PROGRAMA EDUCATIVO INFORMATICA ADMINISTRATIVA

UNIDAD DE APRENDIZAJE ADMINISTRACION DE BASES DE DATOS

Unidad de competencia VII

Conocer las tendencias en el uso e implementación de base de datos

Sistemas Basados en la lógica

ELABORACION
ADRIAN TRUEBA ESPINOSA

PRESENTACIÓN DEL CURSO

Una de las principales actividades del Licenciado en Informática Administrativa es la elaboración de bases de datos, desde su diseño hasta la administración por lo que, cuyas bases deben ser adquiridas en su formación. La administración de bases de datos como una parte de la informática, evoluciona continuamente, sin embargo la administración de las bases de datos es uno de los conocimientos base en la construcción de aplicaciones de mediana y alta complejidad. Esta unidad de aprendizaje pretende introducir al alumno en el manejo y almacenamiento de datos por medios electrónicos

CONTENIDO DEL CURSO

Unidad 1: CONCEPTOS FUNDAMENTALES DE BASES DE DATOS

Unidad 2: MODELO DE DATOS

Unidad 3: METODOLOGÍA DE DISEÑO DE BASE DE DATOS

Unidad 4: CONCEPTOS DE DISEÑO DE APLICACIONES DE BASES DE DATOS

Unidad 5: DISEÑO E IMPLEMENTACIÓN DE BASES DE DATOS

Unidad 6: ADMINISTRACIÓN Y SEGURIDAD EN BASE DE DATOS

Unidad 7. TENDENCIAS ACTUALES EN BASES DE DATOS

METAS A ALCANZAR

Que el alumno conozca los elementos teóricos y prácticos de las bases de datos lógicas

- Concepto de hechos y reglas

- Lógica matemática

OBJETIVO DEL MATERIAL DIDÁCTICO

Conocer las tendencias en el uso e implementación de base de datos lógicas

METODOLOGÍA DEL CURSO

El curso se desarrollará bajo el siguiente proceso de estudio:

1. Exposición de parte del profesor mediante la utilización de este material en diapositivas.
2. Control de lecturas selectas que el profesor asignará para complementar la clase.
3. Investigación de temas, conceptos, procesos de las bases de datos deductivas o logica.
4. Participación en clases
5. Prácticas de laboratorio

UTILIZACIÓN DEL MATERIAL DE DIAPOSITIVAS

El material didáctico visual es una herramienta de estudio que sirve como una guía para que el alumno repase los temas más significativos de “Las bases de datos lógicas”,.

UNIDAD DE COMPETENCIA VII

Conocer las tendencias en el uso e implementación de base de datos

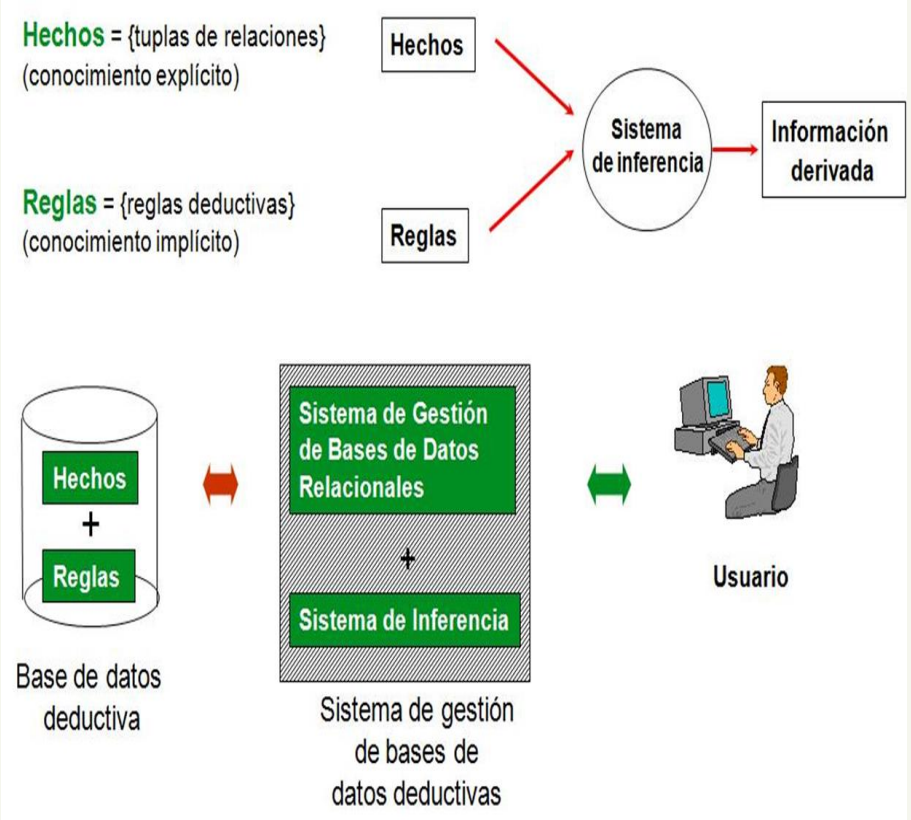
Sistemas Basados en la lógica

Definición:

Un sistema de bases de datos que tenga la capacidad de definir reglas con las cuales deducir o inferir información adicional a partir de los hechos almacenados en las bases de datos se llama ***Sistema de Bases de Datos Deductivas***.

Los fundamentos teóricos de los sistemas de ésta especie es la lógica matemática, ***a menudo se les denomina Bases de Datos Lógicas***.

Una base de datos deductiva en esencia es un programa lógico; mapeo de relaciones base hacia **hechos, y reglas**



Características:

Una Base de Datos lógica debe al menos tener las siguientes características:

1. Tener la capacidad de expresar consultas por medio de reglas lógicas.
2. Permitir consultas recursivas y algoritmos eficientes para su evaluación.
3. Contar con negaciones estratificadas.
4. Soportar objetos y conjuntos complejos.
5. Contar con métodos de optimización que garanticen la traducción de especificaciones dentro de planes eficientes de acceso.
6. Como característica fundamental de una Base de Datos Deductiva es la posibilidad de inferir información a partir de los datos almacenados, es imperativo modelar la base de datos como un conjunto de fórmulas lógicas, las cuales permiten inferir otras fórmulas nuevas.

Los sistemas Bases de Datos Deductivas intentan modificar el hecho de que los datos requeridos residan en la memoria principal (por lo que la gestión de almacenamiento secundario no tiene importancia)

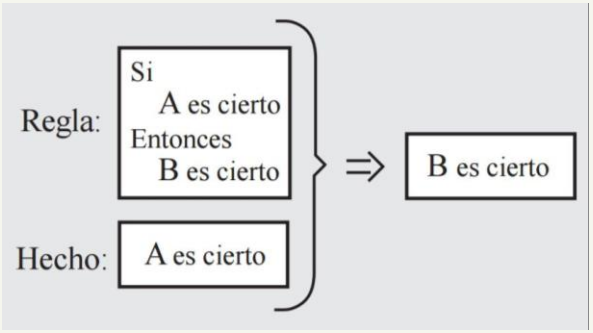
En un sistema de Bases de Datos Deductivas por lo regular se usa un lenguaje declarativo para especificar reglas. Con lenguaje declarativo se quiere decir un lenguaje que define lo que un programa desea lograr.

Una máquina de inferencia (o reglas de deducción) dentro del sistema puede deducir hechos nuevos a partir de la base de datos interpretando dichas reglas.

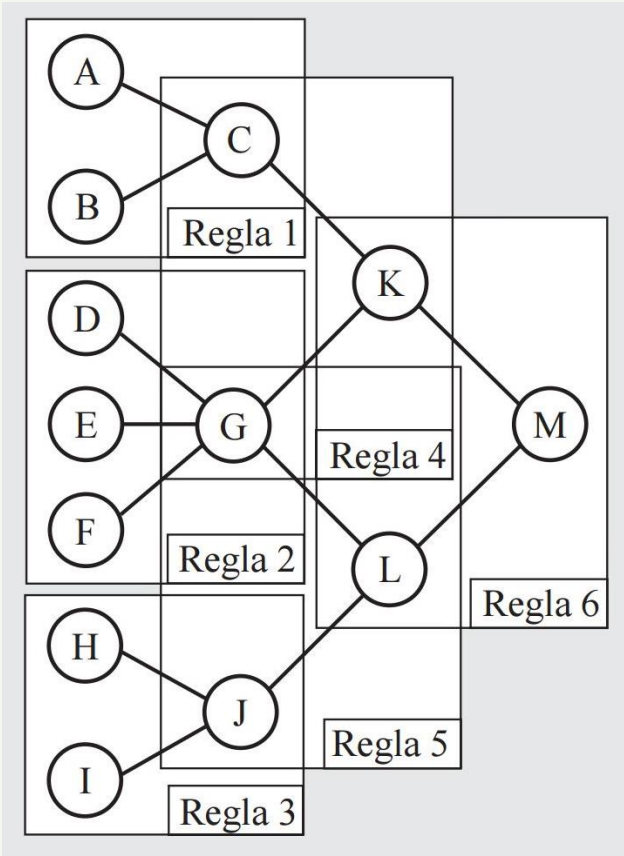
Las Bases de Datos Deductivas extienden la capacidad expresiva de las bases de datos relacionales incluyendo un conjunto de reglas que permiten definir conocimiento implícito

Hechos: Son un conjunto de premisas que intentan inferir una cierta información de una representación del mundo real.

Reglas: Son el conjunto de restricciones o cualidades que exigimos representar, se representan mediante composiciones lógicas.



Regla 1	Regla 2	Regla 3
Si A y B Entonces C	Si D, E y F Entonces G	Si H e I Entonces J
Regla 4	**Regla 5**	**Regla 6**
Si C y G Entonces K	Si G y J Entonces L	Si K y L Entonces M



Maquina de inferencia: Se distinguen dos mecanismos de inferencia:

Encadenamiento hacia atrás (backward chaining)

Inicia con la conclusión que se desea demostrar y gestiona establecer la certeza de los hechos que conducen a ella.

Encadenamiento hacia delante (forward chaining)

Efectúa comparaciones entre las reglas y los hechos disponibles de manera que se establezcan nuevos hechos hasta llegar al objetivo deseado.

