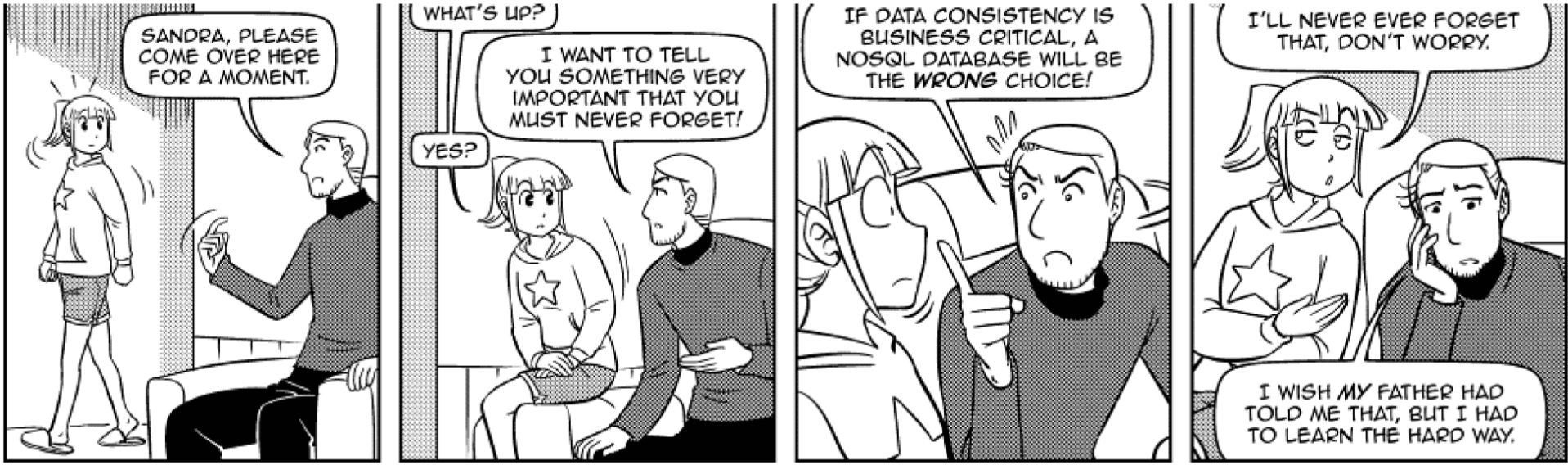
Those who are enamored of practice without theory are like a pilot who goes into a ship without rudder or compass and never has any certainty where he is going. Practice should always be based on a sound knowledge of theory.

Leonardo Da Vinci (1452-1519)

DEPARTAMENTO DE COMPUTACION

Modelo Relacional

Dr. Gerardo Rossel



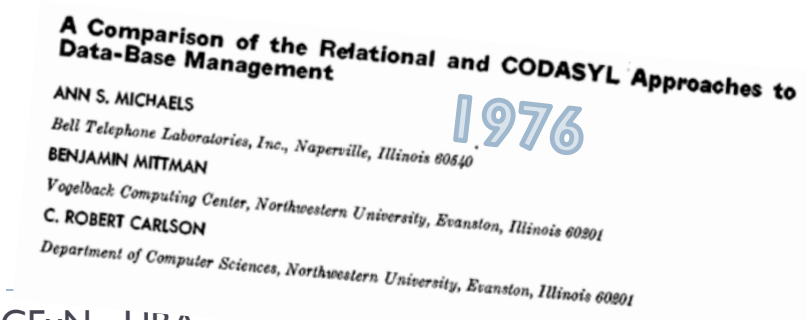
Sandra and Woo by Oliver Knörzer (writer) and Powree (artist) – www.sandraandwoo.com

Modelo Relacional

Los **muertos** que vos matáis **gozan de buena salud**

Modelo Relacional: Contexto Histórico

- 1950...Datos mantenidos en Archivos.
- 1964-1980 Pre-relacional.
  - Modelo Jerárquico - Information Management System (IMS) de IBM (1966).
  - Modelo de Red - CODASYL DBTG: IDS (Integrated Data Store) - Charles Bachman (General Electric)
- 1970-1980: La guerra de los modelos. *CODASYL DBTG* vs *Relational Model*
- 1970 (Junio) Ted Codd: A Relational Model of Data for Large Shared Data Banks in Communications of the ACM.
- 1971 CODASYL DBTG: publica su estándar
- 1974-1977 Surgen dos prototipos clave:
  - **System R**, desarrollado en IBM - **Donald D. Chamberlin y Raymond F. Boyce**.
  - **Ingres**, creado en la Universidad de Berkeley - **Michael Stonebraker** y **Eugene Wong**
- 1979 **Oracle** lanza la primera base de datos relacional comercial del mercado. **Oracle Version 2**



A Comparison of the Relational and CODASYL Approaches to Data-Base Management

ANN S. MICHAELS

Bell Telephone Laboratories, Inc., Naperville, Illinois 60540

BENJAMIN MITTMAN

Vogelback Computing Center, Northwestern University, Evanston, Illinois 60201

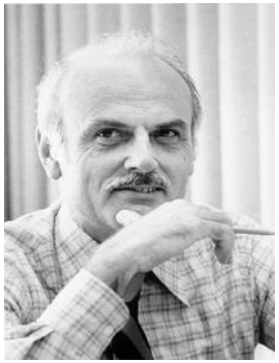
C. ROBERT CARLSON

Department of Computer Sciences, Northwestern University, Evanston, Illinois 60201

1976

Debate: Ted Codd y Charles Bachman

1975 ACM SIGMOD(Special Interest Group on Management of Data).



- Nada tan complicado como la propuesta de DBTG puede ser la forma correcta de administrar datos
- Dependencia del modelo físico
- El modelo de red no tiene fundamentos formales en teoría matemática
- Facilita la independencia entre modelo lógico y físico
- Permite consultas de alto nivel sin manejar punteros

- No se puede construir una implementación eficiente del modelo relacional
- Las aplicaciones necesitan procesar los datos de a un registro por vez
- El modelo relacional es demasiado matemático
- Preocupación por costos de procesamiento en sistemas reales
- Navegación directa entre registros facilita operaciones prácticas

