Dpto. Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

ELEMENTOS DE BASES DE DATOS

Segundo Cuatrimestre 2014

Clase 9:
Teoría Relacional (Parte II)
Diseño – Descomposición – Cubrimientos

Mg. María Mercedes Vitturini
[mvitturi@uns.edu.ar]

Modelo Relacional - Framework

Sabemos que el modelo relacional define una *"colección de tablas"* para representar los datos de un problema y sus relaciones.

- Se espera diseñar *relaciones bien estructuradas*, esto es, con mínima redundancia y que permitan agregar, borrar y modificar tuplas sin causar errores o inconsistencias.
- Estamos analizando cómo respondernos:

¿éste es un buen diseño?
¿existe un diseño relacional de mayor calidad?

Teoría Relacional

- Una forma de encontrar un diseño relacional de calidad es partir de un buen diseño conceptual, i.e. Entidad-Relación.
- Por su lado, *la teoría relacional analiza la calidad de los esquemas de relación en función de los conceptos*:
  - **Dependencias Funcionales**
  - **Llave primaria**

Sabemos que:

- Dado un esquema de relación R, es posible definir un conjunto F de dependencias funcionales $X \rightarrow Y$, con X, $Y \subseteq R$. F es el conjuntos de restricciones esperadas para cualquier r(R) (F puede ser vacío, salvo triviales!).

Hasta ahora vimos que…

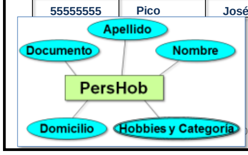
- Dado un esquema R, se define el conjunto F de dependencias funcionales o restricciones para las r(R).
- Dado F, existe F^+, el conjunto de todas las dependencias funcionales lógicamente implicadas de F.
  - Toda relación r(R) que satisface F, entonces satisface F^+.
- Dados R y F, sea $X \subseteq R$, X_F^+ es el conjunto de atributos lógicamente implicados por X bajo las restricciones F.
- Vimos:
  - un algoritmo simple para calcular X^+.
  - los usos de X^+: $X \rightarrow Y$ se deduce de F, X es_superllave / es_llave

Analizamos el esquema…

¿Una clave para *PersHob*?

PersHob

Documento	Apellido	Nombre	Domicilio	Hobby	Hob_Categ
11111111	Lorda	Ana	San Juan 25	Bailar	Arte
11111111	Lorda	Ana	San Juan 25	Cantar	Arte
11111111	Lorda	Ana	San Juan 25	Cocinar	Manualidad
22222222	Cuna	Juan	Mendoza 12	Cantar	Arte
22222222	Cuna	Juan	Mendoza 12	Pescar	Deporte
33333333	Donus	Carlos	Salta 332	Jugar al Tenis	Deporte
44444444	Pico	Clara	Vieytes 54	Bailar	Arte
44444444	Pico	Clara	Vieytes 54	Jugar al Tenis	Deporte
55555555	Pico	José	Vieytes 54	Cocinar	Manualidad
55555555	Pico	José	Vieytes 54	Jugar al Tenis	Deporte



Redundancia – Posibilidad de Inconsistencias
Anomalías de Inserción
Anomalías de Borrado

¿Dependencias Funcionales?

PersHob

Documento	Apellido	Nombre	Domicilio	Hobby	Hob_Categ
11111111	Lorda	Ana	San Juan 25	Bailar	Arte
11111111	Lorda	Ana	San Juan 25	Cantar	Arte
11111111	Lorda	Ana	San Juan 25	Cocinar	Manualidad
22222222	Cuna	Juan	Mendoza 12	Cantar	Arte
22222222	Cuna	Juan	Mendoza 12	Pescar	Deporte
33333333	Donus	Carlos	Salta 332	Jugar al Tenis	Deporte
44444444	Pico	Clara	Vieytes 54	Bailar	Arte
44444444	Pico	Clara	Vieytes 54	Jugar al Tenis	Deporte
55555555	Pico	José	Vieytes 54	Cocinar	Manualidad
55555555	Pico	José	Vieytes 54	Jugar al Tenis	Deporte

PersHob(<u>documento</u>, apellido, nombre, domicilio, <u>hobby</u>, hob_categoria)

Dependencias funcionales: F = {
- documento → documento, apellido, nombre, domicilio,
- hobby → hobby, hob_categoria }

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

¿Qué constituye un mal diseño de base de datos?