Existen principalmente dos tipos de inferencia computacional para interpretar el significado teórico de las reglas:

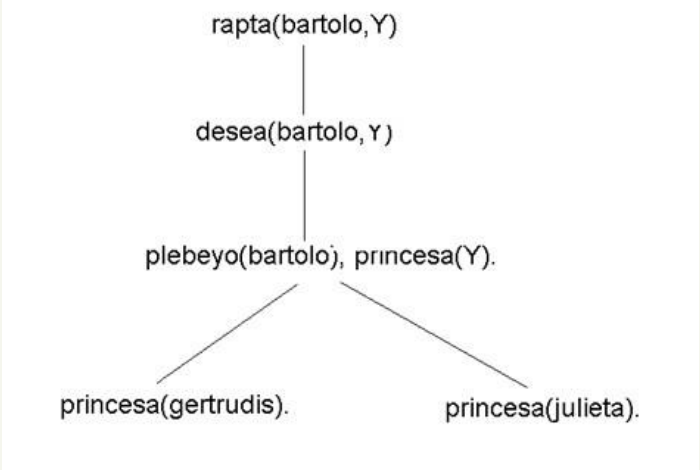
- **Mecanismo de inferencia ascendente**: también llamado encadenamiento hacia delante o resolución ascendente. La máquina de inferencia parte de los hechos y aplica las reglas para generar hechos nuevos. Conviene en primera instancia usar una estrategia de búsqueda para generar sólo los hechos que sean pertinentes a una consulta.

1) rapta(X, Y) :- rufián(X), desea(X, Y).
2) desea(X,Y) :- noble(X), princesa(Y), guapa(Y).
3) desea(X,Y) :- plebeyo(X), princesa(Y).
4) adinerado(X):-noble(X).
5) adinerado(X):-rufian(X),plebeyo(X).
6) rufián(bertoldo).
7) rufián(bartolo).
8) noble(romeo).
9) noble(bertoldo)
10) plebeyo(bartolo).
11) princesa(gertrudis).
12) princesa(julieta).
13) guapa(julieta).

- **Mecanismo de inferencia descendente:** también llamado encadenamiento hacia atrás o resolución descendente. Parte del predicado que es el objetivo de la consulta, intenta encontrar coincidencias con las variables que conduzcan a hechos válidos de la BD. Retrocede desde el objetivo buscado para determinar hechos que lo satisfacen. Si no existieran los hechos que se buscan, el sistema entonces buscará la primera regla cuya cabeza tenga el mismo nombre de predicado que la consulta.

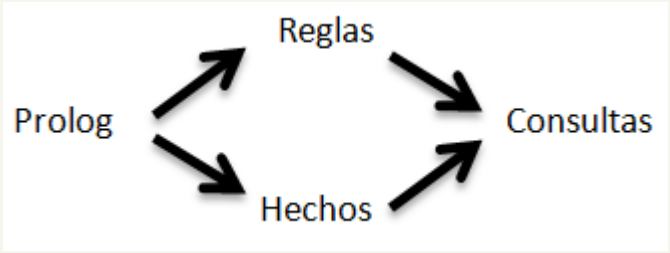
1) rapta(X, Y) :- rufián(X), desea(X, Y).
2) desea(X,Y) :- noble(X), princesa(Y), guapa(Y).
3) desea(X,Y) :- plebeyo(X), princesa(Y).

4) rufián(bertoldo).
5) rufián(bartolo).
6) noble(romeo).
7) noble(bertoldo)
8) plebeyo(bartolo).
9) princesa(gertrudis).
10) princesa(julieta).
11) guapa(julieta).



En un sistema de BDD por lo general se usa un lenguaje declarativo para especificar reglas. Con lenguaje declarativo se hace referencia a un lenguaje que define lo que un programa desea lograr, en vez de especificar los detalles de cómo lograrlo. Una máquina de inferencia (o mecanismo de deducción) dentro del sistema puede deducir hechos nuevos a partir de la BD interpretando dichas reglas. Estas BD están relacionadas con el campo de la programación lógica y el lenguaje PROLOG. Los trabajos sobre BDD basados en lógica han utilizado PROLOG como punto de partida; pero luego los trabajos se inclinaron por la utilización de un subconjunto de PROLOG llamado DATALOG. Aunque la estructura gramatical se parece a la de PROLOG

PROLOG, *Programation et Logique*, es un lenguaje de declarativo



Las BDD son muy usadas en las áreas de: inteligencia artificial, sistemas expertos, representación del conocimiento, tecnología de agentes, sistemas de información, integración de datos, por nombrar algunas. Existe una importante relación entre las BDD y la programación lógica (F. J. Muñoz y J. F. Pose, 2013)

Arquitectura de las Bases de Datos Deductivas fuente (F. J. Muñoz y J. F. Pose, 2013)

Modelo lógico como base de datos

En bases de datos basadas en la lógica se suele dividir la información en dos categorías:

1. Parte *extensional* de la base de datos. Aquellos datos que se corresponden con los hechos del sistema lógico. Los representaremos como predicados con argumentos constantes.

   Por ejemplo, `madre(ana,maría)` dará la información sobre que la madre de Ana es María.

2. Parte *intensional* de la base de datos. Reglas de programa que aparecerán habitualmente en notación Prolog:

   ```
   p(X0) :-  q1(X1),...,qn(Xn).
   ```

   Los predicados intensionales se asimilan generalmente a las vistas en las bases de datos convencionales.

Los objetivos Prolog se verán como consultas a la base de datos.

Aranda L. G. (2009)

Negación estratificada

Un sistema de bases de datos basado en un modelo lógico debe incorporar negación en los cuerpos de las consultas para ser completo con respecto al AR [75]. Al introducir la negación se da el problema de que un programa puede tener distintos significados en función del orden de evaluación de los predicados. Por ejemplo, dado el dominio `[a,b,c]` y el programa:

```
p(a).
p(b).
q(X):- not(p(X)).
```

El valor semántico del predicado `q` tendrá distinto significado:

- Si se calcula el significado de `q` antes que el de `p`, se tendrá: `{q(a),q(b),q(c)}`.
- Si se calcula entre `p(a)` y `p(b)`, se tendrá: `{q(b),q(c)}`.
- Si se calcula después del significado de `p` se tendrá: `{q(c)}`.

Para evitar este problema se impone la restricción de que cuando un predicado se define como negación de otro, se debe calcular el significado del segundo predicado después del significado del primero. En el ejemplo, se debe calcular el significado íntegro de `p` antes que el de `q`.

Aranda L. G. (2009)

2. Bases de datos deductivas: definición y formalización

Base de datos deductiva

Padre

Padre	Hijo
Juan	Luis
Luis	María
Luis	Pedro
Pedro	José

Reglas Deductivas:

Antecesor (x, y) ← Padre (x, y)
Antecesor (x, y) ← ∃z (Padre (x, z) ∧ Antecesor (z, y))

Relación derivada

Antecesor

Antecesor	Descendiente
Juan	Luis
Juan	María
Juan	Pedro
Juan	José
Luis	María
Luis	Pedro
Luis	José
Pedro	José

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Bases de Datos Relacionales

ESQUEMA

Relaciones

$R_i\ (A_{i_1}: D_{i_1}, A_{i_2}: D_{i_2}\ , ..., A_{i_{ni}}: D_{i_{ni}})$

$(1 \leq i \leq m)$ (m relaciones)

Restricciones de Integridad

W_i: W_i es una expresión lógica

$(1 \leq i \leq k)$ (k restricciones de integridad)

Bases de Datos Deductivas

ESQUEMA

Relaciones básicas:

$R_i\ (A_{i_1}: D_{i_1}, A_{i_2}: D_{i_2}\ , ..., A_{i_{ni}}: D_{i_{ni}})$

$(1 \leq i \leq m)$ (m relaciones básicas)

Relaciones derivadas:

$S_i\ (A_{i_1}: D_{i_1}\ , A_{i_2}: D_{i_2}\ , ..., A_{i_{ni}}: D_{i_{ni}})$

$(1 \leq i \leq s)$ (s relaciones derivadas)

Restricciones de Integridad

W_i: W_i es una expresión lógica

$(1 \leq i \leq k)$ (k restricciones de integridad)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Bases de Datos Relacionales

Base de datos

$R_i \subseteq (D_{i_1} \times D_{i_2} \times ... \times D_{i_{ni}})$

$(1 \leq i \leq m)$ (m relaciones)

R_i

A_{i1}	A_{i2}	.....	A_{ini}

Bases de Datos Deductivas

Base de datos

$R_i \subseteq (D_{i_1} \times D_{i_2} \times ... \times D_{i_{ni}})$

$(1 \leq i \leq m)$ (m relaciones básicas)

$S_{i_j} (x_1, x_2, ..., x_{n_i}) \leftarrow W_{i_j}$

$(1 \leq i \leq s)$ (s relaciones derivadas)

$(1 \leq j \leq K_i)$ (K_i reglas para la relación S_i)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Bases de Datos Relacionales

Base de datos

$R_i \subseteq (D_{i_1} \times D_{i_2} \times ... \times D_{i_{ni}})$

$(1 \leq i \leq m)$ (m relaciones)

Si definimos un orden en el conjunto de atributos del esquema de la relación, el concepto de *relación* coincide con el concepto de relación matemática (subconjunto del producto cartesiano de los dominios): se pierde el concepto de atributo de una relación.

Bases de Datos Deductivas

Base de datos

$R_i \subseteq (D_{i_1} \times D_{i_2} \times ... \times D_{i_{ni}})$

$(1 \leq i \leq m)$ (m relaciones básicas)

$S_{i_j}(x_1, x_2, ..., x_{n_i}) \leftarrow W_{i_j}$

$(1 \leq i \leq s)$ (s relaciones derivadas)
$(1 \leq j \leq K_i)$ (K_i reglas para la relación S_i)

En la definición de una regla deductiva, $S \leftarrow W$: W es una fórmula cuyas únicas variables libres son las variables de S.

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Esquema

Relaciones básicas:

PIEZA (codpieza: D1, desc: D2, peso: D3)
CP = {codpieza}

PROV (codprov: D4, nombre: D5, zona: D6)
CP = {codprov}

PRECIOS (codprov: D4, codpieza: D1, precio: D7)
CP = {codprov, codpieza}
CAj = {codprov} → PROV
CAj = {codpieza} → PIEZA

COMP (pieza1: D1, pieza2: D1)
CP = {pieza1, pieza2}
CAj = {pieza1} → PIEZA
CAj = {pieza2} → PIEZA

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Esquema

Relaciones derivadas:

PRECIOS3 (codprov: D4, codpieza: D1, precio: D7)
CP = {codprov, codpieza}
CAj = {codprov} → PROV
CAj = {codpieza} → PIEZA

PRECIOS_EXT (codprov: D4, nombre: D5, codpieza: D1, desc: D2, precio: D7)
CP = {codprov, codpieza}
CAj = {codprov} → PROV
CAj = {codpieza} → PIEZA

COMPONENTE (pieza1: D1, pieza2: D1)
CP = {pieza1, pieza2}
CAj = {pieza1} → PIEZA
CAj = {pieza2} → PIEZA

Restricciones de integridad:

$$\forall x \, \forall y \, (\text{COMPONENTE}(x,y) \rightarrow \neg \, \text{COMPONENTE}(y,x))$$

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Base de Datos

PIEZA

codpieza	desc	peso
pz1	tornillo	10
pz3	tuerca	11
pz8	arandela	8

PROV

codprov	nombre	zona
pv1	Juan	1
pv5	Carlos	3
pv3	Luis	3

PRECIOS

codprov	codpieza	precio
pv1	pz3	10
pv1	pz8	20
pv3	pz8	30
pv5	pz1	50

COMP

pieza1	pieza2
pz1	pz3
pz3	pz8

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Base de Datos

Reglas deductivas:

