fila).
- Anomalías de actualización:
  - Anomalías de inserción:
    - Cuando se inserta sin respetar la dependencia funcional. Ejemplo: añadir un empleado a Informática con un teléfono distinto de 123.
    - Cuando se inserta el consecuente de la dependencia funcional sin el antecedente. Ejemplo: no se puede dar de alta un centro sin dar de alta un empleado.
  - Anomalías de modificación: en aquellos casos con información redundante modificar solamente algunas de sus apariciones. Ejemplo: para modificar el teléfono de Informática es preciso modificar todas sus apariciones.
  - Anomalías de eliminación: cuando se eliminan todas las filas redundantes de una dependencia funcional. Ejemplo: si se eliminan todos los empleados de un centro también se elimina el centro.

3. Conceptos necesarios

3.1. Notación

Necesitamos representar de forma adecuada tanto la estructura de la tabla como las filas de ella. Seguiremos la siguiente notación:

- Consideremos la siguiente relación:

$$profesor(id, nombre, salario, dpto, edificio, presupuesto)$$

- Utilizaremos la letra r para referirnos a un nombre de relación cualquiera.
- Utilizaremos letras mayúsculas para referirnos a nombres de atributos cualesquiera:

$$r(A, B, C, D, ...)$$

- Denotamos los conjuntos de atributos mediante letras griegas α, β, γ, etc.
- Utilizamos la letra ρ para denotar el conjunto de todos los atributos de una relación r.
- Por tanto, en lugar de escribir $r(A, B, C, D, ...)$, escribiremos $r(\rho)$.
- Una instancia de una relación r es un conjunto de tuplas que se adapta al esquema de dicha relación. Se representa de manera tabular.
- Denotamos por t una tupla cualquiera de la instancia.
- La notación $t[\beta]$ representa la proyección de la tupla t sobre los atributos del conjunto β.
- Ejemplo: $t_1[nombre, salario] = (Pepe, 1000)$

3.2. Dependencias funcionales

3.2.1. Claves

- En la bases de datos es habitual imponer una serie de restricciones sobre los datos de una tabla. Por ejemplo:
    - A cada profesor le corresponde un único identificador.
    - Cada profesor pertenece a un único departamento.
    - Cada departamento se encuentra situado en un único edificio.
- Una instancia de r que satisface las restricciones impuestas es una *instancia legal* de r.
- ¿Cómo se expresan formalmente estas restricciones?
    1. Claves.
    2. Dependencias funcionales.
- *Superclave*:
    - Sea $r(\rho)$ una relación, y ρ su conjunto de atributos.
    - Un subconjunto $\alpha \subseteq \rho$ es una superclave de $r(\rho)$ si para par de tuplas t, t' de cualquier instancia legal de $r(\rho)$ se cumple:
    $$t[\alpha] = t'[\alpha] \Rightarrow t[\rho] = t'[\rho]$$
    Es decir, la igualdad de valores para un conjunto de campos determina la igualdad de una tupla completa.
- *Clave candidata*: Un subconjunto $\beta \subseteq \rho$ es una clave candidata de $r(\rho)$ si β es superclave y no existe ningún subconjunto estricto $\alpha \subset \beta$ tal que α sea superclave. Es decir, si quitamos algún campo a β ya no es clave.
- *Clave primaria*: cualquier clave candidata.

3.2.2. Dependencia funcional

- Una instancia de $r(\rho)$ satisface la dependencia funcional $\alpha \to \beta$ si para cada par de tuplas t, t' de la instancia se cumple:
$$t[\alpha] = t'[\alpha] \Rightarrow t[\beta] = t'[\beta]$$
Es decir, el valor de un conjunto de campos determina el valor de otro conjunto de campos.
- ¿De dónde salen las dependencias funcionales?:
    - Muchas son obvias (las dedice el diseñador).
    - Otras son inferidas.

4. Formas normales

- La forma normal de una tabla se refiere a:
    - La forma normal más exigente que satisface dicha tabla.
    - Representa el grado o nivel hasta donde se ha normalizado.
- La forma normal de una BD se refiere a la forma normal más exigente que satisfacen *todas* sus tablas.

- Las formas normales más habituales, por orden ascendente de exigencia de las propiedades deseadas, son:

Forma normal	Nivel de restricción
Primera (1FN)	Muy poco restrictiva
Segunda (2FN)	
Tercera (3FN)	
Boyce/Codd (FNBC)	
Cuarta (4FN)	
Quinta (5FN)	
Sexta (6FN)	Muy restrictiva

- En general, los diseños prácticos exigen, al menos, 3FN.