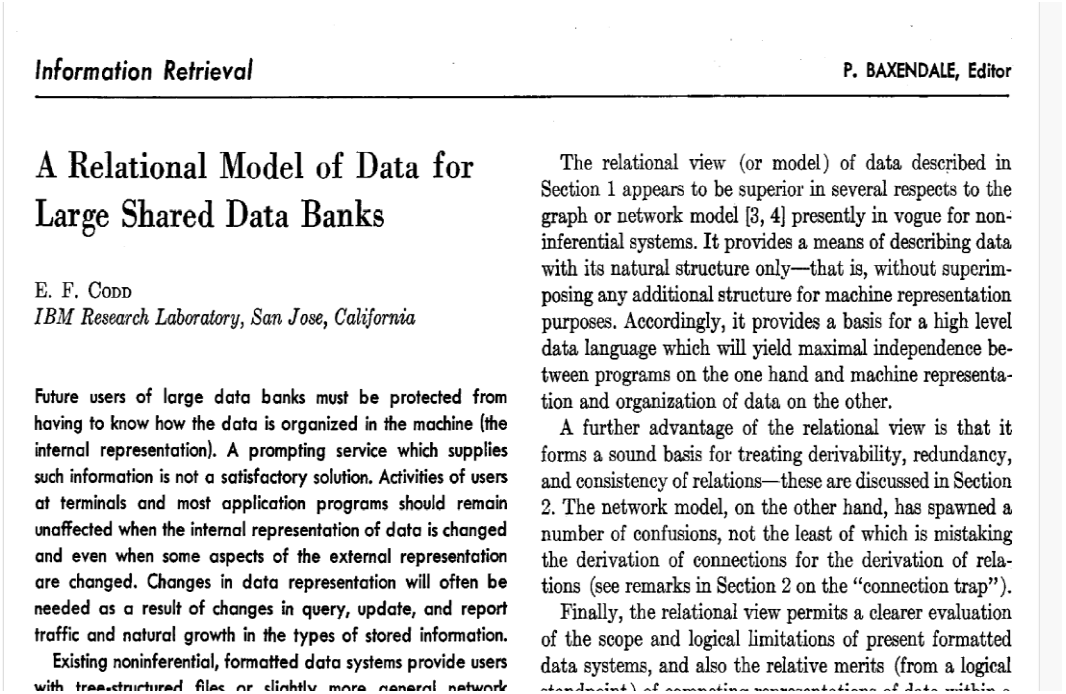
Conferencias

- ACM SIGMOD (Special Interest Group on Management of Data)
  - Fundada en 1975 dentro de la ACM.
  - Se centra en teoría y práctica de los sistemas de bases de datos.
  - Considerada la conferencia más prestigiosa del área, junto con VLDB.
- VLDB (*International Conference on Very Large Data Bases*)
  - Primera edición en 1975 en Framingham, Massachusetts.
  - Foco en bases de datos a gran escala, sistemas distribuidos y big data.
  - Publica la revista PVLDB, una de las principales del campo.
- ICDE (International Conference on Data Engineering)
  - Iniciada en 1984, organizada por IEEE.
  - Orientada a aplicaciones de ingeniería de datos, sistemas y algoritmos.
  - Es la tercera gran conferencia, junto con SIGMOD y VLDB.

Modelo Relacional - Definiciones

'A relational model of data for large shared data banks' (Codd, 1970).

- Representa a la base de datos como un conjunto de relaciones.
- Define el modelo relacional y formas no procedurales de consultar datos en dicho modelo
- Presenta el concepto matemático de relación como su construcción básica , con un fundamento teórico basado en la teoría de conjuntos y la lógica de predicados de primer orden



Information Retrieval P. BAXENDALE, Editor

A Relational Model of Data for Large Shared Data Banks

E. F. CODD
IBM Research Laboratory, San Jose, California

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

Existing noninferential, formatted data systems provide users with tree-structured files or slightly more general network

The relational view (or model) of data described in Section 1 appears to be superior in several respects to the graph or network model [3, 4] presently in vogue for non-inferential systems. It provides a means of describing data with its natural structure only—that is, without superimposing any additional structure for machine representation purposes. Accordingly, it provides a basis for a high level data language which will yield maximal independence between programs on the one hand and machine representation and organization of data on the other.

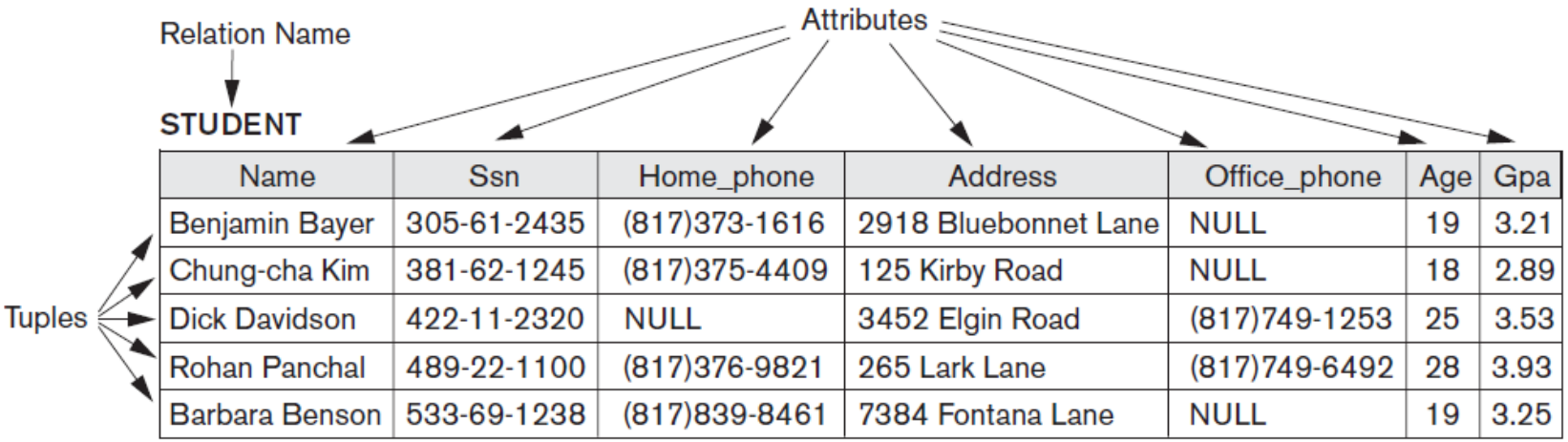
A further advantage of the relational view is that it forms a sound basis for treating derivability, redundancy, and consistency of relations—these are discussed in Section 2. The network model, on the other hand, has spawned a number of confusions, not the least of which is mistaking the derivation of connections for the derivation of relations (see remarks in Section 2 on the "connection trap").