1 PRECIOS3 (x, y, z) ← ∃w (PROV (x, w, 3) ∧ PRECIOS (x, y, z))

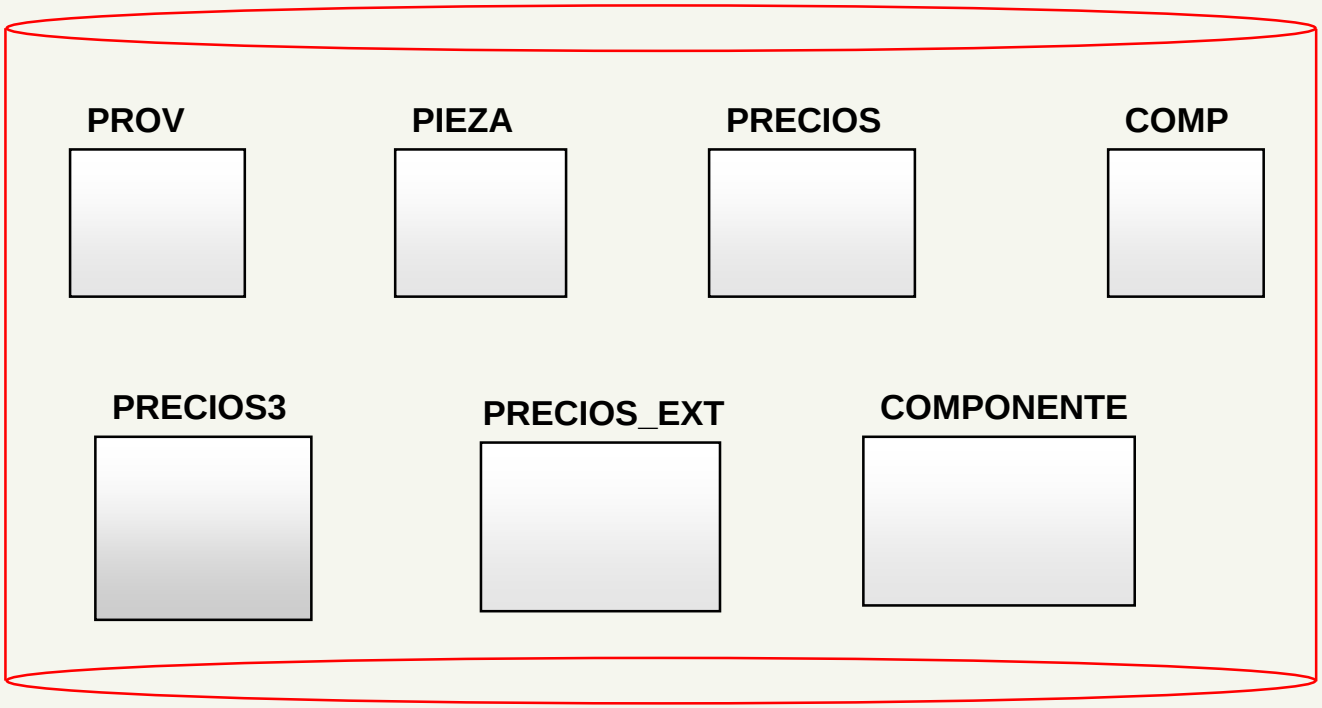
2 COMPONENTE (x, y) ← ∃z (COMP (x, z) ∧ COMPONENTE (z, y))
3 COMPONENTE (x, y) ← COMP (x, y)

4 PRECIOS_EXT (x, n, y, d, p) ← ∃z ∃w (PROV (x, n, z) ∧ PIEZA (y, d, w) ∧ PRECIOS (x, y, p))

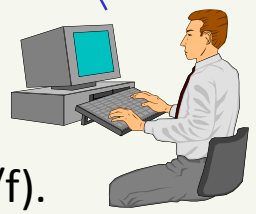
Se asume la notación de la Programación Lógica: todas las variables libres en la regla se suponen cuantificadas universalmente.

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización



El usuario desea manipular las relaciones de la BD independientemente de que sean relaciones básicas o derivadas.



Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Mecanismo de **vistas** del modelo relacional → Definición de información implícita

Relación derivada PRECIOS3 → **VISTA**

SQL92: CREATE VIEW PRECIOS3
AS SELECT codprov, codpieza, precio
FROM PRECIOS, PROV
WHERE (PRECIOS.codprov = PROV.codprov
AND
(PROV.zona=3)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Limitaciones del modelo relacional (SQL92)
en la definición de vistas:

Relación derivada COMPONENTE → **VISTA**

SQL: CREATE VIEW **COMPONENTE** AS
SELECT pieza1, pieza2
FROM COMP
UNION
SELECT pieza1, pieza2
FROM COMP, ~~**COMPONENTE**~~
.....

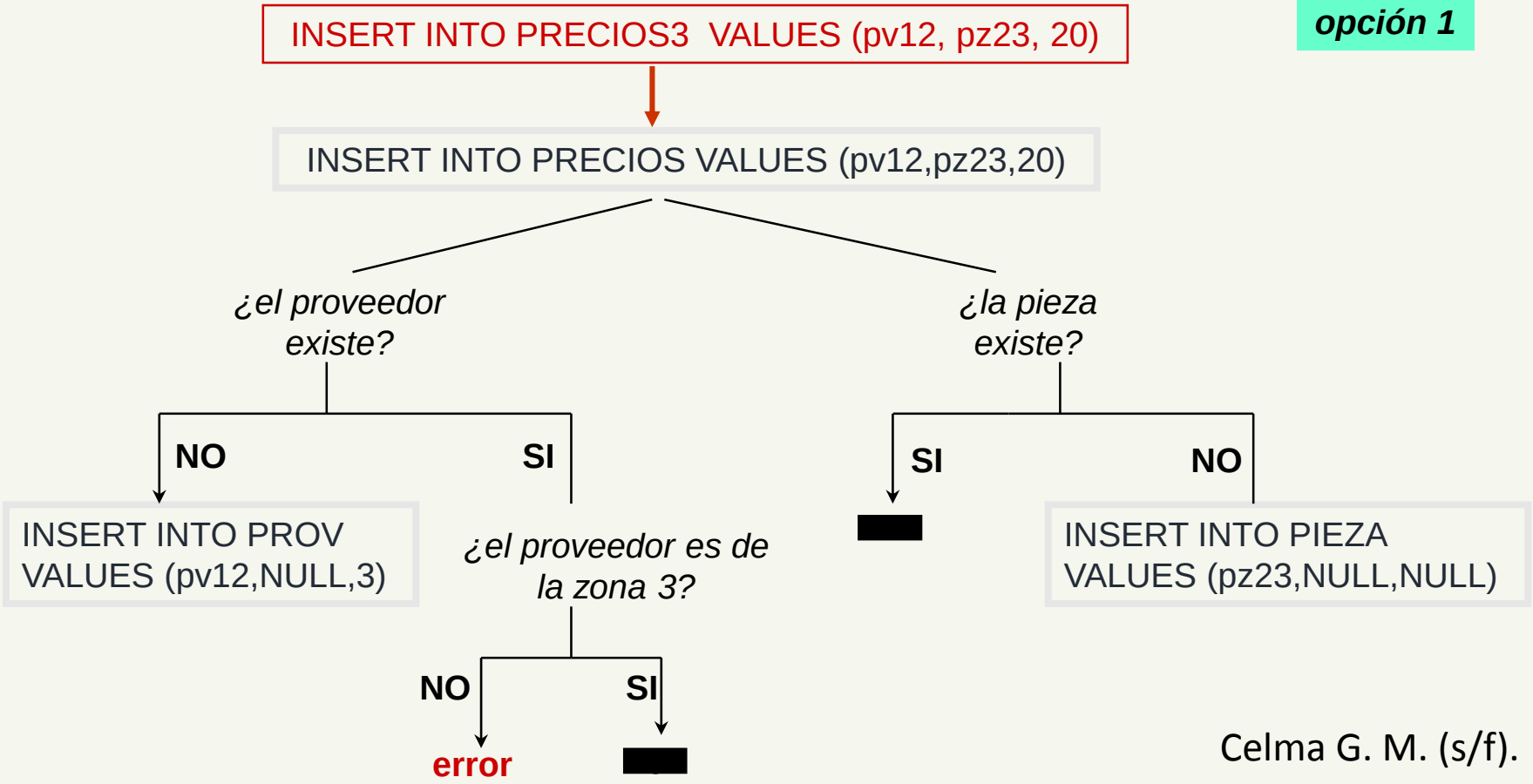
¡ En SQL92 no se pueden definir vistas recursivas!

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Limitaciones del modelo relacional (SQL92) en la actualización de vistas:

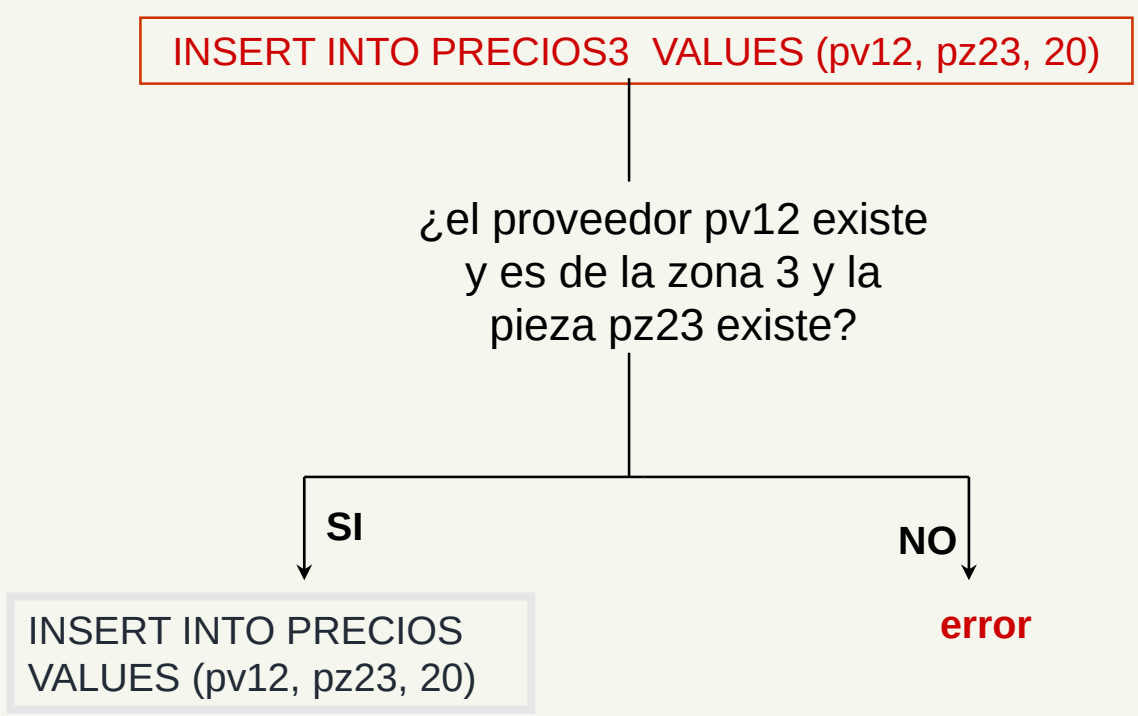
opción 1



Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Limitaciones del modelo relacional (SQL92)
en la actualización de vistas:



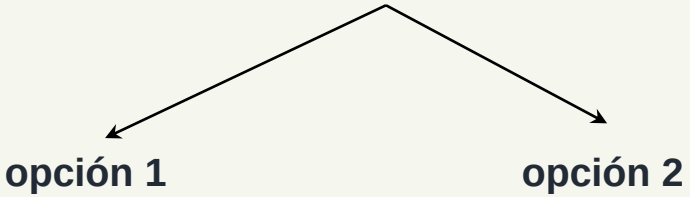
opción 2

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Limitaciones del modelo relacional (SQL92)
en la actualización de vistas:

INSERT INTO PRECIOS3 VALUES (pv12, pz23, 20)



¡Debido a la ambigüedad existente, el SQL92 no permite actualizar vistas definidas a partir de una concatenación de tablas!