- Un mal diseño de la base de datos es aquel que tiene uno o más de los siguientes problemas de:
  - Redundancia en la información.
  - Inconsistencia en los datos.
  - Anomalías de inserción.
  - Anomalías de borrado.

Problemas de diseño

REDUNDANCIA
- Si una persona tiene más de un hobby existen más de una fila para la misma persona, junto con sus datos personales (documento, apellido, nombre y domicilio).

INCOSISTENCIA
- ¿Pueden existir para el mismo número de documento domicilios distintos? ¿Cuál es el correcto?

ANOMALÍAS DE INSERCIÓN
- El problema de conocer todos los datos de una persona, pero no su hobby. Por ser hobby parte de la clave primaria, no es posible ingresar la persona a la tabla.

ANOMALÍAS DE BORRADO
- Si para una persona deseamos borrar la información del único hobby que tiene cargado, debemos borrar todos sus datos

¿Cuál es la solución?

- ¿Cuál es la solución si existen problemas de diseño en el modelo relacional?
  - **Encontrar una descomposición u organización mejor para los atributos.**
- ¿Cualquier descomposición de R es buena?
  - **NO!**
  - Se buscan descomposiciones con las propiedades:
    - **Join sin pérdida,**
    - **preserven dependencias,**
    - **respeten una buena forma normal.**

Descomposición de Esquemas

Descomposición #1

Propiedades:
- Join con pérdida
- Preserva dependencias
- FNBC

Hobby

Hobby	Hob_Categ
Bailar	Arte
Cantar	Arte
Cocinar	Manualidad
Pescar	Deporte
Jugar al Tenis	Deporte

Hobby (<u>hobby</u>, hob_categ)
hobby → hobby, hob_categoria

Persona

Documento	Apellido	Nombre	Domicilio
11111111	Lorda	Ana	San Juan 25
22222222	Cuna	Juan	Mendoza 12
33333333	Donus	Carlos	Salta 332
44444444	Pico	Clara	Vieytes 54
55555555	Pico	José	Vieytes 54

Qué hobbies tiene Ana?

Persona (<u>documento</u>, apellido, nombre, domicilio)
documento → documento, apellido, nombre, domicilio

Descomposición #2

Propiedades:
- Join con pérdida.
- No preserva dependencias.
- 1FN.

Per-Hobby

Documento	Hobby
11111111	Bailar
11111111	Cantar
11111111	Cocinar
22222222	Cantar
22222222	Pescar
33333333	Jugar al Tenis
44444444	Bailar
44444444	Jugar al Tenis
55555555	Cocinar

Per-Hob(<u>documento, hobby</u>)
Sin dependencias funcionales

Persona

Documento	Apellido	Nombre	Domicilio	Hob_Categ
11111111	Lorda	Ana	San Juan 25	Arte
11111111	Lorda	Ana	San Juan 25	Manualidad
22222222	Cuna	Juan	Mendoza 12	Arte
22222222	Cuna	Juan	Mendoza 12	Deporte
33333333	Donus	Carlos	Salta 332	Deporte
44444444	Pico	Clara	Vieytes 54	Arte
44444444	Pico	Clara	Vieytes 54	Deporte
55555555	Pico	José	Vieytes 54	Manualidad
55555555	Pico	José	Vieytes 54	Deporte

Cuáles son los hobbies de Ana?

Persona(<u>documento</u>, nombre, domicilio, <u>hob_categ</u>)
F= {documento→documento, apellido, nombre, hob_categ, documento→hob_categ, documento}

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Descomposición #3

Propiedades:
- Join sin pérdida.
- Preserva dependencias.
- FNBC

Hobby

Hobby	Hob_Categ
Bailar	Arte
Cantar	Arte
Cocinar	Manualidad
Pescar	Deporte
Jugar al Tenis	Deporte

hobby → hob_categoria

Persona

Documento	Apellido	Nombre	Domicilio
11111111	Lorda	Ana	San Juan 25
22222222	Cuna	Juan	Mendoza 123
33333333	Donus	Carlos	Salta 332
44444444	Pico	Clara	Vieytes 54
55555555	Pico	José	Vieytes 54

documento → apellido, nombre, domicilio

Per-Hobby

Documento	Hobby
11111111	Bailar
11111111	Cantar
11111111	Cocinar
22222222	Cantar
22222222	Pescar
33333333	Jugar al Tenis
44444444	Bailar
44444444	Jugar al Tenis
55555555	Cocinar

Descomposiciones

Definición – una **descomposición** de un esquema de relación **$R(A_1,A_2,\ldots, A_n)$** es una colección de subconjuntos R_i de R de la forma $\rho=(R_1,R_2,\ldots,R_k)$ tal que $R=R_1 \cup R_2 \cup \ldots \cup R_k$.

