75.15 / 75.28 / 95.05 - Base de Datos

Teoría del Diseño Relacional
Parte II: Algoritmos

Alberto Fasce, Mariano Beiró

Dpto. de Computación - Facultad de Ingeniería (UBA)

3 de mayo de 2022

Topics

Algoritmos de normalización

- Supongamos que el diseñador de la base de datos definió un conjunto de dependencias funcionales F a partir de la semántica.
- A partir de dichas dependencias, nos interesa generar una descomposición lo menos redundante posible, preservando la información y las dependencias funcionales.
- A continuación describiremos una serie de algoritmos para convertir un esquema de base de datos a 3FN y a FNBC.
- Para ello será necesario introducir algunas definiciones previas relacionadas con los conjuntos de dependencias funcionales.

Inferencia de dependencias funcionales

Axiomas de Armstrong

[ELM16 15.1.1]

- W.W. Armstrong propuso en 1974 tres reglas para inferir dependencias funcionales a partir de otras:
  - Axioma de reflexividad: $Y \subset X \Rightarrow X \to Y$
  - Axioma de aumento: $\forall W : X \to Y \Rightarrow XW \to YW$
  - Axioma de transitividad: $X \to Y \wedge Y \to Z \Rightarrow X \to Z$
- Estos axiomas pueden ser probados a partir de la definición de dependencia funcional (i.e., no son axiomas en *stricto sensu*).
- Los tres axiomas en conjunto son completos: Toda dependencia funcional que se puede inferir de F se puede inferir a través de los axiomas de Armstrong.
- La notación $F \models X \to Y$ indica que la dependencia funcional $X \to Y$ puede inferirse a partir del conjunto de dependencias funcionales F.

Inferencia de dependencias funcionales

Axiomas de Armstrong

Exceso de notación

Cuando trabajemos con nombres de atributos abstractos como A, B, C, D o $A_1, A_2, \ldots$ y escribamos un conjunto de atributos (por ejemplo, $\{B, C, D\}$), dentro de una dependencia funcional, omitiremos las llaves y las comas. En dicho ejemplo, lo denotaremos directamente BCD.

Ejercicio

Muestre que dado el conjunto de dependencias funcionales $F = \{A \rightarrow C, BC \rightarrow E, D \rightarrow B\}$ es posible inferir que $AD \rightarrow E$.

Inferencia de dependencias funcionales

Reglas de inferencia adicionales

- Las siguientes tres reglas se deducen de los axiomas de Armstrong:
  - Regla de unión: $X \to Y \land X \to Z \Rightarrow X \to YZ$
  - Regla de pseudotransitividad: $\forall W : X \to Y \land YW \to Z \Rightarrow XW \to Z$
  - Regla de descomposición: $X \to YZ \Rightarrow X \to Y \land X \to Z$

Clausura de conjuntos de dependencias funcionales y de atributos

- Partimos de una relación $R(A_1, A_2, ..., A_n)$.
- Dado un conjunto de dependencias funcionales F, la clausura de F (F^+) es el conjunto de todas las dependencias funcionales que pueden inferirse de F. Esto es:

$$F^+ = \{(X \rightarrow Y) | F \models (X \rightarrow Y)\}.$$

- Dado un conjunto de atributos X y un conjunto de dependencias F, la clausura de X con respecto a F (X_F^+) es el conjunto de todos los atributos A_i tales que la dependencia funcional $X \rightarrow A_i$ se infiere del conjunto de dependencias F. Esto es:

$$X_F^+ = \{A_i | F \models (X \rightarrow A_i)\}$$

Clausuras de conjuntos de dependencias funcionales y de atributos

Definiciones

- Las clausuras de conjuntos de atributos, X_F^+, son una forma ordenada de construir F^+.
- Esta clausura nos permite dar otra definición de clave candidata: Dado un esquema de relación $R(A_1, A_2, ..., A_n)$ y un conjunto de dependencias funcionales F, CK es clave candidata de R si y sólo si $CK_F^+ = A_1A_2...A_n$ y ningún subconjunto propio cumple con esa propiedad.