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Los sistemas de gestión de bases de datos deductivas deben superar las limitaciones de los sistemas relacionales

PROBLEMAS:

- ✓ Formalización
- ✓ Actualización de la base de datos
- ✓ Construcción de SGBD deductivos

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Bases de Datos Deductivas

ESQUEMA

Relaciones básicas:

$R_i (A_{i_1}: D_{i_1}, A_{i_2}: D_{i_2} ..., A_{i_{n_i}}: D_{i_{n_i}})$

$(1 \leq i \leq m)$ (m relaciones básicas)

Relaciones derivadas:

$S_i (A_{i_1}: D_{i_1}, A_{i_2}: D_{i_2}, ..., A_{i_{n_i}}: D_{i_{n_i}})$

$(1 \leq i \leq s)$ (s relaciones derivadas)

BASE DE DATOS

$R_i \subseteq (D_{i_1} \times D_{i_2} \times ... \times D_{i_{n_i}})$

notación algebraica

$(1 \leq i \leq m)$

$S_{i_j} (x_1, x_2, ..., x_{n_i}) \leftarrow W_{i_j}$

$(1 \leq i \leq s)$
$(1 \leq j \leq K_i)$

notación lógica

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Formalización: Si intentamos representar la información explícita y la información implícita en un mismo lenguaje (lenguaje de 1er orden) obtenemos un programa lógico:

Base de datos deductiva

Hechos

pieza (pz1, tornillo, 10)
...

prov (pv1, Juan, 1)
...

Reglas deductivas

comp (pz1, pz3)
...

precios3 (x, y, z) $\leftarrow \exists w$ (prov (x, w, 3) $\wedge$ precios (x, y, z))

componente (x, y) $\leftarrow \exists z$ (comp (x, z) $\wedge$ componente (z, y))
componente (x, y) $\leftarrow$ comp (x, y)

precios_ext (x,n,y,d,p)$\leftarrow \exists z \exists w$ (prov (x, n, z) $\wedge$ pieza (y, d, w) $\wedge$ precios (x, y, p))

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Lenguaje de definición de reglas

Lenguaje de 1er orden

Base de datos deductiva ≡ Programa lógico

Sistema de gestión de bases de datos deductivas ≡ Sistema de programación lógica

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

MARCO FORMAL: Lógica de primer orden (Programación Lógica)

Esquema de BDD:

(L, RI): - L es un lenguaje de 1er orden
- RI es un conjunto de f.b.f de L (restricciones de integridad)

BDD: (programa lógico)

{A: A es un átomo base} (hechos)

$\cup$

$\{ A \leftarrow L_1 \wedge L_2 \wedge ... \wedge L_n : A$ es un átomo y L_i es un literal$\}$ (reglas)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

BASE DE DATOS

$R_i \subseteq (D_{i_1} \times D_{i_2} \times ... \times D_{i_{n_i}})$

$(1 \leq i \leq m)$

$S_{i_j}(x_1, x_2, ..., x_{n_i}) \leftarrow W_{i_j}$

$(1 \leq i \leq s)$
$(1 \leq j \leq K_i)$

BDD (programa lógico)

{A: A es un átomo base}

$\cup$

{ $A \leftarrow L_1 \wedge L_2 \wedge ... \wedge L_n$: A es un átomo y L_i es un literal}

En la formalización como un programa lógico, las reglas se definen como cláusulas (disyunción de literales). Esta simplificación no quita generalidad ya que el algoritmo de Lloyd [LT84] permite transformar una regla general, S← W, en un conjunto de cláusulas equivalentes en la semántica de la compleción.

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

BDD: (programa lógico)

$$\{A : A \text{ es un átomo base}\} \text{ (hechos)}$$

$$\cup$$

$$\{A \leftarrow L_1 \wedge L_2 \wedge \ldots \wedge L_n : A \text{ es un átomo y } L_i \text{ es un literal}\} \text{ (reglas)}$$

BDD definida: sin negación en el cuerpo de las reglas

BDD normal: con negación en el cuerpo de las reglas

BDD jerárquica: sin recursión

BDD estratificada: con recursión (no en términos de negación)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Esquema: (L, RI) :

Lenguaje L

- **Constantes** → Unión de los dominios de definición de las relaciones del esquema
- **Predicados** → Nombres de relación del esquema

RI

- Restricciones de Integridad (RI) → Fórmulas bien formadas (f.b.f) de L

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Esquema: (L, RI) :

Lenguaje L

Constantes: D1 ∪ D2 ∪ D3 ∪ D4 ∪ D5 ∪ D6 ∪ D7

= {pz1, pz3, …, tornillo, tuerca, …, pv1, pv5, ..., Juan,…}

Predicados: { PIEZA, PROVEEDOR, PRECIOS, PRECIOS_EXT, PRECIOS3, COMP, COMPONENTE}

Variables: { X, Y, Z, …}

Cuantificadores: { ∀ , ∃ }

Conectivas lógicas: { ∧, ∨ , → }

Símbolos de puntuación: { (,),', ... }

RI

Restricciones de Integridad:

$\forall x \forall y (componente(x,y) \rightarrow \neg componente(y,x))$

Fórmulas bien formadas de L

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

BDD:

Hechos

{ pieza (pz1, tornillo, 10), ...,
proveedor (pv1, Juan, 1), ...,
precios (pv1, pz3, 100), ...,
comp (pz1, pz3), ... }

Fórmulas bien formadas de L

∪

Reglas deductivas

precios3 (x, y,z) ← precios (x, y, z) ∧ prov (x, w, 3)

componente (x, y) ← comp (x,z) ∧ componente (z, y)
componente (x, y) ← comp (x, y)

precios_ext (x,n,y,d,p) ← prov (x, n, z) ∧ pieza (y, d, w) ∧ precios (x, y, p)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Esquema de la BDD ➡ Lenguaje de 1er orden L

Extensión de la BDD ➡ Programa lógico (D)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Semántica de una BDD

Programación lógica: semántica operacional: SLDNF
semántica declarativa: comp(D)

Semántica operacional: **procedimiento SLDNF**

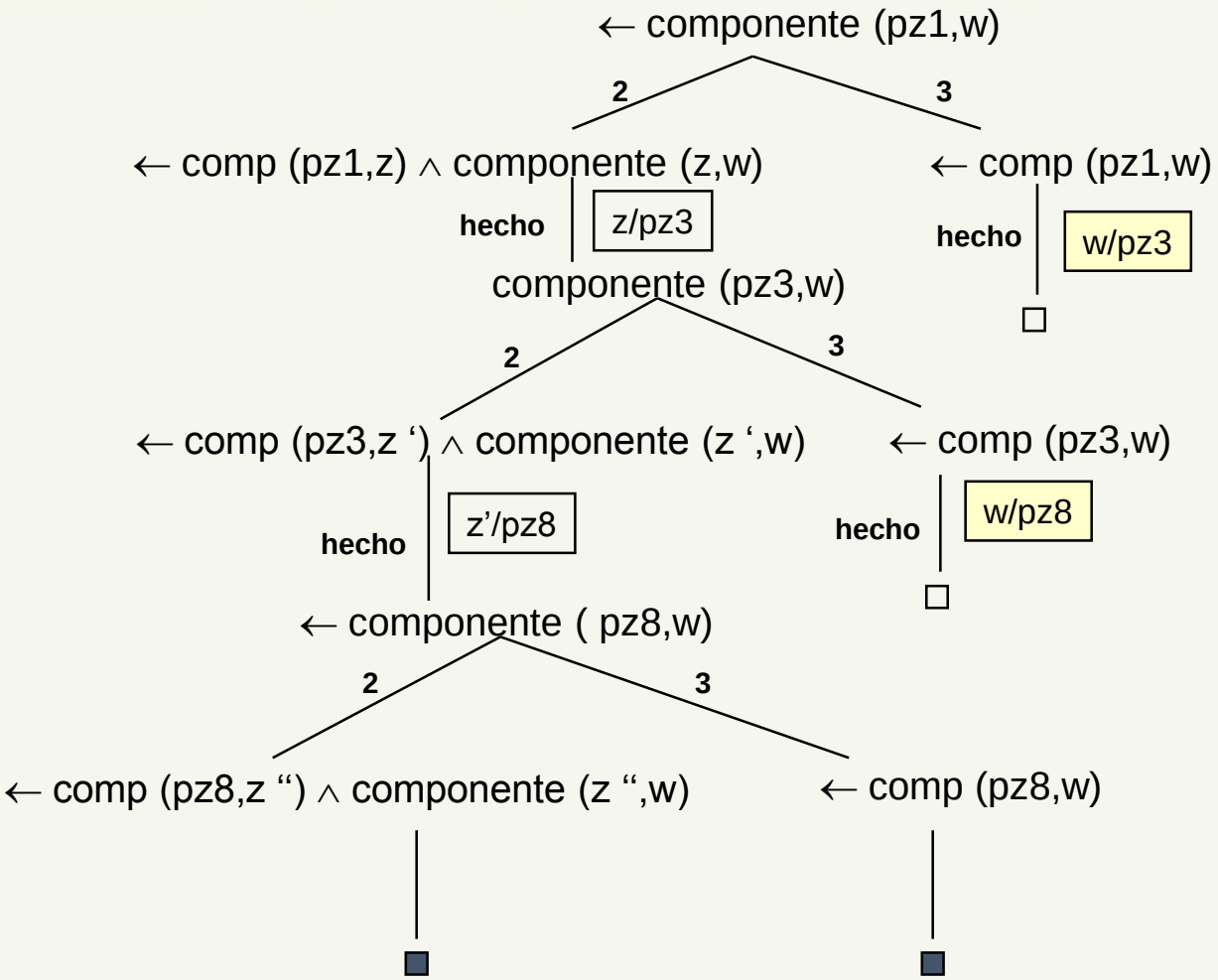
SLDNF: - procedimiento de refutación
- reglas de inferencia:
  - resolución
  - negación como fallo

Semántica declarativa asociada al SLDNF: **compleción de D**

Celma G. M. (s/f).

