



Práctica 2a.

El modelo relacional.



GRUPO DE ESTRUCTURAS DE DATOS

Presentación

• La introducción por Codd

- muy a finales de los años 60

• de la teoría matemática de las relaciones en el campo de las bases de datos

- supuso un importante paso en la investigación de los SGBD
 - suministrando un sólido fundamento teórico para el desarrollo de nuevos sistemas
 - dentro de este enfoque relacional

• El documento de Codd propone un modelo de datos basado en la teoría de las relaciones

- en donde los datos se estructuran lógicamente en forma de relaciones (tablas)
- siendo un objetivo fundamental del modelo mantener la independencia de esta estructura lógica respecto al modo de almacenamiento y a otras características de tipo físico
- En palabras de Codd en 1970:
 - "la vista relacional de los datos ... parece ser superior al modelo en grafos o en red ... Proporciona un medio de describir datos con su estructura natural únicamente, es decir, **sin superponer ninguna estructura adicional con el propósito de su representación en la máquina**".

Presentación

- El trabajo publicado por Codd en ACM presentaba un nuevo modelo de datos que perseguía una serie de objetivos
 - muchos de ellos comunes a otros modelos
 - que se pueden resumir en los siguientes:
 - **Independencia física:** El modo cómo se almacenan los datos no debe influir en su manipulación lógica y, por tanto, los usuarios que acceden a esos datos no han de modificar sus programas por cambios en el almacenamiento físico
 - **Independencia lógica:** Añadir, eliminar o modificar cualquier elemento de la base de datos no debe repercutir en los programas y/o usuarios que están accediendo a subconjuntos parciales de los mismos (*vistas*)
 - **Flexibilidad:** En el sentido de poder ofrecer a cada usuario los datos de la forma más adecuada a la correspondiente aplicación
 - **Uniformidad:** Las estructuras lógicas de los datos presentan un aspecto uniforme (*tablas*), lo que facilita la concepción y manipulación de la base de datos por parte de los usuarios
 - **Sencillez:** Las características anteriores, así como unos lenguajes de usuario muy sencillos, producen como resultado que el modelo de datos relacional sea fácil de comprender y de utilizar por parte del usuario final



GRUPO DE ESTRUCTURAS DE DATOS

Presentación

- Se debe insistir en la importancia que Codd concede al tema de la independencia de la representación lógica de los datos respecto a su almacenamiento interno
 - independencia de **ordenación**
 - independencia de **indización** e
 - independencia de los **caminos de acceso**
- Importancia que Codd expresa desde su primer artículo dedicado al modelo relacional, en cuyo resumen se puede leer:
 - *"... se propone un modelo relacional de datos como una base para proteger a los usuarios de sistemas de datos formateados de los cambios que potencialmente pueden alterar la representación de los datos, causados por el crecimiento del banco de datos y por los cambios en los caminos de acceso"*



GRUPO DE ESTRUCTURAS DE DATOS

Presentación

- Para conseguir los objetivos citados, Codd introduce el concepto de **relación** (tabla) como estructura básica del modelo
 - Todos los datos de una base de datos se representan en forma de relaciones cuyo contenido varía en el tiempo
 - Una *relación*, en terminología relacional, es un conjunto de filas (*tuplas*) con unas determinadas características
- Con respecto a la parte dinámica del modelo, se propone un conjunto de operadores que se aplican a las relaciones
 - Algunos de estos operadores son clásicos de la teoría de conjuntos
 - no hay que olvidar que una relación se define matemáticamente como un conjunto
 - mientras que otros fueron introducidos específicamente para el modelo relacional
 - Todos ellos conforman el **álgebra relacional**
 - definida formalmente por Codd en 1972 en una publicación donde además
 - se compara el álgebra relacional con el **cálculo relacional**, otro lenguaje también propuesto por Codd



GRUPO DE ESTRUCTURAS DE DATOS

Presentación

- Las indiscutibles ventajas del modelo relacional no le han librado de críticas, a veces acerbadas y, a veces también, justificadas
 - especialmente las relativas a la *poca eficiencia* de los primeros prototipos y productos comerciales y a su *falta de semántica*
- A pesar de que desde su introducción en 1970, el modelo relacional se convirtió en uno de los principales temas de investigación en bases de datos
- los primeros sistemas relacionales tardaron unos diez años en aparecer en el mercado
 - llegándose a calificar de *juguetes*
 - más aptos para la investigación o para el desarrollo de bases de datos experimentales o de pequeño tamaño
 - que para el soporte de verdaderos sistemas de información



GRUPO DE ESTRUCTURAS DE DATOS

Presentación

- A partir de 1980, invade el mercado un gran número de productos relacionales
 - iniciándose un fuerte debate entre los defensores del modelo relacional y sus detractores
 - estos últimos consideraban que la falta de eficiencia de los productos relacionales impedía que fuesen una alternativa válida a los sistemas *Jerárquicos* y *Codasyl* imperantes en el momento
- Probablemente, la teoría relacional nació cuando la tecnología existente no podía todavía ofrecer el soporte adecuado
 - para instrumentaciones que respondiesen eficientemente a las necesidades de los usuarios
 - es decir, se podría considerar la teoría relacional como un *niño prematuro* cuya *cuna*
 - la tecnología que había de sustentarlano estaba preparada en el momento de su nacimiento.



