## 3.Dependencias funcionales.

**3.1 Definiciones básicas .** Una dependencia funcional es una restricción inherente a la semántica de los atributos que se expresa en la forma : **$X \rightarrow Y$ (** X e Y son descriptores , esto es, conjuntos de atributos ) y se lee " X implica Y" .

El significado , ya aclarado en la introducción , es que , a cada valor de X , le corresponde uno solo de Y .

Si $\neg \exists\ X' \subset X : X' \rightarrow Y$ , se dice que la dependencia es **total** . En caso contrario , se dice que es **parcial** .

Si $\exists\ Z : Z \cap X = \Phi$ , $Z \cap Y = \Phi$ , $X \rightarrow Z$ , $\neg ( Z \rightarrow X )$ , $Z \rightarrow Y$ ; de la dependencia se dice que es **transitiva .**

Un esquema de relación **R** viene definido por el par ( **T , L** ) , siendo **T** el conjunto de atributos y **L** el conjunto de dependencias .

Una **clave** del esquema es un descriptor **K** ( subconjunto de T ) , tal que :
**K**$\rightarrow$ **T** , siendo ésta una dependencia total .

**3.2 Axiomas de Armstrong .**

Los Axiomas de Armstrong son más bien reglas de inferencia .

Estas reglas permiten deducir todas las dependencias funcionales que tienen lugar entre un conjunto dado de atributos , como consecuencia de las dependencias " dato" , esto es , de las que se asumen como ciertas a partir del conocimiento del problema .

Las dependencias " dato " equivalen a los axiomas de una teoría ( en sentido lógico ) y los Axiomas de Armstrong son el conjunto **completo** de reglas deductivas que nos permite deducir cualquier otra dependencia **cierta** ( teorema de la teoría ) .

Los Axiomas de Armstrong son :

- **Reflexividad :** $\forall X , X \rightarrow X$
- **Proyectividad :** $\{ X \rightarrow Y , Z \subseteq Y \} \Rightarrow X \rightarrow Z$
- **Aumentatividad** : $\{ X \rightarrow Y , Z \supseteq X \} \Rightarrow Z \rightarrow Y$
- **Aditividad** :$\{ X \rightarrow Y , Z \rightarrow V \} \Rightarrow X \cup Z \rightarrow Y \cup V$
- **Transitividad** : $\{ X \rightarrow Z , Z \rightarrow Y \} \Rightarrow X \rightarrow Y$

El carácter correcto de estas reglas , es obvio y , en todo caso , fácilmente demostrable . Lo que ya es más complicado es demostrar el carácter completo del conjunto , demostración que pertenece al campo de la Lógica y no es en todo caso objeto de este curso básico de Bases de Datos .
Lo importante para nosotros es que este resultado ( la completitud del conjunto ) , nos permite abordar y resolver una serie de problemas fundamentales que luego nos conducirán al establecimiento de algoritmos de diseño sencillos y fiables .
Estos problemas fundamentales , son :

- Cierre de un conjunto de dependencias.
- Equivalencia lógica de esquemas .
- Deducción de dependencias .
- Cierre de un descriptor respecto de un conjunto de dependencias.
- .Cálculo de las claves de un esquema .

### 3.3 Cierre de un conjunto de dependencias . Equivalencia de esquemas .

Sea el esquema R ( T , L ) . El cierre del conjunto L de dependencias funcionales es el conjunto de todas las dependencias ciertas ( deducibles a partir de L mediante los Axiomas de Armstrong ) . El cierre se representa por $L^{+}$.

Dos conjuntos de dependencias , L y M , relativas al mismo universo T de atributos , son equivalentes si $L^{+} = M^{+}$ , en cuyo caso R ( T , L ) y S ( T , M ) son esquemas equivalentes ( representaciones alternativas y equivalentes del mismo problema ) .

**3.4 Deducción de dependencias . Cierre de un descriptor respecto de un conjunto de dependencias .**

El problema de deducción puede plantearse en estos términos : Dado un esquema R ( T , L ) , y dada la dependencia $X \rightarrow Y$ ( no perteneciente a L ) , establecer el carácter cierto o falso de la misma .

