Tema III: El enfoque relacional

Departamento de
Informática e Ingeniería
de Sistemas
Universidad Zaragoza
1542

5 MODELO RELACIONAL

5.1 Conceptos básicos: estática del modelo Relacional

M.R. propuesto por Codd (1970) ⟶ *Objetivo:* **independencia de la Estructura Física**

→ de la ordenación
de la indización
de los caminos de accesos

características (ventajas)

- Independencia física
- Independencia lógica
- Uniformidad (tablas)
- Sencillez → *accesibilidad* y *facilidad de uso*
- Flexibilidad
- soporte formal
  - ↑ integridad
  - optimización (consultas y representación)

Tª Normalización
- ☛ ↓ *redundancias*
- ☛ *ayuda diseño*

dinámica →
- ✓ álgebra relacional
- ✓ cálculo relacional (de tuplas y de dominios)

evolución histórica del modelo Relacional.

inconvenientes *(pasado reciente)*
- implementaciones poco eficientes (lentas y voluminosas)
- no soporta muchos de los conceptos

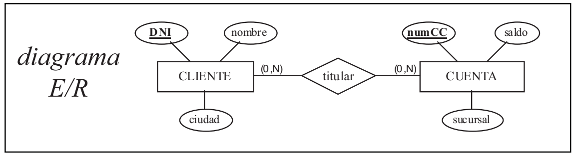
muy superados

evolución

≈ 1970	surge el modelo	prerrelacional
1973-1978	prototipos (Ingres, Sistema R, ..)	
1978	QBE	
1979	ORACLE	relacional
1980	INGRES	
1981	**SQL**	
1982	DB2	
1986	SQL/ANSI	
. . . .		
1992	SQL2	postrelacional
. . . .		
≈ 2000	SQL3, MROO, ...	

aproximación intuitiva al modelo relacional.

diagrama E/R



Esquema relacional (simplificado) de la Base de Datos bancaria

Cuenta

numCC	*Sucursal*	*Saldo*
123	23	2567
194	23	125874
237	14	654875
257	18	25984
100	22	0

Cliente

DNI	*Nombre*	*Ciudad*
17567984	Severino Martínez	Huesca
19465278	Salustiano Pérez	Teruel
25468724	Indalecio López	Zaragoza
18234587	Pedro García	Calatayud
17235465	Antonio Sánchez	Zaragoza

Titular

DNI	*numCC*
17567984	123
18234587	123
25468724	257
25468724	194
17235465	237

introducción al concepto de relación

DOMINIOS

tpDNI: 0..99999999

tpNombre: XXXX…XXX ↔ 40

tpCiudad: Calatayud, Huesca, Teruel, Zaragoza

nombre → Cliente

DNI	*Nombre*	*Ciudad*
17567984	Severino Martínez	Huesca
19465278	Salustiano Pérez	Teruel
25468724	Indalecio López	Zaragoza
18234587	Pedro García	Calatayud
17235465	Antonio Sánchez	Zaragoza

→ atributos

tuplas

cardinalidad = 5

grado = 3

modelo relacional: Dominios y Atributos

Dominio ≡ conjunto <u>finito</u> de valores **homogéneos** y **atómicos,** con un <u>nombre</u> asociado

→ ≡ indivisibles (sin perder semántica)

→ igual ***tipo*** (y representación)

definición
- ✓ intensión { ej. edad = 0..150 }
- ✓ extensión { ej. sexo = (hombre, mujer) }

Dominio compuesto ≡ combinación de dominios simples (+ restricciones integridad)

Atributo ≡ papel de un dominio en una relación — *posibilidad de valores nulos*

☛ el Universo Discurso de una base de datos relacional está formado por un conjunto *finito* y *no vacío* de *atributos* estructurados en *relaciones*

relaciones y esquemas de relación

Relación ≈ subconjunto del *producto cartesiano* de los *dominios* sobre los que se define

→ suele tener un *nombre* asociado

cabecera de relación ≡ conjunto de ***n*** pares *atributo* - *dominio subyacente* $\{(A_i : D_i), \forall i\ 1..\ n\}$

→ **grado**

cuerpo de relación ≡ conjunto de ***m*** tuplas $\{t_1, t_2, ..., t_m\}$

cardinalidad ←

→ conjunto pares *atributo* - *valor* $\{(A_i : V_{ij}), \forall i\ 1..\ n\}$

intensión o <u>esquema de relación</u> R ≡ $R(\{A_i : D_i\}, \forall i\ 1..\ n)$

extensión o <u>estado de relación</u> r(R) ≡ <esquema, cuerpo> → **ocurrencia**

Base Datos Relacional ≡ conjunto de ***variables de relación*** + . . .

clasificación de las relaciones (ISO 92)

clases de relación:

- con nombre
  - Persistentes
    - Base → nivel conceptual
    - Vistas * → nivel externo — **no almacenadas** (virtuales)
    - Instantáneas * → nivel interno — no cambian con los datos se "refrescan" por el sistema
  - Temporales
    - Autónomas
    - Vistas temporales *
    - Instantáneas temporales *
- sin nombre
  - Resultado final de una consulta *
  - Resultados intermedios de una consulta *

* *significa derivada*

concepto de clave en el modelo relacional.

claves *de una relación:*

- ✓ **superclave** ≡ cualquier conjunto de atributos que identifica cada tupla de la relación
- ✓ **clave candidata** ≡ cualquier superclave mínima
- ✓ **clave primaria** ≡ clave candidata elegida por el diseñador. El resto → ***alternativas***
- ✓ **clave ajena** ≡ conjunto de atributos cuyos valores deben coincidir (o ser nulos) con los de la clave primaria de la relación especificada
  - → en SQL92 puede ser clave candidata

→ muy importantes para estudio de la integridad

restricciones en el modelo relacional (1)

Restricciones:

✓ **inherentes**

- no hay dos tuplas iguales
- el orden de las tuplas es irrelevante
- el orden de los atributos no es significativo
- cada atributo (en cada tupla) sólo puede tener un valor del dominio → **1ª Forma Normal**

+ **integridad de entidad** ≡ ningún atributo que forme parte de la clave primaria puede tomar el valor nulo

el modelo no lo exige para claves alternativas

✓ **de usuario (semánticas) básicas**

- especificación de clave primaria, (PRIMARY KEY)
- unicidad (UNIQUE)
- obligatoriedad (NOT NULL) → **pb. Valores NULOS**
- ***integridad referencial*** ⇒ los valores de los atributos de una clave ajena (FOREIGN KEY), deben existir, o ser nulos

restricciones en el modelo relacional (2)

especificación de clave ajena ⇒ indicar acción en operaciones de borrado y modificación

- operación restringida (RESTRICT)
- operación transmisión en cascada (CASCADE)
- operación con puesta a nulos (SET NULL)
- operación con puesta a valor por defecto (SET DEFAULT)

- especificación de restricciones basadas en predicados
  - Intrarrelación (sobre atributos y tuplas)
  - Interrelación
  - de Dominio

  (CHECK y ASSERTION)

  → asociadas a la estática o a la dinámica (TRIGGERS) con posible activación de un procedimiento asociado

  → indicación del momento en que se verifica dentro de la transacción

☛ ***están en el DICCIONARIO, como el resto de las relaciones***

concepto de Base de Datos Relacional y notación

esquema de relación $\equiv$ R ({A_i:D_i}, S)
- S → lista de restricciones (intrarrelación)
- {A_i:D_i} → lista de pares atributo-dominio

esquema de B.D. relacional $\equiv$ < {R_i}, {V_i} >
- {V_i} → lista de restricciones (interrelación)
- {R_i} → lista de esquemas de relación