Clausuras de conjuntos de df's y de atributos

Algoritmo 1: Clausura de un conjunto de atributos X con respecto a F [ELM16 16.1.1]

Algoritmo 1: Clausura de un conjunto de atributos X con respecto a F

Input : Un conjunto de dependencias funcionales F de una relación universal R, un conjunto de atributos X.

Output: La clausura de X con respecto a F, X_F^+.

begin
    $X_F^+ \leftarrow X$;
    repeat
        $oldX^+ \leftarrow X_F^+$;
        foreach $(Y \rightarrow Z) \in F$ **do**
            if $Y \subset X_F^+$ **then**
                $X_F^+ \leftarrow X_F^+ \cup Z$;
            end
        end
    until $oldX^+ = X_F^+$;
end

Cobertura y equivalencia

Definiciones

- Dados dos conjuntos de dependencias funcionales F y G, decimos que el conjunto F cubre a G cuando toda dependencia funcional $X \to Y \in G$ puede ser inferida a partir de F. Es decir:

$$\forall (X \to Y) \in G : F \models X \to Y.$$

- Dos conjuntos de dependencias funcionales F y G son equivalentes cuando cada uno de ellos es cubierto por el otro. En otras palabras, F y G son equivalentes cuando sus clausuras coinciden: $F^+ = G^+$. Lo simbolizaremos $F \equiv G$.

Ejercicio

Muestre que los conjuntos de dependencias funcionales $F_1 = \{A \to B, B \to C, C \to A\}$ y $F_2 = \{A \to C, C \to B, B \to A\}$ son equivalentes.

Cubrimiento minimal de un conjunto de dependencias

Definición

- Dado un conjunto de dependencias F, nos interesará encontrar un conjunto equivalente G que cumpla ciertas propiedades de minimalidad. En particular, nos interesa que:
  - No haya atributos innecesarios del lado izquierdo:

$$\forall (X \to Y) \in G : \nexists (Z \to Y) \in G, Z \subset X, Z \neq X.$$

  - No haya dependencias redundantes:

$$\nexists (X \to Y) \in G : G - \{X \to Y\} \equiv G.$$

- A todo conjunto de dependencias funcionales G que es equivalente a F y cumple estas dos propiedades lo denominamos cubrimiento minimal de F.

Cubrimiento minimal de un conjunto de dependencias

Algoritmo

[ELM16 15.1.3]

- El algoritmo de cubrimiento minimal tiene 3 grandes pasos:
  1. Pasar las dependencias funcionales a forma canónica (descomponer cada dependencia funcional $X \rightarrow Y$ en df's $X \rightarrow A_i$ con $A_i \in Y$.
  2. Eliminar los atributos innecesarios del lado izquierdo de cada dependencia funcional $X \rightarrow A_i$.
  3. Eliminar las dependencias funcionales redundantes.

Algoritmo 2: Cubrimiento minimal de un conjunto de df's F

Input : Un conjunto de dependencias funcionales F.
Output : Un cubrimiento minimal de F, F_{min}.

```
begin
    F_{min} \leftarrow F;
    foreach (X \rightarrow Y) \in F_{min} do
        F_{min} = F_{min} - \{X \rightarrow Y\};
        foreach A_i \in Y do
            F_{min} = F_{min} \cup \{X \rightarrow A_i\}; #Pasamos a forma canónica
        end
    end
    foreach (X \rightarrow A) \in F_{min} do
        foreach B \in X do
            if \{F_{min} - \{X \rightarrow A\}\} \cup \{(X - \{B\}) \rightarrow A\} \equiv F_{min} then
                F_{min} = \{F_{min} - \{X \rightarrow A\}\} \cup \{(X - \{B\}) \rightarrow A\}; #Eliminamos atributos innecesarios del lado izquierdo
            end
        end
    end
    foreach (X \rightarrow A) \in F_{min} do
        if F_{min} - \{X \rightarrow A\} \equiv F_{min} then
            F_{min} = F_{min} - \{X \rightarrow A\}; #Eliminamos las df's redundantes
        end
    end
end
```

Cubrimiento minimal de un conjunto de dependencias

Ejemplo

Ejemplo

Para la relación universal $R(A, B, C, D, E, F, G)$ y el conjunto de dependencias funcionales $F = \{AB \to C, C \to A, BC \to D, ACD \to B, D \to E, D \to G, BE \to C, CG \to B, CG \to D, CE \to A, CE \to G\}$, encuentre un cubrimiento minimal.