Finally, the relational view permits a clearer evaluation of the scope and logical limitations of present formatted data systems, and also the relative merits (from a logical standpoint) of competing representations of data within a

Modelo Relacional - Definiciones

- Intuitivamente, una relación puede pensarse como una **tabla**, con **filas** y **columnas.**
  - Cada **tabla** es una relación y tiene su nombre
  - Cada columna de la tabla representa un **atributo**, asociado a un conjunto de valores posibles que puede tomar. A este conjunto es a lo que llamamos **dominio** del atributo
  - Cada fila, a la que denominaremos **tupla** está formada por un conjunto de valores de datos relacionados
  - Cada fila representa un hecho que se corresponde con una entidad o una interrelación del mundo real

Modelo Relacional



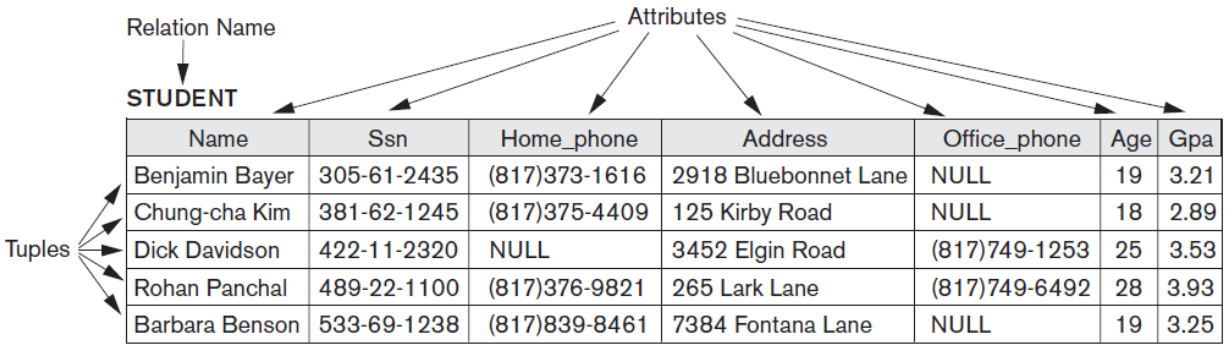
Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Modelo Relacional - Definiciones

- Un **dominio D** es un conjunto de valores atómicos. Por lo que respecta al modelo relacional, atómico significa indivisible.
- Una **relación** se tiene un **esquema** (o intención de la relación) y una **extensión** (estado o instancia)
- El esquema de la relación, que se escribe $R(A_1, A_2, ..., A_n)$ consiste en un nombre de relación **R** y un conjunto de atributos $\{\mathbf{A_1, A_2, ..., A_n}\}$.
- La **aridad** de una relación es la cantidad de atributos que tiene.
- Un atributo $\mathbf{A_i}$ es ***el nombre del rol*** que ejerce algún dominio **D** en un esquema de relación. Si D es el dominio de A_i se escribe como $\mathbf{dom(A_i)}$.

Modelo Relacional - Definiciones

- La extensión de la relación de esquema $R(A_1, A_2, \ldots, A_n)$ denotada como *r(R)* es un **conjunto de tuplas** t_i (i = 1, 2, …, m), donde cada tupla t_i se puede definir como $t_i = <v_1, v_2, \ldots, v_n>$, *donde cada* v_i *representa un valor del atributo* A_i y se cumple que cada v_j es un valor de **dom(A_j).**
  - Una interpretación alternativa seria considerar a una tupla como un conjunto de pares $t_i = \{<A_1:v_{i1}>, <A_2:v_{i2}> \ldots <A_n:v_{in}>\}$ y, para cada par $<A_j:v_{ij}>$, se cumple que v_{ij} es un valor de **dom(A_j).** En esta definición no importaría el orden.
- **Atención** con un valor especial que denominaremos nulo **(null)**.
- Notar entonces que la extensión de una relación es un subconjunto del producto cartesiano de una lista de dominios.
  - $r(R) \subseteq (dom(A_1) \times dom(A_2) \times \ldots \times dom(A_n))$



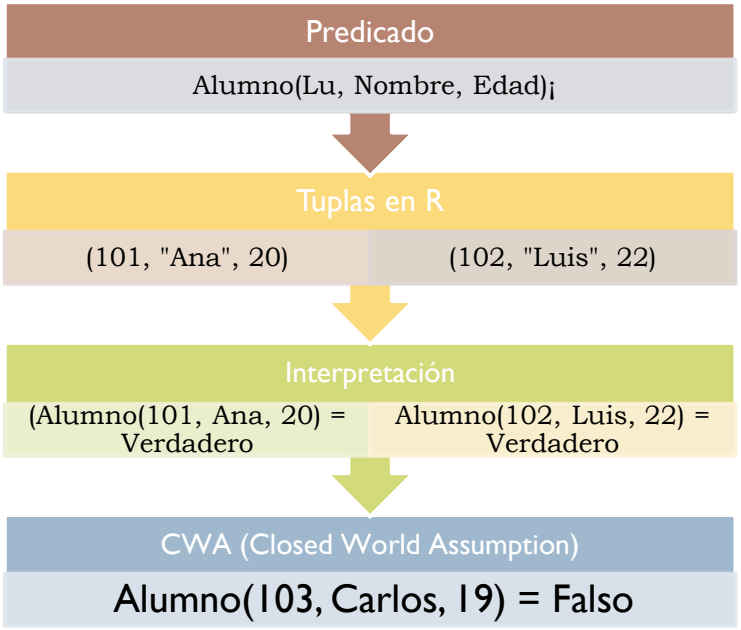
Name	Ssn	Home_phone	Address	Office_phone	Age	Gpa
Benjamin Bayer	305-61-2435	(817)373-1616	2918 Bluebonnet Lane	NULL	19	3.21
Chung-cha Kim	381-62-1245	(817)375-4409	125 Kirby Road	NULL	18	2.89
Dick Davidson	422-11-2320	NULL	3452 Elgin Road	(817)749-1253	25	3.53
Rohan Panchal	489-22-1100	(817)376-9821	265 Lark Lane	(817)749-6492	28	3.93
Barbara Benson	533-69-1238	(817)839-8461	7384 Fontana Lane	NULL	19	3.25

Esquema y Estado

- El **esquema** de una relación R representa una intención y es muy estable, sus cambios son muy infrecuentes
- El **estado** o extensión de una relación *r(R)* ***cambia muy frecuentemente*** acompañando los cambios en el mundo real.
- No hay orden entre las tuplas
  - La relación es un conjunto de tuplas, en el sentido matemático
- La lista de valores de atributos de una tupla tiene orden
- Vamos a considerar que las tuplas son una lista ordenada de valores
- De todos modos, este orden no es esencial , podría cambiarse en la medida que se haga consistentemente y se mantenga la correspondencia de atributos y valores

Modelo Relacional - Interpretación

- Una esquema de relación puede ser interpretado como un predicado y cada tupla puede interpretarse como valores que satisfacen. ese predicado.
- **CWA: closed world assumption.**



Valores en tuplas

- Cada valor de un atributo en una tupla es un valor atómico , indivisible en el contexto del modelo relacional Con esta definición, no son permitidos los atributos multivaluados ni los compuestos
- Valores NULOS
  - Se usa cuando se necesita representar que el valor de un atributo es desconocido o no aplica para una tupla en particular.
  - Es la ausencia de valor.
  - El significado concreto pueda variar según el universo de discurso representado y el modelo planteado.