Presentación

- A pesar de ello, el modelo de datos relacional ha tenido un auge espectacular desde finales de los años 70, y sobre todo en los 80
 - una vez que empezaron a vencerse las dificultades que presentaba su instrumentación
 - y gracias al desarrollo tecnológico que ha permitido una mayor eficiencia de los productos relacionales
- Se han publicado miles de artículos y libros que han ido aclarando y ampliando el modelo originariamente propuesto por Codd
- Han ido apareciendo productos comerciales que corren en las más diversas plataformas con rendimientos muy aceptables, en muchos casos comparables a los de los sistemas soportados en modelos prerrelacionales
 - siendo cada vez menos los autores que ponen en duda que los productos relacionales alcanzan, en cuanto a eficiencia, la altura de los basados en otros modelos

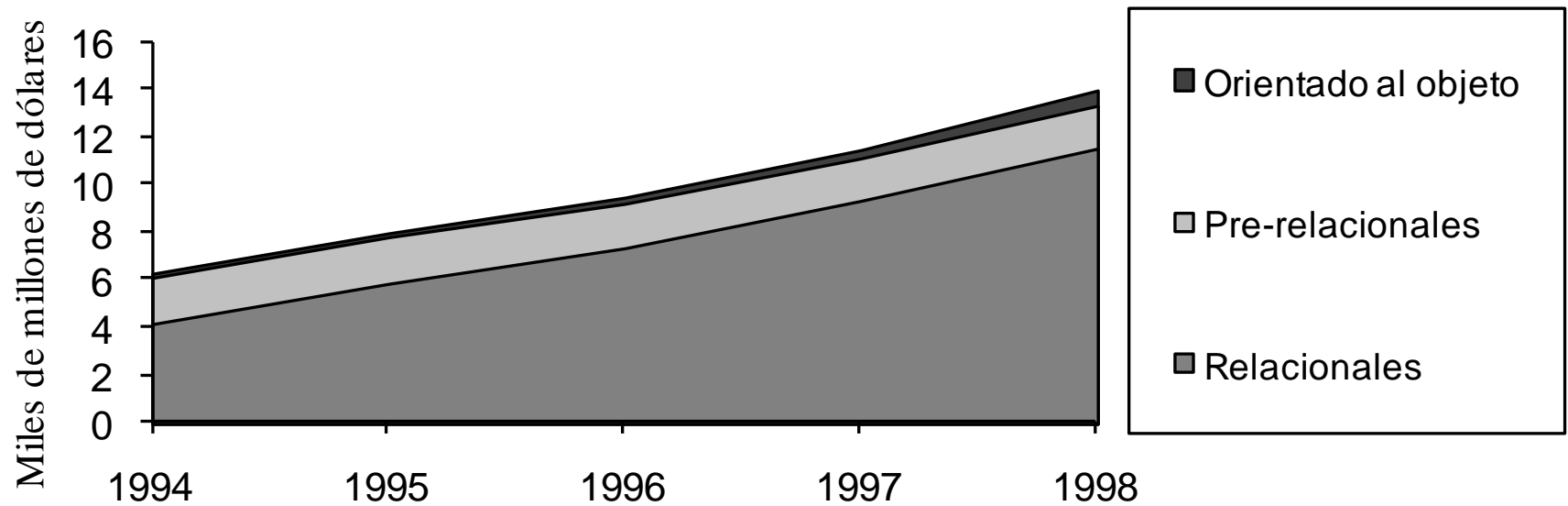


GRUPO DE ESTRUCTURAS DE DATOS

Tecnología de bases de datos	VENTAS MUNDIALES DE SGBD 1991-1999									Crecim. (%) 1994- 1999
	1991	1992	1993	1994	1995	1996	1997	1998	1999	
Prerrelacional	2.000	2.090	2.109	2.050	1.866	1.721	1.701	1.689	1.638	
– Crecimiento	-	4,5	0,9	-2,8	-9,0	-7,8	-1,2	-0,7	-3,0	-4,4
– Cuota de mercado	52,0	45,5	38,8	31,6	24,0	18,4	15,2	12,6	10,3	
Relacional	1.844	2.502	3.328	4.435	5.925	7.652	9.513	11.685	14.254	
– Crecimiento	-	35,7	33,0	33,3	33,6	29,1	24,3	22,8	22,0	26,3
– Cuota de mercado	48,0	54,5	61,2	68,4	76,0	81,6	84,8	87,4	89,7	
SGBD total	3.844	4.592	5.437	6.485	7.791	9.373	11.214	13.374	15.892	
– Crecimiento	-	19,5	18,4	19,3	20,1	20,3	19,6	19,3	18,8	19,6