Soluciones posibles

1- $F^1_{min} = \{AB \to C, C \to A, BC \to D, CD \to B, D \to E, D \to G, BE \to C, CG \to D, CE \to G\}$
2- $F^2_{min} = \{AB \to C, C \to A, BC \to D, D \to E, D \to G, BE \to C, CG \to B, CE \to G\}$

Descomposición a 3FN

[ELM16 15.3.1]

Algoritmo 3: Descomposición a 3FN (Elmasri, Navathe, 2016)

Input : Una relación universal R y un conjunto de dependencias funcionales F.
Output : Una descomposición de R, $D = (R_1, R_2, ..., R_n)$ que preserva la información y las dependencias funcionales, y está en 3FN.

begin
  $D = \emptyset$;
  Encontrar un cubrimiento minimal F_{min} para F;
  foreach $(X|(\exists A)((X \to A) \in F_{min}))$ **do**
    `#(Para cada conjunto X del lado izquierdo)`
    Crear un esquema de relación $R_X(X, A_1, A_2, ..., A_k)$ en donde $X \to A_i$ son las *únicas* df's en F_{min} con el conjunto X en el lado izquierdo;
    $D = D \cup R_X$;
  end
  Hallar todas las claves candidatas de R;
  if *ningún esquema contiene una clave candidata de R* **then**
    Tomar una de las claves candidatas CK de R;
    $D = D \cup R_{CK}(CK)$;
  end
  do
    if *los atributos de una relación $R_i \in D$ están incluidos en los de otra relación R_j (R_i es redundante)* **then**
      $D = D - R_i$;
    end
  while *haya relaciones redundantes en D* ;
end

Proyección de dependencias funcionales

- Al descomponer una relación R con un conjunto de dependencias funcionales F en $D = (R_1(Z_1), R_2(Z_2), ..., R_n(Z_n))$, es necesario saber qué dependencias funcionales se preservan.
- En la descomposición a 3FN que presentamos está garantizada la preservación de todas las dependencias funcionales. Pero en la descomposición a FNBC ésto no está garantizado.
- Las dependencias que se preservan son las que surgen de la proyección de F sobre los atributos Z_i de cada una de las $R_i(Z_i)$.
- La proyección de un conjunto de dependencias funcionales F sobre un conjunto de atributos Z, F_Z, se define como:

$$F_Z^+ = \{X \to Y \in F^+ | X \cup Y \subset Z\}$$

- Las dependencias funcionales preservadas en la descomposición son entonces:

$$F_D^+ = (F_{Z_1} \cup F_{Z_2} \cup ... \cup F_{Z_n})^+$$

Algoritmo de búsqueda de claves candidatas [PIAT06 12.2.4]

- Daremos una descripción de muy alto nivel de un algoritmo[1] para encontrar todas las claves candidatas en una relación $R(A_1, A_2, ..., A_n)$, a partir de un conjunto de dependencias funcionales F.
  1. Calcular un cubrimiento minimal del conjunto de dependencias funcionales F. Inicializar el conjunto de atributos de cálculo $C_a = \{A_1, A_2, ..., A_n\}$.
  2. Hallar los atributos independientes del cálculo, A_{indep}, que son aquellos que no están presentes en ninguna dependencia funcional. Eliminarlos del conjunto de atributos de cálculo: $C_a = C_a - A_{indep}$.
  3. Hallar los conjuntos de términos equivalentes, A_{equiv}, que son aquellos pares (X, Y) de términos que cumplen que $X \rightarrow Y$ y $Y \rightarrow X$ (con $X \cap Y = \emptyset$). De cada conjunto de términos equivalentes dejar sólo uno, y eliminar los restantes de C_a. Calcular la proyección de F_{min} en C_a, F_C.

