Cardinalidad

El objetivo de este apunte es aportar claridad en el estudio de las relaciones entre las entidades que se modelan a través de bases de datos relacionales. Muchas veces, resulta confuso o ambigüo entender en qué momento se utiliza una u otra y posteriormente como representarla de una manera conveniente. Este apunte no debe ser tomado como una receta de procedimientos, sino como un conjunto de análisis y estrategias de planteos que pueden ayudarle.

En primer lugar, tener en cuenta lo siguiente:

- Sólo existen las siguientes formas de establecer relaciones entre entidades: 1 a 1, 1 a N y N a N. No hay nada más que esto en la práctica real generalmente.

- Estos tipos de relaciones (1-1, 1-N, N-N) se piensan independiente del tipo de base de datos. Están en un plano más abstracto. Luego, si se elige una base de datos relacional se implementará de la manera más conveniente, sabiendo que allí solo existen tablas y claves primarias/secundarias-foráneas para hacerlo. Si eligiera otro tipo de bases de datos, se debería modelar la relación acorde a los elementos que estén disponibles para ellos.

Relación 1 a 1

> *"Una instancia de una entidad, sólo puede estar vinculada con una única instancia de otra/misma entidad"*

Ejemplo: "Un usuario tiene su propia información personal"

Piense bien en este ejemplo simplificado y suponga que el usuario se trata de una persona real que hace uso de un sistema. Refinando luego un poco más, puede llegar a considerar lo siguiente:

"Un usuario, es una persona que emplea un sistema. Por lo tanto, es de carácter único e irrepetible. Por consiguiente, la información personal del usuario le pertenece inequívocamente a éste y a ningún otro más".

Para implementar la relación 1 a 1 en una base de datos relacional, una primera aproximación podría ser la siguiente:

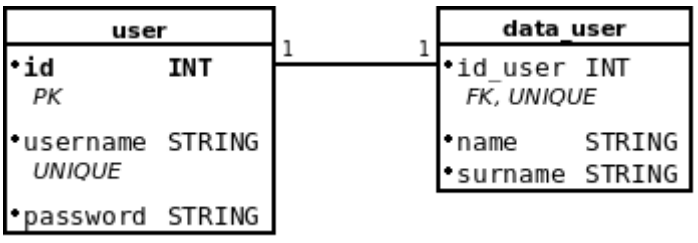

Figura 1. Primera aproximación a un modelado de la relación 1-1.

En la primera tabla *user*, el atributo *id* es la clave primaria de la entidad. Recuerde, que la clave primaria garantiza que el atributo afectado sea *único y no-nulo*. Por otro lado, la tabla *data_user* posee una clave foránea id_user que hace referencia a la clave primaria de la tabla *user.* Sin la presencia de la restricción *UNIQUE en ésta,* no sería posible garantizar la no-repitencia del *id_user*, dando una relación 1-N adversa.

Este diseño, garantiza a nivel de registros la relación 1-1. Sin embargo, esto no siempre alcanza. Por ello, es necesario que no se automatice al modelar relaciones y sea consciente de lo que está representando acorde a los requerimientos. En este ejemplo hay una situación de contexto implícita, que de no tenerse en cuenta, el diseño se comportaría erróneamente en algunas situaciones; si el usuario que está intentando modelar, está pensado para que inicie sesión a través de su *id*, no harían falta restricciones adicionales sobre el atributo *username*. En cambio, si el inicio de sesión fue pensado para efectuarse vía introducción de username/password, la clave primaria afectando al atributo *id* no es suficiente y debería acompañarla garantizando que el nombre de usuario *username* también sea único y no-nulo, porque su intención sería identificar al usuario de manera unívoca también por medio de este campo.

Si usted recurre a referencias bibliográficas comunes y/o populares sobre el modelado de relaciones 1-1, encontrará generalmente importante cantidad de ejemplos de diseños similares a este. Siempre pregúntese si se trata de la única manera de hacerlo, porque la respuesta es casi siempre un no. Existen muchas maneras de representar una relación 1 a 1 en una base de datos relacional. A continuación se le propone un diseño alternativo. Recuerde, que los diseños no están ni bien ni mal, son más o menos adecuados al contexto de lo que necesita resolver y atravesados por múltiples variables condicionantes de diversa naturaleza.