Ejemplo:
- PersHob (dni, apellido, nombre, domicilio, hobby, hob_categ)
  - **R_1 = PERSONA** (dni, apellido, nombre, domicilio)
  - **R_2 = PERSONA_HOBBY** (dni, hobby)
  - **R_3 = HOBBY** (hobby, hob_categ)
- *Obs*ervación: la descomposición no exige que los R_i's sean disjuntos.

Propiedad #1 de una Descomposición: JOIN SIN PÉRDIDA

Join sin Pérdida (JSP)

Un esquema de relación $R(A_1,A_2, \ldots, A_n)$ descompuesto en

$\rho = (R_1, R_2, \ldots, R_k)$

es una descomposición join sin pérdida con respecto al conjunto de df's F *si cada relación r de R satisfaciendo F es tal que:*

$r = \Pi_{R_1}(r) \bowtie \Pi_{R_2}(r) \bowtie \ldots \bowtie \Pi_{R_k}(r)$

- Esto es, toda r(R) es igual al join natural de sus proyecciones en los R_i's.
- Cuando $r \subset \Pi_{R_1}(r) \bowtie \Pi_{R_2}(r) \bowtie \ldots \bowtie \Pi_{R_k}(r)$ entonces hay más información en la descomposición que en la relación original y se dice que es una *descomposición join con pérdida*.

Join con Pérdida

r

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_2	d_1
a_1	b_1	c_3	d_1
a_1	b_1	c_1	d_2

r_1

A	B	D
a_1	b_1	d_1
a_1	b_1	d_2

r_2

B	C
b_1	c_1
b_1	c_2
b_1	c_3

$r_1 \bowtie r_2$

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_1	c_1	d_2
a_1	b_1	c_2	d_1
a_1	b_1	c_2	d_2
a_1	b_1	c_3	d_1
a_1	b_1	c_3	d_2

$r \subset r_1 \bowtie r_2$
Es JCP

Test Join sin Pérdida

Teorema

- Sea $\rho=(R_1,R_2)$ una descomposición en dos subesquemas para R y sea F el conjunto de df's, entonces ρ es una descomposición jsp con respecto a F *si y solo si se verifica que*:
  - **$(R_1 \cap R_2) \rightarrow R_2$, o bien:**
  - **$(R_1 \cap R_2) \rightarrow R_1$.**
- **Importante:** es suficiente con encontrar a una de tales dependencias en F^+.

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Ejemplo

- Dado:
  - *PersHob* (documento, apellido, nombre, domicilio, hobby, hob_categ)
  - F = {documento $\rightarrow$ documento, apellido, nombre, domicilio, hobby $\rightarrow$ hob_categ}
- La descomposición#1 en los subesquemas:
  - *R1 = Personas* (<u>documento</u>, apellido, nombre, domicilio)
  - *R2 = Hobbies* (<u>hobby</u>, hob_categ)

$$(R_1 \cap R_2) \not\rightarrow R_1 \qquad (R_1 \cap R_2) \not\rightarrow R_2.$$

Ejemplo

- La descomposición# 3 en los sub-esquemas:
  - R_1 = PERSONA (<u>documento</u>, apellido, nombre, domicilio), con F1= {documento $\rightarrow$ apellido, nombre, domicilio}
  - R_2 = HOBBY (<u>hobby</u>, hob_categ), con F2={hobby $\rightarrow$ hob_categ}
  - R_3 = PER-HOBBY (<u>documento, hobby</u>)

$(R_1 \cap R_3) \rightarrow R_1 \qquad ((R_1 \cup R_3) \cap R_2) \rightarrow R_2$, luego la descomposición es jsp

Testeo de una descomposición

- Una descomposición válida debería cumplir con la propiedad jsp.
- Para testear si la descomposición de R, $\rho=(R_1,R_2)$ en *dos subesquemas* tiene la propiedad j.s.p
  - Usar el Teorema.
- Para testear si la descomposición de R, $\rho=(R_1,R_2 \ldots R_n)$ tiene la propiedad j.s.p
  - Usar repetidamente el teorema ó
  - Usar el algoritmo Anexo.