4.1. Primera forma normal

- Un tabla está en *primera forma normal* (1FN) si los dominios de sus atributos sólo pueden ser atómicos.
- De esta forma se evitan multivalorados y compuestos.
- Esta restricción se considera parte de la definición formal del Modelo Relacional y de SQL. Es impuesta al pasar del modelo de Entidad/Relación al modelo relacional.
- Ejemplo:
  - Consideremos la tabla *Centros*. Es razonable pensar que un centro puede tener muchos teléfonos.

    Centros:

NombreC	DirecciónC	Teléfonos
Informática	Complutense	123, 321, 213
Matemáticas	Complutense	456
CC. Empresariales	Somosaguas	789, 987

    Evidentemente no está en forma normal. Cuestión: ¿cómo representar los teléfonos?.
  - Veamos varias formas de representar los valores de estos atributos multivaluados. Por supuesto, todo está en 1FN aunque no todo posee la misma calidad. Por lo tanto, 1FN es muy insuficiente por sí sola para determinar un diseño relacional con calidad.
  - Solución 1: Eliminar el atributo *Teléfonos* y crear una nueva relación que asocie en cada fila un centro con un teléfono.

    Centros:

NombreC	DirecciónC
Informática	Complutense
Matemáticas	Complutense
CC. Empresariales	Somosaguas

    Teléfonos:

NombreC	**Teléfono**
Informática	123
Informática	321
Informática	213
Matemáticas	456
CC. Empresariales	789
CC. Empresariales	987

Consecuencias:

- ◦ La clave de la 1ª relación debe formar parte de clave de la 2ª relación.
- ◦ Suceden anomalías cuando se borra un centro (en la tabla *Centros*) y olvidamos borrar los teléfonos asociados.
- ◦ La integridad referencial (FK) asegura evitar estas anomalías. El *NombreC* de *Teléfonos* se hace FK con apunte a *NombreC* de Centros.

- Solución 2: ampliar la clave de la relación de manera que incluya al atributo multivalorado.

NombreC	DirecciónC	**Teléfono**
Informática	Complutense	123
Informática	Complutense	321
Informática	Complutense	213
Matemáticas	Complutense	456
CC. Empresariales	Somosaguas	789
CC. Empresariales	Somosaguas	987

Consecuencias:

- ◦ Inconveniente: añade redundancia que provoca anomalías.

- Solución 3: si se conoce la cardinalidad máxima del atributo multivalorado se pueden crear tantas columnas como la cardinalidad máxima

NombreC	DirecciónC	Teléfono1	Teléfono2	Teléfono3
Informática	Complutense	123	321	213
Matemáticas	Complutense	456	null	null
CC. Empresariales	Somosaguas	789	987	null

Consecuencias:

- ◦ Inconveniente: uso de valores null.

- Si el atributo multivalorado es compuesto, por ejemplo, representar varias direcciones para un empleado: *Empleados(Id_empleado, NombreP,{Direcciones(Calle, Ciudad, CódigoPostal)})*.

- Esta relación se puede descomponer en dos:
  - *Empleados(Id_empleado, NombreP)*
  - *DireccionesP(Id_empleado, Calle, Ciudad, CódigoPostal)*

- Este procedimiento de desanidamiento se puede aplicar recursivamente a cualquier relación con atributos multivalorados:
  - teniendo en cuenta que es necesario propagar:
    - ◦ la clave de la relación original a la clave de la nueva relación
    - ◦ que contiene, además, la clave que identifica unívocamente al atributo multivalorado.

4.2. Segunda forma normal

- Ejemplo:
  - Consideremos la tabla:

Personal-proyectos

Id-empleado	**NúmeroP**	Horas	NombreE	NombreP
123A	P-1	16	Ana Almansa	Proyecto 1
012D	P-1	8	David Díaz	Proyecto 1
012D	P-2	4	David Díaz	Proyecto 2

NO está en 2ª FN pero sí en 1ªFN.

- Dependencias funcionales:
  - $(Id - empleado, NumeroP) \rightarrow_{DF1} Horas$
  - $Id - empleado \rightarrow_{DF2} NombreE$
  - $NumeroP \rightarrow_{DF3} NombreP$
- Problema: Todos los atributos no dependen de la PK completa. Alguno solo de parte de ella (de algún atributo, no de todos).
- Existen anomalías de actualización causadas por DF2 y DF3. Como sus antedecentes no son clave, puede haber varias filas con los mismos valores para estas dependencias funcionales.