La solución ( teórica ) es , obviamente , calcular $L^+$ , y ver si $X \rightarrow Y$ pertenece o no al cierre calculado .

El problema radica en la extraordinaria complejidad computacional del cálculo de un cierre ( el libro de Ullmann citado en la bibliografía básica comenta casi siempre algo acerca de la complejidad computacional de los algoritmos que propone ) , así que no calcularemos $L^+$ en ningún caso .

El cierre de un descriptor X respecto de un conjunto de dependencias L , se representa por $X^+_L$ ( se omite el subíndice si no hay ambigüedad ) . El cierre se define así:

$X \rightarrow X^+ \;\wedge \neg\exists Z \supset X^+ : X \rightarrow Z$

Es decir : X implica a su cierre y no existe ningún superconjunto del mismo que sea implicado por X ( el cierre , en ese sentido , es un conjunto máximo ).

A diferencia del cálculo del cierre de un conjunto de dependencias , el cálculo del cierre de un descriptor es computacionalmente simple y permite resolver el problema antes planteado mediante una simple estrategia . Para saber si X $\rightarrow$Y es cierta en relación a R( T , L ) , calculamos $X^+$. Si Y $\subseteq$X la dependencia es cierta y no lo es en caso contrario.

Si se trata de analizar la equivalencia de dos representaciones del mismo problema , esto es , si R( T , L ) $\approx$ S( T , M ) , hay que establecer la equivalencia de los conjuntos M y L de dependencias funcionales .

M y L serán equivalentes si sus cierres coinciden . $\forall (X \rightarrow Y) \in \{M\text{-}L\}$ se calcula el cierre $X^+_L$ . Si Y es un subconjunto de dicho cierre en todos los casos , pasamos a calcular $V^+_M$ , $\forall (V \rightarrow Z) \in \{L - M\}$ . Si Z es un subconjunto del cierre en todos los casos , L y M son conjuntos equivalentes .

**3.5 Cálculo del cierre de un descriptor respecto de un conjunto de dependencias funcionales .**

Es un proceso iterativo . Sea X el descriptor y L = { $f_1$ , … ,$f_n$ } el conjunto de dependencias . Se calcula la secuencia $X^0$ , … , $X^i$ , … , $X^n=X^+$ en la forma siguiente :

$X^{0=}X$

$X^{i+1}=X^i \cup$ { atributos de la derecha de las dependencias en L implicadas por subconjuntos de $X^i$ } .

Si $X^i=X^{i+1}$ , $X^i$ es el cierre.

Si $X^i=T$ , $X^i$ es el cierre .

Ejemplo:

X=BD , $f_1$= AB$\rightarrow$ C , $f_2$=C $\rightarrow$A , $f_3$=BC$\rightarrow$ D , $f_4$= ACD$\rightarrow$ B , $f_5$=D$\rightarrow$ EG , $f_6$=CG $\rightarrow$BD , $f_7$=BE$\rightarrow$C , $f_8$= CE$\rightarrow$ AG

$X^0$=BD

$X^1$=BDEG (usando $f_5$ )

$X^2$=BDEGC ( usando $f_7$)

$X^3$=BDEGCA ( usando $f_2$ )

Como no hay más atributos , éste es el cierre .

### 3.6 Recubrimiento no redundante .

Si dos conjuntos de dependencias funcionales , L y M , son equivalentes , o sea , tienen sus cierres iguales ; se dice que L es un recubrimiento de M ( y recíprocamente ) .

Dado el conjunto L de dependencias funcionales , M es un recubrimiento no redundante si :

- L$\approx$ M
- Toda dependencia en M es de la forma X$\rightarrow$A (segundos miembros simples )
- Ningún implicante contiene atributos superfluos ( un atributo es superfluo si su supresión no altera el cierre del conjunto ) .

- No hay dependencias redundantes ( una dependencia es redundante si su supresión no altera el cierre del conjunto ) .