Propiedad #2 de una Descomposición: PRESERVA DEPENDENCIAS FUNCIONALES

Preserva Dependencias (PD)

Sea **R** un esquema de relación y **F** el conjunto de df's para **R**, $R_i \subseteq$ **R**. La *proyección de F en R_i*, denotado por $\Pi_{R_i}(F)$, **es el conjunto de df's X$\rightarrow$Y en F^+ tales que XY $\subseteq$ R_i.**

Definición – Sea $\rho = (R_1,R_2,\ldots, R_k)$ una descomposición de R y *F* el conjunto de df's para R. Se dice que ρ ***preserva dependencias*** si:

$$\cup^k_{i=1} \Pi_{R_i}(F^+) = \Pi_{R_1}(F^+) \cup \Pi_{R_2}(F^+) \cup \ldots \cup \Pi_{R_k}(F^+) \;|= F$$

Ejemplo 1

- Sea **R=(CSZ),**
  - ***F* = {CS $\rightarrow$ Z, Z $\rightarrow$ C}**
  - **ρ = (CZ, SZ).**
- Luego:
  - R_1 = (CZ) y **$F_1 = \Pi_{R_1}(F^+)$ = {Z $\rightarrow$ C, Z $\rightarrow$ Z, Z $\rightarrow$ ZC, C $\rightarrow$ C, CZ $\rightarrow$ C CZ $\rightarrow$ Z, CZ $\rightarrow$ CZ },**
  - R_2 = (SZ) y **$F_2 = \Pi_{R_2}(F^+)$ = {S $\rightarrow$ S, SZ $\rightarrow$ S, Z $\rightarrow$ Z, SZ $\rightarrow$ Z, SZ $\rightarrow$ SZ}.**
- ¿Preserva dependencias?
  - No preserva dependencias, es imposible recuperar la dependencia funcional CS$\rightarrow$Z.
- ¿Es join sin pérdida ?
  - **F |= (SZ $\cap$ CZ) $\rightarrow$ CZ ó F|= (SZ $\cap$ CZ) $\rightarrow$ SZ**
  - **$(Z^+)_F$ = ZC.**

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Ejemplo 2

- Sea **R=(ABCD)**, ***F* = {A→B, C→D}** y **ρ = (AB, CD)**.
- Esto es:
  - R_1 = (AB) y $F_1 = \Pi_{R_1}(F^+)$ = {A → B (y d.f. triviales) },
  - R_2 = (CD) y $F_2 = \Pi_{R_2}(F^+)$ = {C → D (y d.f. triviales)}.
- Esta descomposición preserva dependencias puesto que $F = F_1 \cup F_2 = \Pi_{R_1}(F) \cup \Pi_{R_2}(F)$.
- Sin embargo, no es j.s.p pues F no implica ninguna de las siguientes df's:
  - $(AB \cap CD) \to AB$? $\varnothing \to \varnothing$ ó
  - $(AB \cap CD) \to CD$? $\varnothing \to \varnothing$.

Preserva de Dependencias

- Sin embargo, para responder correctamente deberíamos previamente calcular F^+ y proyectar todas las dependencias que se proyectan en cada R_i
- Ejemplo:
  - Sea **R=(ABCD)**, **F = {A→BC, B→AD}** y **ρ = (AD, ABC).**
  - **ρ = (AD, ABC)** preserva dependencias:
    - R_1 = (AD) y $F_1 = \Pi_{R_1}(F^+)$ = {A → D (y d.f. triviales)}
    - R_2 = (ABC) y $F_2 = \Pi_{R_2}(F^+)$ = {A → BC, B → A (y d.f. triviales)}.

Preservación de Dependencias

- Existe un algoritmo alternativo no requiere de F^+.