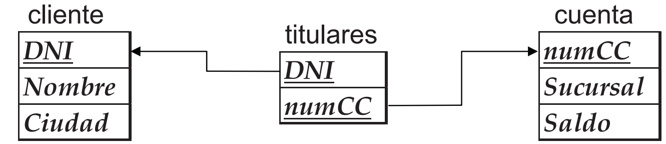
Base de Datos relacional $\equiv$ ***esquema relacional*** + ***ocurrencia válida del esquema***

notación:
- gráfica $\equiv$ diagrama relacional → hay que completarlo con notación textual
- <u>textual</u> (en lenguaje natural, o utilizando un estándar como SQL92)

ejemplo de esquema relacional simple

<u>ejemplo de B.D. Bancaria:</u>

diagrama relacional



Dominios:

tpDNI = 0..99999999;
tpNombre = cadena(50);

esquemas de relación:

cliente = (<u>DNI</u> : tpDNI; Nombre : tpNombre, *UNICO*; Ciudad : cadena(20));
cuenta = (numCC : entero, *clave primaria*; Sucursal : cadena(20) *NO NULO*; saldo : real);
verificar que saldo > 0;
titulares = (<u>DNI</u> : tpDNI; <u>numCC</u> : entero);
clave primaria (DNI, numCC);
clave ajena (DNI), *referencia a* cliente; *borrado* en cascada
clave ajena (numCC), *referencia a* cuenta; *borrado* en cascada

+ restricciones

modelo relacional y arquitectura ANSI (1)

el modelo relacional es un modelo lógico ⇒ <u>no contempla el nivel interno</u> de la B.D.
→ parcialmente en SQL (CREATE INDEX, …)
y en implementaciones concretas

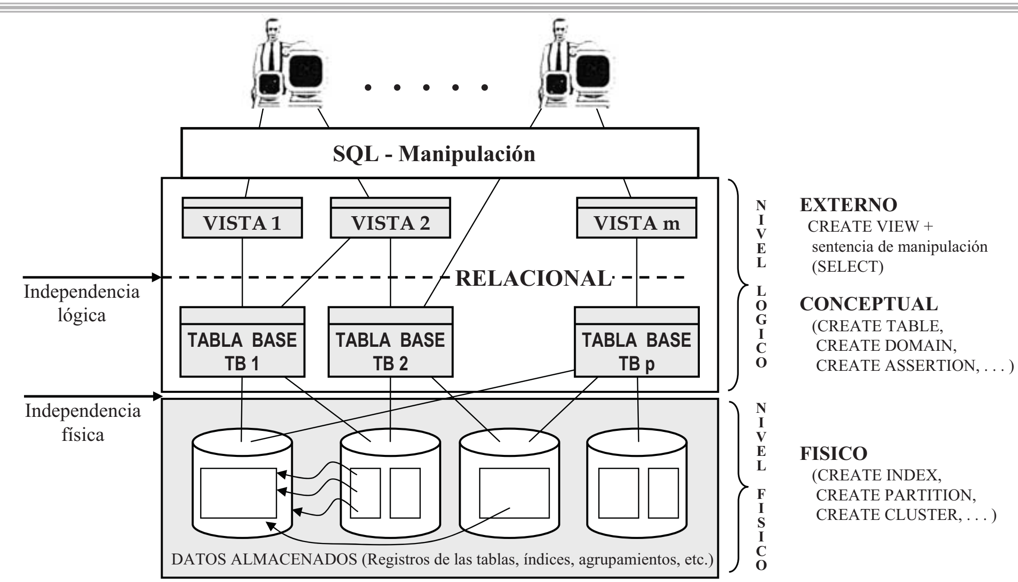
Relaciones base → ***nivel conceptual*** de la B.D. (esquema lógico global)

Vistas *(≡ tablas virtuales)* ≅ ***nivel externo*** de la B.D.

diferencias
- el usuario tiene acceso a más información (autorizaciones, tablas, etc.)
- no todas las vistas son actualizables

Las tablas temporales no tienen correspondencia directa en la arquitectura ANSI

modelo relacional y arquitectura ANSI (2)



5.2 Transformación del modelo E/R en modelo Relacional

las reglas de transformación son bastante ***simples y fáciles de deducir***

✓ ***dominios ER*** → dominios relacionales

✓ ***atributos E/R*** → atributos de relaciones

- AIP → CLAVE PRIMARIA
- AIA → UNICO (o clave alternativa)
- obligatoriedad → NO NULO (por defecto opcionales)

✓ ***tipos de entidad*** → relaciones (tablas)

✓ ***tipos de interrelación***

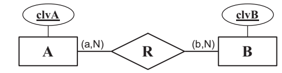
caso general (**N : M**)
- relación con clave primaria ≡ concatenación de las C.P. de los tipos de entidad que participan
- \+ especificación de claves ajenas

casos (**1 : N**) y (**1 : 1**)
- propagación de clave + especificación de clave ajena
- o como el caso general

al tipo correspondiente a la N, o a cualquiera de ellos si es 1:1

transformación de tipo de correspondencia N:M

las cardinalidades mínima y máxima ⇒ ***casos particulares***



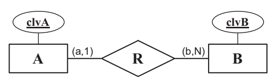
1. R(A(0,N), B(0,N)) → A(clvA, ...); B(clvB, …); R(clvA, clvB)
   clave ajena de B
   clave ajena de A

2. R(A(0,N), B(1,N)) → A(clvA, ...); B(clvB, …); R(clvA, clvB)
   clave ajena de B
   clave ajena de A
   ∀ ocurrencia de **clvA** en A, ∃ en R
   ò ∄(Π_{clvA}(A) - Π_{clvA}(R)), ò Π_{clvA}(R) = Π_{clvA}(A)
   ò clvA **referencia a** R(clvA)

3. R(A(1,N), B(0,N)) → similar al anterior

4. R(A(1,N), B(1,N)) → A(clvA, ...); B(clvB, …); R(clvA, clvB)
   clave ajena de B
   clave ajena de A
   clvB **referencia a** R(clvB)
   clvA **referencia a** R(clvA)

transformación de tipo de correspondencia 1:N



1. R(A(0,1), B(0,N)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A

2. R(A(1,1), B(0,N)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A, NO NULO

3. R(A(0,1), B(1,N)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A
   clvA **referencia a** B(clvA)

4. R(A(1,1), B(1,N)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A, NO NULO
   clvA **referencia a** B(clvA)

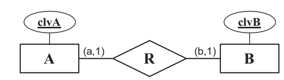
con 3 tablas:

sólo clave primaria

1b. R(A(0,1), B(0,N)) → A(clvA, ...); B(clvB, …); R(clvA, clvB)
   clave ajena de B
   clave ajena de A, NO NULO

el resto de casos, mezcla del caso general y el anterior

transformación de tipo de correspondencia 1:1



si ambas cardinalidades máximas son = 1
⇒ se puede propagar la clave a cualquiera de las relaciones
hacia la tabla más pequeña