Modelo Relacional - Claves

- En este modelo, aparece un concepto importante
  - SuperClave
  - Clave: conjunto minimal de atributos que definen unívocamente a las tuplas.
  - Sea K una clave, E una relación y e_i,e_j tuplas
    - $\forall e_i,e_j \in E: e_i.K = e_j.K \rightarrow e_i = e_j$
- Las relaciones pueden tener varias claves
  - *A las claves de la relación se las denomina Claves Candidatas (Candidate Keys, CK)*
  - Una de ellas será elegida como *Clave Primaria (Primary Key, PK)*
- A su vez, pueden referenciar a claves de otras relaciones
  - Se conocen como *Claves Foraneas (Foreign Keys, FK)*

Restricciones

Las tuplas de las diferentes relaciones usualmente están vinculadas de varias maneras

Las reglas de dominio del universo de discurso representado usualmente determinan restricciones sobre los valores que pueden contener las tuplas de las relaciones en una base de datos, es decir, restricciones sobre los valores actuales del estado de la base de datos completa.

Base de Datos Relacional

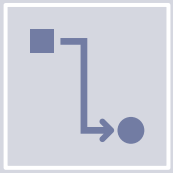
- Un **restricción de integridad (*constraint* o *integrity constraint*)** es básicamente una expression booleana que debe evaluar a Verdadero. (ej. Salario mayor a 0 para tabla Empleados)
- Un **esquema de una base de datos relacional** S es *un conjunto de esquemas de relación* $S = \{R_1, R_2, \ldots, R_m\}$ y un conjunto de **integrity constraints** IC
- Un **estado de una base de datos relacional** DB de S es un conjunto de estados de relación DB = $\{r_1, r_2, \ldots, r_m\}$ tal que cada r_i es un estado de R_i y además staisface las restricciones de integridad espeficadas en IC

Restricciones (Constraints)

- Inherentes al modelo.
- Expresadas en la definición del esquema
  - Claves, No NULL, Dominio
  - Integridad Referencial
  - Integridad de las Relaciones.
- Las restricciones de integridad (*integrity constraints)* se especifican sobre un esquema de base de datos y se espera que se cumplan sobre todo estado válido de la base de datos

Integridad Referencial

Definen restricciones entre dos relaciones

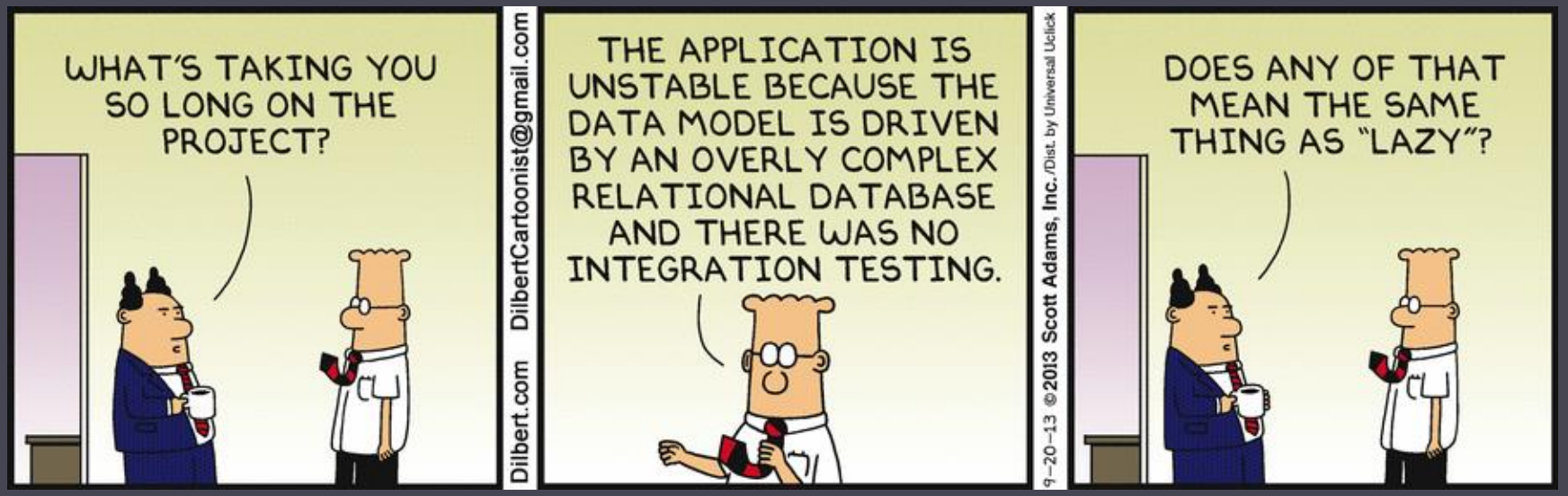


Buscan mantener consistencia entre las tuplas de ambas relaciones

Si una tupla de una relación referencia a otra relación, en la segunda debe existir una tupla que esté referenciada.

Integridad Referencial: Claves Foraneas

- Un conjunto de atributos FK en un esquema de relación $\boldsymbol{R_1}$ es una clave foránea del esquema $\boldsymbol{R_1}$ que referencia a un esquema $\boldsymbol{R_2}$ si:
  - Los atributos de FK tienen los mismos dominios que los atributos de la PK de R_2
  - Dado un valor particular de FK en una tupla t_1 de $r(R_1)$, se cumple una de las siguientes condiciones:
    - Existe una tupla t_2 en $r(R_2)$ cuya PK tiene ese mismo valor
    - El valor FK es nulo
- Decimos que R_1 referencia a R_2 (o que la clave foránea FK de R_1 referencia a la clave primaria PK de R_2)
- Las restricciones de integridad referencial típicamente derivan de interrelaciones entre entidades representadas por los esquemas de relación



Pasaje DER a MR

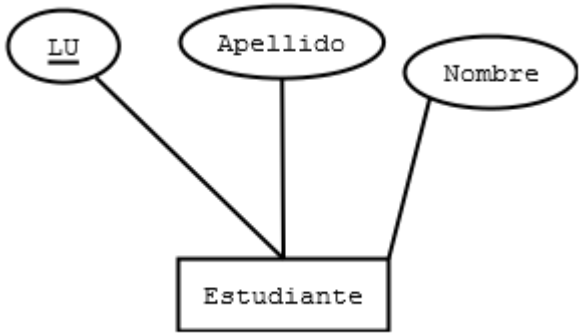
Transformación del MER al MR

En el proceso de diseño, una vez que contamos con un Modelo de Entidad Relación, debemos **generar el correspondiente Modelo Lógico Relacional** (o directamente llamado Modelo Relacional).