Definición: Una **R-operación** sobre un conjunto de atributos Z con respecto a una descomposición y conjunto de df's se define como:

$$Z \cup ((Z \cap R_i)^+ \cap R_i)$$

- Esta operación agrega a Z los atributos A tales que $(Z \cap R_i) \to A$ está en $\Pi_R(F)$.

Algoritmo Clausura sobre ρ

Algoritmo Clausura

Datos de Entrada: Un esquema de relación $R=(A_1 \ldots A_n)$, una descomposición $\rho=(R_1,\ldots,R_k)$, un conjunto de df's F y un conjunto de atributos X.

Datos de Salida: Z (X^+: la clausura de X en ρ).

Comienzo del Algoritmo

$Z \leftarrow X$
Repetir
  $W \leftarrow Z$
  Para cada esquema R_i Hacer
    $Z \leftarrow Z \cup ((Z \cap R_i)^+ \cap R_i)$
    { Esta es la clausura de Z con respecto a F }
Hasta W = Z

Fin del Algoritmo

Ejemplo 3

- Sea R=(ABCD), **ρ=(AB, BC, CD)** y:

  ***F* = {A → B, B → C, C → D, D → A }**
- Veremos si ρ preserva dependencias, esto es:

  ¿ $G = \Pi_{AB}(F) \cup \Pi_{BC}(F) \cup \Pi_{CD}(F) \models F$?
- A → B, B → C y C → D se preservan trivialmente.
- Se preserva **D → A**?.

¿Se preserva D → A?

- Dados:

  R=(ABCD)

  ρ=(AB, BC, CD)

  ***F* = {A → B, B → C, C → D, D → A }**
- Sea Z = {D}. Aplicar la *AB-operación* produce:

  $\{D\} \cup (\,(\{D\} \cap \{A, B\})_F^+ \cap \{A, B\}\,) = \{D\}$
- Similarmente, la *BC-operación* no produce cambios.
- La *CD-operación* produce:

  $\{D\} \cup ((\{D\} \cap \{C, D\})^+ \cap \{C, D\}) = \{D\} \cup (\{D\}^+ \cap \{C, D\})$

  $\{D\} \cup (\{A, B, C, D\} \cap \{C, D\}) = \{D\} \cup \{C, D\} = \{C, D\}$

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

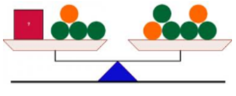
¿Se preserva D → A?

- En el siguiente paso de la *BC-operación* produce {B, C, D}.
- En el último paso, la *AB-operación* produce {A, B, C, D}.
- Por lo tanto, la clausura de D con respecto a G es ABCD.
- Luego ρ = (AB, BC, CD) *preserva dependencias*.
- Por lo tanto, tenemos que:
  - $G = \Pi_{AB}(F) \cup \Pi_{BC}(F) \cup \Pi_{CD}(F)$.
  - $G = \{A \rightarrow B, B \rightarrow A\} \cup \{B \rightarrow C, C \rightarrow B\} \cup \{C \rightarrow D, D \rightarrow C\}$.

Descomposición: Propiedades

- Dados R un esquema de relación, *F* un conjunto de df´s definido sobre R, en ciertos casos R no responde a un buen diseño relacional: problemas de redundancia, inconsistencias, anomalías de inserción y borrado.
- En tal caso la solución será descomponer a R en $\rho(R_1, R_2 ..., R_k)$ una representación mejor para R.
- Una descomposición ρ debe satisfacer:
  - Ser una descomposición JSP ➡ SIEMPRE !!
  - Idealmente, además se espera que ρ preserve las dependencias funcionales de F ➡ DESEABLE !!
  - En una Forma Normal Alta (FNBC, 3FN).

Conjunto de Dependencias Funcionales:
EQUIVALENCIAS Y CUBRIMIENTOS



Sobre el conjunto F de Dependencias Funcionales

Ejemplo :

PeliSocio (nroSocio, nombreSocio, dniSocio, idPelicula, nombrePelicula, fechaAlquiler)

F = {nroSocio → nombreSocio,dniSocio; idPelicula→nombrePelicula; idPelicula,fechaAlquiler→nroSocio, **dniSocio→nroSocio**}

G = {dniSocio → nombreSocio,nroSocio; **nroSocio→dniSocio,nombreSocio**; idPelicula→nombrePelicula; idPelicula,fechaAlquiler→dniSocio}

¿Qué podemos decir de F y G?