- La 2FN evita este tipo de anomalías. Se basa en el concepto de *Dependencia funcional completa.*

- *Dependencia funcional completa*: la dependencia funcional $\alpha \rightarrow \beta$ es completa si no hay dependencia funcional en $\alpha - \{A_i\} \rightarrow Y$ para algún $A_i \in \alpha$.

- *Dependencia funcional parcial*: la dependencia funcional $\alpha \rightarrow \beta$ es parcial si hay dependencia funcional en $\alpha - \{A_i\} \rightarrow \beta$ para algún $A_i \in \alpha$.

- Un tabla está en 2FN si cada atributo que no forme parte de ninguna clave candidata depende funcional y completamente de cada clave candidata.

- ¿Cómo lograr que una tabla esté en 2FN?:

  - El procedimiento es dividir la tabla en tantas nuevas tablas como DFs que no sean completas.
  - El ejemplo anterior se traduce en:
    DF1: *PP1(**Id-empleado**,**NúmeroP**,Horas)*
    DF2: *PP2(**Id-empleado**,NombreE)*
    DF3: *PP3(**NúmeroP**,NombreP)*
  - Este procedimiento asegura que el resultado está, al menos, en segunda forma normal.

4.3. Tercera forma normal

- Consideremos la siguiente tabla:

Empleados-departamentos:

Id-empleado	NombreE	DirecciónE	CódigoD	NombreD	DirectorD
123A	Ana Almansa	c/ Argentales	DS	Sistemas	999Z
012D	David Díaz	c/ Daroca	DS	Sistemas	999Z

- Existen dos dependencias funcionales "claras":

  - $Id - empleado \rightarrow_{DF1} (NombreE, DireccionE, CodigoD)$
  - $CodigoD \rightarrow_{DF2} (NombreD, DirectorD)$

- Existe una dependencia funcional adicional: $Id - empleado \rightarrow_{DF3} (NombreD, DirectorD)$ a través de la transitividad de las anteriores.

- *Dependencia funcional transitiva*: la dependencia funcional $\alpha \rightarrow \beta$ es transitiva si existe un conjunto de campos γ que cumplen:

  1. juntos no forman una clave candidata,
  2. no son subconjunto de ninguna clave candidata y

3. se cumple que $\alpha \to \gamma$ y $\gamma \to \beta$.

- Un tabla está en 3FN si:
  1. satisface la segunda forma normal y
  2. todos los atributos que no forman parte de una CC no dependen transitivamente de ninguna CC.
- El procedimiento para normalizar esta relación consiste en descomponerla en los atributos definidos por la dependencia funcional responsable de la transitividad.
- En este ejemplo se descompone en dos tablas:
  - Para $\to_{DF1}$: *ed1(**Id-empleado**,NombreE,DireccionE,CodigoD)*
  - Para $\to_{DF2}$: *ed2(**CodigoD**,NombreD,DirectorD)*
- Notas: Ni *NombreE* ni *NombreD* se consideran únicos, es decir, en nuestra base de datos puede haber dos personas con el mismo nombre (Javier García García, por ejemplo) y dos departamentos con el mismo nombre.

4.4. Resumen de las tres primeras formas normales

- Una tabla está en FN1 si todos sus campos son atómicos.
- Una tabla está en FN2 si está en FN1 y todos sus campos dependen de la clave entera:
  - Si la clave solo tiene un campo necesariamente está en FN2.
  - Si un campo depende de uno, o varios campos de la clave, pero no de todos los campos de la clave no está en FN2.
- Una tabla está en FN3 si está en FN2 y todos sus campos que no forman parte de la clave dependen directamente, y no a través de otro campo no clave, de la clave.

4.5. Forma normal de Boyce-Codd (FNBC)

4.5.1. Cierre de un conjunto de dependencias

- Para entender la forma normal de Boyce-Codd es preciso introducir nuevos conceptos sobre dependencias funcionales.
- Propiedades de las DFs:
  - Una dependencia funcional $\alpha \to \beta$ es trivial si y sólo si $\beta \subseteq \alpha$:
    - Por ejemplo: $AB \to A$, $C \to C$.
    - Las DFs triviales siempre se satisfacen por cualquier instancia.
  - El hecho de que un conjunto de atributos α sea superclave de una relación $r(\rho)$ se puede expresar mediante la dependencia funcional $\alpha \to \rho$.
  - Podemos añadir atributos en el lado izquierdo o eliminar en el lado derecho de una DF sin alterar su satisfactibilidad:
    - Si se cumple $\alpha \to \beta$, también se cumple $\alpha\gamma \to \beta$
    - Si se cumple $\alpha \to \beta\gamma$, también se cumple $\alpha \to \beta$
- Cierre de un conjunto de DFs:
  - Sea $r(\rho)$ una relación, y F un conjunto de DFs.