1. R(A(0,1), B(0,1)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A, UNICO

2. R(A(1,1), B(0,1)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A, UNICO, NO NULO

3. R(A(0,1), B(1,1)) → A(clvA, ...); B(clvB , clvA, …);
   clave ajena de A, UNICO
   clvA **referencia a** B(clvA)

4. R(A(1,1), B(1,1)) → mejor fusionar las tablas

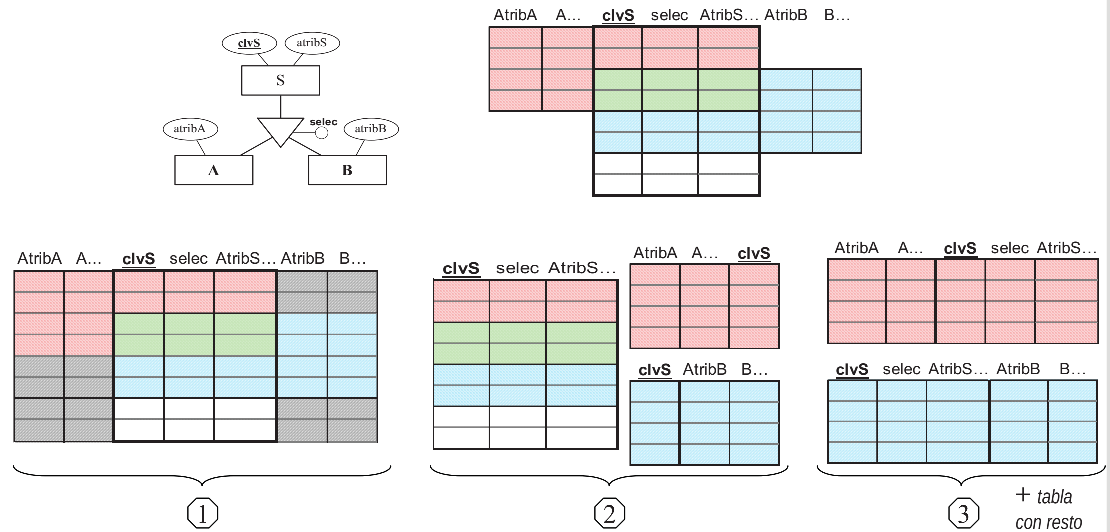
☛ ***no interesa la propagación de clave si***
- hay muy pocas ocurrencias de la interrelación → *daría lugar a muchos "nulos"*
- en el futuro será una interrelación N:M
- hay atributos en la interrelación

otras transformaciones de atributos e interrelaciones

- ✓ los ***atributos de los tipos de interrelación*** se añaden a la traducción de la interrelación
- ✓ ***tipos de interrelación ternaria*** → similar, pero puede haber casos particulares (p.e. 1:N:N)
- ✓ ***tipos de entidad débiles*** → como los fuertes, pero añadiendo a su clave primaria la clave primaria de los tipos de entidad de los que depende + especificación de clave ajena
- ✓ ***tipos de interrelación en exclusión*** → restricciones adicionales (predicados) *+ posibilidad de "trucos" de implementación*
- ✓ ***atributos multivaluados*** → nueva relación cuya clave primaria está formada por la clave primaria y el atributo multivaluado + especificación de clave ajena ⟶ ***casos particulares***

transformación de la generalización / especialización (1)

- ✓ la ***generalización/especialización*** plantea varias posibilidades



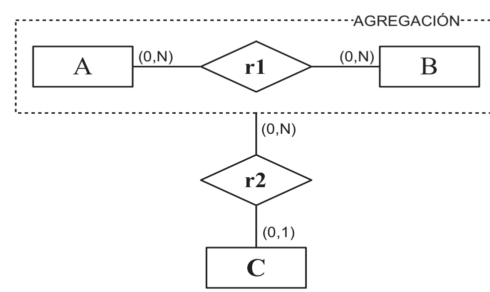
transformación de la generalización / especialización (2)

1. *una única tabla con toda la información* (supertipo y subtipos)
   S(<u>clvS</u>, atribS, selec…, atribA, …, atribB, ...); → ***Acceso más eficiente***
2. *una tabla para cada uno de los tipos de entidad* (supertipo y subtipos)
   S(<u>clvS</u>, atribS, selec, ...);
   A(<u>clvS</u>, atribA, ...); → ***Mejor la semántica***
   B(<u>clvS</u>, atribB, ...);
3. *una tabla para cada uno de los subtipos* (incluyen la del supertipo)
   A(<u>clvS</u>, atribS, selec, ..., atribA, ...); → ***Acceso eficiente peor la semántica***
   B(<u>clvS</u>, atribS, selec, ..., atribB, ...);
   S(<u>clvS</u>, atribS, selec, ...); — *sólo para los que no están en ningún subtipo*

habrá que añadir ***restricciones adicionales*** para implementar la ***semántica*** y los ***casos particulares***

transformación de la agregación

- ✓ la ***agregación*** queda representada por la interrelación

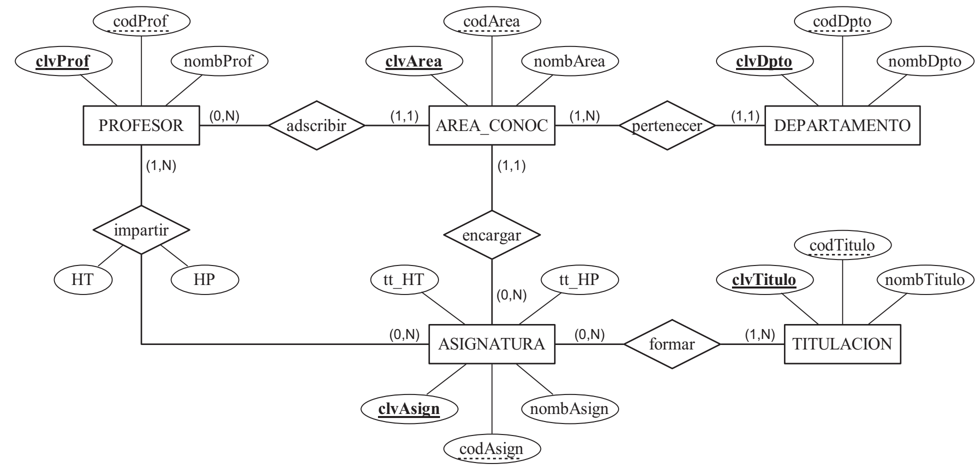


puede haber muchos casos particulares, pero en la práctica sólo algunos y simples

Ejemplos: . . .

transformación de la B.D. de la Universidad (1)

diagrama E/R:



transformación de la B.D. de la Universidad (2)

Atributos de tipos de entidad:

clvProf: tpClave; **AIP**
codProf: tpCódigo; **AIA**
nombProf: tpNombre;

clvArea: tpClave; **AIP**
codArea: tpCódigo; **AIA**
nombArea: tpNombre;

clvDpto: tpClave; **AIP**
codDpto: tpCódigo; **AIA**
nombDpto: tpNombre;

clvAsign: tpClave; **AIP**
codAsign: tpCódigo; **AIA**
nombAsign: tpNombre;
tt_HT, tt_HP: tpHoras;

clvTitulo: tpClave; **AIP**
codTitulo: tpCódigo; **AIA**
nombTitulo: tpNombre;

Atributos de interrelaciones:

impartir.HT, impartir.HP: tpHoras;

Dominios:

tpClave = entero;
tpCódigo = 0..99999;
tpNombre = cadena(40);
tpHoras = 0..400;

Restricciones:

1) Ningún profesor puede impartir docencia en una asignatura que no esté encargada a su área de conocimiento
2) El total de horas impartidas de una asignatura debe ser menor o igual que el correspondiente a la asignatura.