Cubrimientos de conjuntos de DF' s

Definición – dos conjuntos de df's **F** y **G** sobre un mismo esquema **R** son **equivalentes (F ≡ G) si:**

$$F^+ = G^+$$

- Si **F ≡ G** entonces, ***F** es un cubrimiento para **G*** (y viceversa G es un cubrimiento para F).

Lema: Dados dos conjuntos de df's **F** y **G** sobre el esquema **R**, entonces **F ≡ G** si y solo si:

F |= G y

G |= F.

Ejemplos

Ejemplo 1:
- Sea R (ABC)

  F = { A → BC, B → A} y **G = { A → B, B → AC}**
- **F |= G** ? sí
- **G |= F** ? sí

$\boxed{F \equiv G}$ F cubre a G y G cubre a F

Ejemplo 2:
- Sea R (ABCD)

  F = { A → BC, B → AD } y **G = { A → B, B → AC}**
- **F** |= **G** ? sí
- **G** |= **F** ? no

$\boxed{F \not\equiv G}$ F cubre a G, G NO cubre a F

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Qué pasa en este caso

Ejemplo 3:

– Sea R (ABCD)

F = { A → BC, B → CD, A → D} y **G = { A → B, B → CD}**

– **F |= G** ? sí

– **G |= F** ? sí

$\boxed{F \equiv G}$ F cubre a G y G cubre a F

Analicemos en detalle los conjuntos F y G, R (ABCD):

F = { A → BC, B → CD, A → D} y

G = { A → B, B → CD}

¿Qué conjunto de restricciones elegirías? ¿Por qué?

Cubrimiento NoRedundante

Cuando **F |= G,** entonces **F** *cubre a* **G**.

Definiciones

- Un *conjunto F de dependencias funcionales es no redundante* si no contiene ningún subconjunto propio ***F'* (*F'* ⊂ *F*)** tal que ***F'* ≡ *F***. Caso contrario, se dice que *F es redundante*.
- ***F* es un cubrimiento no redundante para *G*** si *F* es un cubrimiento para *G* y *F* es no redundante.

Cubrimientos

- **Definición**: Un conjunto de df's F es *no redundante* si no existe ningún subconjunto propio F' de F equivalente a F, esto es, no existe F' tal que F' ⊂ F y F ≡ F'.
- **Definición**: Un conjunto de df's F es **mínimo** si no existe un conjunto de df's G equivalente a F con menos df's, esto es, no existe G tal que || G || < || F || y G ≡ F.
- **Definición**: Un conjunto de df's F es *óptimo* si es mínimo y no existe un conjunto de df's G equivalente a F de un tamaño menor (el tamaño es la suma de los atributos que aparecen a izquierda y derecha de cada df).

Resultados

- Si esquema de relación R no es de calidad ⇒ tiene problemas de asegurar integridad en los datos que persiste.
- La solución es encontrar otro diseño relacional que resulte de descomponer R en $\rho(R_1, R_2 ..., R_k)$.
  - $\rho(R_1, R_2 ..., R_k)$ debe ser una descomposición j.s.p.
  - $\rho(R_1, R_2 ..., R_k)$ debe respetar alguna Forma Normal alta (3FN o FNBC).
  - Es deseable que $\rho(R_1, R_2 ..., R_k)$ preserve dependencias.
- Las calidad de un esquema R está condicionada por el conjunto de restricciones o dependencias funcionales definidas en R.
- Dados R y F, es deseable encontrar el un conjunto de restricciones F' equivalente a F que sea un conjunto de restricciones óptimo o al menos mínimo.

¿Cómo encontrar las llaves de R?

Dado un esquema de relación **$R(A_1,…,A_n)$**, y un conjunto F de df's mínimo reducido, entonces para cada atributo A_i:

- **Siempre** es parte de una llave si no aparece ni a izquierda ni a derecha en ninguna dependencia de F, o aparece solamente a izquierda en df's de F.
- **Nunca** es parte de una llave si aparece solamente a derecha en las df's de F.
- **Tal vez** es parte de una llave si aparece a izquierda y derecha en dos o más df's de F.

Un conjunto X es llave de R si X^+=R.