- Decimos que F implica la DF $\alpha \rightarrow \beta$ si en toda instancia de $r(\rho)$ en la que se satisfagan las dependencias funcionales de F, también se satisface la dependencia $\alpha \rightarrow \beta$.
  - Ejemplo: El conjunto $\{A \rightarrow B, B \rightarrow C\}$ implica $A \rightarrow C$.
- Sea F un conjunto de DFs de una relación $r(\rho)$. El cierre de F (escrito F^*) es el conjunto de DFs que están implicadas por F.
- Se puede obtener el cierre de cualquier conjunto F mediante los tres axiomas de Armstrong:
  - Reflexividad: Si $\beta \subseteq \alpha$, entonces se cumple $\alpha \rightarrow \beta$.
  - Aumentatividad: Si $\alpha \rightarrow \beta$ entonces $\alpha\gamma \rightarrow \beta\gamma$ para cualquier conjunto γ.
  - Transitividad: Si $\alpha \rightarrow \beta$ y $\beta \rightarrow \gamma$, entonces $\alpha \rightarrow \gamma$.
- Estos tres axiomas son correctos y completos:
  - Correctos: no dan lugar a DFs incorrectas.
  - Completos: son suficientes para obtener F^*.
- Aunque las reglas de reflexividad, aumentatividad y transitividad son suficientes para obtener F^*, suele ser útil la aplicación de otras reglas que se deducen de las primeras:
  - Union: Si $\alpha \rightarrow \beta$ y $\alpha \rightarrow \gamma$, entonces $\alpha \rightarrow \beta\gamma$.
  - Descomposición: Si $\alpha \rightarrow \beta\gamma$, entonces $\alpha \rightarrow \beta$ y $\alpha \rightarrow \gamma$.
  - Pseudotransitividad: Si $\alpha \rightarrow \beta$ y $\gamma\beta \rightarrow \delta$, entonces $\alpha\gamma \rightarrow \delta$.
- Ejemplo:
  - Consideremos la tabla:

id	nombre	salario	dpto	edificio	presupuesto
INF1	Laura Estévez	álgebra	1600	FM	30000
INF2	Juan Herrero	Sistmas inf.	1550	FM	25000
INF3	Javier Guzmán	álgebra	1600	FM	30000

  - Sea $F = \{(id \rightarrow nombre, dpto), (id \rightarrow salario), (dpto \rightarrow edificio), (dpto \rightarrow presupuesto)\}$
  - Enumera algunas de las DFs que forman parte de F^*.
- Calculo de F^*:
  - Existe un algoritmo para calcular F^*.
  - Consiste en la aplicación exhaustiva de los tres axiomas de Armstrong.
  - Este algoritmo no se utiliza en la práctica, ya que el conjunto F^* es demasiado grande.
  - Existe una manera de saber si una dependencia funcional $\alpha \rightarrow \beta$ pertenece a F^* sin tener que enumerar todo el conjunto F^*.
  - La herramienta utilizada será el cierre de un conjunto de atributos.
- Cierre de un conjunto de atributos:
  - Sea $\alpha \subseteq \rho$ un conjunto de atributos en $r(\rho)$ y un conjunto F de dependencias funcionales. El cierre de α (denotado α^*) bajo F es el conjunto de atributos de $r(\rho)$ que está funcionalmente determinado por α.
  - Es decir, el mayor conjunto de atributos posible β tales que $\alpha \rightarrow \beta \in F^*$
  - Ejemplo: Sea $F = \{A \rightarrow CE, C \rightarrow D, CE \rightarrow B\}$ entonces $A^* = \{A, C, E, D, B\}$

- Ejercicio. Dado el siguiente conjunto de dependencias:

  $A \rightarrow B$

  $A \rightarrow C$

  $CG \rightarrow H$

  $CG \rightarrow I$

  $B \rightarrow H$

Calcula $\{AG^*\}$

- Utilidades del cierre de un atributo:

  1. Averiguar si α es superclave de $r(\rho)$: α es superclave si y sólo si $\alpha^* = \rho$.
  2. Comprobar si $\alpha \rightarrow \beta$ pertenece a F^*: $\alpha \rightarrow \beta \in F$ si y sólo si $\beta \subseteq \alpha^*$.
  3. Calcular F^*: para cada $\alpha \subseteq \rho$, y para cada conjunto $\beta \subseteq \alpha^*$, la dependencia $\alpha \rightarrow \beta$ pertenece a F^*.