[1] Extraído del libro “Tecnología y diseño de bases de datos” de Piattini *et al.*

Algoritmo de búsqueda de claves candidatas

4. Construir una clave tentativa K_{tent} con todos los elementos que sean sólo implicantes en F_C (es decir, estén sólo en la parte izquierda). Si $K_{tent}^{+} = C_a$ entonces K_{tent} es clave.
5. Si K_{tent} no resultó clave, entonces se comienzan a agregar otros atributos que sean implicantes pero que puedan ser implicados también. Se agregan alternativamente a K_{tent} todos los posibles subconjuntos de 1 atributo, luego aquellos de 2 atributos, etc, del conjunto $C_a - K_{tent}^{+}$. Con cada uno de ellos se verifica si K_{tent} es clave de C_a calculando la clausura. Al hacer crecer los subconjuntos se deben obviar aquellos que resultaron ser clave de C_a, ya que no van a ser minimales.
6. Por cada K_{tent} encontrado como clave de C_a se unen los atributos independientes, A_{indep} para obtener K, y se agrega K al resultado, CKs.
7. Se calculan otras claves K con otros de los términos equivalentes encontrados en el paso 3, y se agregan todas al resultado, CKs.

Algoritmo de búsqueda de claves candidatas

Ejemplo I

Ejemplo I

Para la relación universal $R(A, B, C, D, E, F, G, H, I, J)$ y el conjunto de dependencias funcionales
$F = \{AB \to C, A \to DE, B \to F, F \to GH, D \to IJ, B \to A, H \to G\}$,
calcule todas las claves candidatas existentes.

Solución

1- $F_{min} = \{AB \to C, A \to D, A \to E, B \to F, F \to H, D \to I, D \to J, B \to A, H \to G\}$
2- $A_{indep} = \emptyset$
3- $A_{equiv} = \emptyset$
4- **CK = {B}**
Observamos que $B^+ = \{A, B, C, D, E, F, G, H, I, J\}$
No hay otras claves candidatas.

Algoritmo de búsqueda de claves candidatas

Ejemplo II

Ejemplo II

Para la relación universal $R(A, B, C, D, E, F, G)$ y el conjunto de dependencias funcionales
$F = \{AB \to F, D \to A, E \to D, D \to E, CF \to B, B \to C\}$, calcule todas las claves candidatas existentes.

Solución

1- $F_{min} = \{AB \to F, D \to A, E \to D, D \to E, CF \to B, B \to C\}$
2- $A_{indep} = \{G\}$; $C_a = \{A, B, C, D, E, F\}$
3- $A_{equiv} = \{E, D\}$
eliminamos $D \to C_a = \{A, B, C, E, F\}$
$\to F_C = \{AB \to F, E \to A, CF \to B, B \to C\}$
4- $K = \{E\}$, pero $E^+ = \{A, E\} \to E$ no es clave.

Algoritmo de búsqueda de claves candidatas

Ejemplo II

Ejemplo II

Para la relación universal $R(A, B, C, D, E, F, G)$ y el conjunto de dependencias funcionales
$F = \{AB \to F, D \to A, E \to D, D \to E, CF \to B, B \to C\}$, calcule todas las claves candidatas existentes.

Solución

5- Agregamos a K otros atributos implicantes, primero de a uno:

- $(EA)^+ = \{A, E\} \neq C_a \to$ no es clave
- $(EB)^+ = \{A, B, C, E, F\} = C_a \to$ es clave
- $(EC)^+ = \{A, C, E\} \neq C_a \to$ no es clave
- $(EF)^+ = \{A, E, F\} \neq C_a \to$ no es clave

Algoritmo de búsqueda de claves candidatas

Ejemplo II

Ejemplo II

Para la relación universal $R(A, B, C, D, E, F, G)$ y el conjunto de dependencias funcionales
$F = \{AB \rightarrow F, D \rightarrow A, E \rightarrow D, D \rightarrow E, CF \rightarrow B, B \rightarrow C\}$, calcule todas las claves candidatas existentes.