Ejemplo: encontrar todas las llaves

Sea R=(ABDEGHI), y un conjunto mínimo F de df's { A → B, BG → H, G → DE, AE → G }.

- **Siempre** son parte de una llave: A (a izquierda solamente) e I (ni a izquierda ni a derecha).
- **Nunca** son parte de una llave: D y H (aparecen a derecha solamente).
- **Tal vez** son parte de esta llave: B, E y G (aparecen a izquierda y derecha).
- Posibles llaves: { AI, AIB, AIE, AIG, AIBE, AIBG, AIEG, AIBEG }.
- Llaves de esta relación: { AIE, AIG }

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Temas de la Clase de Hoy

- Descomposiciones
- Propiedades de las descomposiciones:
  - Join sin pérdida
  - Preserva dependencias.
- Cubrimientos de conjuntos de dependencias.

Bibliografía:

- *Principles of Database and Knowledge Based Systems*. Jeffrey Ullman. Capítulo 7
- *Database System Concepts*. A. Silberschatz. Capítulo 8 (edición 2010)

Algoritmo EsJoinSinPérdida

Algoritmo EsJoinSinPérdida

Datos de Entrada: Un esquema de relación $R=(A_1 \ldots A_n)$, una descomposición $\rho=(R_1,\ldots,R_k)$ y un conjunto de df's F.

Datos de Salida: EsJSP (verdadero o falso).

Comienzo del Algoritmo

Se construye una tabla de n columnas y k filas.
La fila i corresponde al esquema R_i.
La columna j corresponde al atributo A_j.
En la fila i y columna j se coloca a_j si A_j está en R_i. Sino va b_{ij}.
Repetir para cada df $X \rightarrow Y$ en F:
Si existen dos filas que coinciden en todos los atributos de X se igualan todos los atributos de Y en ambas filas (en caso de igualar a's y b's tiene prioridad la a).
Hasta que no haya más cambios o una fila tiene todos a's.
Si alguna fila tiene todos los atributos a's
Entonces EsJSP ← verdadero.
Sino EsJSP ← falso.

Fin del Algoritmo

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y F = { $A \rightarrow C$, $B \rightarrow C$, $C \rightarrow D$, $DE \rightarrow C$, $CE \rightarrow A$ }

	A	B	C	D	E
AB	a_1	a_2	b_{13}	b_{14}	b_{15}
AD	a_1	b_{22}	b_{23}	a_4	b_{25}
AE	a_1	b_{32}	b_{33}	b_{34}	a_5
BE	b_{41}	a_2	b_{43}	b_{44}	a_5
CDE	b_{51}	b_{52}	a_3	a_4	a_5

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y F = { $A \rightarrow C$, $B \rightarrow C$, $C \rightarrow D$, $DE \rightarrow C$, $CE \rightarrow A$ }

	A	B	C	D	E
AB	a_1	a_2	b_{13}	b_{14}	b_{15}
AD	a_1	b_{22}	b_{13}	a_4	b_{25}
AE	a_1	b_{32}	b_{13}	b_{34}	a_5
BE	b_{41}	a_2	b_{43}	b_{44}	a_5
CDE	b_{51}	b_{52}	a_3	a_4	a_5

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y F = { $A \rightarrow C$, $B \rightarrow C$, $C \rightarrow D$, $DE \rightarrow C$, $CE \rightarrow A$ }

	A	B	C	D	E
AB	a_1	a_2	b_{13}	b_{14}	b_{15}
AD	a_1	b_{22}	b_{13}	a_4	b_{25}
AE	a_1	b_{32}	b_{13}	b_{34}	a_5
BE	b_{41}	a_2	b_{13}	b_{44}	a_5
CDE	b_{51}	b_{52}	a_3	a_4	a_5

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y F = { $A \rightarrow C$, $B \rightarrow C$, $C \rightarrow D$, $DE \rightarrow C$, $CE \rightarrow A$ }

	A	B	C	D	E
AB	a_1	a_2	b_{13}	a_4	b_{15}
AD	a_1	b_{22}	b_{13}	a_4	b_{25}
AE	a_1	b_{32}	b_{13}	a_4	a_5
BE	b_{41}	a_2	b_{13}	a_4	a_5
CDE	b_{51}	b_{52}	a_3	a_4	a_5