En millones de dólares

Total ingresos por las ventas mundiales de SGBD

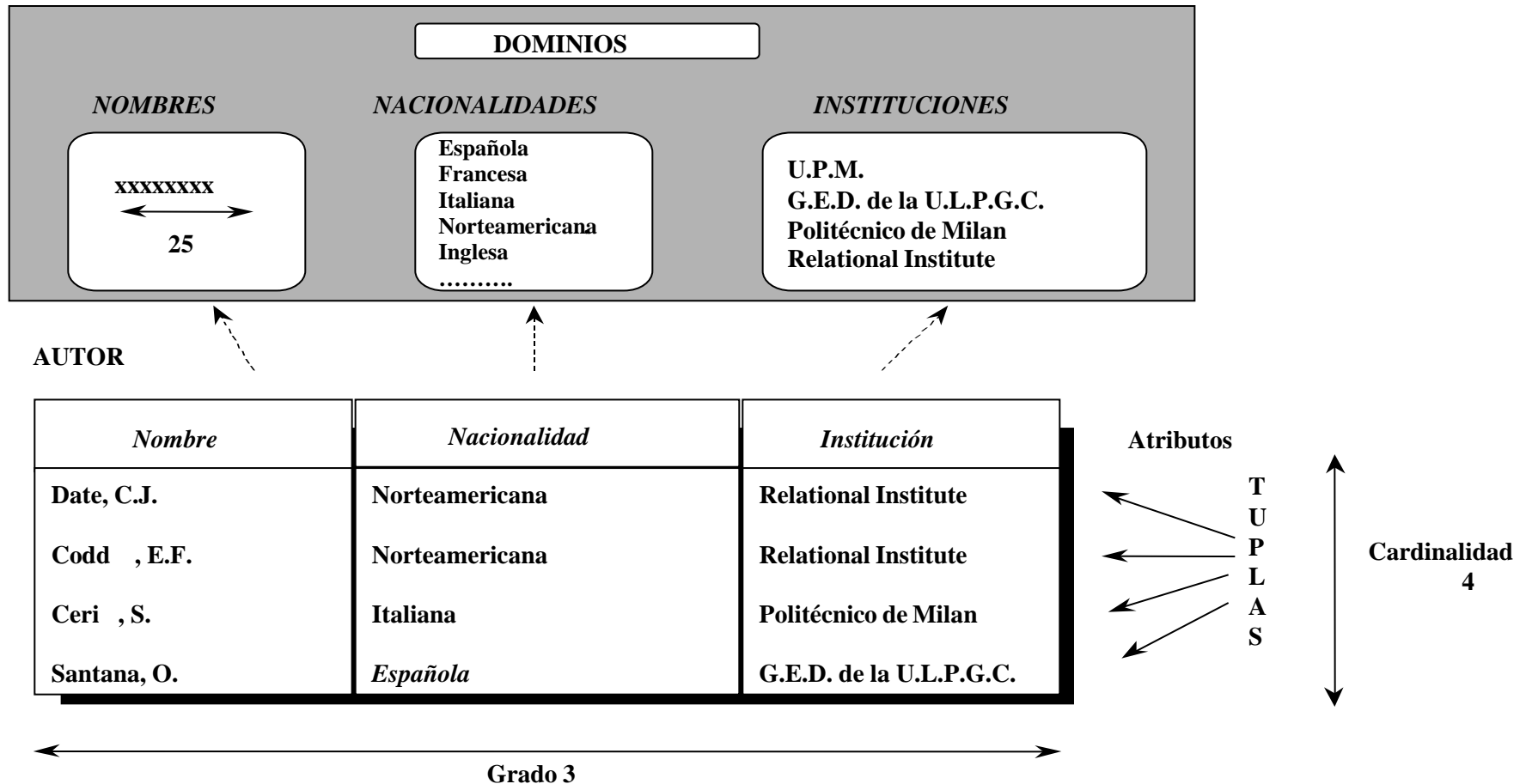




GRUPO DE ESTRUCTURAS DE DATOS

Estructura del modelo relacional

- Como ya se ha señalado, la relación es el elemento básico del modelo relacional, y *se puede representar* como una *tabla*



Relación .vs. Tabla

- Insistimos en que una relación se *puede representar* en forma de tabla
 - aunque tiene una serie de *restricciones inherentes al modelo relacional* que la distinguen de la tabla
 - y que se analizarán más adelante
- Obsérvese también que en una tabla se puede distinguir:
 - una **cabecera** que define la estructura de la tabla
 - es decir, sus atributos con los dominios subyacentes y el resto de metainformación estructural
 - y un **cuerpo** que está formado por un conjunto de tuplas que varían en el tiempo

Relación .vs. Tabla

- Esta representación de la relación como una tabla ha sido el origen de que los productos relacionales y los usuarios utilicen habitualmente el nombre de *tabla*
 - en principio ajeno a la teoría relacional
- para denominar las relaciones y, como consecuencia de ello
 - se llame *filas* a las tuplas
 - y *columnas* a los atributos
- si bien, la terminología es irrelevante y un producto no es más o menos relacional por utilizar una u otra terminología

Comparación terminológica

- A continuación se compara la terminología relacional con la que corresponde a las tablas y a los ficheros





GRUPO DE ESTRUCTURAS DE DATOS

Dominio

- Un dominio D es un conjunto finito de valores, V_1, V_2, \dots, V_n
 - homogéneos
 - y atómicos
- caracterizado por un nombre
- decimos valores **homogéneos** porque son todos del mismo tipo
- y **atómicos** porque son indivisibles en lo que al modelo se refiere
 - es decir, si se descompusiesen, perderían la semántica a ellos asociada
 - Por ejemplo, el dominio de nacionalidades tiene los valores: Española, Francesa, Norteamericana, etc..., que son todos del mismo tipo y que no son divisibles sin perder su semántica
 - así, si descompusiéramos el valor "ESPAÑOLA" en las letras "E", "S", "P", etc..., se perdería la semántica, ya que estas letras consideradas aisladamente han dejado de tener el significado que tiene "ESPAÑOLA" como un valor de la nacionalidad
- Esta exigencia de atomicidad sólo es válida para los llamados dominios simples
 - Codd en 1990 en la versión 2 del modelo relacional distingue entre dominios simples y dominios compuestos cuya definición se verá más adelante

Dominio

- Todo dominio ha de tener un nombre
 - por el cual nos podemos referir a él
- y un tipo de datos
 - Así el tipo de datos del dominio de *nacionalidades* es una tira de caracteres de longitud diez
 - También se le puede asociar una unidad de medida, como metros, kilos, etc..., y ciertas restricciones
- Los dominios pueden definirse por intensión o por extensión
 - Por ejemplo, el dominio de las edades de las personas activas se puede definir por intensión como entero de longitud dos comprendido entre 18 y 65
 - mientras que la definición del dominio de nacionalidades por intensión sería muy pobre semánticamente, ya que permitiría toda combinación de 10 letras aun cuando no constituyesen un nombre válido de nacionalidad
 - por ello, sería preferible definir este dominio por extensión con los nombres de las distintas nacionalidades que admitiésemos en nuestra base de datos

Dominio

- Se podría pensar que un dominio es igual que una relación de grado 1
 - Sin embargo esto no es cierto, ya que el dominio contiene todos los posibles valores que puede tomar un atributo y es estático
 - estos valores no varían en el transcurso del tiempo
 - y si variasen se consideraría un dominio distinto
 - En cambio la relación es dinámica por su misma naturaleza
- Además, los dominios juegan un importante papel propio, característico en ciertas operaciones, como veremos más adelante

