TEMA II
LA BASE DE DATOS RELACIONAL - RDBMS

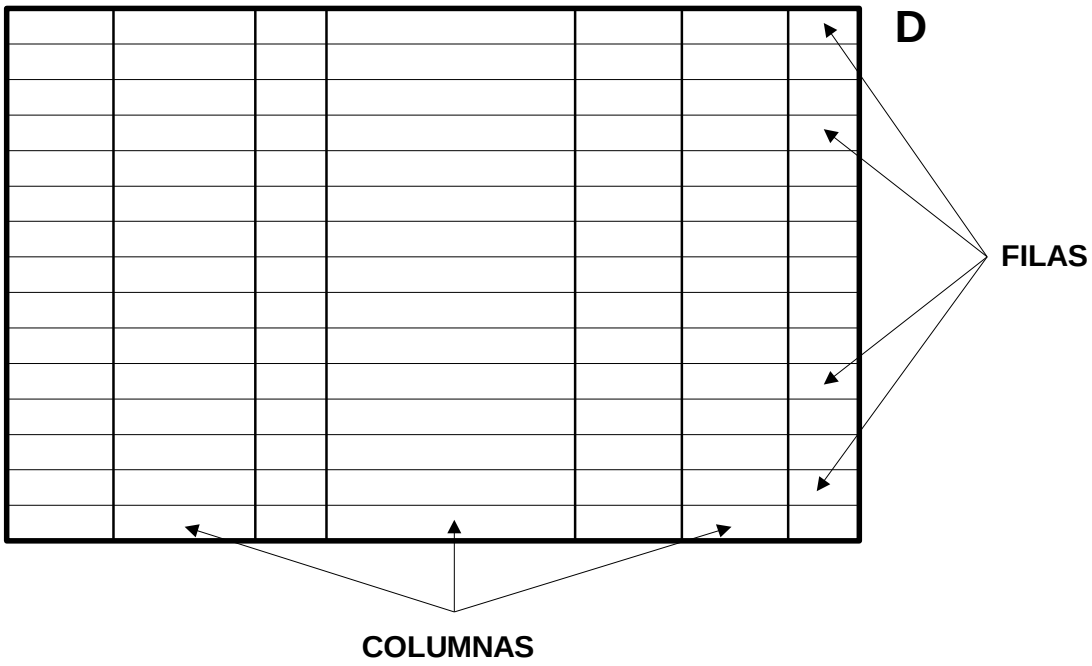
INTRODUCCIÓN

En el tema anterior, se han visto generalidades sobre bases de datos sin considerar en particular algún tipo de arquitectura de implementación. En este tema, se desarrollará la arquitectura relacional y a través de un ejemplo se mostrará el pasaje de un modelo conceptual al modelo físico de datos para el modelo relacional.

Cualquiera sea el tipo de arquitectura utilizada, siempre será necesario establecer conjuntos de relaciones entre dos o más elementos conceptuales. Se observa claramente que los elementos conceptuales si bien son abstractos, representan "algo", mientras que las relaciones no resultan tan tangibles ya que no pueden ser medidas o clasificadas con claridad como los atributos de un individuo (de una clase o una entidad). Una relación no existe como algo separado o tangible. Los datos son abstractos, pero más aun lo son las relaciones. Dentro de la computadora, los datos pueden existir como señales magnéticas o eléctricas. Pero cuando los datos se convierten en parte de una base de datos, ésta incluirá las relaciones que existen entre las entidades de datos, de modo tal que las relaciones también se deben convertir en algo concreto; vale decir que se da a la relación un nivel equivalente a los datos a los cuales se aplica. La relación se formula del mismo modo que los datos. Por lo tanto, esta relación se convierte en algo tan concreto como aquellos datos. La forma de trabajar con ambos, datos y relación, es la tabla o matriz que se desarrollará a continuación.

LA MATRIZ DE DATOS

La base de datos relacional fue creada por personas orientadas matemáticamente. Ellos deseaban obtener una visión uniforme de los datos. Todos los datos acerca de una población se ingresan en una sola tabla. Esta tabla se denomina también *matriz*. Como ejemplo de este tipo de tablas, se presenta el de la siguiente figura:



La citada tabla es bidimensional; consiste en filas y columnas:

- **Una *fila* comprende todos los datos que están en la misma línea horizontal**
- **Una *columna* comprende todos los datos dispuestos sobre la misma línea vertical**

Un dato simple, es denominado ***celda***. Una fila consiste en una cierta cantidad de celdas horizontales. Una columna es el conjunto de celdas verticales. Se puede identificar una celda, dando como dirección la fila y la columna en que se halla ubicada.

Se utilizarán letras mayúsculas en negrita como símbolo de una tabla. Se denomina **D** a la tabla o matriz de la figura anterior.

La matriz **D** es análoga a un archivo. Cada fila corresponde a un registro, consistente en un grupo de campos. Por lo tanto, un campo y una celda son conceptos comparables. La posición de una celda dentro de una fila, es importante. Se identificará qué representa dicha celda por medio de su posición dentro de la fila. Todas las filas consisten en conjuntos ordenados de celdas. Por otro lado, la posición de una fila dentro de una matriz, no es importante; dos matrices se denominan equivalentes, siempre que consistan en las mismas filas, independientemente del orden en el cual éstas aparezcan en cada una de las matrices.

Relación uno a uno

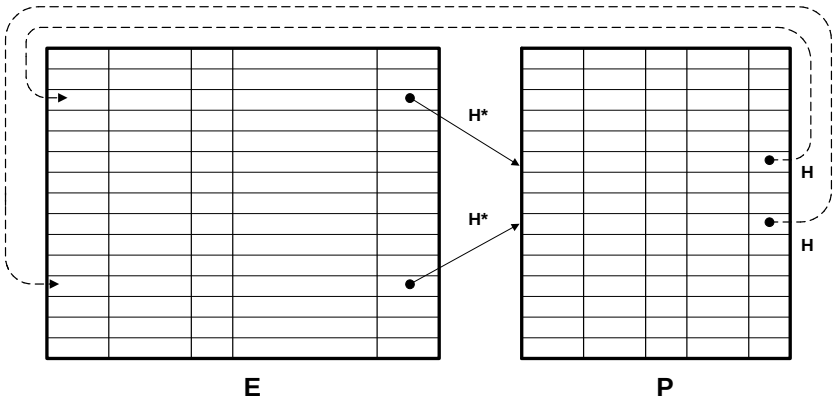
Se considerarán ahora dos poblaciones. Cada una de ellas, está representada por su propia matriz. Para una forma de relación uno a uno, se trata de un problema simple, ya que para cada fila de una matriz, existirá una fila correspondiente en la otra matriz, y todo lo que se tiene que hacer, es establecer punteros entre las filas correspondientes.

Ejemplo:

Se considerarán las poblaciones de ***estudiante*** y ***persona***. La población ***estudiante*** es un subconjunto de ***persona***. El vínculo que mantienen es a través de una relación de funcionalidad 1-1. En la siguiente figura, puede verse que la población de **personas**, se halla representada por la matriz **P** de la derecha. Los estudiantes están representados por la matriz **E** de la izquierda. Para relacionar un estudiante con una persona, se usará la forma de relación **H***. Por lo tanto, **H*** relaciona una fila de **E** con una fila diferente de **P**. Esta asociación resulta en un puntero desde una fila de una población hacia otra fila en la otra población.