- Ejemplo:

  - Consideremos la relación:

id	nombre	salario	dpto	edificio	presupuesto
INF1	Laura Estévez	álgebra	1600	FM	30000
INF2	Juan Herrero	Sistmas inf.	1550	FM	25000
INF3	Javier Guzmán	álgebra	1600	FM	30000

  - Dados: $(id \rightarrow nombre, dpto), id \rightarrow salario, dpto \rightarrow edificio, dpto \rightarrow presupuesto$
  - ¿Es *id* superclave?. $id^* = \{id, nombre, dpto, salario, edificio, presupuesto\}$. Por tanto, *id* es superclave.
  - ¿se deduce $dpto \rightarrow edificio, salario$, a partir de estas DFs?. $dpto^* = \{dpto, edificio, presupuesto\}$. Como $salario \notin dpto^*$, no se deduce.

4.5.2. Formulación de la forma normal de Boyce-Codd (FNBC)

- La FNBC es más estricta que la reformulación de la 3FN:

  - La FNBC evita otras redundancias que la 3FN no puede.
  - Pero la FNBC no siempre es posible conseguirla.

- Un esquema de relación $r(\rho)$ está en forma normal de Boyce-Codd con respecto a un conjunto de dependencias funcionales F si y sólo si para todo $\alpha \rightarrow \beta \in F^*$ se cumple uno de los siguientes requisitos:

  - $\alpha \rightarrow \beta$ es una DF trivial.
  - α es superclave de ρ.

- Ejemplo:

  - Consideremos la siguiente relación *r(profesor,DNI-profesor,asignatura)*.
  - Sea F formado por: $profesor \rightarrow DNI-profesor$ y $DNI-profesor \rightarrow profesor$
  - Sin embargo, la relación no está en BCNF, ya que las siguientes relaciones pertenecen a F^*: $profesor \rightarrow DNI-profesor$ y $DNI-profesor \rightarrow profesor$
  - Ninguna de ellas es trivial, y ni *profesor* ni *DNI-profesor* forman superclaves.

- Transformación a BCNF:

  - Partimos de una relación $r(\rho)$ y un conjunto F de dependencias funcionales.
  - Paso 1: Determinar una dependencia $\alpha \rightarrow \beta \in F^*$ que no cumpla las condiciones de la BCNF. Si no hay, finalizar.
  - Paso 2: Descomponer $r(\rho)$ en dos relaciones:
    - $r1(\alpha \bigcup \beta)$

```
para cada α ⊆ ρ hacer:
    calcular α* según F
    si α* no contiene atributos de (ρ–α), o bien
       ρ ⊆ α* entonces pasar al siguiente α.
    sino devolver α → (α* – α) ∩ ρ
devolver nada
```

Figura 1: Test sobre BCNF

        - $r2(\rho - (\beta - \alpha))$
    - Paso 3: Aplicar recursivamente el proceso de transformación a *r1* y a *r2* por separado.

- Ejemplo:
    - Sea $r(A, B, C, D, E)$ y el siguiente conjunto de dependencias funcionales: $F = \{A \to B, BC \to D\}$
    - La dependencia $A \to B$ no cumple las condiciones de la BCNF:
        - $A \to B$ no es trivial.
        - A no es superclave porque $A^* = \{A, B\}$
    - Por tanto, descomponemos la tabla (A, B, C, D, E) en dos tablas: (A, B) y (A, C, D, E).
    - La relación $r1(A, B)$ ya está en BCNF.
    - En general, cualquier relación con dos atributos está en BCNF.
    - Pasamos a la tabla $r2(A, C, D, E)$.
    - ¿Existe una DF en F^* que no cumpla las dos condiciones de BCNF?.
    - No podemos encontrarla en F, ya que $F = \{A \to B, BC \to D\}$, y ninguna de estas dos dependencias tiene sentido en (A, C, D, E).
    - Sin embargo, la dependencia $AC \to D$ está contenida en el cierre de F y no cumple ninguna de las dos condiciones porque:
        - $AC \to D$ no es trivial.
        - AC no es superclave en (A, C, D, E), porque $\{AC\}^* = \{A, C, B, D\}$. ¿Cómo hemos obtenido esta dependencia? Lo veremos a continuación.
    - Descomponemos *(A,C,D,E)* en dos: *(A,C,D)* y *(A,C,E)*.
    - Tanto *(A, C, D)* como *(A, C, E)* están en BCNF. ¿Por qué lo sabemos? Lo veremos a continuación.
    - Por tanto, hemos finalizado. Resultado de la descomposición: *(A, B) (A,C,D) (A,C,E)*