Dominio y Atributo

- Un atributo A es el papel que juega un determinado dominio D en una relación
 - se dice que D es el dominio de A
 - Así, el atributo *Nacionalidad* de la tabla AUTOR, definido sobre el dominio de *Nacionalidades* nos indica que dicho dominio tiene el papel de nacionalidad del autor en la referida tabla
- El universo del discurso de una base de datos relacional está compuesto por un conjunto finito y no vacío de atributos estructurados en relaciones
 - cada atributo toma sus valores de un único dominio (dominio subyacente) y varios atributos pueden tener el mismo dominio subyacente
 - Es muy usual dar el mismo nombre al atributo y al dominio subyacente
 - En el caso de que sean varios los atributos de una misma tabla definidos sobre el mismo dominio
 - habrá que darles nombres distintos, ya que una tabla no puede tener dos atributos con el mismo nombre

Dominios compuestos

- Además de los dominios simples, que acabamos de definir, en los trabajos posteriores de algunos autores
 - Codd y Date en 1990
- se introduce el concepto de **dominio compuesto** que es muy importante desde un punto de vista práctico para el diseño de bases de datos
- Un **dominio compuesto** se puede definir como una combinación de dominios simples a la que se puede aplicar ciertas restricciones de integridad
 - Por ejemplo, un usuario puede necesitar manejar, además de los tres dominios *Día*, *Mes* y *Año*, un dominio compuesto por ellos denominado *Fecha*, al que podríamos aplicar las adecuadas restricciones de integridad a fin de que no aparecieran valores no válidos para la fecha
 - según las aplicaciones, puede ser conveniente tratarlos en conjunto o por separado

Atributos compuestos

- Al igual que es posible definir dominios compuestos, existen también atributos compuesto
 - así el atributo *Fecha* tomaría sus valores del dominio compuesto de igual nombre
- En estos momentos, muy pocos productos relacionales reconocen el concepto de dominio
 - ni, menos aun, lo tratan adecuadamente
- ni tampoco, en general, los productos son capaces de trabajar con atributos compuestos
- Sin embargo, aún en esas condiciones se debe atender obligatoriamente a los principios teóricos impuestos conceptualmente por los dominios subyacentes

Definición matemática de Relación

- Matemáticamente,
 - una **relación**, definida sobre los n dominios D_1, D_2, \dots, D_n , no necesariamente distintos, es un subconjunto del producto cartesiano de estos dominios, donde cada elemento de la relación (*tupla*) es una serie de n valores ordenados
 - En esta definición matemática de relación
 - que es la que aparece en los primeros trabajos de Codd
 - no se alude a los atributos, es decir, al papel que juegan los dominios en la relación y
 - además, en ella el orden de los valores dentro de una tupla es significativo

Elementos de una relación

- A fin de evitar estos inconvenientes se puede dar otra definición de relación más adecuada desde el punto de vista de las bases de datos
 - para lo cual es preciso distinguir en la noción de relación los siguientes elementos:
 - **Nombre:** Las relaciones se identifican por un nombre; si bien ciertas relaciones que no necesitan identificarse (por ejemplo, resultados intermedios) pueden no tener nombre
 - **Cabecera de relación:** Conjunto de n pares atributo-dominio subyacente $\{ (A_i : D_i) \}_{i=1}^n$ donde n es el **grado**
 - se corresponde con la primera fila cuando la relación se percibe como una tabla
 - el conjunto A de atributos sobre los que se define la relación se llama **contexto** de la misma
 - El término conjunto tiene aquí su preciso significado matemático, es decir, no existen elementos repetidos y el orden de los elementos es irrelevante

Elementos de una relación

- **Cuerpo de la relación:** Conjunto de m tuplas $\{ t_1, t_2, \dots, t_m \}$, donde cada tupla es un conjunto de n pares atributo-valor $\{ (A_i : V_{ij}) \}$, siendo V_{ij} el valor j del dominio D_i asociado al atributo A_i
 - el número de tuplas m es la **cardinalidad**
- El término conjunto vuelve a tener aquí su preciso significado matemático, es decir, no existen elementos repetidos y el orden de los elementos es irrelevante
- Así como la cabecera de relación es invariante, su cuerpo varía con el transcurso del tiempo, al igual que la cardinalidad

Esquema de relación

- El **esquema de relación** estará constituido por el nombre R (si existe) y la cabecera, denotandose:
- $$R (\{ A_i : D_i \}^{n}_{i=1})$$
 - Después de estudiar las restricciones se podrá dar una definición más completa y precisa de este concepto
- El esquema de relación representa la parte definitoria y estática y se denomina también **intensión**
 - Se corresponde con lo que hemos llamado *tipo* en el modelo Entidad/Interrelación
 - Aunque a veces se dice que es *relativamente estática*, dado que una relación puede modificarse añadiendo o suprimiendo algún atributo cuando varía el universo del discurso al cual ésta representa
 - sería preferible decir en este caso que se trata de una nueva relación

Definición de relación

- El **estado de relación** $r(R)$, al que denominaremos simplemente **relación**, está constituido por el esquema y el cuerpo de relación:
- $\langle \text{esquema, cuerpo} \rangle$
 - El cuerpo se denomina también **extensión** u *ocurrencia de relación*
 - Es preciso observar, sin embargo, que lo que se llama ocurrencia en el modelo E/R sería un sola tupla de la relación
 - Esta terminología no homogénea y, en general, poco precisa suele inducir a confusión, en especial a los que están iniciándose en el tema
- Una **base de datos relacional** es una base de datos percibida por los usuarios como una colección de relaciones que pueden variar en el tiempo

ESQUEMA DE RELACION (INTENSION):

AUTOR (*Nombre: Nombres, Nacionalidad: Nacionalidades, Institución: Instituciones*)

RELACION (EXTENSION, ESTADO u OCURRENCIA):