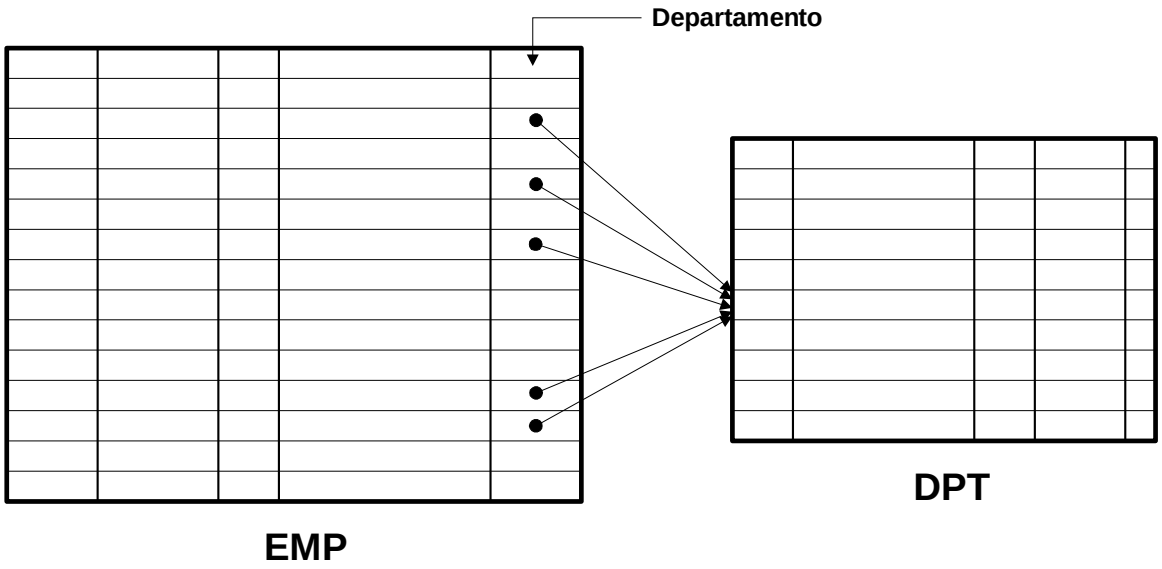
Hay una diferencia definida en los medios por los cuales se asocia una fila con otra; se trata de algo simbólico más que físico. Cada *estudiante* (fila), contiene una celda destinada a almacenar la *persona* de su identidad. Esta representación de la persona, es por ejemplo el documento (clave). Para encontrar esta fila, se debe buscar en la matriz de personas, entrando por la clave que identifica un individuo en particular.

Puede ser necesario proveer un efecto recíproco, vale decir, dado una persona, encontrar los datos del estudiante (si es que la persona lo es). Esto se lleva a cabo de la misma manera. Se necesita ahora que cada fila de **P** contenga un número de estudiante (clave) que podría ser el número de libreta universitaria. Este número de libreta universitaria permite encontrar la fila que describe al estudiante que es esa persona. Una instancia de dicha relación **H** se ilustra mediante la línea de trazos de la figura.



Muchos a uno

La relación muchos a uno, puede tratarse de la misma manera. Considérese un empleado asociado a un departamento en particular. Es posible construir dos matrices, una para la población **empleado** y otra para la población **departamento**. En el siguiente ejemplo, se llamarán **EMP** y **DPT** respectivamente. Esto puede observarse en la figura siguiente:



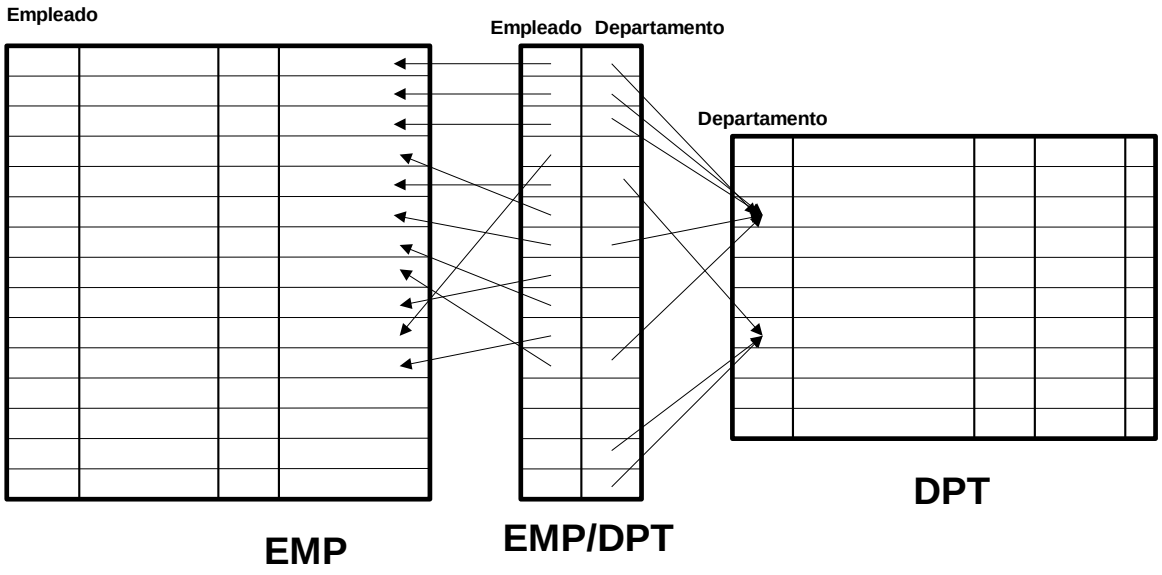
Varios empleados se encuentran asociados exactamente con un departamento; por lo tanto, es posible colocar en cada fila de empleados, un número de departamento. Varias filas tendrán el mismo número de departamento como se observa en el gráfico. Todos estos empleados trabajan o están asociados al mismo departamento. Esta disposición tiene la ventaja de que sólo existe un número de departamento y de que la información del departamento no se repite para cada empleado. Solamente como comentario se acota que esta estructura de tablas no permitiría llevar un registro histórico de empleados, vale decir, la lista de todos los departamentos por los que pasa un empleado a lo largo de su carrera administrativa, pues, para ello, debería haber tantas columnas en la tabla **EMP** como departamentos distintos hayan sido asignado cada uno de los empleados. Puede presentarse el caso de que un empleado haya pasado tan sólo por un departamento, y otros por **n** departamentos distintos; consecuentemente el escenario mencionado constituye otro tipo de funcionalidad en la relación.

Uno a Muchos

Ya se ha visto cómo encontrar la descripción del departamento siempre que se haya localizado a un empleado. Para invertir esta situación, dado un departamento, se puede desear encontrar todos los empleados que trabajan en el mismo. No es posible llevar a cabo esto con dos tablas si se desea mantener un número fijo de filas y columnas en cada una de ellas. Se hace la salvedad que, a través de una aplicación, puede recorrerse la tabla de empleados y levantar aquellas filas en las que la celda que indica el departamento en el que trabaja, coincida con el seleccionado. Existe otra alternativa: **crear otra tabla** cuyo sólo propósito sea **asociar** empleados con departamentos, y se la denominará **Matriz Relacional**.

El uso de una matriz adicional para indicar la relación que existe entre dos tipos de individuos, se ilustra en la siguiente figura. Nuevamente se tienen las dos matrices, **EMP** y **DPT** para empleados y departamentos. La matriz agregada **EMP/DPT** contiene solamente dos columnas, una para el número clave del empleado y otra para el número clave del departamento. Cada número de departamento puede aparecer en varias filas diferentes, mientras que cada número de empleado es único.

Para encontrar en qué departamento trabaja un empleado, se necesitará consultar solamente **EMP/DPT**. Inversamente, si se desea encontrar los empleados asignados a un departamento, también se puede consultar esta tabla. Para obtener información combinada acerca de un empleado, se debe entrar en **EMP/DPT** para localizar la descripción del departamento y su propia descripción identificando las filas correspondientes en **EMP** y **DPT**. Se pueden usar estas tablas de un modo similar para clarificar la información acerca de todos los miembros del departamento o bien para hacer más eficientes las búsquedas en términos de tiempo.



Nótese que existen tantas filas de la matriz relacional, como filas hay en la tabla de empleados. Esto resulta lógico, pues todos los empleados deben estar relacionados con un departamento.