Transformaremos el Diagrama de Entidad Relación en un Modelo Lógico Relacional, mediante un conjunto de reglas de derivación o transformación.

Transformación del MER al MR

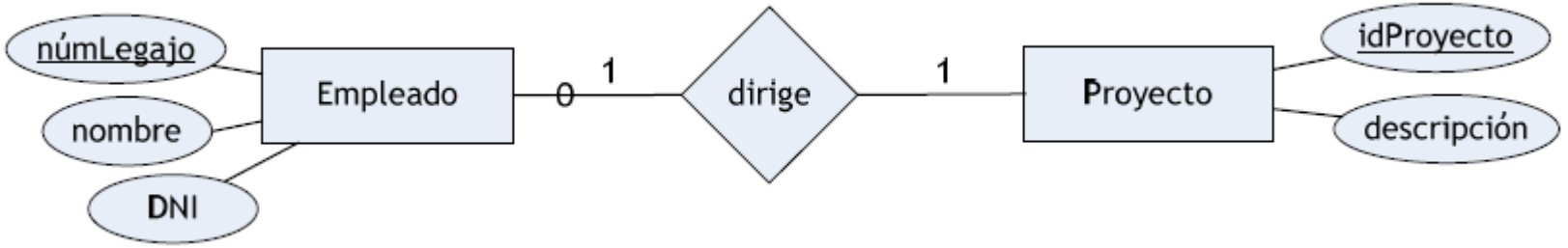
- Entidades



Estudiante(LU, Apellido, Nombre)

Transformación del MER al MR

- Interrelaciones 1 : 1



Empleado(<u>númLegajo</u>, nombre, DNI)
PK = {númLegajo}
CK ={númLegajo, DNI}

Proyecto(<u>idProyecto</u>, descripción, <u>núm</u>Legajo)
PK = CK = {idProyecto}
FK = {númLegajo}

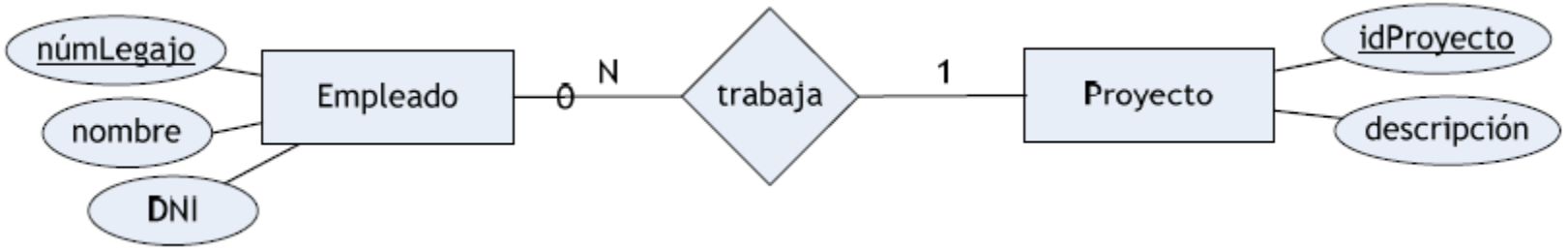
Establecemos las siguientes **restricciones adicionales:**

Empleado.númLegajo puede no estar en *Proyecto.númLegajo*

Proyecto.númLegajo debe estar en *Empleado.númLegajo*

Transformación del MER al MR

- Interrelaciones 1:N



Empleado(<u>númLegajo</u>, nombre, DNI, <u>idProyecto</u>)
PK = {númLegajo}
CK = {númLegajo, DNI}
FK = {idProyecto}
Proyecto(<u>idProyecto</u>, descripción)
PK = CK = {idProyecto}

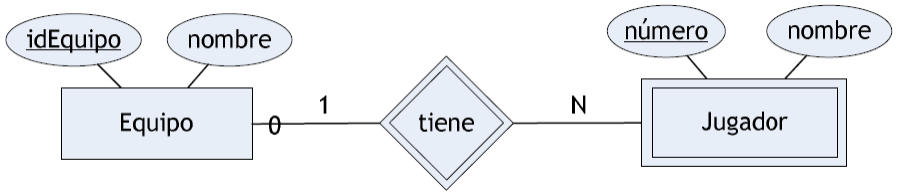
Establecemos las siguientes **restricciones adicionales**:

Empleado.idProyecto es nulo o está en *Proyecto.idProyecto*

Proyecto.idProyecto debe estar en *Empleado.idProyecto*

Transformación del MER al MR

- Entidades Débiles
  - También se transforman como relaciones
  - La transformación de atributos descriptivos sigue las mismas reglas que en las entidades fuertes.
  - La clave primaria de la relación se conforma **conjuntamente** por el **atributo identificatorio de la entidad padre** y por el **atributo identificatorio parcial de la entidad débil**



Equipo(idEquipo, nombre)

Jugador(idEquipo, número, nombre)

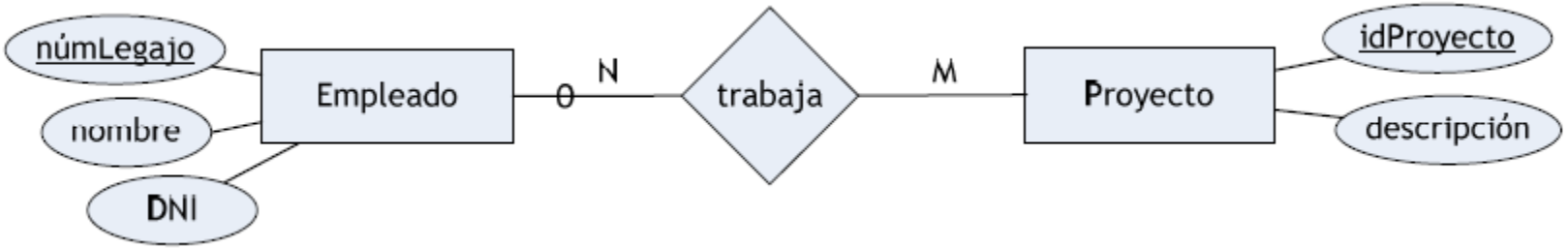
Equipo	PK= CK={idEquipo}
Jugador	PK= CK = {(idEquipo,número)}
	FK = {idEquipo}

Equipo.idEquipo puede no estar en *Jugador.idEquipo*

Jugador.idEquipo debe estar en *Equipo.idEquipo*

Transformación del MER al MR

- Interrelaciones N:M



Empleado(<u>númLegajo</u>, nombre, DNI)
PK = {númLegajo}
CK ={númLegajo, DNI}

Proyecto(<u>idProyecto</u>, descripción)
PK = CK = {idProyecto}

Trabaja(<u>númLegajo, idProyecto</u>)
PK = CK = {(númLegajo, idProyecto)}
FK = {númLegajo, idProyecto}