Ejemplo :

Calcular un recubrimiento no redundante de :

L={ AB$\rightarrow$ C , C$\rightarrow$ A , BC $\rightarrow$D , ACD$\rightarrow$ B , D$\rightarrow$ EG , BE$\rightarrow$ C , CG$\rightarrow$ BD , CE$\rightarrow$ AG }

Aplicando el axioma de proyectividad :

$L_1$={ AB$\rightarrow$ C , C$\rightarrow$ A , BC$\rightarrow$ D , ACD$\rightarrow$ B , D$\rightarrow$ E , D$\rightarrow$ G , BE$\rightarrow$ C , CG$\rightarrow$ B , CG$\rightarrow$ D , CE$\rightarrow$ A , CE$\rightarrow$ G }

Cuando hay redundancies obvias , pueden ( y deben ) ser eliminadas cuanto antes , pues esto contribuirá a simplificar el problema en su conjunto .

Un rápido examen, permite observar que , ya que C$\rightarrow$ A , CE $\rightarrow$A es redundante y puede eliminarse . Asimismo , ACD$\rightarrow$B puede sustituirse por CD$\rightarrow$B .

Nos queda el conjunto $L_2$ en el que , por comodidad de notación , vamos a identificar cada elemento mediante un número de orden :

$L_2$= { AB$\rightarrow$ C (1 ) , C$\rightarrow$ A ( 2 ) , BC$\rightarrow$ D ( 3 ) , CD$\rightarrow$ B ( 4 ) , D$\rightarrow$ E ( 5 ) , D$\rightarrow$ G ( 6 ) , BE$\rightarrow$ C ( 7 ) , CG$\rightarrow$ B ( 8 ) , CG$\rightarrow$ D ( 9 ) , CE$\rightarrow$ G ( 10 ) }

Eliminación de atributos superfluos:

Si una dependencia viene implicada por un atributo simple , es obvio que éste no es superfluo .

Analicemos los implicantes compuestos : AB $\rightarrow$ C . Si A fuese superfluo , B no podrá serlo , y recíprocamente .

Supongamos que se elimina A . Nos queda B$\rightarrow$C . Si es cierta , A es superfluo

Calculamos $B^+_{L2}$ y se obtiene B *A no es superfluo*.

Si eliminamos B ,nos queda A$\rightarrow$C . El cierre $A^+_{L2}$ es A . *B no es superfluo.*

Repitiendo este proceso para todos los implicantes compuestos , puede verse que no hay atributos superfluos *(y es un ejercicio muy recomendable para fijar los conceptos )*.

Eliminación de dependencias redundantes :

Podemos seguir el orden que queramos , llegando eventualmente a recubrimientos no redundantes distintos , pero equivalentes .

Para ver si (1) es redundante , la elimino y calculo $AB^{+}_{L2-\{1\}}$ :

$AB^{+}_{L2-\{1\}}=AB$ . Como C no pertenece al cierre , (1) no es redundante .

Repetimos el proceso:

$C^{+}_{L2-\{2\}}= C$ , así que (2) no es redundante .

$BC^{+}_{L2-\{3\}}=ABC$ , así que (3) no es redundante .

$CD^{+}_{L2-\{4\}}=CDAEGB$ , así que (4) es redundante y se suprime , siendo $L_3=L_2-\{4\}$

$D^{+}_{L3-\{5\}}=DG$ , así que (5) no es redundante .

$D^{+}_{L3-\{6\}}=DE$ , así que (6) no es redundante .

$BE^{+}_{L3-\{7\}}=BE$ , así que (7) no es redundante .

$CG^{+}_{L3-\{8\}}=CGAD$ , así que (8) no es redundante .

$CG^{+}_{L3-\{9\}}=CGABD$ , así que (9 ) es redundante y se suprime , siendo $L_4=L_3-\{9\}$

$CE^{+}_{L4-\{10\}}=CEA$ , así que (10) no es redundante .

Un recubrimiento no redundante de L es , por tanto :

$AB\rightarrow C$ ; $C \rightarrow A$ ; $BC\rightarrow D$ ; $D\rightarrow E$ ; $D\rightarrow G$ ; $BE\rightarrow C$ ; $CG\rightarrow B$ ; $CE\rightarrow G$

En problemas de diseño y cálculo de claves , trabajaremos siempre con recubrimientos no redundantes .