• • •

transformación de la B.D. de la Universidad (3)

Dominios:

tpClave = entero; tpNombre = cadena(40);
tpCódigo = 0..99999; tpHoras = 0..400;

Esquemas de relación:

Departamento = (<u>clvDpto</u>: tpClave; codDpto: tpCódigo, *UNICO, NO NULO;* nombDpto: tpNombre, *NO NULO*);

eliminar pertenecer *y propagar* clvDpto *a relación* AreaConoc

AreaConoc = (<u>clvArea</u>: tpClave; codArea: tpCódigo, *UNICO, NO NULO;* nombArea: tpNombre, *NO NULO*
clvDpto: tpClave, *NO NULO, clave ajena de* Departamento);

eliminar adscribir *y propagar* clvArea *a relación* Profesor

Profesor = (<u>clvProf</u>: tpClave; codProf: tpCódigo, *UNICO, NO NULO;* nombProf: tpNombre, *NO NULO*);
clvArea: tpClave, *NO NULO, clave ajena de* AreaConoc);

eliminar encargar *y propagar* clvArea *a relación* Asignatura

Asignatura = (<u>clvAsign</u>: tpClave; codAsign: tpCódigo, *UNICO, NO NULO;* nombAsign: tpNombre, *NO NULO;*
clvArea: tpClave, *NO NULO, clave ajena de* AreaConoc; tt_HT, tt_HP: tpHoras, *NO NULOS*);

transformación de la B.D. de la Universidad (4)

Titulacion = (<u>clvTitulo</u>: tpClave; codTitulo: tpCódigo, *UNICO, NO NULO;* nombTitulo: tpNombre, *NO NULO*);

ImparteAsign = (<u>clvProf</u>: tpClave; <u>clvAsign</u>: tpClave; HT, HP: tpHoras, *NO NULOS*);
clave ajena (clvProf), *referencia a* Profesor; *borrado* en cascada
clave ajena (clvAsign), *referencia a* Asignatura; *borrado* en cascada

AsignTitulo = (<u>clvAsign</u>: tpClave; <u>clvTitulo</u>: tpClave);
clave ajena (clvAsign), *referencia a* Asignatura; *borrado* en cascada
clave ajena (clvTitulo), *referencia a* Titulacion; *borrado* en cascada

restricciones:

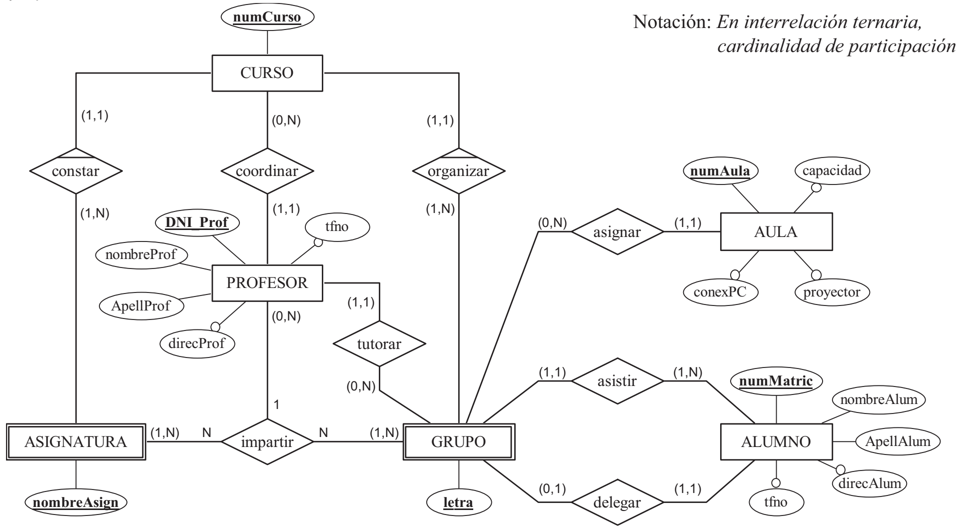
verificar que ∀ *ocurrencia de* clvDpto *en* Departamento, ∃ *en* AreaConoc
verificar que ∀ *ocurrencia de* clvAsign *en* Asignatura, ∃ *en* AsignTitulo
verificar que ∀ *ocurrencia de* clvAsign *en* Asignatura, ∃ *en* ImparteAsign

verificar que ∀ *ocurrencia en* ImparteAsign, *la* clvArea *del* Profesor *coincide con la de la* Asignatura

verificar que *la suma de HT, y la de HP en* ImparteAsign, *agrupando por asignatura es menor o igual que el valor de* tt_HT *y* tt_HP *correspondiente a la* Asignatura

transformación de la B.D. del colegio (1)

***diagrama E/R*:**



transformación de la B.D. del colegio (2)

Dominios:

tpNombre = cadena(40);
tpDirección = cadena(40);
tpTfno = 0..999999999;

Atributos de tipos de entidad:

DNI_Prof: cadena(9); **AIP**
nombreProf: tpNombre;
apellProf: tpNombre;
direcProf: tpDireccion;
profesor.tfno: tpTfno;

numMatric: 0..9999; **AIP**
nombreAlum: tpNombre;
apellAlum: tpNombre;
direcAlum: tpDireccion;
alumno.tfno: tpTfno;

Atributos de tipos de entidad (cont.):

numCurso: 0..8; **AIP**
letra: 'A'..'G'; **AIP**

nombreAsign: tpNombre; **AIP**

numAula: 0..99; **AIP**
capacidad: 0..150;
conex_PC: booleano;
proyector: booleano;

Restricciones:

1) Un alumno sólo puede ser delegado del grupo al que asiste.
• • •

transformación de la B.D. del colegio (3)

Dominios:

tpDNI = cadena(9);	tpNombre = cadena(40);	tpNombAsign = cadena(16);
tpCurso = 1..8;	tpTfno = 0..999999999;	tpNumMatric = 0..99999;
tpIdAula = 1..69;	tpIdGrupo = 'A'..'G';	tpCapacAula = 10..100;

Esquemas de relación:

Profesor = (<u>DNI_Prof</u>: tpDNI; tfno: tpTfno; nombreProf: tpNombre, NO NULO);
ApellProf: tpNombre, NO NULO; direcProf: tpNombre);

eliminar coordinar *y propagar* DNI_Prof *a relación* Curso

Curso = (<u>numCurso</u>: tpCurso; ProfCoord: tpDNI, NO NULO, *clave ajena de* Profesor);

añadir a entidad débil asignatura *la clave de entidad fuerte* numCurso
eliminar constar *y propagar* numCurso *a relación* Asignatura *(ya hecho)*

Asignatura = (<u>nombreAsign</u>: tpNombre; <u>numCurso</u>: tpCurso, *clave ajena de* Curso);
verificar que ∃ numCurso *en* Asignatura ∀ *ocurrencia de* Curso

Aula = (<u>numAula</u>: tpIdAula; capacidad: tpCapacAula; conexPC, proyector: booleano);

transformación de la B.D. del colegio (4)

eliminar asignar *y propagar* numAula *a relación* Grupo
eliminar tutorar *y propagar* DNI_Prof *a relación* Grupo
eliminar delegar *y propagar* numMatric *a relación* Grupo
añadir a entidad débil Grupo *la clave de entidad fuerte* numCurso
eliminar organizar *y propagar* numCurso *a relación* Grupo *(ya hecho)*

Grupo = (<u>letra</u>: tpIdGrupo; tfno: <u>numCurso</u>: tpCurso; numAula: tpIdAula, NO NULO;
profTutor: tpDNI, NO NULO; deleGrupo: tpNumMatric, NO NULO, UNICO);
clave ajena (numCurso), *referencia a* Curso; *borrado* en cascada
clave ajena (numAula), *referencia a* Aula; *borrado* en cascada
clave ajena (profTutor), *referencia a* Profesor; *borrado* en cascada
clave ajena (deleGrupo), *referencia a* Alumno; *borrado* en cascada
verificar que ∃ numCurso *en* Grupo ∀ *ocurrencia de* curso
verificar que ∃ (deleGrupo, numcurso, letra) *en* Alumno

eliminar asistir *y propagar* (numCurso, letra) *a relación* Alumno

Alumno = (<u>numMatric</u>: tpNumMatric; nombreAlum, ApellAlum: tpNombre, NO NULOS;
direcAlum: tpNombre; tfno: tpTfno; numCurso: tpCurso, NO NULO; letra: tpIdGrupo, NO NULO);
clave ajena (numCurso, letra), *referencia a* Grupo; *borrado* en cascada
verificar que ∃ (letra , numCurso) *en* Alumno ∀ *ocurrencia de* Grupo