Establecemos las siguientes **restricciones adicionales**:

Empleado.númLegajo puede no estar en *Trabaja.númLegajo*

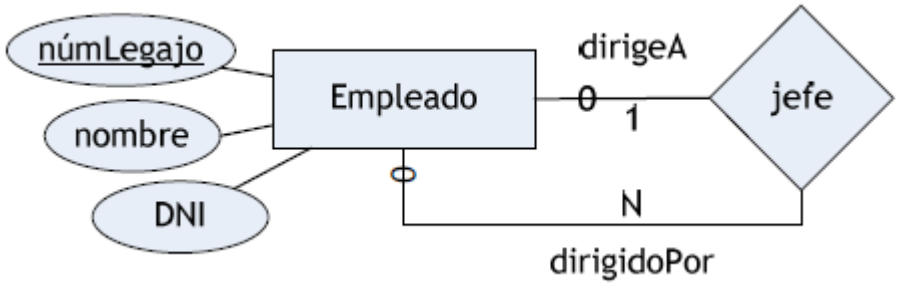
Proyecto.idProyecto debe estar en *Trabaja.idProyecto*

Trabaja.númLegajo debe estar en *Empleado.númLegajo*

Trabaja.idProyecto debe estar en *Proyecto.idProyecto*

Transformación del MER al MR

- Interrelaciones Unarias



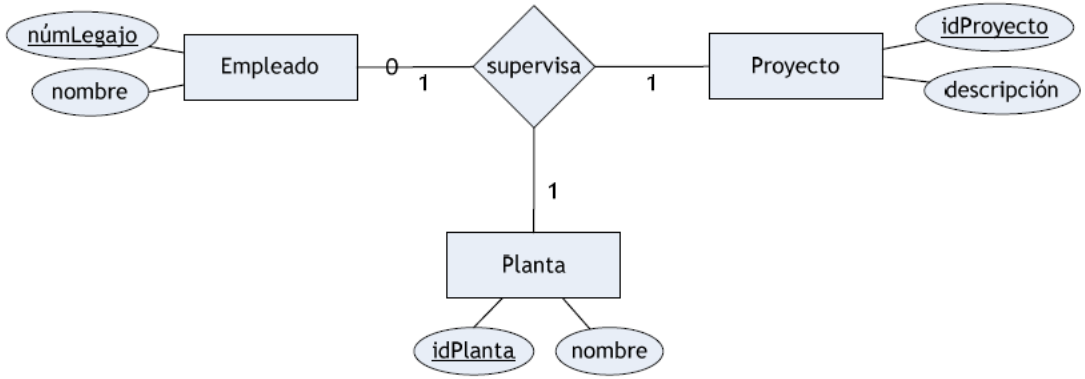
Empleado(<u>númLegajo</u>, nombre, DNI, <u>núm</u>LegajoJefe)
PK = {númLegajo}
CK = {númLegajo, DNI}
FK = {númLegajoJefe}

Empleado.númLegajoJefe puede ser nulo o debe estar en *Empleado.númLegajo*

Empleado.númLegajo puede no estar en *Empleado.númLegajoJefe*

Transformación del MER al MR

- Ternarias 1:1:1



Empleado(<u>númLegajo</u>, nombre)
PK = CK = {númLegajo}

Proyecto(<u>idProyecto</u>, descripción)
PK = CK = {idProyecto}

Planta(<u>idPlanta</u>, nombre)
PK = CK = {idPlanta}

Supervisa(<u>númLegajo, idProyecto</u>, <u>idPlanta</u>)
PK = {(númLegajo, idProyecto)}
CK = {(númLegajo, idProyecto), (númLegajo, idPlanta), (idProyecto, idPlanta)}
FK = {númLegajo, idProyecto, idPlanta}

Supervisa.númLegajo debe estar en *Empleado.númLegajo*

Supervisa.idProyecto debe estar en *Proyecto.idProyecto*

Supervisa.idPlanta debe estar en *Planta.idPlanta*

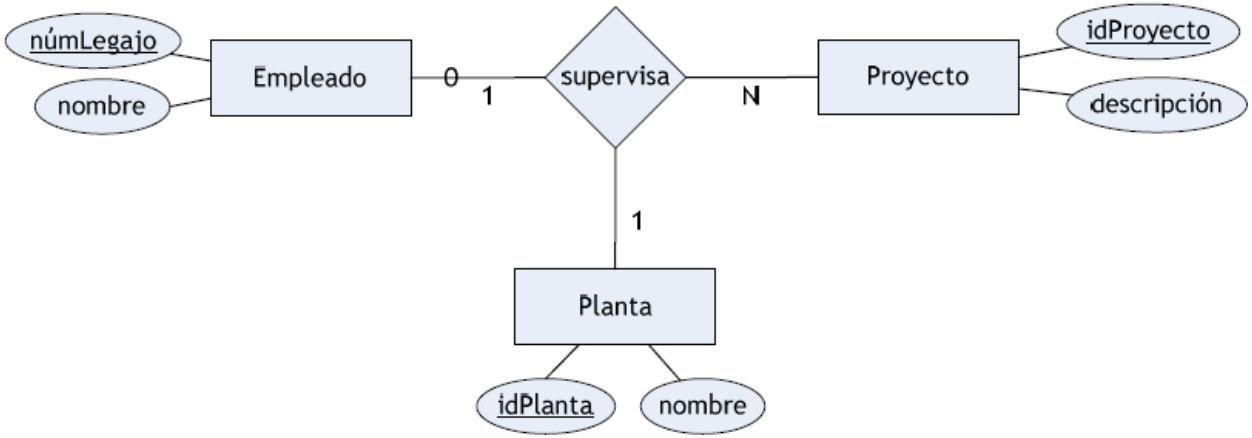
Empleado.númLegajo puede no estar en *Supervisa.númLegajo*

Proyecto.idProyecto debe estar en *Supervisa.idProyecto*

Planta.idPlanta debe estar en *Supervisa.idPlanta*

Transformación del MER al MR

- Ternarias 1:1:N
  - Cambia el problema y ahora un empleado en una planta puede supervisar varios proyectos