- Transformación a BCNF:
    - ¿Cómo saber si existe una dependencia en F^* que no cumple las condiciones de la BCNF sin necesidad de calcular F^*?
    - Para la descomposición de la relación inicial (primera llamada recursiva) puede demostrarse que si F no contiene DFs que violen las condiciones de BCNF, entonces F^* tampoco las tiene.
    - Sin embargo, esto no es cierto para las relaciones que resultan de descomponer la relación inicial en dos. Es necesario aplicar un algoritmo más sofisticado.
    - Dada una relación $r(\rho)$ y un conjunto de dependencias funcionales F, queremos saber si existe una dependencia en F^* que no cumpla las condiciones de BCNF. Ver figura 1.

- Ejemplo:
    - Sea *(A, B, C, D, E)*

- Volvemos al ejemplo anterior: $F = \{A \to B, BC \to D\}$
- Como estamos descomponiendo la relación inicial, sólo es necesario buscar dependencias funcionales que no cumplan las condiciones de BCNF en F.
  - En nuestro caso, $A \to B$ no cumple las condiciones.
- Para (A, B):

  $\{A\}^* = \{A, B\}$ contiene todo $\{A, B\}$ $\{B\}^* = \{B\}$ no contiene atributos de $\{A, B\} - \{B\}$
  $\{A, B\}^* = \{A, B\}$ contiene todo $\{A, B\}$

  - Aplicamos el algoritmo de búsqueda de DFs que violen las condiciones de BCNF en la tabla *(A, B)*
  - Por tanto *(A,B)* está en BCNF. Eso ya lo sabíamos, porque cualquier relación con dos atributos está en BCNF.
- Para *(A, C, D, E)*:
  - Aplicamos el algoritmo de búsqueda de DFs que violen las condiciones de BCNF.
    $\{A\}^* = \{A, B\}$
    $\{C\}^* = \{C\}$
    $\{D\}^* = \{D\}$
    $\{E\}^* = \{E\}$
    $\{A, C\}^* = \{A, C, B, D\}$
    $\{A, D\}^* = \{A, D, B\}$
    $\{A, E\}^* = \{A, E, B\}$
    $\{C, D\}^* = \{C, D\}$
    $\{C, E\}^* = \{C, E\}$
    $\{D, E\}^* = \{D, E\}$
    $\{A, C, D\}^* = \{A, C, D, B\}$
    $\{A, C, E\}^* = \{A, C, E, B, D\}$
    $\{A, D, E\}^* = \{A, D, E, B\}$
    $\{C, D, E\}^* = \{C, D, E\}$
    $\{A, C, D, E\}^* = \{A, C, D, E, B\}$
  - Aplicamos el algoritmo de búsqueda de DFs que violen las condiciones de BCNF: $\{A, C\}^* = \{A, C, B, D\}$
  - La DF que no cumple las condiciones de BCNF es: $AC \to D$
  - Descomponemos *(A, C, D, E)* en *(A, C, D)* y *(A, C, E)*.
- Comprobamos *(A, C, D)*:

  $\{A\}^* = \{A, B\}$

  $\{C\}^* = \{C\}$

  $\{D\}^* = \{D\}$

  $\{A, C\}^* = \{A, C, B, D\}$

  $\{A, D\}^* = \{A, D, B\}$

  $\{C, D\}^* = \{C, D\}$

  Está en BCNF.

- Comprobamos *(A, C, E)*:

  $\{A\}^* = \{A, B\}$

  $\{C\}^* = \{C\}$

  $\{E\}^* = \{E\}$

$\{A, C\}^* = \{A, C, B, D\}$
$\{A, E\}^* = \{A, E, B\}$
$\{C, E\}^* = \{C, E\}$
$\{A, C, E\}^* = \{A, C, E, B, D\}$

Está en BCNF.