Solución

5- Agregamos de a dos, evitando partir de EB:

- $(EAC)^+ = \{A, C, E\} \neq C_a \rightarrow$ no es clave
- $(EAF)^+ = \{A, E, F\} \neq C_a \rightarrow$ no es clave
- $(ECF)^+ = \{A, B, C, E, F\} = C_a \rightarrow$ es clave

No hay grupos de tres a agregar, que no incluyan claves ya encontradas.
$\rightarrow$ Hemos encontrado $\{E, B\}$ y $\{E, C, F\}$

Algoritmo de búsqueda de claves candidatas

Ejemplo II

Ejemplo II

Para la relación universal $R(A, B, C, D, E, F, G)$ y el conjunto de dependencias funcionales
$F = \{AB \rightarrow F, D \rightarrow A, E \rightarrow D, D \rightarrow E, CF \rightarrow B, B \rightarrow C\}$, calcule todas las claves candidatas existentes.

Solución

6- Agregamos los atributos independientes, A_{indep}:
$\rightarrow \{E, B, G\}$ y $\{E, C, F, G\}$
7- Agregamos las del término equivalente D:
$\rightarrow$ **CKs = {{E,B,G}, {E,C,F,G}, {D,B,G}, {D,C,F,G}}**

Algoritmo de búsqueda de claves candidatas

Ejercicio

Ejercicio

Para la relación universal $R(A, B, C, D, E, F, G, H, I, J)$ y el conjunto de dependencias funcionales
$F = \{AB \to C, BD \to EF, AD \to GH, A \to I, H \to J\}$, calcule todas las claves candidatas existentes.

Solución

Hay una única clave candidata $CK = \{ABD\}$.

Descomposición a FNBC

[ELM16 15.3.2; GM09 3.3.4]

Algoritmo 4: Descomposición a FNBC (García-Molina, 2009)

Input : Una relación universal R y un conjunto de dependencias funcionales F.
Output : Una descomposición de R, $D = (R_1, R_2, ..., R_n)$ que preserva la información y está en FNBC.

begin
    $D = \{R\}$;
    while ($\exists R_i(Z) \in D$ *tal que* $R_i(Z)$ *no está en FNBC)* **do**
        Encontrar una dependencia funcional $X \rightarrow Y$ contenida en R_i que viole la FNBC;
        Calcular X^+;
        $D = D - \{R_i(Z)\}$; `#Eliminamos la relación que viola la FNBC`
        $D = D \cup \{R_{i1}(X^+)\}$; `#Agregamos una relación para representar la dependencia funcional y otros atributos implicados por X`
        $D = D \cup \{R_{i2}(Z - (X^+ - X))\}$; `#Agregamos una relación sin los atributos implicados por X`
    end
end

Algoritmo Chase

[ELM16 15.2.3; GM09 3.4]

- Hemos presentado algoritmos de descomposición a 3FN y FNBC que preservan la información.
- La preservación de información implica que dada una relación R y una descomposición de R en $D = (R_1(Z_1), R_2(Z_2), ..., R_n(Z_n))$, toda instancia r de R puede recuperarse como:

$$r = \pi_{Z_1}(r) * \pi_{Z_2}(r) * ... * \pi_{Z_n}(r)$$

- El algoritmo de Chase nos permite verificar la preservación de información de una descomposición aún sin saber cómo la misma se obtuvo.
- Fue propuesto en 1979 por D. Maier, A. Mendelzon y Y. Sagiv, en simultáneo con A. Aho, C. Beeri y J. Ullman.