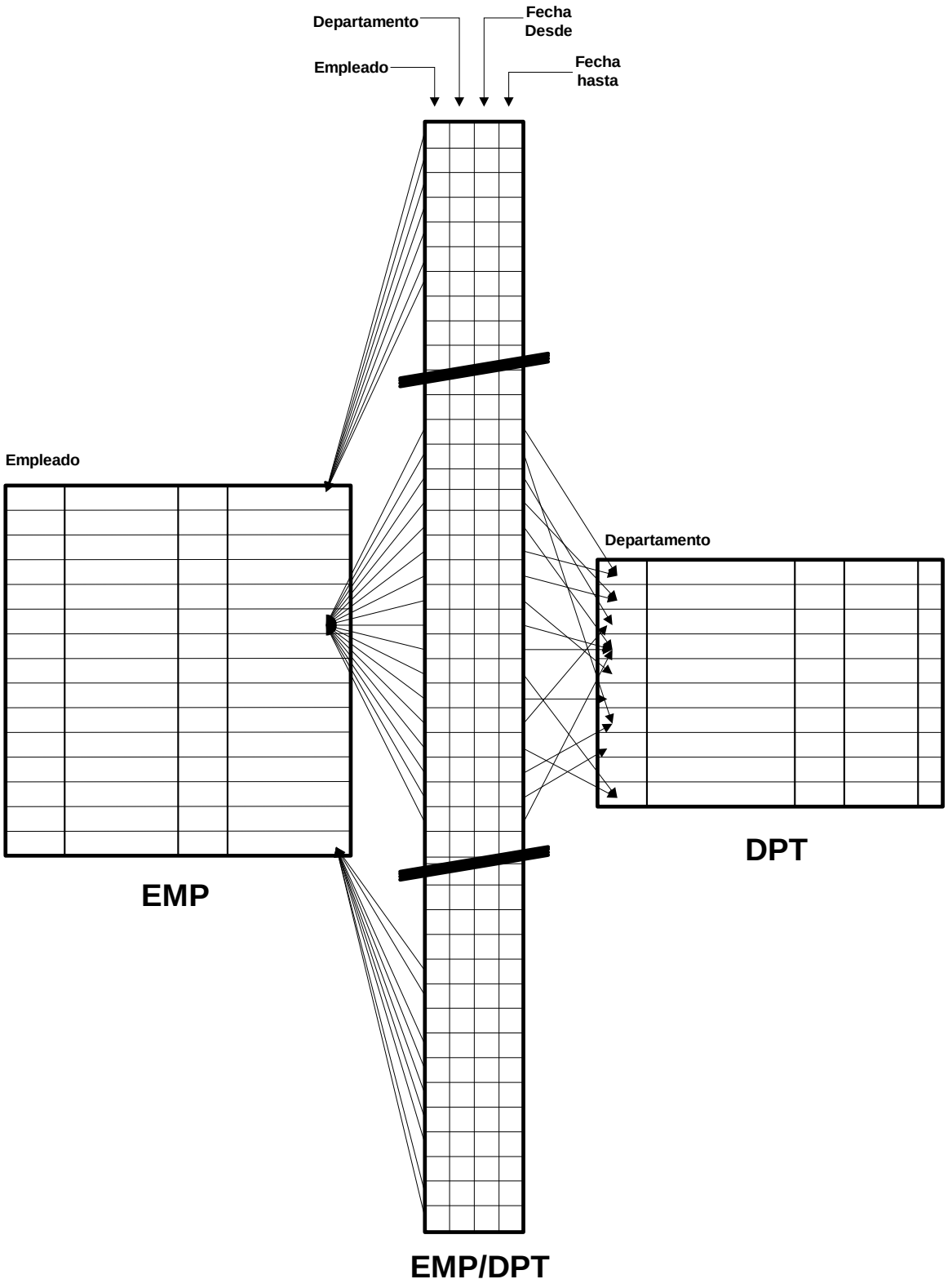
Muchos a Muchos

Resulta relativamente fácil extender este concepto, a partir de la relación uno a muchos a la relación muchos a muchos. Una tabla de relación como esta, sigue siendo necesaria para relacionar dos tablas de datos. Considérese ahora, que se desea tener en cuenta la historia del empleado en lo que respecta a las distintas reparticiones por las que fue pasando a lo largo de su carrera. Ello conduce a una relación de muchos a muchos, puesto que un empleado por ejemplo, puede haber recorrido varios departamentos y aun puede repetirse su situación pero para dos fechas diferentes, vale decir que la fecha formará parte del identificativo.

En el gráfico de la siguiente página, puede observarse la manera de resolver tal caso. Nótese que para una fila de la tabla **EMP** puede haber varias en la tabla de relación **EMP/DPT**, y que ocurre la misma situación haciendo referencia a la tabla **DPT**.

A manera de acotación, si se desea obtener la lista de los empleados que pertenecen a un departamento en la actualidad, puede recurrirse a la tabla de relación y obtener el subconjunto de

las filas en que figure el departamento y que además, la celda que representa la *fecha hasta* (fecha de baja del empleado en ese departamento) no contenga ningún valor. Asimismo, si la lista deseada es la historia de un empleado en particular, puede recurrirse a la tabla, y consultar mediante el número del empleado, el subconjunto de filas en las que coincida el número del empleado buscado sin importar ninguna otra columna. Obviamente, en ambos casos, tanto para determinar los datos del empleado y los del departamento, debe recurrirse a las tablas **EMP** y **DPT**.


Departamento
Fecha
Desde
Empleado
Fecha
hasta
Empleado
Departamento
EMP
DPT
EMP/DPT

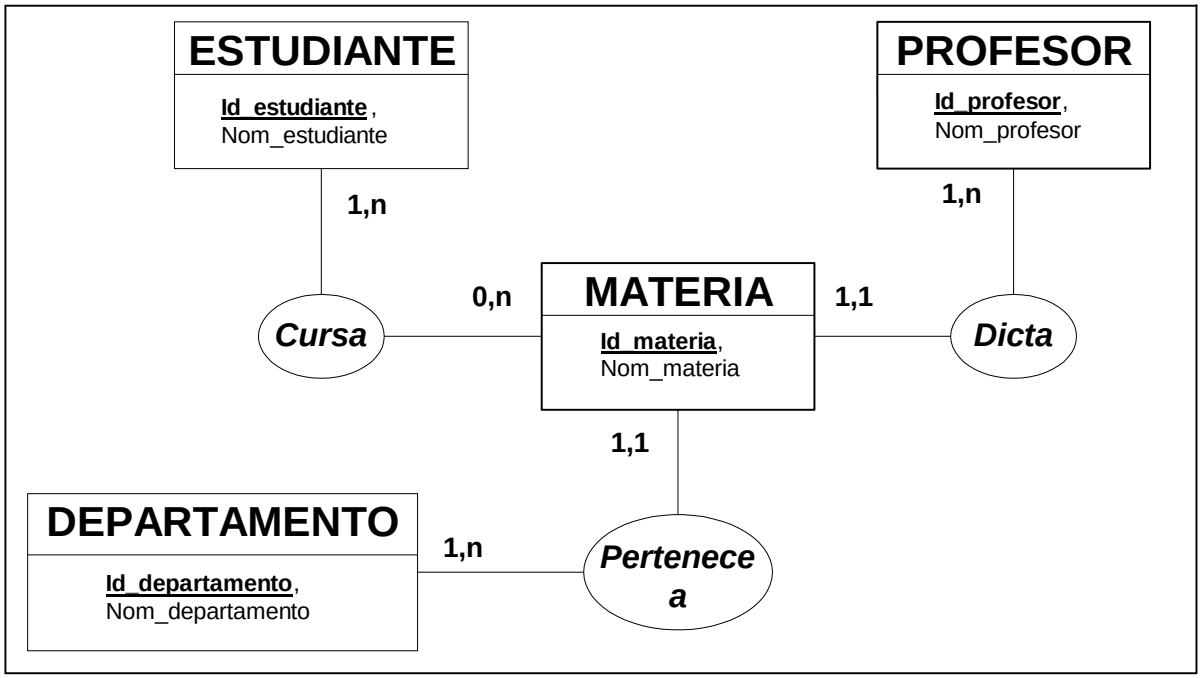
La Base de Datos *"Alumnado"*

Con el propósito de ejemplificar una base de datos simple, se considerará un sistema de alumnado de cierta facultad. Se definirá como objetivo principal, el dictado de clases a los estudiantes. Las clases son por lo tanto el elemento de mayor importancia. Se hará una serie de simplificaciones iniciales dadas por las siguientes reglas de gestión:

- Un PROFESOR dicta al menos una MATERIA
- Una MATERIA es dictada por uno y solamente un PROFESOR
- Un ALUMNO puede estar inscripto y cursar varias MATERIAs
- Una MATERIA depende de uno y solamente un DEPARTAMENTO

Modelo conceptual de datos de *"Alumnado"*

De acuerdo a las reglas de gestión, MCD correspondiente es:



El Registro Compuesto

Una forma directa de construir una base de datos, es generar un registro compuesto. El registro compuesto, tiene toda la información con respecto a cada conjunto existente de formas de relación que afecten a todas las variables - estudiantes, profesores, materias y departamentos-. La cuestión es cómo hacer esto de modo de obtener un agrupamiento significativo.