Procedimiento SLDNF

¿ De qué piezas se compone la pieza pz1?



2 componente (x, y) ← comp (x, z) ∧ componente (z, y)
3 componente (x, y) ← comp (x, y)

w = pz3
w = pz8

Celma G. M. (s/f).

Procedimiento SLDNF

¿ De qué piezas se compone la pieza pz1?

← componente (pz1,w)

resolución ------→ componente (x,y) ← comp (x,z) ∧ componente (z,y) **regla 2**

x/pz1 y/w

← comp (pz1,z) ∧ componente (z,w)

comp (pz1,pz3) **hecho**

resolución ------→ z/pz3

← componente (pz3,w)

componente (x,y) ← comp (x,y) **regla 3**

resolución ------→ x/pz3 y/w

← comp (pz3,w)

resolución ------→ comp (pz3,pz8) **hecho**

éxito w/pz8

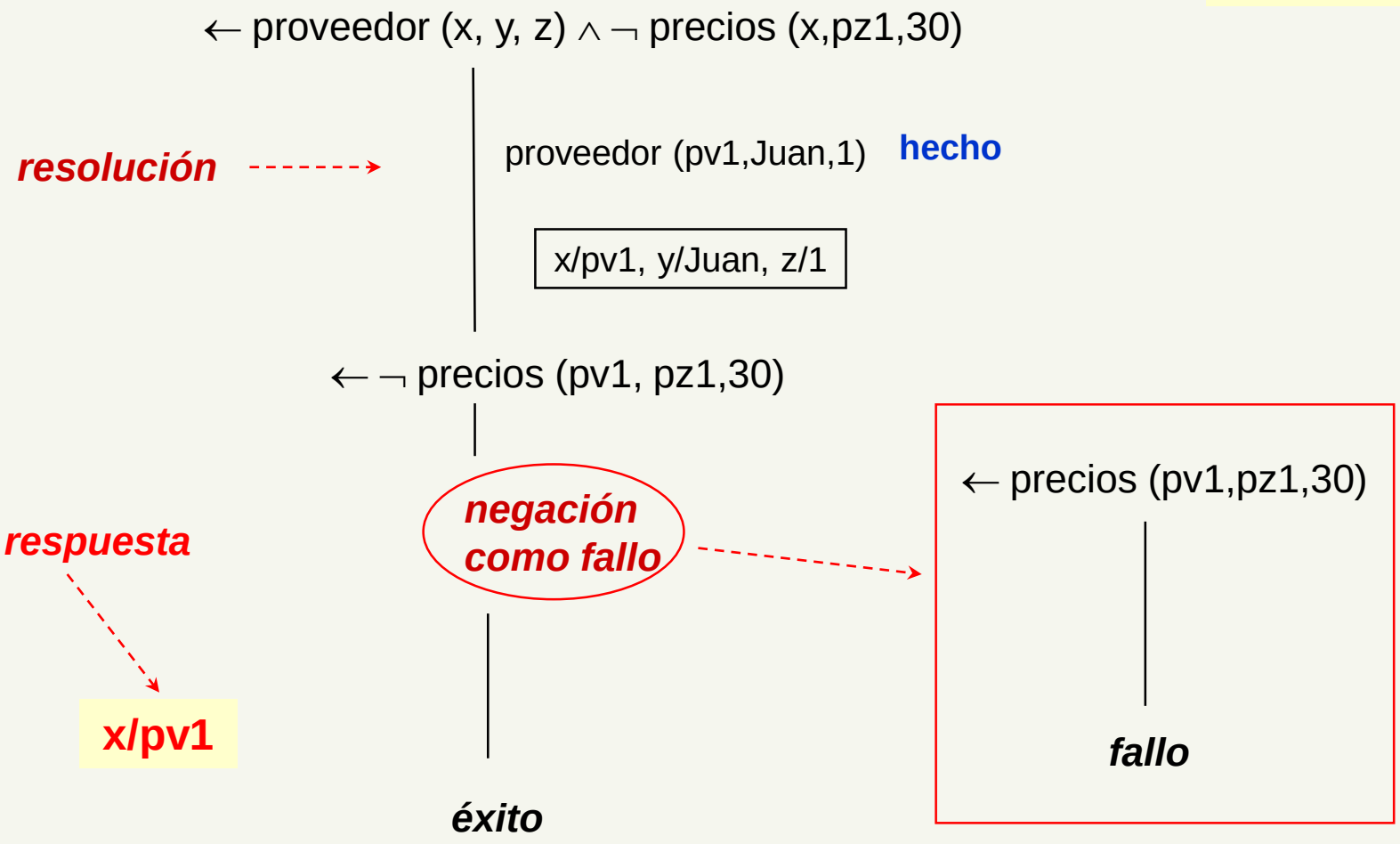
respuesta

w = pz8

Celma G. M. (s/f).

Procedimiento SLDNF

¿Qué proveedores no suministran la pieza pz1 a 30€?

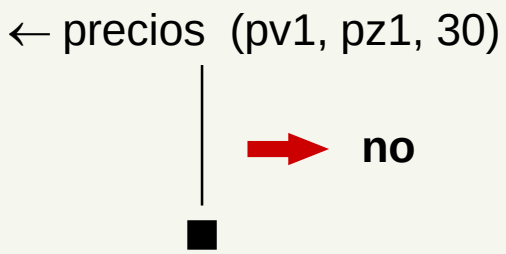


Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Semántica declarativa: teoría de la compleción de D (comp(D))

¿Suministra el proveedor pv1 la pieza pz1 a 30€?



$D \not\models$ precios (pv1, pz1, 30)

$D \not\models \neg$ precios (pv1, pz1, 30)

SLDNF no infiere consecuencias lógicas de D

SLDNF infiere consecuencias lógicas de comp(D)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

D:

{ precios (pv1, pz3,10),
precios (pv1, pz8,20),
precios (pv3, pz8, 30),
precios (pv5, pz1,50),
, ... }

$D \not\models$ precios (pv1, pz1, 30)

$D \not\models \neg$ precios (pv1, pz1, 30)

Comp(D):

{ precios (pv1, pz3,10),
precios (pv1, pz8, 20),
precios (pv3, pz8, 30),
precios (pv5, pz1, 50),
$\forall x \, \forall y \, \forall z$ (precios (x, y, z) $\rightarrow$
$(x=pv1 \wedge y=pz3 \wedge z=10) \vee (x= pv1 \wedge y= pz8 \wedge z= 20)$
$\vee (x= pv3 \wedge y= pz8 \wedge z= 30) \vee (x= pv5 \wedge y= pz1 \wedge z= 50)$), ... }

Comp(D): $\models \neg$ precios (pv1, pz1, 30)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

$\forall x\ \forall y\ \forall z$ (precios (x, y, z) $\rightarrow$
(x=pv1 $\wedge$ y=pz3 $\wedge$ z=10) $\vee$ (x= pv1 $\wedge$ y= pz8 $\wedge$ z= 20) $\vee$
(x= pv3 $\wedge$ y= pz8 $\wedge$ z= 30) $\vee$ (x= pv5 $\wedge$ y= pz1 $\wedge$ z= 50))

(precios (pv1, pz1,30) $\rightarrow$
(pv1=pv1 $\wedge$ pz1=pz3 $\wedge$ 30=10) $\vee$ (pv1=pv1 $\wedge$ pz1=pz8 $\wedge$ 30=20) $\vee$
(pv1=pv3 $\wedge$ pz1=pz1 $\wedge$ 30=30) $\vee$ (pv1=pv5 $\wedge$ pz1=pz1 $\wedge$ 30=50))

$\neg$ (pv1=pv1 $\wedge$ pz1=pz3 $\wedge$ 30=10)
$\neg$ (pv1=pv1 $\wedge$ pz1=pz1 $\wedge$ 30=20)
$\neg$ (pv1=pv3 $\wedge$ pz1=pz8 $\wedge$ 30=30)
$\neg$ (pv1= pv5 $\wedge$ pz1=pz1 $\wedge$ 30=50)

$\neg$ precios (pv1, pz1, 30)

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

comp (D) =

Axiomas de hechos

{pieza (pz1, tornillo, 10), ...,
proveedor (pv1, Juan, 1), ...,
precios (pv1, pz3, 100), ...,
comp (pz1, pz3), ... }

$\cup$

Axiomas de reglas

precios3 (x, y,z) $\leftarrow$ precios (x, y, z) $\wedge$ prov (x, w, 3)

componente (x, y) $\leftarrow$ comp (x,z) $\wedge$ componente (z, y)
componente (x, y) $\leftarrow$ comp (x, y)

precios_ext (x,n,y,d,p) $\leftarrow$ prov (x, n, z) $\wedge$ pieza (y, d, w) $\wedge$ precios (x, y, p)

$\cup$

Celma G. M. (s/f).

Axiomas de compleción

$\cup$

$\forall x\ \forall y\ \forall z$ pieza $(x, y, z) \rightarrow (x = pz1 \wedge y = tornillo \wedge z = 10)$

$\vee$

$(x = pz3 \wedge y = tuerca \wedge z = 11)$

$\vee$

$(x = pz8 \wedge y = arandela \wedge z = 8)$

...

$\forall x\ \forall y\ \forall z$ precios3 $(x, y, z) \rightarrow$ precios $(x,y,z) \wedge$ prov $(x,w,3)$

...

$\forall x\ \forall y$ componente $(x,y) \rightarrow$ comp(x,y)

$\vee$

$\exists z$ (comp$(x,z) \wedge$ componente (z,y))

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

Teoría de la compleción: comp(D)

$p(t_1,.., t_n) \leftarrow L_1 \wedge L_2 \wedge ... \wedge L_m \Rightarrow \; p(x_1,..., x_n) \leftarrow \exists y_1... \exists y_d \; (x_1=t_1 \wedge ... \wedge x_n=t_n \wedge L_1 \wedge L_2 \wedge ... \wedge L_m)$

Com(D) =

{ $p(x_1,..., x_n) \leftarrow E_1$

. . . $(E_i = \exists y_1... \exists y_d \; (x_1=t_1 \wedge ... \wedge x_n=t_n \wedge L_1 \wedge L_2 \wedge ... \wedge L_m))$ — Axiomas sobre p en D

$p(x_1,..., x_n) \leftarrow E_k$,

$\forall x_1, ..., x_n \; (p(x_1, ..., x_n) \rightarrow (E_1 \vee ... \vee E_k))$, — Axiomas de compleción de p

$\forall x_1, ..., x_n \neg p(x_1, ..., x_n)$: no existen sentencias de p en D

$\forall x \; =(x, x)$, — Axiomas de la igualdad

$\neg =(A, B), \neg =(A, C), \neg =(A, a), ..., \neg =(P100, P200)$

$\forall x \; (=(x, A) \vee =(x, B) \vee ... \vee =(x, P200))$ **}** — Axioma de cierre de dominio

Celma G. M. (s/f).