Algoritmo Chase

Principio de funcionamiento

- Analicemos la siguiente descomposición de la relación $R(ABCD)$ con el conjunto de df's $F = \{A \rightarrow C, B \rightarrow D, AB \rightarrow CD\}$:
  - $R_1(AC)$
  - $R_2(BD)$
  - $R_3(AB)$
- ¿Será que estamos preservando toda la información?
- El algoritmo chase utiliza una tabla denominada *tableau*, con tantas filas como relaciones y tantas columnas como atributos:

	A	**B**	**C**	**D**
$\mathbf{R_1}$				
$\mathbf{R_2}$				
$\mathbf{R_3}$				

Algoritmo Chase

Principio de funcionamiento

- $R(ABCD)$, $F = \{A \to C, B \to D, AB \to CD\}$:
  - $R_1(AC)$
  - $R_2(BD)$
  - $R_3(AB)$
- El algoritmo parte de una hipotética tupla (a_1, a_2, a_3, a_4) de la junta $r_1 \bowtie r_2 \bowtie r_3$ que se proyecta a cada una de las r_i: si una relación r_i contiene un atributo A_j, entonces en la posición (i, j) de la tabla escribimos el valor abstracto a_j.

	A	**B**	**C**	**D**
R$_1$	a_1		a_3	
R$_2$		a_2		a_4
R$_3$	a_1	a_2		

Algoritmo Chase

Principio de funcionamiento

- $R(ABCD)$, $F = \{A \to C, B \to D, AB \to CD\}$:
  - $R_1(AC)$
  - $R_2(BD)$
  - $R_3(AB)$
- Ahora rellenamos las demás posiciones con valores b_{ij}.

	A	**B**	**C**	**D**
R₁	a_1	b_{12}	a_3	b_{14}
R₂	b_{21}	a_2	b_{23}	a_4
R₃	a_1	a_2	b_{33}	b_{34}

son iguales

Algoritmo Chase

Principio de funcionamiento

- $R(ABCD)$, $F = \{A \to C, B \to D, AB \to CD\}$:
  - $R_1(AC)$
  - $R_2(BD)$
  - $R_3(AB)$
- La dependencia funcional $A \to C$ está reflejada en la primera fila. Como en la tercera fila también tenemos A, reemplazamos b_{33} por a_3 para no violar la dependencia funcional.

	A	**B**	**C**	**D**
$\mathbf{R_1}$	a_1	b_{12}	a_3	b_{14}
$\mathbf{R_2}$	b_{21}	a_2	b_{23}	a_4
$\mathbf{R_3}$	a_1	a_2	a_3	b_{34}

Algoritmo Chase

Principio de funcionamiento

- $R(ABCD)$, $F = \{A \rightarrow C, B \rightarrow D, AB \rightarrow CD\}$:
  - $R_1(AC)$
  - $R_2(BD)$
  - $R_3(AB)$
- La dependencia funcional $B \rightarrow D$ está reflejada en la segunda fila. En la tercera, reemplazamos b_{34} por a_4 para no violar la dependencia funcional.

	A	**B**	**C**	**D**
$\mathbf{R_1}$	a_1	b_{12}	a_3	b_{14}
$\mathbf{R_2}$	b_{21}	a_2	b_{23}	a_4
$\mathbf{R_3}$	a_1	a_2	a_3	a_4

Algoritmo Chase

Principio de funcionamiento

- $R(ABCD)$, $F = \{A \to C, B \to D, AB \to CD\}$:
  - $R_1(AC)$
  - $R_2(BD)$
  - $R_3(AB)$
- Observamos que $AB \to CD$ sólo se encuentra en la tercera fila.

	A	**B**	**C**	**D**
$\mathbf{R_1}$	a_1	b_{12}	a_3	b_{14}
$\mathbf{R_2}$	b_{21}	a_2	b_{23}	a_4
$\mathbf{R_3}$	a_1	a_2	a_3	a_4

Descomposición sin pérdidas

Volví a la tupla original

- La tercera línea del *tableau* indica que $(a_1, a_2, a_3, a_4) \in r$, y que por lo tanto podemos reconstruir r sin pérdida de información.

Algoritmo Chase

Principio de funcionamiento

	A	B	C	D
R_1	a_1	b_{12}	a_3	b_{14}
R_2	b_{21}	a_2	b_{23}	a_4
R_3	a_1	a_2	a_3	a_4

- Observaciones:
  - El *tableau* representa una instancia r de la relación universal R que se construye a partir de instancias de las relaciones R_i.
  - Si logramos construir una tupla completa de r (la fila verde), entonces no hemos perdido información en la descomposición.
  - Si no hubieramos incluído R_3 en la descomposición, aún preservándose todas las dependencias funcionales no se hubiera preservado la información.