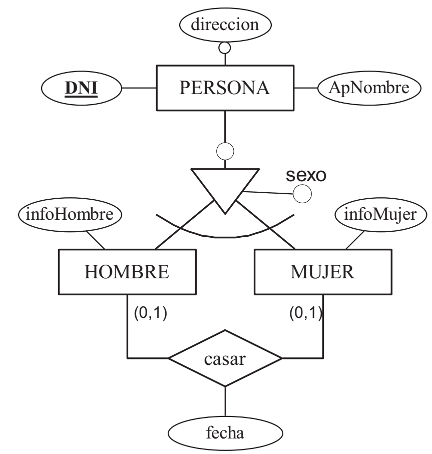
transformación de la B.D. del colegio (5)

impartir = (DNI_Prof: tpDNI , ***NO NULO***; <u>numCurso</u>: tpCurso; <u>letra</u>: tpIdGrupo; <u>nombreAsign</u>: tpNombre);
clave ajena (numCurso, letra), *referencia a* Grupo; *borrado* en cascada
clave ajena (DNI_Prof), *referencia a* Profesor; *borrado* en cascada
clave ajena (nombreAsign, numCurso), *referencia a* Asignatura; *borrado* en cascada

verificar que ∃ (letra , numCurso) *en* impartir ∀ *ocurrencia de* Grupo
verificar que ∃ (nombreAsign, numCurso) *en* impartir ∀ *ocurrencia de* Asignatura

- ***Verificar el esquema relacional obtenido y añadir las restricciones que faltan***
- ***Haga pequeñas modificaciones en el enunciado y estudie cómo afectan al esquema relacional, así como posibles mejoras a la solución propuesta***
- ***Transforme el resto de los esquemas E/R diseñados***

otros ejemplos de transformación



¿y como multivaluado ?

5.3 Dinámica del modelo Relacional: Álgebra relacional

*la **componente dinámica** de un modelo* ≡ ***reglas de transformación de estado** aplicables a las ocurrencias de la B.D. (operadores y restricciones)*

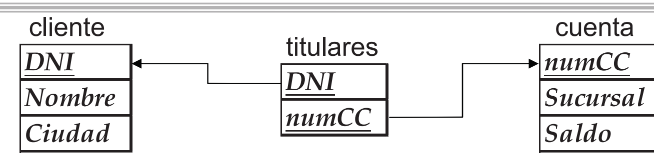
operadores básicos { **Selección** *<condición>* / **Acción** *<objetivo>* → *se expresan mediante* **LMD**

LMD relacionales { • operan sobre conjuntos de relaciones (tuplas) • suelen ser lenguajes de especificación } ≡ no-navegacionales y no-procedurales

- algebraicos (***álgebra relacional***) ⇒ expresiones basadas en operadores algebraicos los operandos y resultados son relaciones
- predicativos (***cálculo relacional***) ⇒ expresiones basadas en lógica de predicados que especifican el objetivo (conjunto de tuplas)
  - ✓ cálculo relacional de tuplas
  - ✓ cálculo relacional de dominios

ejemplos de consultas en álgebra y cálculo relacional (1)

Ejemplos:



1) *numCC, Sucursal y saldo* de todas las cuentas con saldo > 100000 :

Álgebra → $\sigma_{saldo>100000}$ (cuenta)

cálculo de tuplas → { t | t ∈ cuenta ∧ t [saldo] > 100000 }

cálculo de dominios → { < *a, b, c* > | < *a, b, c* > ∈ cuenta ∧ *c* > 100000 }

2) *nombre* de todos los clientes del banco:

Álgebra → Π_{nombre} (cliente)

cálculo de tuplas → { t | ∃ s ∈ cliente (t [nombre] = s [nombre]) }

cálculo de dominios → { < *n* > | ∃ *d, c* (< *d, n, c* > ∈ cliente) }

ejemplos de consultas en álgebra y cálculo relacional (2)

3) *DNI* de todos los clientes del banco que tienen cuenta:

- *Álgebra* ⟶ $\prod_{DNI}$ (cliente ⋈ titulares)
- *cálculo de tuplas* ⟶ { d | ∃ t ∈ titulares (d [DNI] = t [DNI]) }
- *cálculo de dominios* ⟶ { < *d* > | ∃ *c* (< *d, c* > ∈ titulares }

4) *nombre y numCC* de todos los clientes :

- *Álgebra* ⟶ $\prod_{nombre,\ numCC}$ (cliente ⋈ titulares)
- *cálculo de tuplas* ⟶ { s | ∃ c ∈ clientes (s [nombre] = c[nombre] ∧ ∃ t ∈ titulares (c [DNI] = t [DNI] ∧ s [numCC] = t [numCC])))}
- *cálculo de dominios* ⟶ { < *a, b* > | ∃ *d, c* (< *d, a, c* > ∈ cliente ∧ < *d, b* > ∈ titulares)}

introducción al Álgebra Relacional

☛ operandos y resultados son ***relaciones*** R_i y los ***operadores*** O_j / $O_j(R_i) \rightarrow R_k$

operadores básicos	*unión*	∪	R1 ∪ R2
	diferencia	−	R1 − R2
	producto cartesiano	×	R1 × R2
	selección	σ	$\sigma_{condición}$ (R)
	proyeccion	∏	$\prod_{atributos}$ (R)
operadores derivados	*intersección*	∩	R1 ∩ R2
	cociente	÷	R1 ÷ R2
	join	⋈	R1 ⋈ R2

☛ además de añaden otros operadores (*asignación, tratamientos aritméticos, agrupación*, etc.) para facilitar la descripción de consultas más complejas ⟶ álgebra relacional extendida

☛ normalmente sólo se usan para obtener información (otros operadores para inserción, eliminación, etc.)

álgebra relacional: operador selección

Sean:

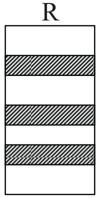
$R(A) = (A_1 : D_1, \dots , A_n : D_n)$ la *intensión* (esquema de relación) de una relación R de grado n, y

$r(R) = \{ t_i \}$ con 1<=i<=m, y $t_i = \langle v_{i1}, v_{i2}, \dots , v_{in} \rangle$ / $v_{ij} \in D_j$, la *extensión* de R.

operadores unarios

ⓐ ***Selección, o restricción*** σ *sobre una relación **R**, según un predicado **p***

R



$\sigma_p(R)$ ≡ selección de las tuplas de la relación R que verifican el predicado *p*

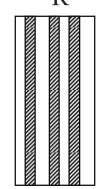
$\sigma_p(R) = R'$, con $r'(R') = \{ t_i \in r(R) \mid p(t_i) \}$, donde *p* es un predicado de selección formado por una expresión lógica de cláusulas de la forma Ai θ Aj , o Ai θ cte. , siendo θ operador de comparación. Los operadores lógicos serán AND, OR y NOT

☛ el grado de R', n′ , es igual a n, y su cardinalidad, m′ <= m

álgebra relacional: operador proyección

ⓑ ***Proyección*** ∏ *de una relación **R** sobre un conjunto de atributos **X***

R



$\prod_X(R)$ ≡ selección de los atributos *X* de la relación R

$\prod_X(R) = R'(X)$, con $r'(R') = \{ t_i(X) \mid X \subset A \}$, es decir, las tuplas de la relación original definidas sobre los atributos X, eliminando las duplicadas.