AUTOR

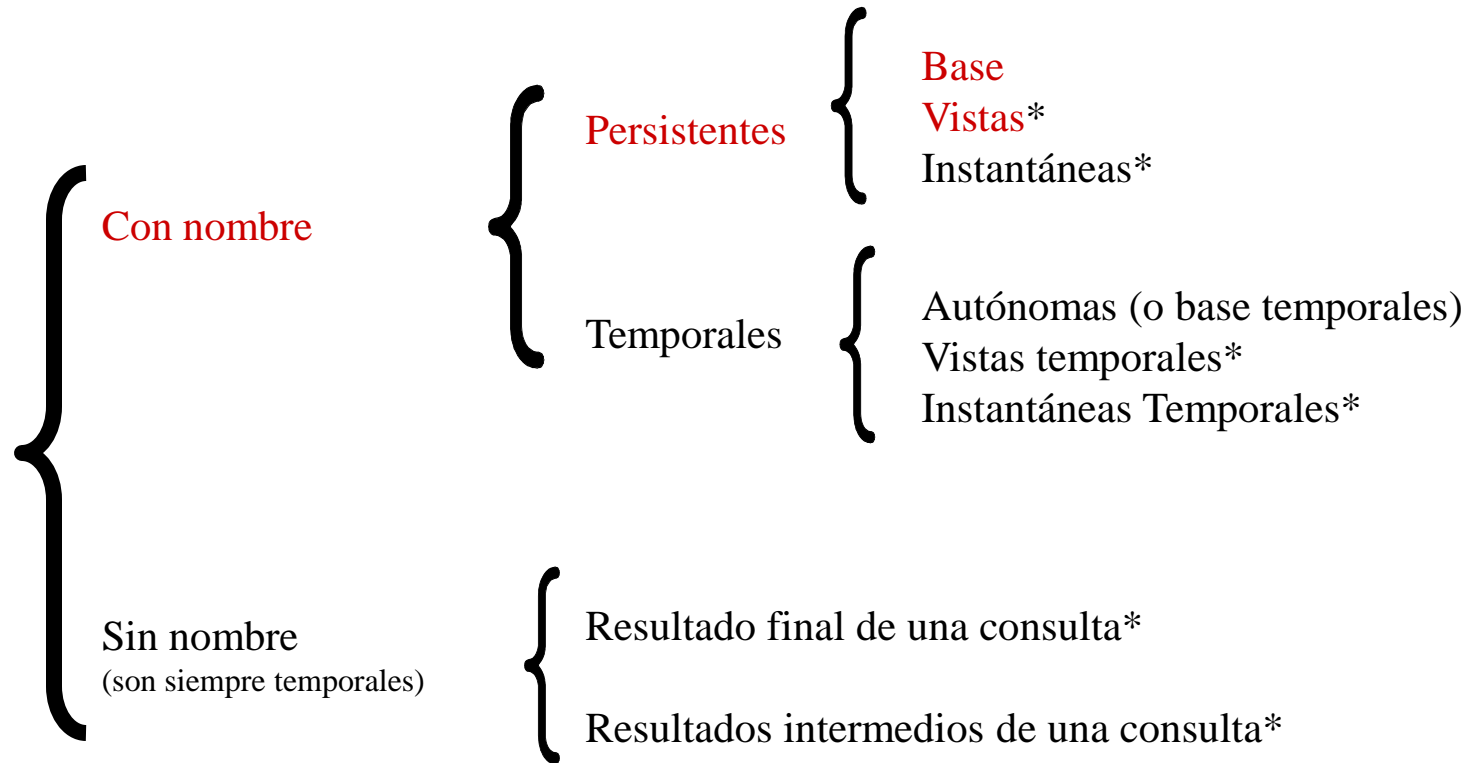
<i>Nombre</i>	<i>Nacionalidad</i>	<i>Institución</i>
Date, C.J.	Norteamericana	Relational Institute
Santana, O.	<i>Española</i>	G.E.D. de la U.L.P.G.C.
Ceri, S.	Italiana	Politécnico de Milan