Algoritmo Chase

Principio de funcionamiento

- Veamos otro ejemplo: $R(ABCDE)$ con un conjunto de dependencias funcionales $F = \{B \rightarrow C, C \rightarrow D, D \rightarrow A, B \rightarrow E\}$ es descompuesta en:
  - $R_1(AB)$
  - $R_2(BCD)$
  - $R_3(DE)$
- El *tableau* inicial tendrá el siguiente aspecto:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	b_{13}	b_{14}	b_{15}
$\mathbf{R_2}$	b_{21}	a_2	a_3	a_4	b_{25}
$\mathbf{R_3}$	b_{31}	b_{32}	b_{33}	a_4	a_5

Algoritmo Chase

Principio de funcionamiento

- $R(ABCDE)$, $F = \{B \to C, C \to D, D \to A, B \to E\}$:
  - $R_1(AB)$
  - $R_2(BCD)$
  - $R_3(DE)$
- Procesando la dependencia $B \to C$ observamos que:
  - B aparece en dos relaciones.
  - En una de ellas C no aparece. Remplazamos allí el b_{13} por a_3.

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	a_3	b_{14}	b_{15}
$\mathbf{R_2}$	b_{21}	a_2	a_3	a_4	b_{25}
$\mathbf{R_3}$	b_{31}	b_{32}	b_{33}	a_4	a_5

Algoritmo Chase

Principio de funcionamiento

- $R(ABCDE)$, $F = \{B \to C, C \to D, D \to A, B \to E\}$:
  - $R_1(AB)$
  - $R_2(BCD)$
  - $R_3(DE)$
- Luego, al procesar $C \to D$ tenemos que:
  - C aparece en dos relaciones.
  - En una de ellas D no aparece. Remplazamos allí el b_{14} por a_4.

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_2}$	b_{21}	a_2	a_3	a_4	b_{25}
$\mathbf{R_3}$	b_{31}	b_{32}	b_{33}	a_4	a_5

Algoritmo Chase

Principio de funcionamiento

- $R(ABCDE)$, $F = \{B \rightarrow C, C \rightarrow D, D \rightarrow A, B \rightarrow E\}$:
  - $R_1(AB)$
  - $R_2(BCD)$
  - $R_3(DE)$
- Ahora procesamos $D \rightarrow A$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	a_3	a_4	b_{25}
$\mathbf{R_3}$	a_1	b_{32}	b_{33}	a_4	a_5

Algoritmo Chase

Principio de funcionamiento

- $R(ABCDE)$, $F = \{B \rightarrow C, C \rightarrow D, D \rightarrow A, B \rightarrow E\}$:
  - $R_1(AB)$
  - $R_2(BCD)$
  - $R_3(DE)$
- Y por último, $B \rightarrow E$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_3}$	a_1	b_{32}	b_{33}	a_4	a_5

Algoritmo Chase

Principio de funcionamiento

- $R(ABCDE)$, $F = \{B \rightarrow C, C \rightarrow D, D \rightarrow A, B \rightarrow E\}$:
  - $R_1(AB)$
  - $R_2(BCD)$
  - $R_3(DE)$

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_3}$	a_1	b_{32}	b_{33}	a_4	a_5

- Observamos que ninguna fila nos quedó llena de elementos a_i.
- La descomposición por lo tanto no preserva la información.

Algoritmo Chase

Principio de funcionamiento

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	a_3	a_4	b_{15}
$\mathbf{R_3}$	a_1	b_{32}	b_{33}	a_4	a_5

- El mismo *tableau* sirve como contraejemplo de una instancia de R que al ser descompuesta en instancias de $R_1(Z_1), R_2(Z_2), ..., R_n(Z_n)$ pierde información.
- Es decir que, llamando r a la instancia de R determinada por el *tableau*:

$$\pi_{Z_1}(r) * \pi_{Z_2}(r) * ... * \pi_{Z_n}(r) \neq r$$

Algoritmo Chase

Ejercicio

Ejercicio

Dada la relación universal $R(ABCDE)$ con el siguiente conjunto de dependencias funcionales $F = \{A \to C, B \to C, C \to D, DE \to C, CE \to A\}$, descompuesta en:

- $R_1(AD)$
- $R_2(AB)$
- $R_3(BE)$
- $R_4(CDE)$
- $R_5(AE)$

Determine si la misma es con o sin pérdidas.

