Ingeniería Reversa de Bases de Datos:
Propuesta de Herramienta Abierta Basada en Modelo Semántico

Marcelo Colman
Gustavo Larriera
Raúl Ruggia

siLAB - Laboratorio de Sistemas de Información
Facultad/Escuela de Informática
Noviembre 1997

Contexto

- **Area:**
  - Bases de Datos
  - Ingeniería Reversa
- **Problemática planteada:**
  - Obtener *especificaciones conceptuales* a partir de *BDs implementadas*, con los *objetivos* de:
    - re-documentar, migraciones, construir meta-bases

☞ Ingeniería Reversa de Bases de Datos (DBRE)

Temario

- **Introducción**
- **Breve estado del arte**
- **Nuestra propuesta de DBRE**
- **Conclusiones y perspectivas**

Introducción

- **La Ingeniería Reversa de BDs (DBRE):**
  - Genera *especificaciones conceptuales* de BDs :
    - Archivos, Jerárquicas, Codasyl, Relacionales
- **Problemas técnicos planteados:**
  - La *información en la estructura* de la BD suele ser *pobre*
  - La *información en las instancias* no siempre puede tomarse como totalmente válida
  - La *interpretación de consultas y programas* sigue siendo un *problema abierto*

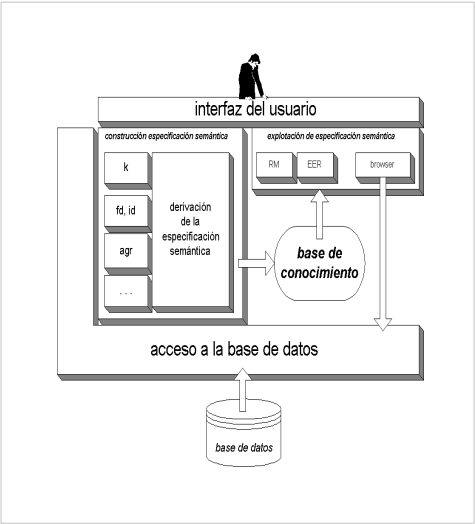
Breve estado del arte

- **Dos generaciones de propuestas:**
  - **Hasta principios de los ‘90s**
    - Algoritmos con ***hipótesis fuertes sobre la estructura*** de la BD
    - *Información de entrada:* estructura de la BD
  - **A partir de los ‘90s**
    - Algoritmos más complejos con ***pre-condiciones mínimas***: NFs bajas, nombramiento de atributos no relevante
    - *Información de entrada:* consultas SQL y datos
- **Visión general:**
  - Algoritmos a aplicar, de diferentes *familias*
  - Métodos *no* son *totalmente automatizables*

Nuestra propuesta en DBRE

- **Herramienta:**
  - Integra diferentes *familias* de algoritmos
- **Modelo semántico:**
  - Captura la semántica de la BD
  - Más *expresivo* que el Modelo Entidad-Relación
- **Nuevos algoritmos:**
  - Deducción de *jerarquías* a partir de GROUP-BY

Arquitectura de la herramienta



- **Módulos**
  - ***construcción*** de especificación semántica
  - ***explotación*** de especificación semántica
  - ***acceso*** a la BD
- **Base de conocimiento**
- **Interfaz al usuario**

Módulos (1)

- **Los *módulos de construcción* analizan la BD y construyen su especificación semántica**
- ***Tipos* de módulos de construcción**
  - *algoritmos de reingeniería*: detectores de claves, dependencias, y agrupamientos
  - *algoritmos de derivación*: construyen la base de conocimiento a partir de la información obtenida de los algoritmos de reingeniería

Módulos (2)

- **Los *módulos de explotación* de especificación semántica acceden a ella para:**
  - generar el esquema conceptual (Modelo Entidad-Relación como caso particular)
  - (re)generar un esquema relacional
  - browsing de los objetos de la BD

Módulos (y 3)

- **El *módulo de acceso* a la BD provee**
  - acceso a la BD desde los algoritmos de reingeniería
  - acceso a la BD desde los módulos de explotación
  - acceso a los datos correspondientes a objetos definidos en la especificación semántica

Base de conocimiento (1)

- **Información de la especificación semántica**
  - ***Objetos* de la base**
    - **objetos simples o átomos**: *p.e. atributos*
    - **objetos compuestos o moléculas**: *p.e. entidades, relationships*
  - ***Vínculos* entre objetos de la base**
    - **tipo de vínculo**: *p.e. generalización, asociación, etc.*
    - **origen** (cómo fue detectado el vínculo)
- ***Aserciones de trabajo*:** dependencias funcionales, dependencias de inclusión, condiciones de join, agrupamientos, cardinalidades, etc.