Para un registro compuesto, se necesita una forma de relación compuesta; para el ejemplo, esta asociación existe entre un estudiante y un profesor dentro de una única materia. Una asociación múltiple de este tipo, es un "individuo" en el contexto de un registro compuesto. Debe tenerse en cuenta que no se trata simplemente de alguna asociación entre un estudiante y un profesor; un estudiante pue-

de cursar dos materias dictadas por el mismo profesor, tratándose en este caso de dos asociaciones distintas. Cada tupla de esta forma de relación, es un individuo que tiene un registro para describirlo.

La figura siguiente, permite observar porciones de un registro típico de esta población. Debe recordarse que se desea tener un registro para cada asociación, que describa los individuos que toman parte en dicha asociación; esto vale para sus muchos campos. Tal como se plantea el problema, no parece que se tenga una clave única para este registro: ninguno de lo participantes - estudiantes, materias o profesor -, provee unicidad para esta asociación. Existen dos formas para resolver el problema:

- Utilizar un registro compuesto consistente en una identificación concatenada con elementos significativos de materia y estudiante
- Asignar al estudiante un número único distinto para cada una de las materias que cursa. En la figura, esto aparece como una matrícula ID estudiante-materia, denotando la clave compuesta

Por otra parte, se necesita también una identificación única para cada materia, ID materia, que es el atributo que la identifica. Éste será distinto para cada descripción de materia. La combinación a adoptar en referencia a la relación con los estudiantes que la cursan, podría consistir en la matrícula Id_ materia, más el número ordinal del estudiante según su inscripción en tal materia. Por otra parte, y en virtud de ser estudiante, un elemento fundamental y obviamente una entidad, deberá existir alguna manera de identificarlo unívocamente, y no a partir del número ordinal de inscripción en una materia, que dependiendo de ésta última, cambiará. El identificativo adoptado para el estudiante será Id_estudiante. Algo similar, ocurrirá con los profesores y con los departamentos.

ID Estudiante Materia	ID Materia	Nombre Materia	ID Departamento	Nombre Departamento	ID Estudiante	Nombre Estudiante	ID Profesor	Nombre Profesor

Para identificar al estudiante, se puede utilizar una codificación particular de la Facultad. Puede usarse el tipo y número de documento, número de libreta universitaria, etc. Para identificar al profesor, también debe recurrirse a alguna forma de clave.

Puede observarse que este registro compuesto, tiene toda la información que se puede necesitar para muchas aplicaciones. Ese precisamente es el problema, hay demasiada información repetitiva. La descripción de la materia, se encuentra repetida para cada estudiante. Lo mismo ocurre con la descripción del profesor y con la de los departamentos. Esto no resulta eficiente en términos de espacio.

Descomposición

Partiendo de un registro compuesto, se puede ahorrar mucho espacio conservando los datos sobre profesores, estudiante, materias y departamentos en conjuntos y separados. Si se procede de

este modo, es necesario encontrar una forma de recolectar los datos más tarde, para obtener nuevamente el registro compuesto o un registro menor. Por ejemplo, el encargado de confeccionar las listas de alumnos regulares o en condiciones de cursar, o bien de hacer las actas de examen, puede desear saber solamente cuántos estudiantes están inscriptos en una determinada materia, y qué profesor imparte la misma. Todo lo que necesita, son los nombres de los estudiantes y del profesor y/o los números, y no todo el conjunto de información que también estaría disponible. Análogamente, para efectuar una interrogación acerca de qué materias está cursando un estudiante, el registro compuesto no solamente es ineficiente con respecto al espacio de almacenamiento, sino también con respecto a una recuperación rápida.

La base de datos relacional de *"Alumnado"*

El elemento básico de una base de datos relacional (BDR) es denominado tupla, forma abreviada de n-tupla, y es definida como un conjunto ordenado de n valores. Existe una tupla para cada individuo de interés en cada población de aplicaciones.

El conjunto de tuplas que describe a la población se define como la relación. Si bien se ha utilizado el término "relación" para describir cómo dos individuos actúan o están situados uno con respecto al otro, se trata aquí de un contexto diferente; no debe surgir confusión entre estos dos usos del mismo término. La relación para la BDR es una colección de tuplas, y será representado por una matriz. Existe una colección particular correspondiente a la población de aplicación. Cada tupla, se identifica para correlacionarla con un individuo. El componente o los componentes necesarios en cada tupla para este propósito, constituyen la clave primaria. Si bien las tuplas tienen una clave, la colección de tuplas, no es necesariamente un conjunto ordenado. Dos relaciones pueden verse como idénticas si contienen las mismas tuplas, independientemente del orden en el cual éstas aparezcan.

Una relación lleva por sí misma a la normalización -la base para la descomposición- o sea a la conversión de una relación en distintas sub-relaciones, las cuales tomadas en conjunto, preservan el contenido de información de la relación original pero en un formato más eficiente. Una base de datos relacional simple, es por lo tanto, un agrupamiento de sub-relaciones, obtenido a partir de la relación original, y tal cual se requiere para una sola aplicación o conjunto de aplicaciones. Una base de datos relacional, es una colección de estas relaciones simples.

La tabla de relación

Una relación es presentada a menudo, para la comunicación entre usuarios, como una tabla. Una tabla de relación de este tipo para el ejemplo, se muestra en la figura siguiente.

Id Materia	Nombre Materia	Id Depto.	Nombre Depto.	Id Profesor	Nombre Profesor	Id Estudiante	Nombre Estudiante
10725	Ing. de Software	100	Sistemas	1	CERI, José	666	SCRUTTI, Claudio
						127	MORETTI, A. R.
						…..	…..
						25466	ROSSI, Fabiana C.
10726	Bases de datos	100	Sistemas	2	CERI, José	1433	SANTO, Gabriel M.

			25466	ROSSI, Fabiana c.
			…..	…..
			3155	DEYRO, B.M.W.

La tabla se descompone horizontalmente en filas y verticalmente en columnas. Cada fila contiene una tupla, la cual corresponde a un registro. Cada una de las columnas, representa un atributo. Cuando se observa una tupla correspondiente a un individuo, el contenido de cada componente representa un valor de atributo que califica a ese individuo.

La forma usual de utilizar la base de datos relacional, es en primer término, armar la matriz compuesta, una tabla que contiene toda la información requerida para la aplicación. Cada fila, contiene una tupla compuesta. Esta tabla única, es ineficiente y repetitiva, como se ha visto en el Ingeniería de Software I, por lo que se hace necesario normalizarla para simplificar y descomponer la tabla en varias tablas más simples. De acuerdo a lo especificado inicialmente, el identificativo de cada ocurrencia de la relación establecida entre el estudiante y la materia que cursa estará dada según la siguiente tabla:

Id Materia Estudiante	Id Materia	Nombre Materia	Id Depto.	Nombre Depto.	Id Profesor	Nombre Profesor	Id Estudiante	Nombre Estudiante
10725-1	10725	Ing. de Software	100	Sistemas	1	CERI, José	666	SCRUTTI, Claudio
10725-2	10725	Ing. de Software	100	Sistemas	1	CERI, José	127	MORETTI, A. R.
…..	10725	Ing. de Software	100	Sistemas	1	CERI, José	…..	…..
10725-n	10725	Ing. de Software	100	Sistemas	1	CERI, José	25466	ROSSI, Fabiana C.
10726-1	10726	Bases de datos	100	Sistemas	2	CERI, José	1433	SANTO, Gabriel M.
10726-2	10726	Bases de datos	100	Sistemas	2	CERI, José	25466	ROSSI, Fabiana c.
…..	10726	Bases de datos	100	Sistemas	2	CERI, José	…..	…..
10726-n	10726	Bases de datos	100	Sistemas	2	CERI, José	3155	DEYRO, B.M.W.