Algoritmo Chase

Ejercicio

- El *tableau* inicial tendrá el siguiente aspecto:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	b_{12}	b_{13}	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	b_{23}	b_{24}	b_{25}
$\mathbf{R_3}$	b_{31}	a_2	b_{33}	b_{34}	a_5
$\mathbf{R_4}$	b_{41}	b_{42}	a_3	a_4	a_5
$\mathbf{R_5}$	a_1	b_{52}	b_{53}	b_{54}	a_5

Algoritmo Chase

Ejercicio

- Después de aplicar $A \rightarrow C$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	b_{12}	b_{13}	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	b_{13}	b_{24}	b_{25}
$\mathbf{R_3}$	b_{31}	a_2	b_{33}	b_{34}	a_5
$\mathbf{R_4}$	b_{41}	b_{42}	a_3	a_4	a_5
$\mathbf{R_5}$	a_1	b_{52}	b_{13}	b_{54}	a_5

Algoritmo Chase

Ejercicio

- Después de aplicar $B \to C$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	b_{12}	b_{13}	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	b_{13}	b_{24}	b_{25}
$\mathbf{R_3}$	b_{31}	a_2	b_{13}	b_{34}	a_5
$\mathbf{R_4}$	b_{41}	b_{42}	a_3	a_4	a_5
$\mathbf{R_5}$	a_1	b_{52}	b_{13}	b_{54}	a_5

Algoritmo Chase

Ejercicio

- Después de aplicar $C \rightarrow D$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	b_{12}	b_{13}	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	b_{13}	a_4	b_{25}
$\mathbf{R_3}$	b_{31}	a_2	b_{13}	a_4	a_5
$\mathbf{R_4}$	b_{41}	b_{42}	a_3	a_4	a_5
$\mathbf{R_5}$	a_1	b_{52}	b_{13}	a_4	a_5

Algoritmo Chase

Ejercicio

- Después de aplicar $DE \rightarrow C$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	b_{12}	b_{13}	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	b_{13}	a_4	b_{25}
$\mathbf{R_3}$	b_{31}	a_2	a_3	a_4	a_5
$\mathbf{R_4}$	b_{41}	b_{42}	a_3	a_4	a_5
$\mathbf{R_5}$	a_1	b_{52}	a_3	a_4	a_5

Algoritmo Chase

Ejercicio

- Por último, aplicando $CE \rightarrow A$:

	A	**B**	**C**	**D**	**E**
$\mathbf{R_1}$	a_1	b_{12}	b_{13}	a_4	b_{15}
$\mathbf{R_2}$	a_1	a_2	b_{13}	a_4	b_{25}
$\mathbf{R_3}$	a_1	a_2	a_3	a_4	a_5
$\mathbf{R_4}$	a_1	b_{42}	a_3	a_4	a_5
$\mathbf{R_5}$	a_1	b_{52}	a_3	a_4	a_5

- $\Rightarrow$ **La descomposición es sin pérdidas.**

Bibliografía

[ELM16] Fundamentals of Database Systems, 7th Edition.
R. Elmasri, S. Navathe, 2016.
Capítulo 15 (Algoritmo de descomposición a 3FN)

[PIAT06] Tecnología y diseño de bases de datos.
M. Piattini, E. Marcos, C. Calero, B. Vela, 2006.
Capítulo 12 (Algoritmo de extracción de claves candidatas)

[GM09] Database Systems, The Complete Book, 2nd Edition.
H. García-Molina, J. Ullman, J. Widom, 2009.
Capítulo 3 (Algoritmo de descomposición a FNBC)