☛ el grado de R' es n′ < n, y su cardinalidad, m′ <= m

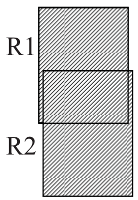
operadores binarios ⇒ *tienen como operandos dos relaciones, y el resultado es otra relación*

☛ en algunos operadores (∪, −, ∩) los operandos deben ser **compatibles**; es decir, deben estar definidos sobre el mismo conjunto de dominios (**semánticamente equivalentes**)

para cambiar los nombres de los atributos y su orden, se puede utilizar el operador unario: **RENOMBRAR** (R; A'), donde A' describe la transformación de atributos de R

álgebra relacional: operadores unión y diferencia

ⓒ ***Unión*** $\cup$ *de dos relaciones **compatibles** R1 y R2*

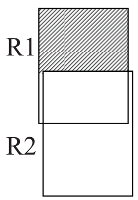


R1 $\cup$ R2 ≡ relación formada por las tuplas de ambas relaciones R1 y R2

R1 $\cup$ R2 = R', con r'(R') = { t_i | $t_i \in$ r1(R1) **or** $t_i \in$ r2(R2) }

☛ el grado de R' es `n' = n1 = n2`, y
su cardinalidad es `m' >= m1`, `m' >= m2`, y `m' <= m1+m2`

ⓓ ***Diferencia*** $-$ *de dos relaciones **compatibles** R1 y R2*



R1 – R2 ≡ relación formada por las tuplas de R1 que no aparecen en R2

R1 – R2 = R', con r'(R') = { t_i | $t_i \in$ r1(R1) **and** $t_i \notin$ r2(R2) }

☛ el grado de R' es, `n' = n1 = n2`, y su cardinalidad, `m' <= m1`

álgebra relacional: operador producto cartesiano

ⓔ ***Producto cartesiano generalizado*** $\times$ *de dos relaciones R1 y R2* ***(cross JOIN)***

R1 × R2 ≡ relación formada por todas las tuplas obtenidas concatenando una tupla de R1 con otra de R2

R1 × R2 = R', con r'(R') = { $<v_{i1}, \ldots, v_{in1}, v_{j1}, \ldots, v_{jn2}>$ | $< v_{i1}, \ldots, v_{in1} > \in$ r1(R1), $< v_{j1}, \ldots, v_{jn2} > \in$ r2(R2) }

☛ el grado de R' es `n' = n1 + n2`, y su cardinalidad `m' = m1 * m2`

☛ *Para distinguir los atributos que tienen el mismo nombre en las dos relaciones, se antepone el nombre de la relación: R1.X, R2.X*

álgebra relacional: ejemplo de operaciones básicas

Sean: R1(A1, A2); R2(A3, A4); R3 = **RENOMBRAR** (R2; A1=A3, A2=A4)

r(R1) =

A1	A2
a	1
b	2
b	1
c	3

r(R2) =

A3	A4
x	1
b	2

$\sigma_{A1=b}$(R1) =

A1	A2
b	1
b	2

Π_{A1}(R1) =

A1
a
b
c

R1 $\cup$ R3 =

A1	A2
a	1
b	2
b	1
c	3
x	1

R1 – R3 =

A1	A2
a	1
b	1
c	3

R1 $\cap$ R3 =

A1	A2
b	2

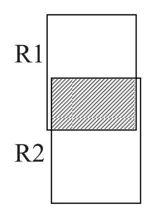
R1 × R2 =

A1	A2	A3	A4
a	1	x	1
a	1	b	2
b	2	x	1
b	2	b	2
b	1	x	1
b	1	b	2
c	3	x	1
c	3	b	2

álgebra relacional: operador intersección

operadores derivados ⟶ *se introducen para simplificar las expresiones, ya que pueden ser descritos con los operadores básicos*

ⓕ ***Intersección*** $\cap$ *de dos relaciones **compatibles** R1 y R2*



R1 $\cap$ R2 ≡ relación formada por las tuplas que pertenecen a R1 y a R2

R1 $\cap$ R2 = R', con r'(R') = { t_i | $t_i \in$ r1(R1) **and** $t_i \in$ r2(R2) }

☛ el grado de R' es `n' = n1 = n2`, y su cardinalidad `m' <= m1`, `m' <= m2`

Obsérvese que: R' = (R1 $\cup$ R2) – ((R1 – R2) $\cup$ (R2 – R1)), ò bien
R' = R1 – (R1 – R2), o R' = R2 – (R2 – R1)

álgebra relacional: operador división

(g) ***División*** $\div$ *de dos relaciones R1 y R2*

> R1 ÷ R2 ≡ relación formada por todas las tuplas *(cociente)* tales que su producto cartesiano por R2 *(divisor)* está incluída en R1 *(dividendo)*

supóngase, para simplificar, que los k últimos atributos de R1 coinciden con los de R2

$R1 \div R2 = R'$, con $R'(A') = (A_1 : D_1, \ldots, A_{n1-k} : D_{n1-k})$, y

$r'(R') = \{ <v_{i1}, \ldots, v_{i(n1-k)}> \mid \forall <v_{j(n1-k+1)}, \ldots, v_{jn1}> \in r2(R2),$
$\exists <v_{i1}, \ldots, v_{i(n1-k)}, v_{j(n1-k+1)}, \ldots, v_{jn1}> \in r(R1)\}$

☛ el grado de R' es `n' = n1 - n2`, y su cardinalidad `m' <= m1 / m2`

Obsérvese que: $R1 \div R2 = \underbrace{\Pi_{A'}(R1)}_{\text{las que hay}} - \underbrace{\Pi_{A'}((\underbrace{\Pi_{A'}(R1) \times R2}_{\text{tuplas que debería haber}}) - R1)}_{\text{las que no pueden ser}}$

ej. de interpretación: Obtener los *alumnos* que están *matriculados* en ***todas*** las *asignaturas*

álgebra relacional: ejemplo de operación división

r(R4) =

A1	A2	A3	A4
a	1	x	1
a	1	b	2
b	2	x	1
b	2	b	2
b	1	x	1
c	3	b	2

r(R2) =

A3	A4
x	1
b	2

⇨ R4 ÷ R2 =

A1	A2
a	1
b	2

$$R4 \div R2 = \underbrace{\Pi_{A1,A2}(R4)}_{R1} - \underbrace{\Pi_{A1,A2}((\underbrace{\underbrace{\underbrace{\Pi_{A1,A2}(R4)}_{R1} \times R2}_{R5}) - R4}_{R6})}_{R7}$$

$\Pi_{A1,A2}(R4) =$ = R1 =

A1	A2
a	1
b	2
b	1
c	3

R1 × R2 = = R5 =

A1	A2	A3	A4
a	1	x	1
a	1	b	2
b	2	x	1
b	2	b	2
b	1	x	1
b	1	b	2
c	3	x	1
c	3	b	2

R5 – R4 = = R6 =

A1	A2	A3	A4
b	1	b	2
c	3	x	1

$\Pi_{A1,A2}(R6) =$ = R7 =

A1	A2
b	1
c	3

R1 – R7 =

A1	A2
a	1
b	2

álgebra relacional: operadores JOIN (1)

(h) ***θ-Combinación, o θ-JOIN*** $\bowtie_{\theta}$ *de dos relaciones R1 y R2*

donde θ es un predicado de selección que relaciona atributos de R1 y R2

> $R1 \bowtie_{\theta} R2 \equiv$ relación formada por todas las tuplas obtenidas concatenando *(combinando)* a una tupla de R1 otra de R2, y que verifican el predicado θ

si k y l representan dos atributos cualquiera de R1 y R2, respectivamente, sobre los que se define θ,