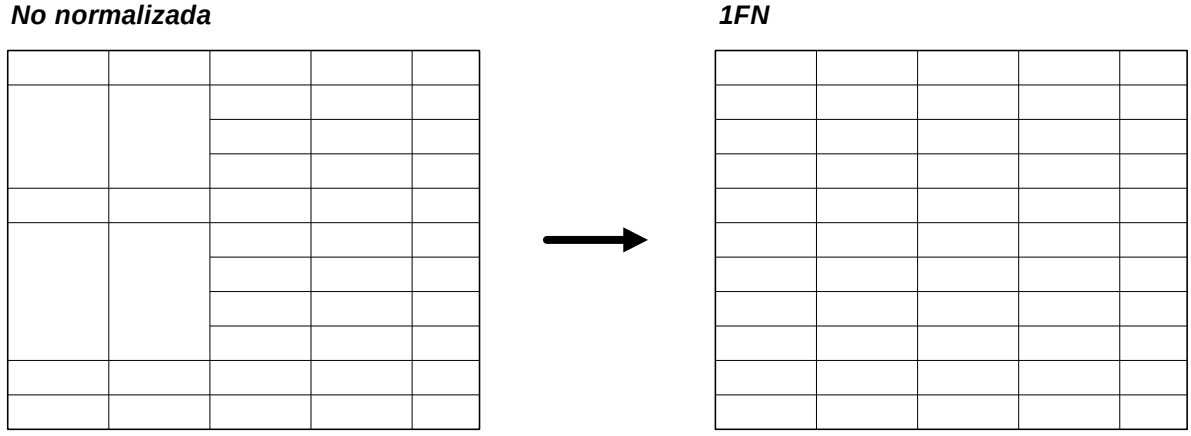
Subtuplas

Para simplificar aún más el modelo, se omitirá por ahora la presencia de la entidad **Departamento**, considerándose solamente las restantes. Se definirá ahora una subtupla como uno o más componentes de una tupla compuesta, asociados entre sí de algún modo funcional. En la figura anterior se distinguen tres subtuplas separadas por líneas verticales gruesas. Cada subtupla corresponde a una de las entidades de la aplicación; existen subtuplas para materia, estudiantes y profesores.

Subsiste un campo que es único y que por lo tanto, sirve como clave. Se trata de la matrícula **ID Materia - Estudiante**, un número compuesto en la figura; la primera parte es el número que identifica unívocamente a cada materia; la segunda parte es el número ordinal del estudiante dentro de la clase. El segundo número puede ser asignado de acuerdo con el orden en el que se han registrado los estudiantes. Por lo tanto, la matrícula 10725-1, corresponde a la materia identificada como 10725, y al estudiante que se ha registrado primero en dicha materia.

En la figura, los datos acerca de la materia así como acerca del profesor, permanecen constantes en todas las filas que pertenecen a una misma materia. Existe una fila diferente que describe a cada estudiante. En una tabla es común dejar espacios vacíos en las filas correspondientes a subtuplas duplicadas; la primera fila contiene la subtupla a ser duplicada en las filas en blanco que siguen debajo, como se muestra en la primer figura. Éste se repite sistemáticamente en la última figura.

Una tabla con partes de sus filas en blanco, como la de la izquierda, obviamente es no normalizada. En las técnicas de las bases de datos relacionales, es un requerimiento que todas las tablas, y por lo tanto todas las matrices, tengan valores en cada componente de cada tupla.



La tabla no normalizada de la izquierda, ha sido convertida en primera forma normal (**1FN**) una vez que todos los campos en todas las filas contengan un valor. Se dice que la tabla MEP (Materia - Estudiante - Profesor) está en su primera forma normal, ya que cumple con este requerimiento. Un requerimiento adicional de la primera forma normal, es que toda fila, tenga una única clave identificadora. Para el ejemplo, se trata en este caso de la matrícula ID materia-estudiante. Es posible elegir una clave compuesta a partir de atributos existentes sin necesidad de una clave sintética e independiente. Para poner esto más claro, considérese que una única clave para cada fila, podría consistir en una combinación de número de matrícula y número de estudiante. Se ha utilizado un número de matrícula de cinco dígitos; si el número de estudiante es el número de documento, se trataría de ocho dígitos adicionales. Más aun, no se tendría información del "orden de llegada" con respecto al estudiante (si realmente es de interés).

Descomposición

El primer paso en la descomposición, es identificar las subtuplas apropiadas. Dado que el ejemplo, incluye materias, profesores y estudiantes, deben construirse subtuplas adecuadas para cada una de estas entidades, como se ve en la siguiente figura. Se puede observar el conjunto original en su primera forma normal como la tabla MEP. Las líneas horizontales gruesas, separan las tuplas de una clase con respecto a las tuplas de la clase siguiente. Las líneas verticales, definen las subtuplas. Las subtuplas han sido divididas a su vez en componentes, pero las columnas no tienen denominación alguna.

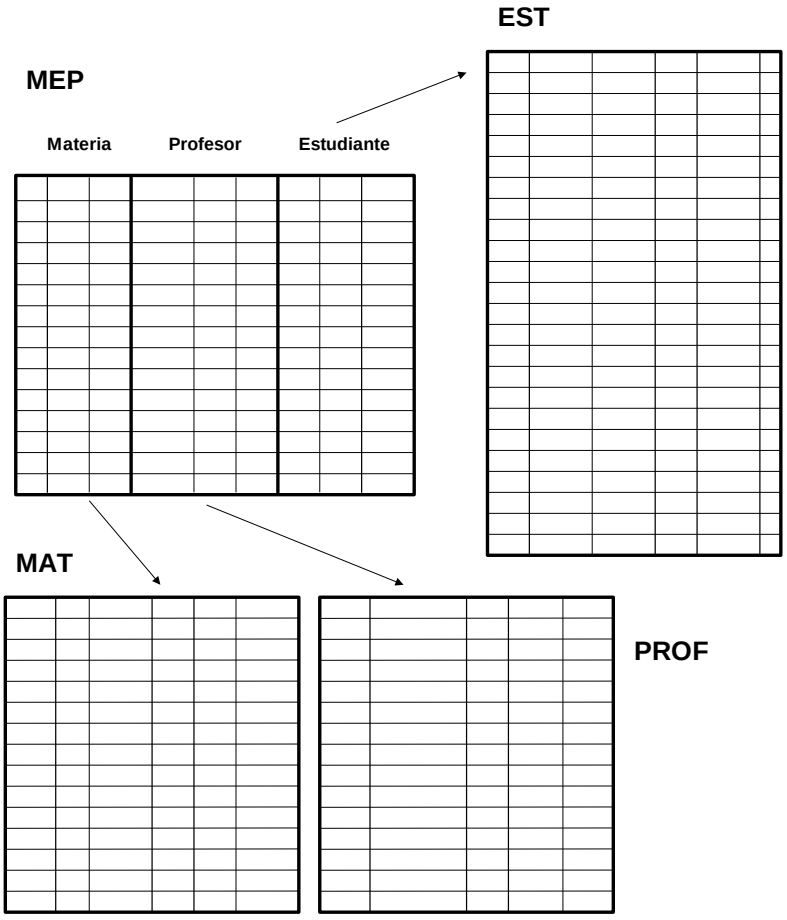
A partir de la tabla MEP se puede ahora obtener tres tablas más pequeñas cortando dichas tablas a lo largo de las líneas verticales gruesas. Antes de hacerlo, se debe asegurar que cada tabla tenga

su columna clave, con un atributo que pueda identificar en forma unívoca a cada fila. Esto da lugar a tres tablas:

- La tabla de Materias **MAT**
- La tabla de Profesores **PROF**
- La tabla de Estudiantes **EST**

Como ya se ha explicado, cada tabla puede tener muchas filas duplicadas; éstas deben ser eliminadas. Cuando no existan duplicaciones, las tablas revisadas, aparecerán en la forma de figura siguiente, representando las nuevas relaciones:

- **MAT** contiene una tupla para cada Materia identificable
- **PROF** contiene una tupla para cada Profesor que dicta clases
- **EST** contiene una tupla para cada Estudiante inscripto por lo menos en una materia.