2. Bases de datos deductivas: definición y formalización

BDD	PL
hechos >>> reglas	hechos ~ reglas
relaciones de D: - básicas - derivadas	predicados de P
lenguaje libre de funciones	no
hechos y reglas son independientes del dominio	no
eficiencia ≡ tiempo de acceso a los datos	eficiencia ≡ tiempo del sistema de inferencia

Celma G. M. (s/f).

Bases de Datos Deductivas

1. Lógica y Bases de Datos: introducción.
2. Bases de datos deductivas: definición y formalización.
3. Actualización de vistas.
    - 3.1. Introducción al problema
    - 3.2. Estudio avanzado del problema
4. Comprobación de restricciones de integridad en esquemas con vistas
    - 4.1 Introducción al problema
    - 4.2 Estudio avanzado del problema
5. SGBD deductivos.
    - 5.1. Evaluación de consultas
    - 5.2. SQL3

Celma G. M. (s/f).

3. Actualización de vistas

3.1 Introducción al problema.

Actualización sobre relación derivada → Actualización(es) sobre relación(es) básicas(s)

"Dada una base de datos D (D=BDI ∪ BDE) y un requisito de actualización insertar (A) (resp. borrar (A)) donde A es una tupla de una relación derivada, encontrar una transacción T sobre EDB tal que T(D) satisfaga el requerimiento de actualización"

Ejemplo: DELETE FROM PRECIOS_EXT WHERE codprov=pv1

Celma G. M. (s/f).

PIEZA

codpieza	desc	peso
pz1	tornillo	10
pz3	tuerca	11
pz8	arandela	8

PRECIOS

codprov	codpieza	precio
pv1	pz3	10
pv1	pz8	20
pv3	pz8	30
pv5	pz1	50

PROV

codprov	nombre	zona
pv1	Juan.....	1
pv5	Carlos	3
pv3	Luis	3

PRECIOS_EXT

codprov	nombre	codpieza	desc	precio
pv1	Juan...	pz3	tuerca	10
pv1	Juan...	pz8	arandela	20
pv3	Luis...	pz8	arandela	30
pv5	Carlos.. .	pz1	tornillo	50

T1={borrar (PROV (pv1,Juan,1))}

T2={borrar (PIEZA (pz3,tuerca,11),
borrar (PIEZA (pz8,arandela,8))}

T3={borrar (PRECIOS (pv1,pz3,10),
borrar (PRECIOS (pv1,pz8,20))}

Celma G. M. (s/f).

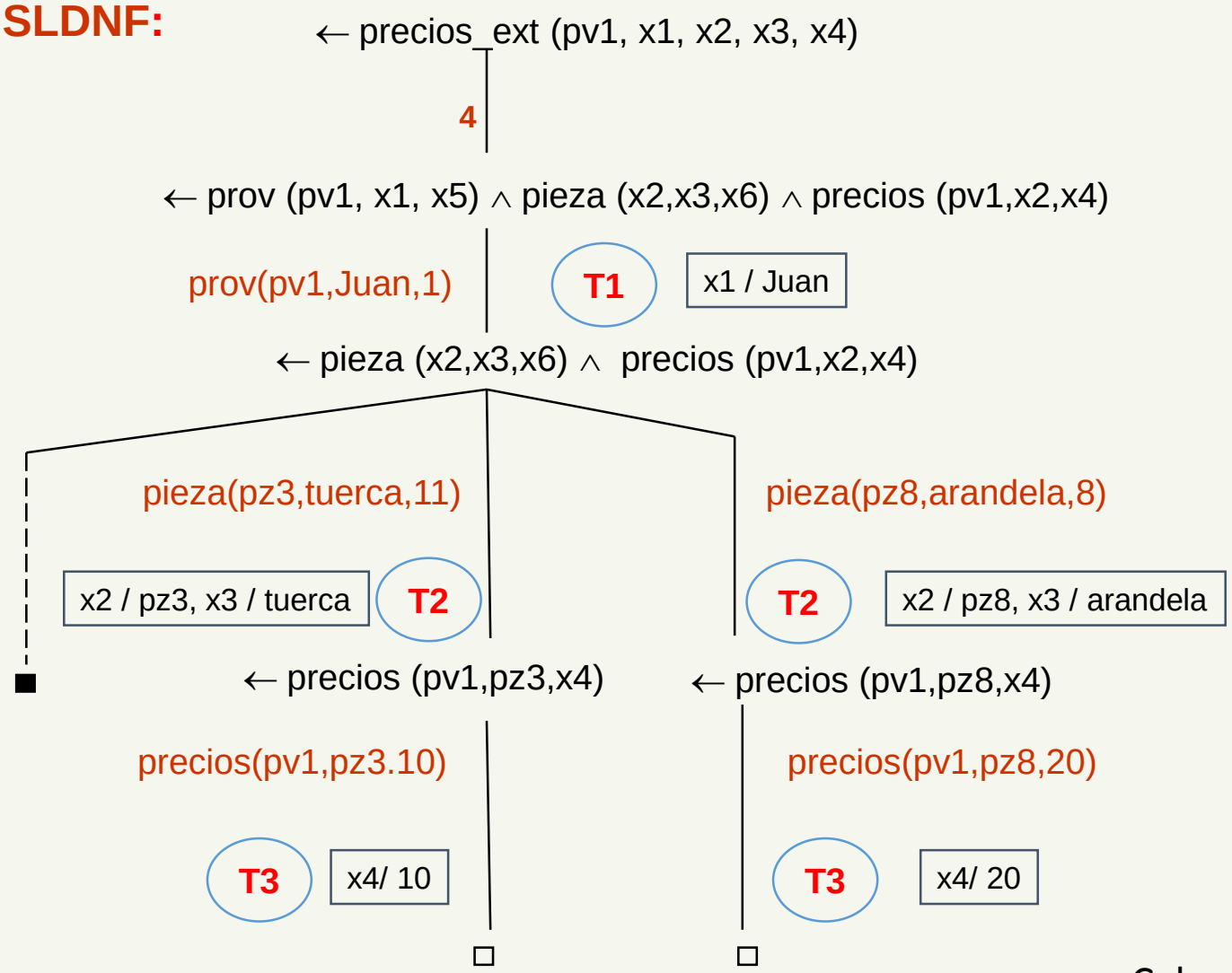
3. Actualización de vistas

Métodos para la actualización de bases de datos deductivas

Utilización de los procedimientos de evaluación de consultas para determinar los posibles caminos de derivación del conocimiento que se desea a actualizar

Celma G. M. (s/f).

SLDNF:



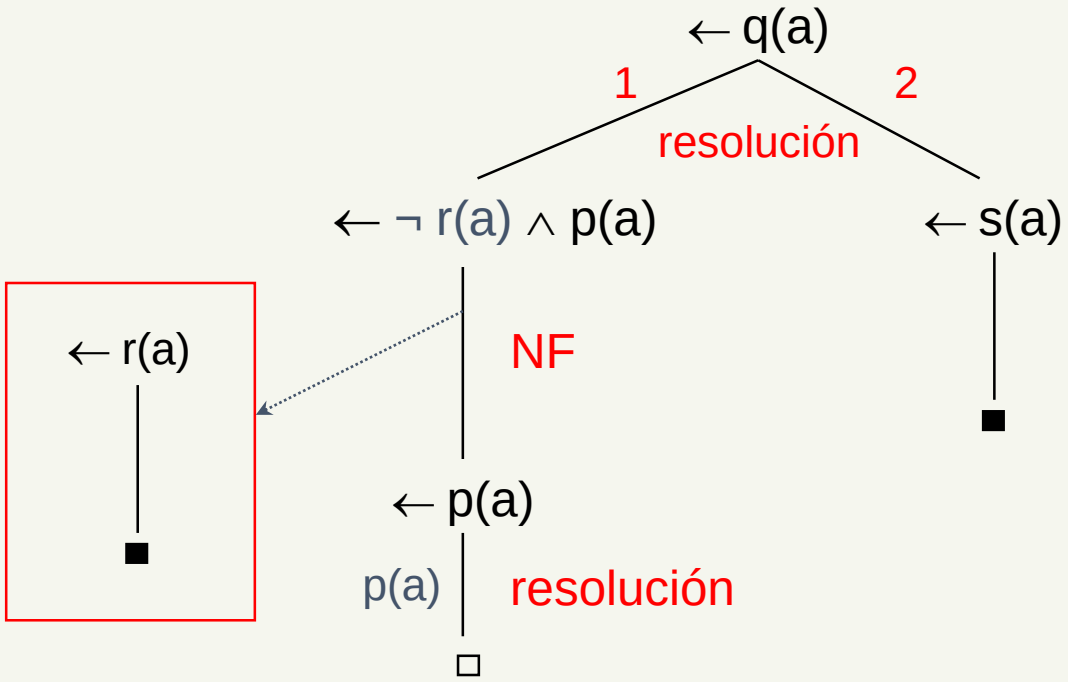
Celma G. M. (s/f).

3. Actualización de vistas

BDD:
1. $q(x) \leftarrow \neg r(x) \wedge p(x)$
2. $q(x) \leftarrow s(x)$

$p(a)$

Act: borrar(q(a))



T_1 = { insertar (r(a)) }

T_2 = { borrar (p(a)) }

Celma G. M. (s/f).