GRUPO DE ESTRUCTURAS DE DATOS

Clases de relaciones

* Relaciones derivadas



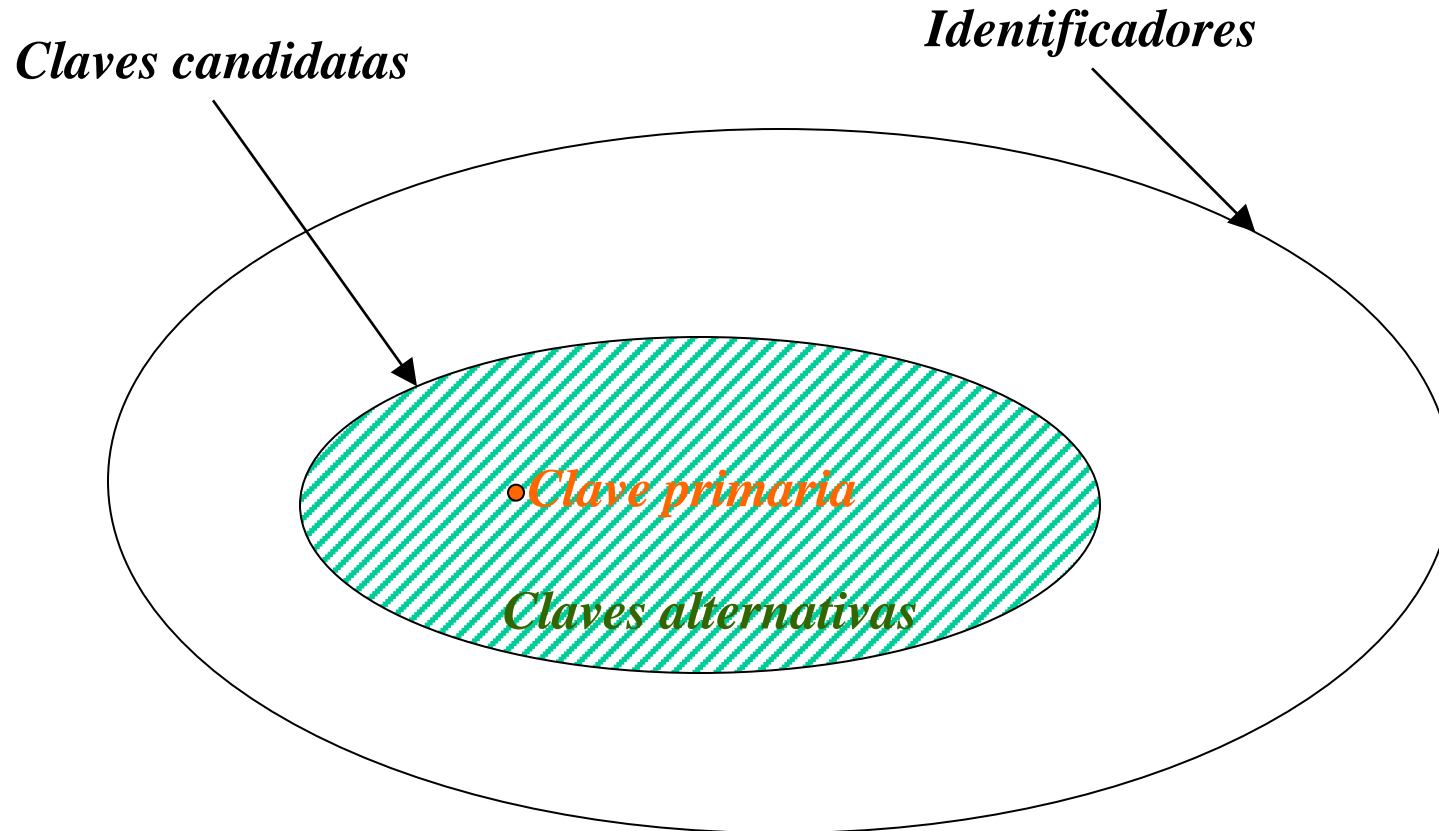
Claves candidatas

- Una **clave candidata** de una relación es un conjunto de atributos que identifican unívoca y mínimamente cada tupla de la relación
 - Por la propia definición de relación, siempre hay, al menos, una clave candidata
 - ya que al ser una relación un conjunto, no existen dos tuplas iguales y, por tanto, el conjunto de todos los atributos siempre tiene que identificar unívocamente a cada tupla
 - Si no se cumpliera la condición de minimalidad
 - estaríamos ante un identificador que no es clave candidata
 - se habrían de eliminar del conjunto de atributos aquellos que impidan cumplir la condición de minimalidad, a fin de que constituya una clave candidata

Clave primaria y claves alternativas

- Una relación puede tener más de una clave candidata, entre las cuales se debe distinguir:
 - **Clave primaria**: es aquella *clave candidata* que el usuario escogerá, posiblemente por consideraciones ajenas al modelo relacional, para identificar y hacer referencia a las tuplas de la relación
 - Cuando sólo exista una clave candidata, ésta será la clave primaria
 - **Claves alternativas**: son aquellas *claves candidatas* que no han sido escogidas como clave primaria y por lo tanto no se usarán para identificar ni hacer referencia a las tuplas de la relación
 - Una relación puede tener cero o más claves alternativas

Claves de una relación





GRUPO DE ESTRUCTURAS DE DATOS

Claves ajenas

- Se denomina **clave ajena** de una relación R2 a un atributo (quizá compuesto) cuyos valores deben concordar con los de la clave primaria de una relación R1
 - R1 y R2 no son necesariamente distintas
- Obsérvese que no se requiere lo inverso
 - la clave primaria correspondiente a una clave ajena dada
 - podría contener un valor que no aparezca de momento como valor de esa clave ajena
- La clave ajena y la correspondiente clave primaria deben definirse sobre el mismo dominio subyacente
 - el cual puede ser compuesto
- Se dice que R2 referencia a R1 y se puede expresar así:

$R2 \longrightarrow R1$ { Etiquetando opcionalmente la flecha con el nombre de la clave ajena

Claves ajenas

- La clave ajena **no** tiene por qué servir de **identificador** de las tuplas de su relación
 - por lo cual **no** es muy *afortunado* aplicar el término **clave** a las *claves ajenas*
- Adviértase que las claves ajenas **deben** **aceptar** *nulos* **en ocasiones**
- A veces se dice que las concordancias de clave ajena con clave primaria son el adhesivo que da coherencia a las bases de datos relacionales
 - puesto que permiten representar ciertas interrelaciones entre tuplas
 - pero hay que tener en cuenta que no todas las interrelaciones entre tuplas se representan por este mecanismo
 - y que en realidad esta concordancia es sólo un caso especial (muy importante) de un mecanismo más general, a saber, la concordancia de cualesquiera dos atributos definidos sobre el mismo dominio

Restricciones inherentes:

Obligatoriedad de la clave primaria

- De la definición matemática de relación, se deduce inmediatamente una serie de características propias de una relación que se han de cumplir obligatoriamente
 - por lo cual se trata de restricciones inherentes
 - y como ya hemos señalado, diferencian una *relación* de una *tabla*
 - **No hay dos tuplas iguales** (*de donde se deduce la obligatoriedad de la clave primaria*)
 - Esta restricción, inherente al modelo, no es respetada por muchos productos comerciales
 - los cuales permiten la definición de clave primaria pero no hacen esta definición obligatoria
 - Las tablas que admiten filas duplicadas no son conjuntos (en su sentido matemático), sino “multiconjuntos”

Restricciones inherentes: Orden de tuplas y atributos

- **El orden de las tuplas no es significativo**
- **El orden de los atributos no es significativo**
 - Si nos atuviésemos a la definición matemática de relación como "subconjunto del producto cartesiano de n dominios no necesariamente distintos"
 - el orden de los atributos sería significativo, es decir, si cambiásemos el orden de los atributos tendríamos una relación distinta
 - Ante el inconveniente que esto supondría para el usuario y las ventajas de poder alterar el orden de los atributos sin que cambie la relación
 - es conveniente definir la relación como lo hemos hecho desde el punto de vista de las bases de datos en la segunda definición
 - teniendo así una definición consistente con la característica de que el orden de los atributos es irrelevante
 - Muchos autores siguen conservando la primera definición, aunque introducen esta restricción