$R1 \bowtie_{\theta} R2 = R'$, con $r'(R') = \{ <v_{i1}, \ldots, v_{in1}, v_{j1}, \ldots, v_{jn2}> \mid \forall\, i, j \;\; <v_{i1}, \ldots, v_{in1}> \in r1(R1),$
$<v_{j1}, \ldots, v_{jn2}> \in r2(R2)$, y se verifica $v_{ik}\ \theta\ v_{jl}$ }

☛ el grado de R' es `n' = n1 + n2`, y su cardinalidad `m' <= m1 * m2`

Obsérvese que: $R1 \bowtie_{\theta} R2 = \sigma_{\theta}(R1 \times R2)$

álgebra relacional: operadores JOIN (1)

Equi-JOIN $\bowtie_{=}$ *de dos relaciones R1 y R2*

> $R1 \bowtie_{=} R2 \equiv R1 \bowtie_{\theta} R2$, con θ el operador =, definido sobre los atributos comunes de R1 y R2

JOIN Natural $\bowtie$ *de dos relaciones R1 y R2*

> $R1 \bowtie R2 \equiv R1 \bowtie_{=} R2$, seguido de la eliminación de atributos repetidos
> ≡ *equi-JOIN, seguido de proyección sobre la unión de los atributos de R1 y R2*

☛ *Es uno de los operadores más usados, ya que permite completar la información de una tabla con la de otra, referenciada en la primera mediante una clave ajena*

álgebra relacional: ejemplo de operaciones JOIN

r(R8) =

A1	A2	A3
a	1	x
a	1	y
b	2	x
b	2	z
b	1	s
c	3	y

r(R9) =

A3	A4
x	7
y	9

$R8 \bowtie R9$ =

A1	A2	A3	A4
a	1	x	7
a	1	y	9
b	2	x	7
c	3	y	9

$R8 \bowtie R9 \; = \; \prod_{A1,A2,A3,A4}(\sigma_{R8.A3=R9.A3}(R8 \times R9))$

$R8 \underset{4*A2>A4}{\bowtie} R9$ =

A1	A2	A3	Á3	A4
b	2	x	x	7
b	2	z	x	7
c	3	y	x	7
c	3	y	y	9

$R8 \underset{4*A2>A4}{\bowtie} R9 \; = \; \sigma_{4*A2>A4}(R8 \times R9)$

$R8 \underset{=}{\bowtie} R9$ =

A1	A2	A3	Á3	A4
a	1	x	x	7
a	1	y	y	9
b	2	x	x	7
c	3	y	y	9

$R8 \underset{=}{\bowtie} R9 \; = \; \sigma_{R8.A3=R9.A3}(R8 \times R9))$

álgebra relacional: tratamiento de los valores nulos

el ***valor nulo*** (o *ausente*) es consecuencia de
- ✓ ***valores desconocidos*** *de atributos*
- ✓ *nuevos **atributos añadidos** a una relación*
- ✓ ***atributos inaplicables*** *a una tupla concreta*

☛ Su tratamiento complica las operaciones y obliga a la ***redefinición de los operadores*** (aritméticos, comparación, lógicos, relacionales, estadísticos, etc.)

→ *No existe una total unidad de criterio en las implementaciones de SGBD*
$\Rightarrow$ *hay que asegurarse al implementar sobre un SGBD concreto*

☛ **los valores nulos no son comparables** *(un valor nulo no es ni igual ni distinto de otro valor)*

☛ el operador $\boxed{\textbf{ES-NULO}\ (v)}$ devuelve verdad si *v* es el valor nulo, y falso en caso contrario

lógica trivaluada

AND	V	Q	F
V	V	Q	F
Q	Q	Q	F
F	F	F	F

OR	V	Q	F
V	V	V	V
Q	V	Q	Q
F	V	Q	F

	NOT
V	F
Q	Q
F	V

*la lógica **cuatrivaluada** usa:*
verdad (V), falso(F),
desconocido aplicable(A),
desconocido no-aplicable(I)

5.4 Interrogación de B.D. con álgebra relacional extendida

para diseñar consultas a una B.D. R. se añaden otros operadores (*álgebra relacional extendida*) →
- *simplificar expresiones complejas*
- *permitir tratamientos especiales* (aritméticos, estadísticos, etc.)

Asignación ← ò :=

R1 **:=** R2 ≡ asocia a la relación R1 la ocurrencia correspondiente a la expresión relacional R2

se puede utilizar para el renombrado de atributos

Agrupación **AGRUPAR** *una relación R respecto a un subconjunto de atributos X*

AGRUPAR $_F$(R, *X*) ≡ agrupa las tuplas de R según los valores de los atributos *X*, y aplica a cada uno de los grupos la lista de funciones de agregación *F*

el resultado consta de los atributos *X*, más uno por cada una de las funciones *F*, y tantas tuplas como "grupos" se hayan obtenido (= cardinalidad de $\prod_X(R)$)

funciones de agregación típicas: ***CONTAR***, ***SUMA***, ***MEDIA***, ***MAX***, ***MIN***, *etc.*

ejemplo de operaciones de agrupamiento

r(R) =

A1	A2	A3
a	1	x
a	1	y
b	2	x
b	2	z
b	1	s
c	3	y

R'(A2, N) **:= AGRUPAR** $_{contar}$(R, *A2*) ⟶ R' =

A2	N
1	3
2	2
3	1

R'(A3, S) **:= AGRUPAR** $_{suma(A2)}$(R, *A3*) ⟶ R' =

A3	S
x	3
y	4
s	1
z	2

☛ *no se suelen eliminar los duplicados antes de aplicar la función de agregación*

☛ *¡¡ Pb. tratamiento de **nulos** !!* ⟶ *no se suelen contar*

extensión de atributos y operadores de JOIN externo

Extensión de atributos **EXTENDER** *una relación R con un conjunto de atributos X*

EXTENDER (R, *X*) ≡ añade a R los atributos especificados por *X* (pares atributo-dominio)

JOIN externo izquierdo, derecho, o completo ${}^{+}\!\bowtie$ $\bowtie^{+}$ ${}^{+}\!\bowtie^{+}$ *left, right, o full* ***outer***

R1 ${}^{+}\!\bowtie$ R2 ≡ JOIN natural de R1 y R2 completado con las tuplas de R1 que no tienen correspondencia en R2 (dichos atributos de R2 tomarán el valor nulo)

R1 $\bowtie^{+}$ R2 ≡ JOIN natural de R1 y R2 completado con las tuplas de R2 que no tienen correspondencia en R1 (dichos atributos de R1 tomarán el valor nulo)

R1 ${}^{+}\!\bowtie^{+}$ R2 ≡ JOIN natural de R1 y R2 completado con las tuplas de R1 que no tienen correspondencia en R2 (dichos atributos de R2 tomarán el valor nulo), y con las tuplas de R2 que no tienen correspondencia en R1 (serán nulos)

ejemplo de operaciones de JOIN externo

r(R1) =

A1	A2	A3
a	1	x
a	1	y
b	2	x
b	2	z
b	1	s
c	3	y

r(R2) =

A3	A4
x	7
y	9

r(R3) =

A3	A4
x	7
y	9
t	6

R1 ${}^{+}\!\bowtie$ R2 =

A1	A2	A3	A4
a	1	x	7
a	1	y	9
b	2	x	7
b	2	z	
b	1	s	
c	3	y	9

R1 $\bowtie^{+}$ R3 =

A1	A2	A3	A4
a	1	x	7
a	1	y	9
b	2	x	7
c	3	y	9
		t	6

R1 ${}^{+}\!\bowtie^{+}$ R3 =

A1	A2	A3	A4
a	1	x	7
a	1	y	9
b	2	x	7
b	2	z	
b	1	s	
c	3	y	9
		t	6

otros operadores de álgebra relacional extendida. Ejemplos

cierre transitivo **CT** *de R sobre un subconjunto de atributos compatibles X*

CT (R, *X*) ≡ relación obtenida aplicando la transitividad sobre *X* hasta la saturación

unión externa $\cup^{+}$ *devuelve relación con todos los atributos* . . .