Otra manera de describir la creación de la subtupla es la siguiente. Barriendo la tabla MEP, descomponer cada fila en subtuplas; distribuir las mismas en tres subtablas y eliminar cualquier subtupla para la cual exista un duplicado en la tabla.

Los requerimientos para la subtabla prevalecen respecto de la tabla original:

- **Cada nueva tabla está en la primera forma normal**
- **Cada tupla tiene una clave**

La eficiencia ganada en espacio da como resultado menores requerimientos de éste, y menor número de columnas para cada sub-relación, así que:

- Cada sub-relación tiene menor número de columnas
- Cada relación tiene por lo general menos filas, dado que el número de entidades para cada sub-relación es a menudo menor que el de la relación original

La tabla de estudiantes **EST** es la mayor y consiste en tantas filas como estudiantes haya. Esto contrasta con MEP que contiene una fila por estudiante y por materia que éste curse, lo cual es necesariamente mucho mayor.

Relacionando las subtablas

La relación original MEP relacionaba las tres entidades: Materia, Estudiante y Profesor. **MAT**, **EST** y **PROF** proveen toda la información previa excepto por el hecho de que no indican qué estudiantes cursan cada materia, ni qué profesor enseña en cada materia por lo que obviamente, falta algo más que permita restaurar la información original proveniente del registro compuesto.

La siguiente figura representa esquemáticamente cómo la relación adicional **MAT/EST** relaciona estudiante y materias. Las entradas en **MAT/EST** están dispuestas por materias y dentro de éstas por estudiantes. Nuevamente valen los requerimientos para una relación:

- **Está en la primera forma normal**
- **Cada fila tiene un único identificador**

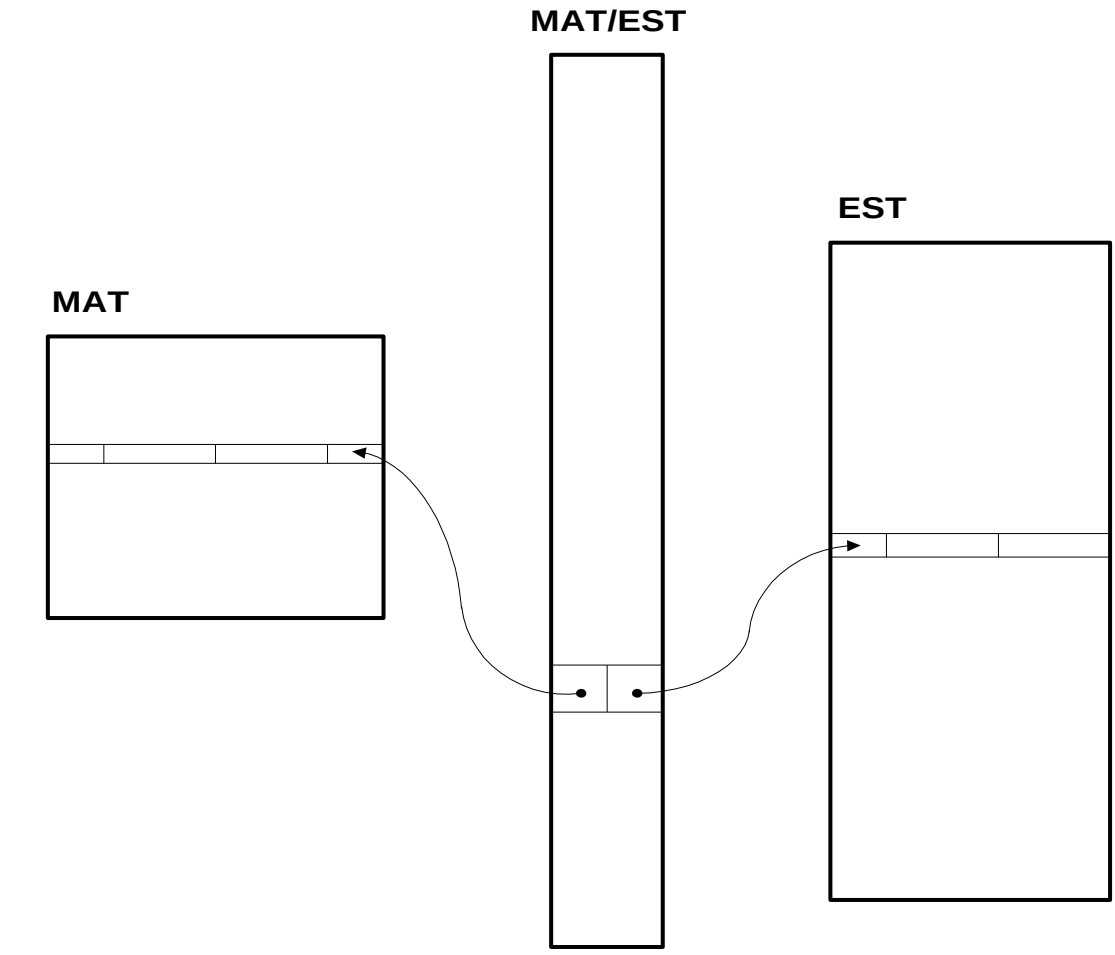
Se tiene en este caso una clave compuesta por los identificadores de materia y de estudiante. Cada fila, necesita contener solamente eso, dado que su propósito es relacionar las otras dos sub-relaciones. Es sin embargo un detalle importante, que una fila pueda contener mayor información, tal como el registro de asistencia y las notas en la materia. Debe enfatizarse que se trata sin embargo de información adicional y no un requerimiento de esta tabla de relación.

Profesores

Mientras **MAT/EST** relaciona materias con estudiantes, se ve que esto es insuficiente para restaurar toda la información originariamente contenida en el MEP. Como mínimo se necesita una tabla adicional, tal como la **MAT/PROF**. Esta tabla relaciona profesores y materias. En esta ocasión existe una entrada para cada materia, y contiene como segundo campo en su fila, un identificador del profesor que dicta la materia.

Las dos relaciones **MAT/EST** y **MAT/PROF** son suficientes para reconstruir el contenido de información de MEP. Puede verificarse que el número total de campos en las cinco sub-relaciones, es mucho menor que el de los campos de MEP. Esta es la fuente del mejoramiento de eficiencia; la mis-

ma no se contradice por el hecho de que algunos componentes puedan encontrarse en dos o más relaciones.