Restricciones inherentes: Normalización

- Por supuesto, todo atributo simple estará definido sobre un dominio simple; además
- **Cada atributo sólo puede tomar un único valor del dominio** sobre el que está definido, **no** admitiéndose por tanto los *grupos repetitivos*
 - Se dice que una relación que cumple esta condición está **normalizada**
 - o también que está en *primera forma normal*
 - Toda relación ha de estar normalizada, en caso contrario no es realmente una relación

Restricciones inherentes: Integridad de entidad

- Además de las anteriores restricciones inherentes
 - derivadas de la misma definición de relación
- existe otra restricción inherente que es la **regla de integridad de entidad**, la cual impone que:
 - *"Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo"*
 - esto es, un valor desconocido o inexistente
 - protegiendo de esta forma la identificación y la capacidad de hacer referencia a las tuplas de las relaciones
 - Esta restricción se considera inherente al modelo porque toda relación ha de tener obligatoriamente una clave primaria

Restricciones semánticas: Unicidad

- **Unicidad** (UNIQUE): Indica que los valores de un conjunto de atributos (uno o más) no pueden repetirse en una relación
- Esta restricción permite la definición de *claves alternativas*

No nulidad

- **No nulidad** (NOT NULL): Indica que el atributo no admite valores nulos

Restricciones semánticas:

Clave primaria

- **Clave primaria** (PRIMARY KEY): Permite declarar un atributo o un conjunto de atributos como *clave primaria* de una relación
- Toda declaración de *clave primaria* lleva implícitas las siguientes *restricciones*:
 - sus valores no se podrán repetir (*unicidad*) y
 - no se admitirán valores nulos en ninguno de sus componentes (*no nulidad*)
 - La *no nulidad* de los componentes de la clave primaria es una restricción inherente del modelo relacional (*integridad de entidad*)
 - también conocida como *primera regla de integridad*

Restricciones semánticas:

Clave primaria

- Se debe distinguir entre la restricción inherente de obligatoriedad de la clave primaria y la restricción semántica que le permite al usuario indicar qué atributos forman parte de la misma
 - La declaración de uno o varios atributos como clave primaria de una relación es una restricción semántica
 - que responde a la necesidad del usuario de imponer que los valores de *ese conjunto concreto de atributos*, que constituyen la clave primaria, no se repitan en la relación ni tampoco tomen valores nulos

Restricciones semánticas:

Claves ajenas

- **Integridad referencial** (FOREING KEY): Si una relación R2 (relación que referencia) tiene un atributo que actúa como clave primaria de la relación R1 (relación referenciada), todo valor de dicho atributo debe
 - o bien concordar con un valor de la clave primaria de R1
 - o bien ser nulo
- El atributo de R2 es, por tanto, una clave ajena en la relación R2
- Las relaciones R1 y R2 no son necesariamente distintas
- Además, cabe destacar que la clave ajena puede ser también parte (o la totalidad) de la clave primaria de R2
- Obsérvese que los nombres de los atributos que son clave ajena de una relación no tienen porqué coincidir con los nombres de la clave primaria de la relación referenciada

Restricciones semánticas: Reglas para claves ajenas

- Además de definir las claves ajenas, hay que determinar si se permiten o no valores nulos y las consecuencias que pueden tener ciertas operaciones:
 - **borrado y**
 - **modificación**
- realizadas sobre tuplas de la relación referenciada
- pudiéndose distinguir las opciones que se exponen a continuación

Restricciones semánticas: Reglas para claves ajenas

- **Operación restringida (RESTRICTED):** El borrado de tuplas de la relación referenciada (o la modificación de su clave primaria) sólo se permite si no existen tuplas con este valor en la relación referenciante
 - En caso de que existan, el sistema impide la operación
- **Operación con propagación o transmisión en cascada (CASCADE):** El borrado de tuplas de la relación referenciada (o la modificación de su clave primaria) lleva consigo el borrado (o modificación) en cascada de las tuplas de la relación referenciante

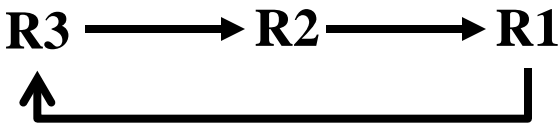
Restricciones semánticas: Reglas para claves ajenas

- **Operación con puesta a nulo (SET NULL):** El borrado de tuplas de la relación referenciada (o la modificación de su clave primaria) lleva consigo *poner a nulos* los valores de la clave ajena de la relación referenciante
 - Esta opción, obviamente, sólo es posible cuando el atributo que es clave ajena admite valores nulos
- La opción seleccionada en caso de borrado es independiente de la de modificación
 - es decir, las opciones de borrado y de modificación pueden ser distintas

Restricciones semánticas: Reglas para claves ajenas

- Las tres opciones expuestas no agotan todas las posibilidades
 - Podría plantearse una opción para poner en la clave ajena un valor por defecto señalado previamente
 - o en general, podría invocarse un procedimiento definido previamente
- pero en general las tres opciones que se han dado cubren una buena parte de las situaciones que se plantean en la práctica
- Sea $R2 \longrightarrow R1$ { Con propagación en la regla de eliminación (modificación)
y $R3 \longrightarrow R2$ { Con propagación en la regla de eliminación (modificación)
 - la eliminación (o modificación) de tuplas de $R1$ puede propagarse hasta eliminar (o modificar) las tuplas de $R3$
 - Si la relación referencial entre $R3$ y $R2$ no es en cascada haría fracasar el intento de propagación de $R1$ a $R2$

Restricciones semánticas: Reglas para claves ajenas

- Ante ciclos referenciales 

```
graph LR; R3 --> R2; R2 --> R1; R1 --> R3
```
- a) deben permitirse nulos al menos en una clave ajena del ciclo o bien
- b) no se realizará la verificación de restricciones en las inserciones individuales, hasta que se hayan insertado todas las tuplas del ciclo referencial
 - porque en caso contrario no podrá insertarse la primera tupla de la base de datos

Restricciones semánticas adicionales:

Restricciones de rechazo

- Además de las restricciones que acabamos de exponer, existen otras restricciones que podríamos llamar de **rechazo**
 - en las que el usuario formula una condición mediante un predicado definido sobre un conjunto de atributos, de tuplas o dominios
 - el cual debe ser verificado por los correspondientes objetos en toda operación de actualización para que el nuevo estado constituya una ocurrencia válida del esquema
 - en caso de que la operación intente violar la condición se impide que la operación se lleve a cabo
 - es decir, se produce un **rechazo** de la operación

Restricciones semánticas adicionales:

Restricciones de rechazo

- Se podrían distinguir dos tipos de restricciones de rechazo, según la condición afecte:
 - a un único elemento de la base de datos (por ejemplo, a una relación)
 - o a más de uno
- **Verificación** (CHECK): Comprueba, en toda operación de actualización, si el predicado es cierto o falso y, en el segundo caso, rechaza la operación. La restricción de verificación se define sobre un único elemento (incluyéndose en la definición de dicho elemento) y puede o no tener nombre
- **Aserción** (ASSERTION): Actúa de forma idéntica a la anterior, pero se diferencia de ella en que puede afectar a varios elementos (por ejemplo, a dos relaciones distintas) y su definición, por tanto, no va unida a la de un determinado elemento por lo que siempre ha de tener un nombre, ya que la aserción es un elemento más del esquema que tiene vida por sí mismo

Restricciones semánticas adicionales: Restricciones de rechazo

- Las restricciones de rechazo que se definen sobre un solo dominio, relación o atributo
 - diremos que son restricciones *intraelemento*
- y las que se definen sobre varios elementos
 - se dice que son restricciones *interelementos*
- También es posible definir restricciones de transición haciendo referencia en el predicado a los valores anteriores a la operación de actualización y a los nuevos valores

Restricciones semánticas adicionales: Disparadores

- Los **disparadores** (*triggers*) son restricciones en las que el usuario puede especificar libremente la respuesta (acción) ante una determinada condición
- Así como las anteriores restricciones son *declarativas*, los disparadores son *procedimentales* (al menos en lo que a la acción se refiere)
 - siendo preciso que el usuario escriba el procedimiento que ha de aplicarse en caso de que se cumpla la condición
- Los disparadores no sólo se utilizan para garantizar la integridad de la base de datos
 - sino que tienen también un importante campo de aplicación en las bases de datos activas
- En ellos la acción se activa cuando la condición se cumple, al contrario de lo que ocurre en otro tipo de restricciones



GRUPO DE ESTRUCTURAS DE DATOS

Esquema de relación

- En un esquema de relación deberemos especificar:
 - los atributos y dominios sobre los que se define la relación
 - así como las restricciones de integridad que se deben cumplir para que la relación constituya una ocurrencia válida del esquema
 - es decir, aquellas restricciones que afectan a cada uno de los elementos que forman parte del correspondiente esquema de relación (restricciones intraelementos)
- Por tanto, podremos definir un esquema de relación como:
 - $R \langle A:D, S \rangle$
 - siendo R el nombre de la relación, A la lista de atributos, D los dominios sobre los que están definidos los atributos y S las restricciones de integridad intraelementos
 - Es habitual omitir, en la definición de un esquema de relación, las restricciones de integridad
 - pero ésta **no** sería una definición completa de esquema de relación

Esquema relacional

- El esquema de la base de datos relacional será una colección de esquemas de relación y de restricciones de integridad interelementos
- Esto se puede representar:
- $$E < \{R_i\}, \{I_i\} >$$
 - donde E es el nombre del esquema relacional, $\{R_i\}$ es el conjunto de esquemas de relación e $\{I_i\}$ representa el conjunto de restricciones de integridad interelementos
- Podemos definir una Base de Datos Relacional -“variable relacional” siguiendo la terminología de Date en 1995- como un *esquema relacional* junto con una *ocurrencia válida* de dicho esquema
 - es decir, *una ocurrencia que cumple todas las restricciones descritas en el esquema*

Esquema de relación: Notación I

- Una notación que se puede seguir para definir una relación en el esquema relacional es la siguiente:
 - Supóngase que se quiere definir dos relaciones
 - R con una clave candidata y
 - S con dos claves candidatas y una restricción de valor no nulo
 - cuyos respectivos esquemas son
 - $\{(A_1, D_1), (A_2, D_2), \dots, (A_r, D_r)\}$ y $\{(B_1, E_1), (B_2, E_2), \dots, (B_t, E_t)\}$
 - Además, R tiene una clave ajena que referencia a S

R \longrightarrow S



GRUPO DE ESTRUCTURAS DE DATOS

Esquema de relación: Notación I

- $R (A_1:D_1, A_2:D_2, \dots, A_r:D_r)$
CP: $\{A_i, \dots, A_m\}$
CAj: $\{A_p, \dots, A_q\}$ referencia a S
f: $A_p \rightarrow B_j$
...
 $A_q \rightarrow B_n$
 - borrado en cascada
 - modificación a nulos
- $S (B_1:E_1, B_2:E_2, \dots, B_t:E_t)$
CP: $\{B_j, \dots, B_n\}$
CAlt: $\{B_q, \dots, B_r\}$
VNN: $\{B_s, \dots, B_t\}$

Esquema de relación: Notación II

- Sin embargo, la notación que se usará con más frecuencia para poder expresar el esquema de una relación sin concretar las características específicas de ningún SGBD relacional, es esta *notación pseudo-SQL*:

Create Table <nombre relación> (< nombre atributo> <tipo>|<dominio> [not null]
[{,<nombre atributo> <tipo>|<dominio> [not null]}])

Primary Key (<clave primaria>)

[{Alternative Key (<clave alternativa>)}] --frecuentemente se sustituye Alternative Key por Unique

[{Foreign Key (<clave ajena>) references to <objetivo>

delete of <objetivo> <efecto> --efecto puede ser restricted, cascade o set null

update of <clave primaria del objetivo> <efecto> }]