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Ejemplo

Sean R=(ABCDE), ρ=(AB,AD,AE,BE,CDE) y
F = { A → C, B → C, C → D, DE → C, CE → A }

	A	B	C	D	E
AB	a_1	a_2	b_{13}	a_4	b_{15}
AD	a_1	b_{22}	b_{13}	a_4	b_{25}
AE	a_1	b_{32}	a_3	a_4	a_5
BE	b_{41}	a_2	a_3	a_4	a_5
CDE	b_{51}	b_{52}	a_3	a_4	a_5

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y
F = { A → C, B → C, C → D, DE → C, CE → A }

	A	B	C	D	E
AB	a_1	a_2	a_3	a_4	b_{15}
AD	a_1	b_{22}	a_3	a_4	b_{25}
AE	a_1	b_{32}	a_3	a_4	a_5
BE	b_{41}	a_2	a_3	a_4	a_5
CDE	b_{51}	b_{52}	a_3	a_4	a_5

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y
F = { A → C, B → C, C → D, DE → C, CE → A }

	A	B	C	D	E
AB	a_1	a_2	a_3	a_4	b_{15}
AD	a_1	b_{22}	a_3	a_4	b_{25}
AE	a_1	b_{32}	a_3	a_4	a_5
BE	a_1	a_2	a_3	a_4	a_5
CDE	a_1	b_{52}	a_3	a_4	a_5

Ejemplo

Sean R=(ABCDE), ρ=(AB, AD, AE, BE, CDE) y
F = { A→ C, B → C, C → D, DE → C, CE → A }

	A	B	C	D	E
AB	a_1	a_2	a_3	a_4	b_{15}
AD	a_1	b_{22}	a_3	a_4	b_{25}
AE	a_1	b_{32}	a_3	a_4	a_5
BE	a_1	a_2	a_3	a_4	a_5
CDE	a_1	b_{52}	a_3	a_4	a_5

JSP

Ejemplo

Sea R = (ABC), F = {A → B} y ρ = (AB,AC).

	A	B	C
AB	a_1	a_2	b_{13}
AC	a_1	b_{22}	a_3

	A	B	C
AB	a_1	a_2	b_{13}
AC	a_1	a_2	a_3

AB ∩ AC = A

¿ F |= A → AB ?   ¿ F |= A → AC ?

SI   NO

Es JSP

Ejemplo 2

Sea R = (ABC), F = {A → B} y ρ = (AB,BC).

	A	B	C
AB	a_1	a_2	b_{13}
BC	b_{21}	a_2	a_3

	A	B	C
AB	a_1	a_2	b_{13}
BC	b_{21}	a_2	a_3

AB ∩ BC = B

¿ F |= B → A B?   ¿ F |= B → BC ?

NO   NO

No es JSP

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.

Algoritmo PreservaDependencias

Algoritmo PreservaDependencias
Datos de Entrada: Un esquema de relación $R=(A_1 \ldots A_n)$, una descomposición $\rho=(R_1,\ldots,R_k)$ y un conjunto de df's F.
Datos de Salida: PD (verdadero o falso).
Comienzo del Algoritmo
Sea $G = \cup^{k}_{i=1} \Pi_{R_i}(F^+)$
$H \leftarrow F$
$PD \leftarrow$ Verdadero
Repetir
Sea $X \rightarrow Y$ la primera df de H
$H \leftarrow H - \{ X \rightarrow Y \}$
Si $Y \not\subset X^+_G$ entonces $PD \leftarrow$ Falso
Hasta (No PD) o ($H = \varnothing$)
Fin del Algoritmo

Sobre las descomposiciones

- Un esquema de relación R no es de calidad, i.e. tiene problemas de redundancia y posibilidad de inconsistencias, anomalías de inserción y borrado, la solución es encontrar otro diseño relacional que resulte de descomponer R en $\rho(R_1, R_2 \ldots, R_k)$.
- $\rho(R_1, R_2 \ldots, R_k)$ debe ser una descomposición j.s.p.
- $\rho(R_1, R_2 \ldots, R_k)$ debe respetar algununa Forma Normal alta (3FN o FNBC)
- Es deseable que $\rho(R_1, R_2 \ldots, R_k)$ preserve dependencias

Estas transparencias proveen sólo una referencia a los temas. Para su estudio debe remitirse a la bibliografía.