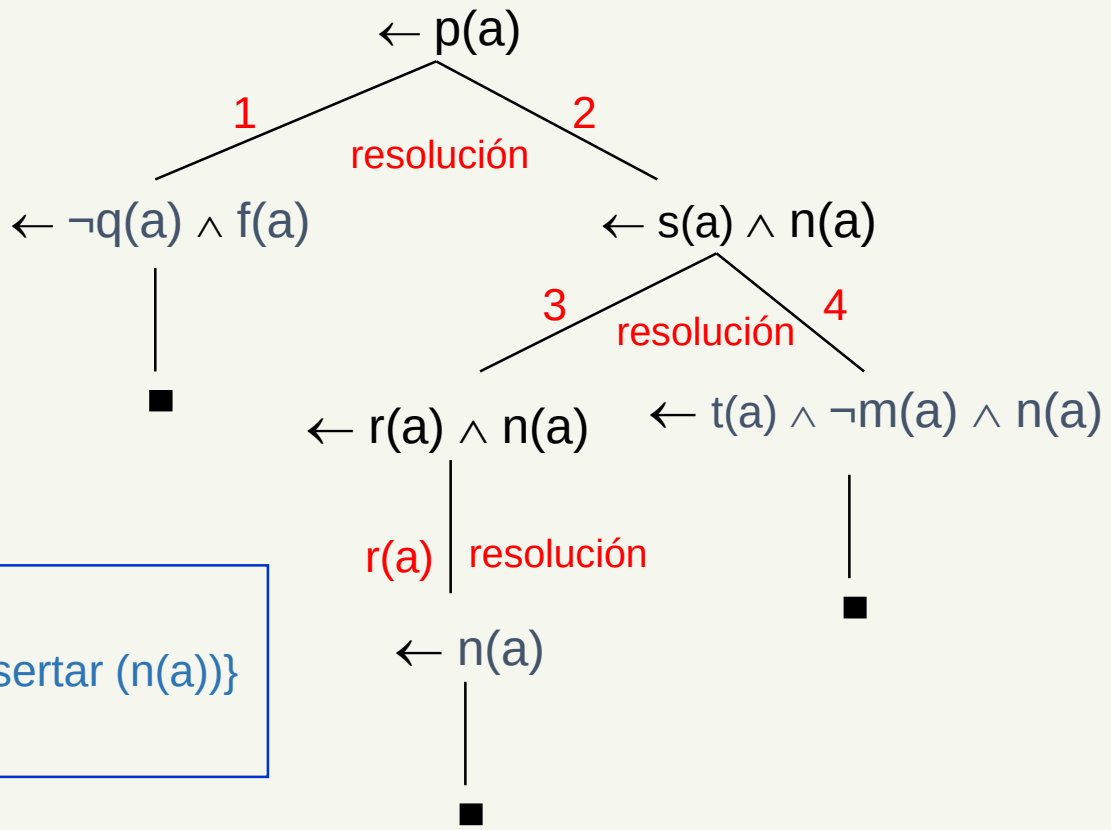
3. Actualización de vistas

Act: insertar(p(a))

BDD:

1. $p(x) \leftarrow \neg q(x) \wedge f(x)$
2. $p(x) \leftarrow s(x) \wedge n(x)$
3. $s(x) \leftarrow r(x)$
4. $s(x) \leftarrow t(x) \wedge \neg m(x)$

$r(a), q(a), m(a)$



T_1= {insertar (f(a)), borrar (q(a))}
T_2 = {insertar (t(a)), borrar (m(a)), insertar (n(a))}
T_3 = {insertar (n(a))}

Celma G. M. (s/f).

3. Actualización de vistas

Procedimientos de borrado e inserción de una tupla de una relación derivada:

Hipótesis:

- reglas deductivas sin recursión (BDD jerárquicas)
- procedimiento de evaluación: SLDNF
- regla de selección de literales en un paso de derivación: seleccionar primero los literales derivados positivos y sólo seleccionar un literal negativo cuando es base.

Características:

- sólo actualizan la base de datos explícita
- procedimientos recursivos (se llaman mutuamente)

Celma G. M. (s/f).

Procedimiento de borrado: Borrado (D, A, τ)

Entrada: una base de datos D y un requisito de borrado borrar(A) donde A es una tupla de una relación derivada

Salida: un conjunto de transacciones τ

Inicio

t:= un árbol SLDNF para $D \cup \{ \leftarrow A\}$

τ := { $[T_1, ..., T_n]$: (debe existir un T_i por cada rama de éxito del árbol t)

T_i = borrar(C) donde C es un hecho de D utilizado como cláusula de entrada en una rama de éxito de t

o

T_i = insertar(B) tal que B es básico y ¬B tiene éxito en una rama no fallada de t

o

$T_i \in \gamma$ tal que ¬B tiene éxito en una rama no fallada de t, B es derivado y γ es la salida de la llamada al procedimiento de inserción con argumentos de entrada D y B }

Fin

Celma G. M. (s/f).

Procedimiento de inserción: Inserción (D, A, τ)

Entrada: una base de datos D y un requisito de inserción insertar(A) donde A es una tupla de una relación derivada

Salida: un conjunto de transacciones τ

Inicio

τ := { [T_1, ..., T_n]: r:= una rama (derivación*) SLDNF fallada para D ∪ { ← A}

← $L_1 \wedge ... \wedge L_n$ es el objetivo que falla en r, L_i (i=1.. n) es base**

T_i = insertar (B) si L_i = B y B es básico (hecho) y B ∉ D

o

T_i = borrar(B) si L_i = ¬ B y B es básico (hecho) y B ∈ D

o

$T_i \in \gamma$ si L_i = ¬ B y B es derivado y γ es la salida de la llamada al procedimiento de borrado con argumentos de entrada D y B }

Fin

***** seleccionar primero los literales derivados positivos y sólo seleccionar un literal negativo cuando es base (los literales se pueden seleccionar en cualquier orden)

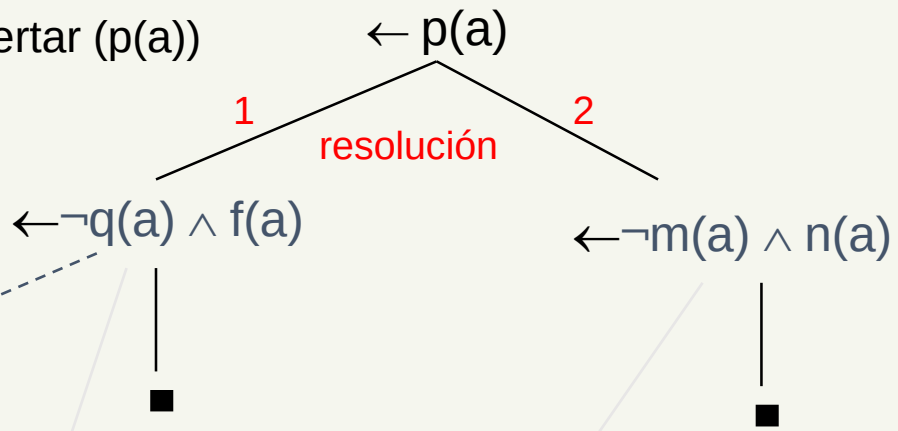
** se deben buscar las derivaciones que cumplan esta propiedad porque a partir de ellas se pueden encontrar transacciones

Celma G. M. (s/f).

BDD:
1. $p(x) \leftarrow \neg q(x) \wedge f(x)$
2. $p(x) \leftarrow \neg m(x) \wedge n(x)$
3. $q(x) \leftarrow r(x) \wedge \neg t(x)$

$r(a), m(a)$

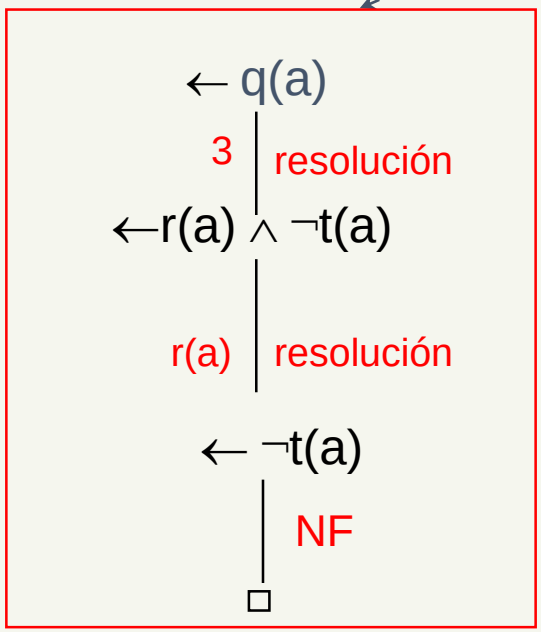
Act: insertar (p(a))



T_1={borrar(m(a)), insertar(n(a))}

T_1 = {borrar (r(a)), insertar (f(a))}

T_2 = {insertar (t(a)), insertar (f(a))}



borrar(q(a))

γ_1= {insertar(t(a))}

γ_2= {borrar(r(a))}

Celma G. M. (s/f).

3. Actualización de vistas

3.2 Estudio avanzado del problema.

Enunciado del problema:

Dados el esquema (L,RI) de una base de datos deductiva, un estado de base de datos D de ese esquema tal que $\forall W \in RI$ se cumple que D *satisface* W, y dado un requisito de actualización U tal que U no es *cierto* en D entonces encontrar una transacción T tal que $\forall W \in RI$, D' = T(D) *satisface* W y U es *cierto* en D'.

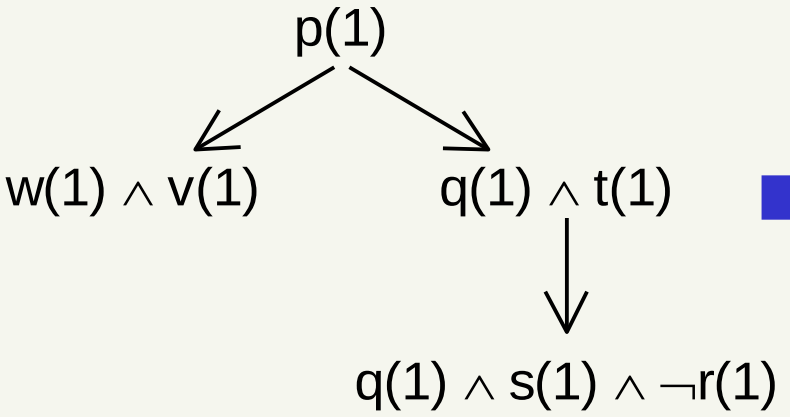
Celma G. M. (s/f).

3. Actualización de vistas

Ejemplo 1

1. $p(x) \leftarrow q(x) \wedge t(x)$
2. $p(x) \leftarrow w(x) \wedge v(x)$
3. $t(x) \leftarrow s(x) \wedge \neg r(x)$

Actualización: U = p(1)



1) $\{w(1), v(1)\} \subseteq BDE$
2) $\{q(1), s(1)\} \subseteq BDE$ y $\{r(1)\} \notin BDE$
3) $\{p(1)\} \subseteq BDE$
4) $\{q(1), t(1)\} \subseteq BDE$

Obtener transacciones que aseguren una de estas cuatro situaciones

Celma G. M. (s/f).

3. Actualización de vistas

Caracterización del problema:

1) Tiempo de generación de la solución.
2) Variables cuantificadas existencialmente
3) Recursividad
4) Información asumida
5) Tratamiento de restricciones de integridad

Celma G. M. (s/f).

3. Actualización de vistas

1) Tiempo de generación de la solución.

- Tiempo de ejecución: el árbol de derivación para el requisito de actualización se genera cuando la actualización es solicitada.
- Tiempo de definición: el árbol de derivación para un requisito de actualización se estudia cuando se define el esquema de la base de datos, lo que supone una mejora ya que determinadas tareas sólo se realizan una vez.
- Mixto: en este caso una parte de la solución se genera en tiempo de definición del esquema y se completa en tiempo de ejecución.

Celma G. M. (s/f).

3. Actualización de vistas

En el Ejemplo 1:

- un método que obtuviese la solución en tiempo de ejecución estudiaría el árbol de derivación de la actualización p(1) para encontrar una solución.
- un método que trabajase en tiempo de definición del esquema estudiaría el requisito genérico p(x) para obtener soluciones que luego se instanciarían en tiempo de ejecución.

