**Practico numero 4**

4) Dado el siguiente conjunto de dependencias funcionales:

G = {A → CD, B → AF, F → CH, AF → B}

definido sobre R(ABCDF H), decir si las siguientes son superllaves para G y encontrar todas las llaves candidatas.

a) ABD

|  |  |  |
| --- | --- | --- |
| Siempre | Talvez | Nunca |
|  | A B F | C D H |

Combinamos los atributos de Siempre con los de Talvez.

1 atributo:

A = ACD ⊄ R, no es llave

A partir de “A” puedo generar “A” por eso primero escribo A = A, luego vemos si podemos aplicar alguna dependencia funcional para ver si podemos determinar algún otro atributo, como no podemos generar todo R entonces no es una llave.

B = BAFCDH ⊆ R es llave

F = FCH ⊄ R, no es llave

2 atributos:

AF = AFCDHB es llave.

Luego las llaves candidatas son: B, AF.

a) ABD no es llave

b) ABF es super llave porque ⊆ AF ⊆ R

c) ACDH no es llave

d ) AF es llave

7)Para los siguientes conjuntos de dependencias funcionales encontrar un cubrimiento mínimo

reducido (CMR). Una vez obtenido el CMR, calcule a partir de este todas las llaves candidatas.

a) G1 = {A → C, AB → DE, AB → CDI, AC → I} definido en R1(ABCDEI).

**Cubrimiento mínimo:**

**1.1 Cubrimiento cerrado en atributos**: (A)G1+ =ACI, (AB)G1+=ABCDEI, (AC)G1+= ACI

Luego el cubrimiento cerrado de atributos es M1={A->ACI, AB -> ABCDEI, AC->ACI}

**1.2 Eliminar dependencias funcionales redundantes**. Tomo una dependencia y la tacho del conjunto Mi. Luego, calculo la clausura del atributo a izquierda de esa dependencia con respecto al conjunto Mi restante:

¿A->AC es redundante? Para ello calculamos la clausura de A tachando A->AC de la dependencia de m1.

(A)m1\(A->AC)+= A ⊉ AC, no es redundante

A partir de A solo podemos generar A, A ⊉ AC significa que A no incluye AC.

AB -> ABCDEI es redundante?

(AB)m1\(AB -> ABCDEI)+= ABCI ⊉ ABCDEI no es redundante

AB solo puede generar ABCI pero no ABCDEI

¿AC->ACI es redundante?

(AC)m1\(AC->ACI)+ = ACI ⊆ ACI es redundante

Luego el cubrimiento mínimo es M2={ A->ACI, AB -> ABCDEI}

**Cubrimiento Reducido**

2.1 **Reducir a izquierda a** M2={ A->ACI, AB -> ABCDEI}

(A)M2+= ACI ⊉ ABCDEI luego B no es extraño a izquierda

(B) M2+= B ⊉ ABCDEI luego A no es extraño a izquierda

Luego, el cubrimiento mínimo reducido a izquierda es M3 ={ A->ACI, AB -> ABCDEI}

2.2 **Reducir a derecha ídem a reducir a izquierda, pero con el lado derecho:**

¿A->ACI A es extraño a derecha?

**b)** G2 = {A → DF, D → AC, BF → C, AF → CE, D → F C} definido en R2(ABCDEF).

**Cubrimiento mínimo**

1. Del lado derecho agregamos todo lo que podemos determinar con el lado izquierdo:

A+ = ADFCE D+=DACFE BF+=BFC AF+=AFCED D+=DFCAE

**M1 = {A->ADFCE, D->DACFE, BF->BFC, AF->AFCED}**

2. Eliminamos dependencias redundantes o sea calculamos la clausura del lado izquierdo, pero sin su dependencia:

¿A->ADFCE es redundante?

A+M1\{A->ADFCE} = A ⊉ ADFCE por lo tanto no es redundante

¿D->DACFE es redundante?