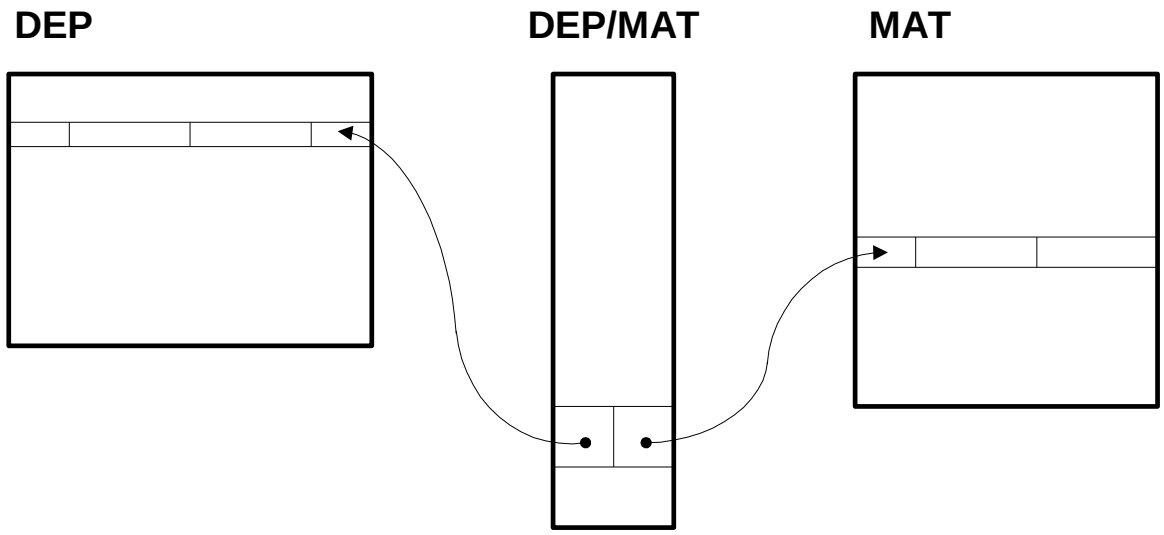
Recuperación

Con **MAT/EST**, **MAT y EST**, es posible reconstruir información acerca de cuales estudiantes cursan una determinada materia. Si se proveyeran otros atributos en **MAT/EST**, se podría incluso recuperar registros de asistencia, notas, etc. Si las necesidades se presentan en sentido inverso, por ejemplo si el encargado de las listas necesita encontrar las materias a las que está asistiendo un determinado estudiante, las tablas que se tienen son suficientes para obtener esta información, dado que se puede hacer la búsqueda en la tabla **MAT/EST** y recuperar solamente aquellas tuplas que contienen el número ID de estudiante deseado en el segundo campo. Cada una de estas tuplas, tiene un valor de campo Materia, que señala el nombre de una materia en la cual está registrado el estudiante.

El método para establecer este tipo de requerimientos es simbólico y matemáticamente muy simple. El método para efectuar la búsqueda, puede no ser tan directo, y los procesos que lo logran dentro del motor de la base de datos, dependen del fabricante del software. Por otra parte, la presencia de otra relación, que se llamará **EST/MAT**, que lista primero a los estudiantes y después las materias a las que dicho estudiante concurre (ordenada por ID estudiante), permitirá hallar directamente los datos del estudiante.

Descomposición adicional

La operación original, descompuso la base de datos del ejemplo, en cinco relaciones. Tres de ellas eran descriptivas (entidades) y las otras dos relacionaban a los correspondientes individuos (relaciones). Es posible efectuar una descomposición adicional, que involucra a las materias con los departamentos. Como se ve, el criterio seguido, es análogo al aplicado anteriormente. La situación, puede observarse en el diagrama siguiente.



La tabla de Departamentos, **DEP** lista la información de los departamentos. Se crea entonces otra tabla de relación que vincula cada materia con un departamento determinado (**DEP/MAT**). No disminuye en este caso el tamaño de la tabla de materias, pero sí sus columnas, en las que figuraba la información adicional correspondiente a los departamentos. La cantidad de filas existentes en **DEP/MAT**, es la misma que en la tabla **MAT**.

Depuración de tablas relacionales

Del ejemplo planteado, definitivamente han quedado las siguientes tablas:

1. **ESTUDIANTE (***Id_estudiante*, Nom_estudiante**)**
2. **PROFESOR (***Id_profesor*, Nom_profesor**)**
3. **MATERIA (***Id_materia*, Nom_materia**)**
4. **DEPARTAMENTO (***Id_departamento*, Nom_departamento**)**
5. **EST_MAT (**relación entre estudiante y materia**)**
6. **PROF_MAT (**relación entre profesor y materia**)**
7. **DEP_MAT (**relación entre departamento y materia**)**

Entonces, para cada uno de los símbolos utilizados en el modelo conceptual de datos, existirá una tabla en el modelo físico (una por cada entidad y una por cada relación). En la realidad esto no ocurre, ya que no todas las relaciones generan tablas. Su generación depende de la funcionalidad de la relación y de la parcialidad de participación de las entidades en la relación en cuestión. En la enumeración presentada, 5, 6 y 7, representan relaciones. Sus funcionalidades son:

EST_MAT (m,n): ya que un estudiante puede cursar muchas materias y una materia puede ser cursada por muchos estudiantes.

PROF_MAT (1,n): ya que una materia es dictada por uno y solamente un profesor, mientras que un profesor puede dictar varias materias.

DEP_MAT (1,n): ya que una materia pertenece a uno y solamente un departamento, mientras que un departamento tiene varias materias.

Para los dos últimos casos, expresan una dependencia funcional, ya que a partir del conocimiento de la materia, pueden ser determinados el ***profesor que la dicta*** y el ***departamento al que pertenece***. No ocurre esta situación con **EST_MAT**, ya que ni a partir de estudiante o de materia puede determinarse el otro componente de la ocurrencia de la relación. Esta última situación es la que determina la existencia de una TABLA DE RELACIÓN, mientras que en los otros dos casos, tal tabla no es necesaria.

Se plantea ahora la cuestión de cómo poder establecer el vínculo entre materia y profesor por un lado, y por otro, el de materia y departamento. Para ambos casos el razonamiento es el mismo. Al existir una dependencia funcional (cardinalidad 1,1 entre Materia con Profesor y Materia con Departamento), al expresar físicamente esta situación, el identificativo de la entidad que se determina a través de la dependencia funcional, pasará como atributo de la entidad condicionante, recibiendo el nombre de Clave Ajena (*foreign key*). De esta forma, la tabla MATERIA quedará conformada por los atributos:

Atributos
- *Id_materia*
- *Nom_materia*
- *Id_profesor*
- *Id_departamento*

Integridad
- *Clave primaria*: *Id_materia*
- *Clave ajena* ***Id_profesor*** ***REFERENCIANDO a*** **PROFESOR**
- *Clave ajena* ***Id_departamento*** ***REFERENCIANDO a*** **DEPARTAMENTO**

Es de notar que el valor de un atributo que se dice que es clave ajena, deberá existir como clave primaria en la tabla a la que referencia, aunque existe la posibilidad de declarar explícitamente que se admita clave ajena nula. Se concluye entonces que en los casos de dependencia funcional, la entidad que condiciona tomará como atributo y clave ajena a la clave de la entidad dependiente.

Considerando el caso de **EST_MAT**, y generalizando, cualquier relación de funcionalidad **m,n** (muchos a muchos), generará una TABLA DE RELACIÓN, cuya clave primaria estará conformada por la concatenación de las claves de las entidades que son colección de la relación, y cada una, será clave ajena referenciando precisamente a las tablas en donde son claves primarias. Así, para el caso de la tabla **EST_MAT**, ésta estará compuesta por los siguientes atributos:

Atributos
- *Id_estudiante*
- *Id_materia*

Integridad
- *Clave primaria*: *Id_estudiante + Id_materia*
- *Clave ajena* ***Id_estudiante*** ***REFERENCIANDO a*** **ESTUDIANTE**
- *Clave ajena* ***Id_materia*** ***REFERENCIANDO a*** **MATERIA**

Considerándose la metodología desarrollada en Ingeniería de Software I, los elementos que se transformarán en tablas serán:

- Las entidades
- Las relaciones que sean de funcionalidad **m,n** (muchos a muchos)
- Las relaciones del tipo **0,1 – 0,1**
- Las relaciones que poseen atributos (que necesariamente serán **m,n** o bien **0,1 – 0,1**)
- Las relaciones en las que participen más de 2 entidades
- Para el caso de las jerarquías de clasificación pueden:
    - Generar solamente la tabla correspondiente a la entidad padre
    - Generar la tabla de la entidad padre y tablas para cada uno de sus hijos
    - Generar solamente las tablas de las entidades hijo

LA BASE DE DATOS RELACIONAL

Durante la década de los ‘80 han aparecido las bases de datos relacionales, las cuales se han hecho más populares que las jerárquicas y las de redes que las precedieron. Con el fin de introducir un cierto orden en la información que surge cada vez a más velocidad sobre las bases de datos relacionales, CODD estableció en 1985 una serie de principios de los cuales al menos seis deben satisfacerse para que una base de datos pueda llamarse totalmente relacional. Estos fueron precedidos de una regla general global, llamada Regla Cero.