Celma G. M. (s/f).

3. Actualización de vistas

2) Variables existencialmente cuantificadas.

Dada una regla deductiva de una base de datos normal, a las variables que aparecen en el cuerpo de la regla y no aparecen en la cabeza se les denomina variables existencialmente cuantificadas.

$$\forall x_1 \ldots \forall x_i \ldots \forall x_m \, (A \leftarrow L_1 \wedge \ldots \wedge L_n)$$
$$\equiv \quad (x_i \text{ no aparece en } A)$$
$$\forall x_1 \ldots \forall x_{i-1} \, \forall x_{i+1} \ldots \, \forall x_m \, (A \leftarrow \exists x_i \, (L_1 \wedge \ldots \wedge L_n))$$

La presencia de variables existencialmente cuantificadas en las reglas deductivas puede provocar la aparición del problema llamado falta de valores durante la generación de las transacciones que resuelven un requisito de actualización.

Celma G. M. (s/f).

3. Actualización de vistas

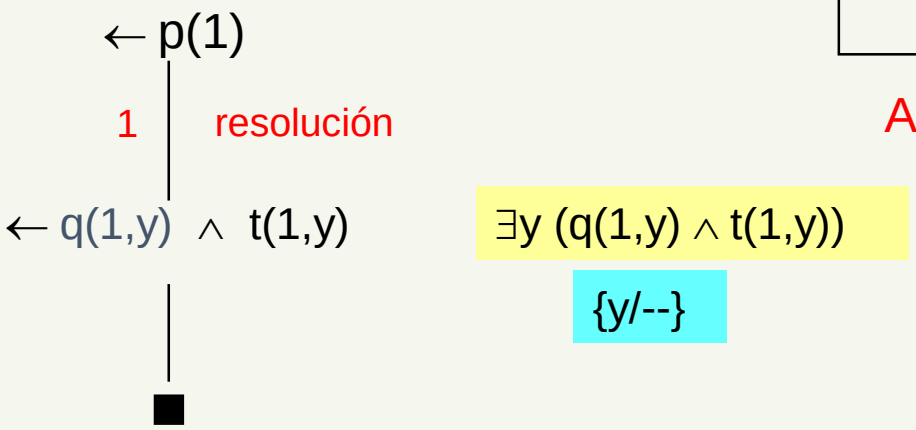
Ejemplo 2:

BDD: 1. $p(x) \leftarrow q(x,y) \wedge t(x,y)$

….

t(1,2)

Actualización: U = p(1).



Una solución sencilla a este problema consiste en utilizar el valor nulo para la variable de la que se desconoce el valor o bien un valor cualquiera proporcionado por el usuario o extraído sin criterio de la base de datos. Aunque en ocasiones esta solución es la única posible, en otras se puede elegir un valor tal que la transacción obtenida sea más sencilla.

Celma G. M. (s/f).

3. Actualización de vistas

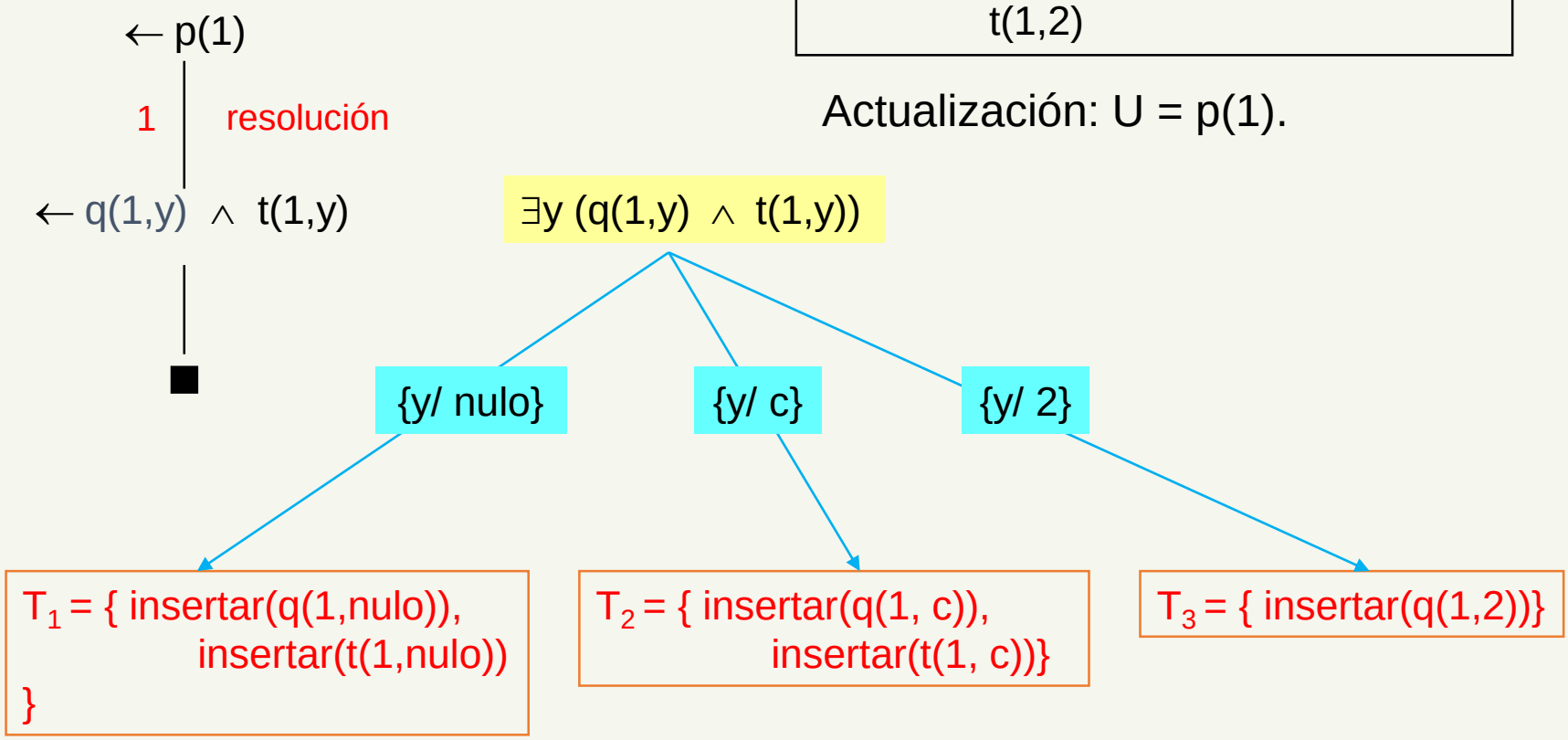
Ejemplo 2:

BDD: 1. $p(x) \leftarrow q(x,y) \wedge t(x,y)$

....

t(1,2)

Actualización: U = p(1).



Celma G. M. (s/f).

3. Actualización de vistas

3) Recursividad.

La presencia de reglas recursivas en la base de datos puede complicar la generación de la transacción, ya que el árbol de derivación puede ser infinito para un determinado requisito de actualización, lo que supone la existencia de infinitas transacciones posibles para satisfacerlo.

Celma G. M. (s/f).

3. Actualización de vistas

Ejemplo 3:

BDD: 1. $p(x,y) \leftarrow q(x,y)$

2. $p(x,y) \leftarrow q(x,z) \wedge p(z,y)$

Actualización: U = p(1,1).

Para satisfacer este requisito hay infinitas transacciones posibles:

T_1: {insertar(q(1,1))}

T_2: {insertar(q(1,2)), insertar(q(2,1))}

T_3: {insertar(q(1,2)), insertar(q(2,3)), insertar(q(3,1))}

…

Ventajas y Desventajas

Ventajas

Las principales ventajas al utilizar una Bases de Datos lógica son las siguientes:

- Tener la capacidad de expresar consultas por medio de reglas lógicas.
- Permitir consultas recursivas y algoritmos eficientes para su evaluación.
- Contar con negaciones estratificadas.
- Soportar objetos y conjuntos complejos.
- Contar con métodos de optimización que garanticen la traducción de especificaciones dentro de planes de acceso eficientes.

Como la característica fundamental de una BDD es la posibilidad de inferir información a partir de los datos almacenados, es imperativo modelar la BD como un conjunto de fórmulas lógicas, las cuales permiten inferir otras nuevas fórmulas.

Desventajas

La explotación de las reglas de deducción en una BDD plantea algunos problemas:

- Replantear correctamente, en un contexto deductivo, las convenciones habituales en una BD (representaciones de informaciones negativas, eficacia de las respuestas a las interrogaciones, cierre del dominio).

- Desarrollar procedimientos eficaces de deducción. La posibilidad de caer en bucles infinitos es un problema muy importante.

Posibles aplicaciones

A pesar de ser un tipo de BD que no se ha extendido mucho, se pueden encontrar algunas investigaciones en las que se plantean nuevas formas para aprovechar el potencial de las mismas. Por ejemplo:

- Algunos proyectos utilizan las características de este tipo de BD para la captación de datos y eliminación de "ruidos" (en dichos datos) aplicando reglas escritas por ingenieros conocedores tanto de las características de los sensores como de la interpretación de los datos.
- Modelado de empresas: este dominio implica modelar la estructura, los procesos y las restricciones dentro de una empresa. Los datos relacionados con ella utilizarán el modelo ER extendido que contiene cientos de entidades y vínculos y miles de atributos.
- Prueba de hipótesis o dragado de datos: este dominio implica formular una hipótesis, traducirla a un conjunto de reglas y una consulta; y luego ejecutar la consulta contra los datos para probar la hipótesis. El proceso se repite reformulando las reglas y la consulta. Esto se ha aplicado al análisis de datos de genoma en el campo de la microbiología. El dragado de datos consistió en identificar las secuencias de ADN a partir de auto radiografías digitalizadas de bajo nivel obtenidas de experimentos con bacterias E. coli.

E. coli es la abreviatura de Escherichia coli, un tipo de bacteria que vive en el intestino. La mayoría de las E. coli son inofensivas. Sin embargo, algunos tipos pueden producir enfermedades y causar diarrea.

Bibliografía

F. J. Muñoz y J. F. Pose, 2013. Comparación Conceptual entre Bases de Datos Deductivas y Bases de Datos Relacionales Difusas

MATA MUELA, T. MURILLOGOMEZ, J.M. Y TELLEZ DECEPEDA, J.L. 2009. Modelos avanzados de base de datos.

•

TEJADA CASTILLO, L.VEJARANO SANDOVAL, Y. YALVAREZ ALVAREZ G. 2008.Base de datos deductivas.

Aranda L. G. (2009) Un sistema de bases de datos deductiva. Universidad Complutense de Madrid , España

Celma G. M. (s/f). La Lógica en el desarrollo de las Bases de Datos, Departamento de sistemas informáticos y computación. Universidad politécnica de valencia