Base de conocimiento (2)

```
●      MOLECULA              ATOMO
       -----------------     --------------------
●      gen_of/spec_of        has_attr
● M    part_of/comp_of
●      assoc
       -----------------     --------------------
● A    attr_of               is_part_of/comprises
●                            super_cat/sub_cat
●
```

```
r(Tbl)
a(Tbl, Atr, Unique?, NotNull?)
q(Tbl1, Latr1, Tbl2, Latr2)
id(Tbl1, Latr1, Tbl2, Latr2, Confiab)
fd(Tbl, Latr1, Latr2, Confiab)
...
```

Base de conocimiento (y 3)

```
gen_of(Molec1, Molec2, Origen)
assoc(Molec1, Molec2, Card, Origen)
attr_of(Atom, Molec, Monoval?, Uniq?, Origen)
...
spec_of(M1,M2,Orgn) :- gen_of(M2, M1, Orig).
...
has_attr(M,A,Mv?,Uq?,Orgn) :- attr_of(A,M,Mv?,Uq?,Orgn).
```


Interfaz al usuario

- **Acceso al usuario a los diferentes módulos:**
  - módulos de reingeniería
  - módulos de explotación

Ejemplo (1)

PRS(<u>id</u> nom dom zip ciudad)
DPT(<u>dep</u> emp cargo proy)
ASG(<u>emp dep proy</u> fecha nomproy)
HST(<u>nro fecha</u> sueldo)

- *Equi_joins*:
  - HST[nro]⊗PRS[id]
  - DPT[emp]⊗HST[nro]
  - ASG[dep]⊗DPT[dep]
  - DPT[proy]⊗ASG[proy]

```
SELECT Nro, Nom, Fecha, Sueldo
FROM Hst, Prs
WHERE Hst.Nro = Prs.Id
```


Ejemplo (2)

- **Los *detectores de dependencias* [p.e. Petit et al 96] detectan:**

***Dependencias de inclusión (IDs)*:**

- HST[nro]<<PRS[id]
- DPT[emp]<<HST[nro]
- ASG[emp]<<HST[nro]
- DPT[proy]<<ASG[proy]

***Dependencias funcionales (FDs)*:**

- DPT: emp → cargo proy
- ASG: proy → nomproy
- ...

Ejemplo (3)

- ***Aserciones de trabajo*** **cargadas por los algoritmos de reingeniería (frag.):**

```
/* r(tbl) */

r(prs).
r(dpt).
r(asg).
r(hst).
...
```

```
/* a(tbl, atr, uniq?, nnull?) */

a(prs, id, true, true).
a(prs, nom, false, true).
a(prs, dom, false, true).
a(prs, zip, false, true).
a(prs, ciudad, false, true).
...
```


Ejemplo (4)

```
/* q(tbl1, latr1, tbl2, latr2) */
q(hst, [nro], prs, [id]).
q(dpt, [emp], hst, [nro]).
...
```

```
/* id(tbl1, latr1, tbl2, latr2, confiab) */
id(hst, [nro], prs, [id], 80).
id(dpt, [emp], hst, [nro], 80).
...
```

```
/* fd(tbl, latr1, latr2, confiab) */
fd(dpt, [nro], [cargo proy], 80).
fd(asg, [proy], [nomproy], 80).
...
```

Ejemplo (5)

- **La *especificación semántica*, cargada por los algoritmos de derivación (frag.):**

```
molec(prs).
molec(dpt).
...
atom(id, type_numeric).
atom(nom, type_string).
...
attr_of(id, prs, monoval, uniq, origDDL, ...).
...
assoc(dpt, prs, 1-N, derivAlg, ...).
...
```


Ejemplo (y 6)

- **Reglas de generación de esquema E-R (frag.):**

```
entity_type(O) :-              molec(O),
                               not assoc(O,_,_,_).
relationship_type(O) :-        molec(O),
                               assoc(O,_,_,_).
link_Ent_Rel(R,E,Card) :-      molec(R),
                               molec(E),
                               assoc(R,E,Card,_).
attributes(O,A,D,C,U) :-       molec(O),
                               atom(A,D),
                               has_attr(O,A,C,U, _).
specializat(E1, E2) :-         entity_type(E1),
                               entity_type(E2),
                               spec_of(E1,E2, _).

...
```

Agrupamientos

- **Los algoritmos detectores de agrupamientos:**
  - Analizan cláusulas *GROUP BY*
  - Detectan información útil para generar *jerarquías en dimensiones* de esquemas multidimensionales
  - *No han sido considerados en otras propuestas*, posiblemente debido a que los modelos EER no pueden expresar dichas jerarquías

Conclusiones

- **Panorama general:**
  - Area activa dentro de las Bases de Datos
  - Múltiples aplicaciones
  - Todavia existen problemas abiertos
  - Pocas implementaciones
- **Trabajo realizado:**
  - Implementación en curso
  - Modelo semántico de alta expresividad
  - Integración de algoritmos

Perspectivas

- **Temas en investigación:**
  - Tratamiento de esquemas con optimizaciones físicas
  - Análisis de más tipos de expresiones SQL
- **Nuestro trabajo inmediato:**
  - Finalización del prototipo
  - Browser para “navegar” en la BD
  - Conección con otros proyectos:
    - Ambientes CASE (InCo - Facultad de Ingeniería).

siLAB - Laboratorio de Sistemas de Información
Facultad/Escuela de Informática
e-mail: *silab@ei.edu.uy*
Internet: *www.silab.ei.edu.uy*