### 3.7 Determinación de las claves de un esquema .

Ahora ya podemos formular la base del procedimiento . Sea el esquema R ( T , L ) con T un conjunto de n atributos .

Para calcular las claves , vamos generando los posibles descriptores a partir de los elementos de T, y vamos calculando los cierres respecto de L . Si $X^{+}_{L}=T$, X es clave ( y ya ignoramos sus superconjuntos ) .

Lo malo es que el número de descriptores a explorar ( en el caso peor ) es $2^n$, y es que el problema de calcular todas las claves de un esquema es del tipo NP-completo , por lo que todos los algoritmos existentes para su computación requieren de mucho tiempo para su ejecución .
Afortunadamente , los problemas de diseño sólo requieren del conocimiento de al menos una clave , y , determinar una clave , no suele ser complicado .
En este curso , y por limitación del tiempo disponible para impartir este tema , nos limitaremos a calcular una clave de R ( T , L ) para luego “rediseñar” el esquema en forma de proyecciones normalizadas .

**3.8 Cálculo de una clave del esquema R ( T , L ) .**

Lo primero que debe hacerse , es calcular un recubrimiento no redundante de L .Esto “despeja” un poco los datos de partida y facilita el cálculo de la(s) clave(s) , sea cual sea el procedimiento empleado .
Ya hemos dicho que el cálculo de todas las claves no es una necesidad en la práctica del diseño . Sólo tendría un sentido si lo que se pretende es establecer el nivel de normalización de un esquema ( enfoque analítico ) , para validar un diseño.
Pero , ya que partimos de algoritmos que aseguran la normalización del resultado , no tiene más interés en este curso .
Ahora bien , para diseñar correctamente , necesitamos conocer al menos una clave . Afortunadamente , esto no es muy complicado .
Sea L un conjunto no redundante de dependencias funcionales . Distinguiremos cuatro conjuntos de atributos :
**I**={ atributos que aparecen sólo como implicantes }
**D**={ atributos que aparecen sólo como implicados }
**ID**={ atributos que aparecen como implicantes e implicados }
**N**={ atributos que no aparecen en ninguna dependencia }
Es fácil razonar lo siguiente :

- Los atributos de **I** forman parte de todas las claves .
- Los atributos de **N** forman parte de todas las claves .
- Los atributos de **D** no forman parte de ninguna clave .
- Los atributos de **ID** pueden formar parte o no de alguna clave .

Un procedimiento sencillo para determinar una clave , es por tanto el siguiente :

1. Calcular $Z=I \cup N$ . Z es el núcleo del proceso ( intersección de todas las claves , pudiendo ser en algún caso el conjunto vacío )
2. Calcular $Z^+$ . Si este cierre es T , Z es la clave (única) del esquema. Si no es así , sigue la búsqueda de una clave .
3. $Z^0=Z$ , $Z^i=Z^{i-1} \cup \{ A_j \}$ , $A_j \in ID$
4. Calcular $( Z^i )^+$ . Si este cierre es T , $Z^i$ es , en general una superclave del esquema ( es clave o contiene alguna clave )
5. La(s) clave(s) contenida(s) en $Z^i$ se extrae(n) por eliminación de posibles atributos superfluos (distintos de los del núcleo y del último añadido ) .

En el ejemplo del apartado anterior para cálculo de un recubrimiento no redundante , es fácil ver :

ID ={ A B C D E G }

$I=\phi$ , $N=\phi$ , $D= \phi$

$Z^0 =\phi$ .

$Z^1=A$ , $A^+=A$

$Z^2=AB$ , $AB^+=T$

AB es una clave .

(Nota:

Con relación al punto 3,en el que el núcleo se va incrementando con atributos del conjunto **ID**, si se diese el caso de que el cierre de $Z^i$ incluyese elementos de **ID**, es obvio que será innecesario añadirlos.

Esto es una heurística que puede agilizar el proceso).