Regla 0:

Gestión de una base de datos relacional. Todo sistema que se anuncie como un sistema de gestión de base de datos relacional, debe ser capaz de manejar bases de datos exclusivamente con sus capacidades relacionales.

Regla 1: Representación de la información.

Toda la información de una base de datos relacional, se representa explícitamente en el ámbito lógico y exactamente de una forma: mediante valores en tablas.

Regla 2: Garantía de accesibilidad lógica.

Todos y cada uno de los datos de una base de datos, relacional tienen la garantía de ser accesibles lógicamente mediante el recurso de una combinación de: el nombre de la Tabla, el valor de la clave primaria y el nombre de la columna.

Regla 3: Representación sistemática de la información que falta.

Los valores nulos (que son distintos de la cadena vacía de caracteres o de la cadena de caracteres en blanco, y distintos de cero o de cualquier otro número) tienen la existencia en los sistemas de gestión de bases de datos totalmente relacionales, para representar la información que falta y la información que no es aplicable, de forma sistemática e independiente del tipo de dato.

Regla 4: Sub-lenguaje de datos completo.

Un sistema relacional puede soportar varios lenguajes y varios modos de uso terminal. Sin embargo, debe haber, al menos, un lenguaje cuyas instrucciones puedan expresarse por alguna sintaxis bien definida, como cadenas de caracteres, y que sea completo, soportando todos los términos siguientes:

Definición de Datos
Definición de Vistas
Manejo de Datos
Limitaciones de integridad
Autorización o permisos
Límites de transacción (inicio y fin para hacer permanentes los cambios y deshacer los cambios no permanentes)

Regla 5: Inserción, actualización y borrado de alto nivel.

La capacidad de manejar una relación de base o una relación derivada como un único operador, se aplica no sólo a la recuperación de datos, sino también a la inserción, a la actualización y al borrado de datos.

Regla 6: Independencia de los datos físicos.

Los programas de aplicaciones y las actividades terminales, permanecerán lógicamente inalterados siempre que se realicen cambios en las representaciones de almacenamiento o en los métodos de acceso.

Regla 7: Independencia de los datos lógicos.

Los programas de aplicaciones y las actividades finales permanecerán lógicamente inalterados cuando se llevan a cabo cambios en las tablas de base que conservan la información de cualquier tipo que permita teóricamente su inalterabilidad.

Regla 8: Independencia de la integridad.

Las limitaciones de integridad, específicas de una base de datos en particular, deben ser definibles en un sub-lenguaje de definición de datos y almacenables en el catálogo o diccionario.

Existen otras reglas más esbozadas acerca de los requerimientos de una base de datos relacional, pero las antes expuestas, son las de mayor importancia.

El SQL (Lenguaje de Consulta Estructurado), es un lenguaje estándar universal de manejo de datos, basado en el cálculo aplicado a los predicados y aplicable a las bases de datos relacionales. El hecho de que los desarrolladores supieran de antemano lo que debía ser SQL y lo que requeriría que hiciese, le dio una fuerte base teórica. Esta fue probablemente la primera vez que ocurre en el desarrollo de un lenguaje para computadoras, porque la mayoría de los lenguajes de programación son el resultado de una idea básica que se complementa con una gran cantidad de parches sobre la marcha para resolver los problemas a medida que surgen. Este hecho, el de especificar las necesidades de SQL antes de desarrollar los mecanismos del mismo, dio lugar a un lenguaje elegantemente parsimonioso que consta de relativamente pocos comandos que se pueden utilizar para satisfacer la mayoría de las necesidades de una base de datos muy compleja. Su sencillez hace que SQL sea adecuado tanto para el usuario ocasional, como para el que hace desarrollos profesionales. Se pueden realizar consultas "ad hoc", y también puede ser embebido en lenguajes de programación.

Definiciones

Teniendo en cuenta el hecho de que SQL es un auxiliar del sistema de gestión de bases de datos relacionales, se definen los siguientes términos dentro del contexto relacional.

Una Tabla es la estructura principal de la base de datos. Es una matriz rectangular con las siguientes propiedades:

1. Es homogénea en sus columnas; en otras palabras, en cualquier columna que se seleccione, los elementos son todos de la misma clase, mientras que los elementos de columnas distintas no tienen porqué ser de la misma clase.

2. Cada elemento, es un único número o una cadena de caracteres – **1FN -** (por lo tanto, al elemento de cualquier fila y columna que se haya especificado, no se encontrará un conjunto de número o un grupo de ellos).

3. Todas las filas de una tabla, deben ser distintas (no se permiten duplicaciones) **– 1FN -** .

4. El orden de las filas dentro de una tabla, es indiferente.

5. A las columnas de una tabla, se les asigna nombres distintos, y el orden de las columnas dentro de una tabla, es indiferente.

A estos tipos de tablas, se las llama relación. Si tiene n columnas, entonces se la llama una "relación de grado n".

Al conjunto de todos los nombres de las columnas de una tabla, se le llama esquema de relaciones (estructura de la tabla), y la Tabla, es una relación sobre el esquema de relaciones. En general, una base de datos, consta de más de una tabla, cada una de las cuales, tiene su propio conjunto de atributos o nombres de columnas. La estructura de una base de datos, es la reunión de todos los esquemas de relaciones de las tablas de la base de datos. Dos esquemas de relaciones distintos, pueden tener algunos nombres de columnas -atributos- en común. Una tabla, es un ejemplo de una relación sobre un esquema de relaciones; es decir, si se cambia de alguna manera los datos de una tabla de relación, se tendrá una relación distinta con el mismo esquema de relaciones. Resumiendo:

Una relación denotará una ***tabla***.
A un atributo se le llamará ***Columna***.
A un registro único, se le llamará ***fila***.
Al valor individual de la intersección de cualquier fila y columna, se le llamará ***dato***.

Las propiedades de las Tablas

Las tablas, deben tener un nombre, y ese nombre no puede ser una palabra clave de SQL. Hay tres tipos de tablas en las bases de datos relacionales: las tablas de base, las tablas virtuales y las tablas temporales. Las tablas temporales se comportan como tablas de base pero presentan restricciones como ser que solamente las puede ver el usuario que la creó. Por otra parte, su vida es efímera ya que son removidas cuando se desconecta la sesión. Una tabla de base, se ajusta a la definición de tabla dada anteriormente. Una tabla virtual, también llamada vista, existe tan sólo como una definición en el catálogo o diccionario de datos. Cada vez que se accede a una vista, se recupera la definición del diccionario y se ejecuta la definición que hay en la consulta, con lo que se crea una tabla con las columnas que están en el listado de la definición de la vista. A las vistas, se las llama tablas virtuales, porque no existen por derecho propio en la base de datos en la forma en que existen las tablas de base. Por el contrario, la vista se reconstruye a partir de los datos de las tablas de base subyacentes cada vez que se consulta la vista. Mientras que una vista, en su forma más sencilla, puede ser una parte de la tabla base, también puede ser el resultado de unir una parte de una o de varias tablas entre sí. Una vista puede ser también una parte de otra vista, en la que esta última sea una parte de una tabla base.

El valor NULL

SQL incluye el concepto de valor NULL para referirse a la información que es incompleta o de la que no se dispone. Se puede pensar y decir que este valor, es simplemente una instancia más en el dominio de un atributo. Hay distintas opiniones sobre la utilidad o la lógica que subyace en el concepto de los valores NULL. En la actualidad, los valores NULL son en gran medida, una parte del SQL, y en el futuro es probable que aparezcan modificados o adornados, pero nunca eliminados de su sintaxis. Sirven para rellenar un espacio en blanco en una matriz de datos, pero deben utilizarse con precaución. Deberá tenerse en cuenta las siguientes reglas, así como otros usos y precauciones especiales que sobre los NULL se indiquen en los contextos en los que puedan presentarse:

Un valor NULL de un dato numérico, no es lo mismo que cero
Un valor NULL de un dato tipo carácter, no es lo mismo que espacio o blanco
Un valor NULL no tiene por qué ser igual a otro valor NULL
Un valor NULL no se puede utilizar en una instrucción de proyección (SELECT)
El tratamiento que de los NULL hacen las funciones agregadas, no es uniforme