D+M1\{D->DACFE} =D ⊉ DACFE por lo tanto no es redundante

¿BF->BFC es redundante?

BF+M1\{BF->BFC} = BF ⊉ BFC por lo tanto no es redundante

¿AF->AFCED es redundante?

AF+M1\{AF->AFCED} = AFDCE ⊇ AFCED por lo tanto es redundante

Luego el conjunto sin las dependencias redundantes quedaría así:

**M2 = {A->ADFCE, D->DACFE, BF->BFC}**

**Cubrimiento mínimo Reducido**

**3. Reducimos a izquierda**, sea una df XA->Y, A es extraño a izquierda si podemos generar X->Y, si lo es modificamos el conjunto y seguimos operando con el conjunto modificado.

En A->ADFCE, D->DACFE no puede haber extraños a izquierda.

Veamos BF->BFC

¿F es extraño a izquierda?

B+M2 = B ⊉ BFC, por lo tanto, B no es extraño a izquierda

¿B es extraño a izquierda?

F+M2 = F ⊉ BFC, por lo tanto, F no es extraño a izquierda

**M3 =** **M2 = {A->ADFCE, D->DACFE, BF->BFC}**

**4. Reducimos a derecha,** sea una df D->AC ¿A es extraño a derecha? “Tachamos” la A y vemos que generamos con D.

Eliminamos atributos trivialmente extraños a derecha:

**M4 = {A->DFCE, D->ACFE, BF->C}**

Si a la derecha hay un solo atributo no hay extraño a derecha.

En A->DFCE

¿D es atributo extraño a la derecha? “Tachamos” la D y vemos que generamos con A

A+M4\{ A->DFCE } u { A->FCE } = AFCE ⊉ DFCE, entonces D no es extraño a derecha

¿F es atributo extraño a la derecha?

A+M4\{ A->DFCE } u { A->DCE }= ADCEF ⊇ DFCE, entonces F es atributo extraño a derecha

**M5 = {A->DCE, D->ACFE, BF->C**}

¿C es atributo extraño a derecha?

A+M5\{A->DCE} u{A->DE} = ADECF ⊇ DCE, entonces C es atributo extraño a derecha

**M6 = {A->DE, D->ACFE, BF->C**}

¿E es atributo extraño a derecha?

A+M6\{A->DE}u{A->D} = ADCFE ⊇ DE, entonces D es atributo extraño a derecha

**M7 = {A->D, D->ACFE, BF->C}**

En D->ACFE:

¿A es atrib extraño a der?

D+M7\{D->ACFE}u{D->CFE} = DCFE ⊉ ACFE, entonces A no es atrib extr a der

¿C es atrib ext a der?

D+M7\{D->ACFE}u{D->AFE} = DAFE ⊉ ACFE, entonces C no es atrib ext a der

¿F es atrib extr a der?

D+M7\{D->ACFE}u{D->ACE} = DACE ⊉ ACFE, entonces F no es atrib ext a der

¿E es atrib ext a der?

D+M7\{D->ACFE}u{D->ACF} = DACF ⊉ ACFE, entonces E no es atrib ext a der

**M8 = M7 = {A->D, D->ACFE, BF->C}**

**¡Excelente!**

**Calculo de llaves:**

**M8 = {A->D, D->ACFE, BF->C} R(ABCDEF)**

|  |  |  |
| --- | --- | --- |
| Siempre | Talvez | Nunca |
| B | A D F | C E |

B+F = B ⊉ R por lo tanto no es llave

Combinaciones de 2 atributos:

BA+F= BADCFE ⊇ R por lo tanto es llave

BD+F= BDACFE ⊇ R por lo tanto es llave

BF+F= BFC ⊉ R por lo tanto no es llave

Combinaciones de 3 atributos de BF + atributos de talvez:

BFA+F es combinación de BA por lo tanto es super llave

BFD es combinación de BD por lo tanto es super llave

Luego las llaves candidatas son {ba,bd}

**¡Excelente!**