Empleado(<u>númLegajo</u>, nombre)
PK = CK = {númLegajo}

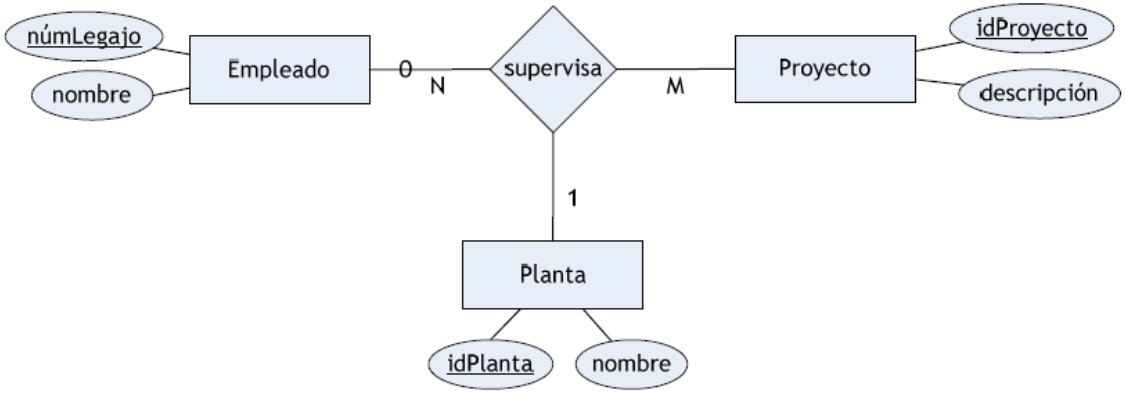
Proyecto(<u>idProyecto</u>, descripción)
PK = CK = {idProyecto}

Planta(<u>idPlanta</u>, nombre)
PK = CK = {idPlanta}

Supervisa(<u>númLegajo, idProyecto</u>, <u>idPlanta</u>)
PK = {(númLegajo, idProyecto)}
CK = {(númLegajo, idProyecto), (idProyecto, idPlanta)}
FK = {númLegajo, idProyecto, idPlanta}

Transformación del MER al MR

- Ternarias 1:N:M
  - Cambia el problema y ahora un proyecto en una planta puede ser supervisado por varios empleados



Empleado(<u>númLegajo</u>, nombre)
PK = CK = {númLegajo}

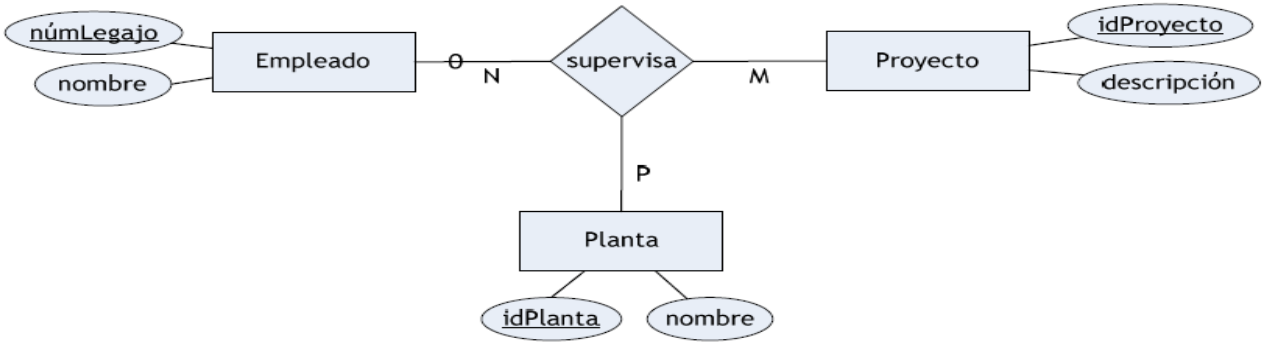
Proyecto(<u>idProyecto</u>, descripción)
PK = CK = {idProyecto}

Planta(<u>idPlanta</u>, nombre)
PK = CK = {idPlanta}

Supervisa(<u>númLegajo, idProyecto</u>, <u>idPlanta</u>)
PK = CK = {(númLegajo, idProyecto)}
FK = {númLegajo, idProyecto, idPlanta}

Transformación del MER al MR

- Ternarias N:M:P
  - Cambia el problema y ahora un empleado puede supervisar un proyecto en más de una planta.



Empleado(<u>númLegajo</u>, nombre)
PK = CK = {númLegajo}

Proyecto(<u>idProyecto</u>, descripción)
PK = CK = {idProyecto}

Planta(<u>idPlanta</u>, nombre)
PK = CK = {idPlanta}

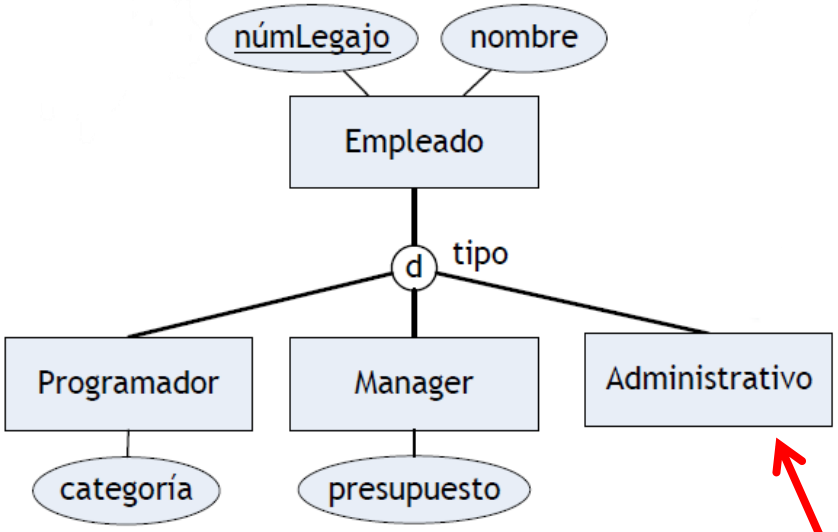
Supervisa(<u>númLegajo, idProyecto, idPlanta</u>)
PK = CK = {(númLegajo, idProyecto, idPlanta)}
FK = {númLegajo, idProyecto, idPlanta}

Transformación del MER al MR

- RESUMEN TERNARIAS
  - Siempre se genera un esquema aparte para la interrelación.
  - La clave del esquema dependerá de la cardinalidad.

Transformación del MER al MR

- Jerarquías- Disjunta



Se le agrega el discriminante como atributo, es el que permite particionar el conjunto de empleados