para el diseño de *consultas complejas* → ***descomponer*** en pasos intermedios

ejemplo: *Considérese el siguiente esquema (simplificado) de una B.D. relacional*

Pieza (<u>clvPieza</u>: *entero*; nombPieza, color: *tpNombre*);
Proveedor (<u>clvProv</u>: *entero*; nombProv: *tpNombre*);
suministrar (<u>clvProv</u>: *entero*; <u>clvPieza</u>: *entero*)
clvProv *clave ajena de* Proveedor; clvPieza *clave ajena de* Pieza;

ejemplos de consulta a una B.D. con álgebra relacional (1)

1 *Piezas de color 'verde'*

$R1 = \sigma_{color = \text{'verde'}}(\text{Pieza})$

2 *Nombre de los proveedores que suministran al menos una pieza de color 'verde'*

$R2 = \Pi_{nombProv}(\text{Proveedor} \bowtie \text{Suministrar} \bowtie R1)$

3 *Nombre de los proveedores que no suministran "tuercas"*

$R3 = \sigma_{nombPieza = \text{'tuerca'}}(\text{Pieza})$ ≡ *piezas que son tuercas*

$R4 = \Pi_{clvProv}(\text{Suministrar} \bowtie R3)$ ≡ *proveedores de tuercas*

$R5 = \Pi_{clvProv}(\text{Proveedor}) - R4$ ≡ *proveedores que no suministran tuercas*

$R6 = \Pi_{nombProv}(\text{Proveedor} \bowtie R5)$ ≡ *nombre de proveedores que no suministran tuercas*

o también

$R5 = \Pi_{nombProv}(\text{Proveedor} \bowtie (\Pi_{clvProv}(\text{Proveedor}) - \Pi_{clvProv}(\text{Suministrar} \bowtie R3)))$

ejemplos de consulta a una B.D. con álgebra relacional (2)

④ *Clave, nombre, y suministradores de las piezas existentes*

a) si todas tienen algún suministrador (faltaría la restricción en el esquema relacional):

$R6 = \prod_{clvPieza,\ nombPieza,\ clvProv} (\text{Pieza} \bowtie \text{Suministrar})$

b) si puede haber piezas que no tienen suministrador:

$R7 = \prod_{clvPieza,\ nombPieza,\ clvProv} (\text{Pieza} \overset{+}{\bowtie} \text{Suministrar})$

o también

$R8 = \prod_{clvPieza,\ nombPieza} (\text{Pieza} \bowtie \text{Suministrar})$ ≡ *piezas que tienen proveedor*

$R9 = \prod_{clvPieza,\ nombPieza} (\text{Pieza}) - R8$ ≡ *piezas que no son suministradas por nadie*

$R7 = (R8 \bowtie \text{Suministrar}) \cup$ **EXTENDER** (R9, *clvProv: entero*)

ó R6

o también

$R9 = \prod_{clvPieza,\ nombPieza} (\text{Pieza}) - \prod_{clvPieza,\ nombPieza} (R6)$ ≡ *piezas que no son suministradas*

$R7 = R6 \cup$ **EXTENDER** (R9, *clvProv: entero*)

ejemplos de consulta a una B.D. con álgebra relacional (3)

⑤ *Nombre de los proveedores que suministran al menos todas las piezas suministradas por el proveedor de clave 25*

$R1 = \prod_{clvPieza} (\sigma_{clvProv = 25} (\text{Suministrar}))$ ≡ *piezas suministradas por el proveedor 25*

$R2 = \prod_{clvProv} (\text{Proveedor}) \times R1$ ≡ *suministros si todos los proveedores suministrasen R1*

$R3 = R2 - \text{Suministrar}$ ≡ *suministros de R1 que no se realizan*

$R4 = \prod_{clvProv} (R3)$ ≡ *proveedores que no suministran todas las piezas de R1*

$R5 = \prod_{clvProv} (\text{Suministrar}) - R4$ ≡ *suministradores de todas las piezas de R1*

$R6 = \prod_{nombProv} (\text{Proveedor} \bowtie R5)$

o también

ejemplos de consulta a una B.D. con álgebra relacional (4)

⑤ *Nombre de los proveedores que suministran al menos todas las piezas suministradas por el proveedor de clave 25*

$R1 = \prod_{clvPieza} (\sigma_{clvProv = 25} (\text{Suministrar}))$ ≡ *piezas suministradas por el proveedor 25*

$R5 = \text{Suministrar} \div R1$ ≡ *suministradores de todas las piezas de R1*

$R6 = \prod_{nombProv} (\text{Proveedor} \bowtie R5)$

⑥ *Nombre de los proveedores que suministran 'tuercas' y 'tornillos'*

$R1 = \prod_{clvPieza} (\sigma_{nombPieza = 'tuerca'} (\text{Pieza}))$ ≡ *piezas que son tuercas*

$R2 = \prod_{clvPieza} (\sigma_{nombPieza = 'tornillo'} (\text{Pieza}))$ ≡ *piezas que son tornillos*

$R3 = \prod_{clvProv} (\text{Suministrar} \bowtie R1)$ ≡ *proveedores de tuercas*

$R4 = \prod_{clvProv} (\text{Suministrar} \bowtie R2)$ ≡ *proveedores de tornillos*

$R5 = \prod_{nomProv} (\text{Proveedor} \bowtie (R3 \cap R4))$

ejemplos de consulta a una B.D. con álgebra relacional (5)

⑥ *Nombre de los proveedores que suministran 'tuercas' y 'tornillos'*

$R1 = R2 = \text{Suministrar} \bowtie \text{Pieza}$ ≡ *piezas y sus suministradores*

$R3(clvProv) = \prod_{R1.clvProv} (\sigma_p (R1 \times R2))$ ≡ *proveedores que suministran tuercas y tornillos*

Siendo ***p* = (R1.clvProv = R2.clvProv) AND (R1.nombPieza = 'tuerca') AND (R2.nombPieza = 'tornillo')**

$R5 = \prod_{nomProv} (\text{Proveedor} \bowtie R3)$

⑦ *Clave, nombre, y nº de piezas que suministra cada proveedor*

R1(clvProv, numPiezas) **:= AGRUPAR** $_{contar(clvPieza)}$(Suministrar, *clvProv*) ≡ *de los proveedores que suministran piezas*

$R2 = \prod_{clvProv,\ nombProv,\ numPiezas} (\text{Proveedor} \overset{+}{\bowtie} R1)$

ejemplos de consulta a una B.D. con álgebra relacional (6)

8 *Clave de las piezas que son suministradas por más de un proveedor*

9 *Clave de las piezas que a lo sumo tienen un proveedor*

10 *Clave de los proveedores que sólo suministran un tipo de piezas*

11 *Clave de los proveedores que suministran alguna pieza en exclusiva (sólo ellos)*

12 *Clave de los proveedores "prescindibles", pues no suministran ninguna pieza en exclusiva*

13 *Parejas (sin repetir ninguna) de proveedores que comparten el suministro de alguna pieza*

14 *Proveedor(es) que más piezas suministra del mismo tipo (tuercas, tornillos, etc.)*

15 *Parejas (sin repetir ninguna) de proveedores suministran exactamente las mismas piezas*

16 *Clave de los proveedores que suministran piezas de todos los colores*

17 *Clave de los proveedores que suministran más de dos piezas*

18 *Pares de suministradores (S_1, S_2) tales que S_1 suministra todas las piezas que suministra S_2*

19 • • • • •