Empleado(<u>númLegajo</u>, nombre, tipo)
PK = CK = {númLegajo}

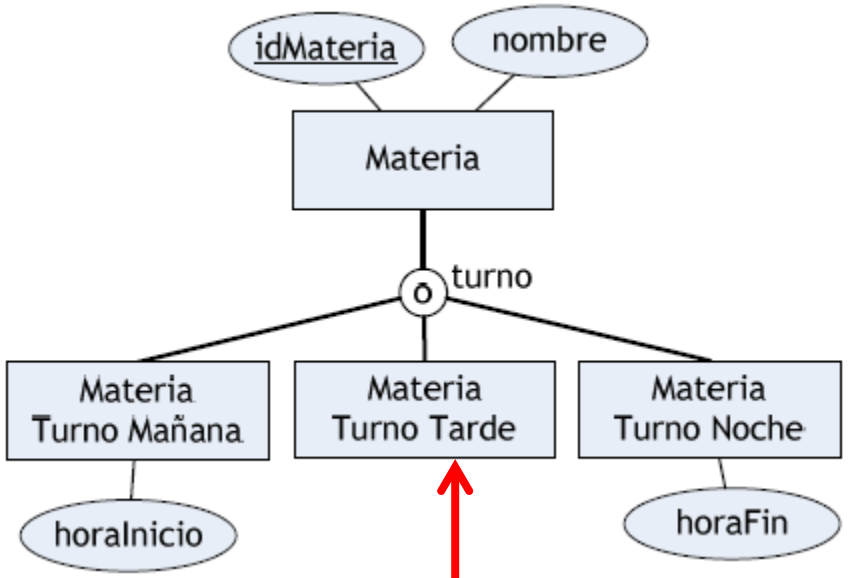
Programador(<u>númLegajo</u>, categoría)
PK = CK = FK = {númLegajo}

Manager(<u>númLegajo</u>, presupuesto)
PK = CK = FK = {númLegajo}

No tiene atributos ni relaciones no es necesario generar un esquema

Transformación del MER al MR

- Jerarquías con solapamiento



No se agrega el discriminante

Materia(<u>idMateria</u>, nombre)
PK = CK = {idMateria}

MateriaTurnoMañana(<u>idMateria</u>, horaInicio)
PK = CK = FK = {idMateria}

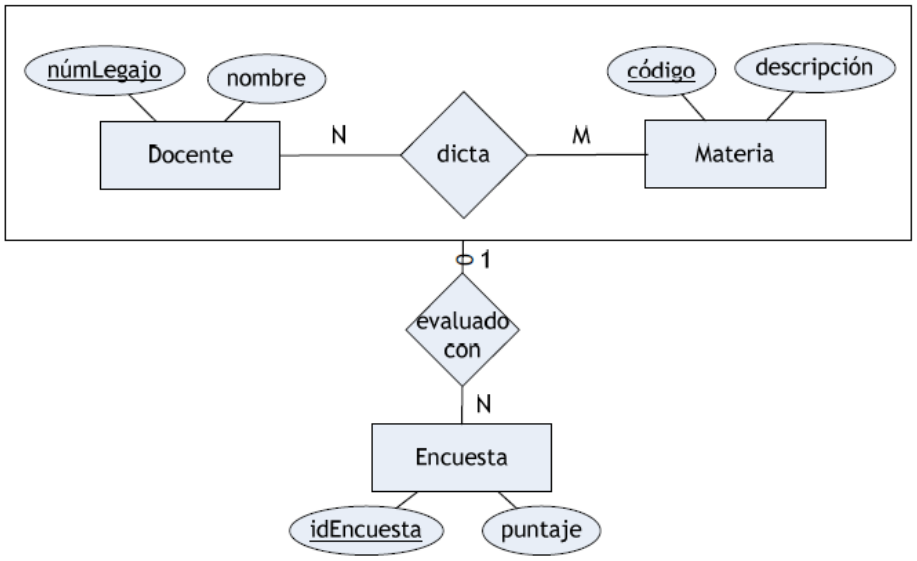
MateriaTurnoTarde(<u>idMateria</u>)
PK = CK = FK = {idMateria}

MateriaTurnoNoche(<u>idMateria</u>, horaFin)
PK = CK = FK = {idMateria}

En este caso **si es necesario** generar un esquema

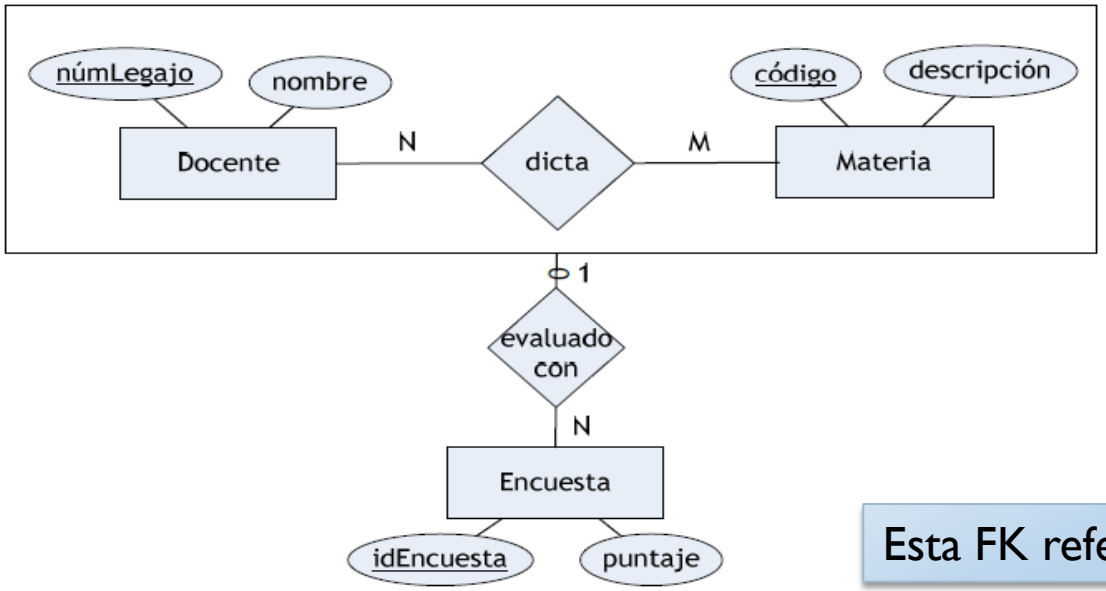
Transformación del MER al MR

- Agregación
  - En la notación de la materia sólo permitimos agregación en interrelaciones N:M.
  - Las agregaciones se transforman considerando a la agregación como si fuera una entidad.



Transformación del MER al MR

- Agregación



Esta FK referencia a Dicta

Docente(<u>númLegajo</u>, nombre)
PK = CK = {númLegajo}

Materia(<u>código</u>, descripción)
PK = CK = {código}

Dicta(<u>númLegajo</u>, <u>código</u>)
PK = CK = {(númLegajo,código)}
FK = {númLegajo, código}

Encuesta(<u>idEncuesta</u>, puntaje, <u>númLegajo, código</u>)
PK = CK = {idEncuesta}
FK = {(númLegajo, código)}

Bibliografía

- Fundamentals of Database Systems *Elmasri/Navathe 7th Ed., Addison Wesley*
- Database System Concepts. *4ta Edición Abraham Silberschatz, Henry F. Korth y S. Sudarshan*
- DatabaseManagement Systems, *Ramakrishnan/Gherke 3rd Ed.*
- A Practical Approach to Design, Implementation, and Management. *Thomas Connolly/Carolyn Begg 4ta Ed.*
- Apunte de la Materia