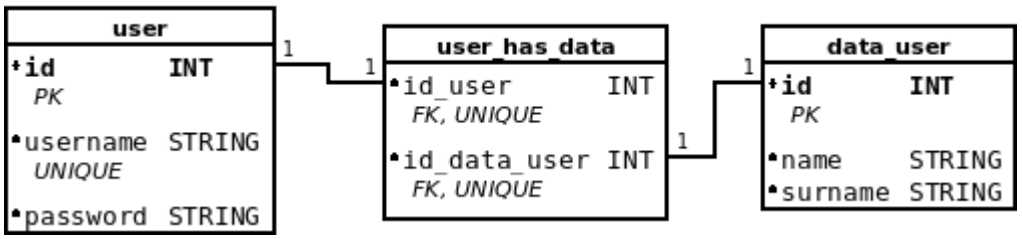

Figura 2. Diseño alternativo para el modelado de una relación 1-1.

Observe este segundo diseño. Aunque no parezca, a primera vista parece el clásico ejemplo de tablas intermedias para modelar relaciones N-N. Sin embargo, no es así. La tabla *user* es idéntica al diseño anterior, por lo tanto, no requiere mayor explicación. La tabla *data_user*, tiene un ligero cambio. En vez de tener una clave foránea referenciando la clave primaria de la tabla *user*, tiene su propia clave primaria y no se ve ninguna vinculación directa entre ellas. Esto quiere decir que ambas entidades son *autónomas*. Esa es la particularidad de este diseño. La forma de relación se delega en la entidad auxiliar *user_has_data.* Ahora bien, para que la relación sea 1-1, se tienen que dar dos restricciones en las claves foráneas. Ambas deben ser únicas, porque justamente *UN* registro de *user*, sólo debe relacionarse con *UN* registro de *user_data.* Si usted deja libre alguno de los atributos, estaría habilitando que existan registros en la tabla intermedia que repitan algún ID moviéndose hacia a otro tipo de cardinalidad entre ambas entidades.

El objetivo de esta lectura y el análisis sobre dos diseños es para hacernos el hábito de introducirnos la pregunta de porqué elijo utilizar un diseño u otro. Resulta una buena práctica identificar las ventajas y desventajas de cada uno, acorde al contexto de requerimientos y del desarrollo.

Ventajas	Desventajas
● Entidades desacopladas, no condicionan el orden de inserción en las tablas. ● Permite secuencializar mejor el ingreso de datos al modelo cuando los datos están muy relacionados. ● En presencia de agregados de atributos, este diseño resulta más flexible, dado que no promueve redefiniciones de tablas. El agregado de atributos se puede resolver agregando tablas nuevas. ● Se habilita el guardado de información específica a la relación entre entidades en la tabla intermedia, sin invadir las entidades individuales. ● El diseño es más adecuado para implementar borrado lógico de relaciones. ● Posibilita hacer más granular el control de acceso a información en sistemas multiusuario.	● Incrementa el número de JOINs en el código SQL para efectuar lecturas de datos. ● El diseño incrementa el número de tablas. ● El borrado físico de registros requiere de un análisis más exhaustivo a la hora de ser implementado. ● Requiere de más tiempo de desarrollo.

Relación 1-N

“Una instancia de una entidad, sólo puede puede estar vinculada a una o más instancias de otra/misma entidad”.

Ejemplo: “Un usuario puede tener varias cajas de ahorro”.

Una primera aproximación de implementación podría ser:

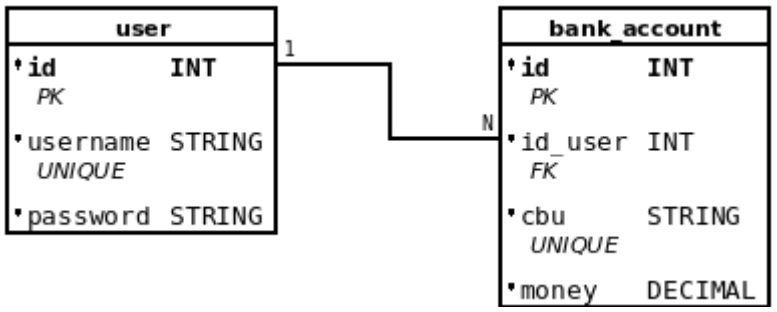

Figura 3. Diseño simplificado para el modelado de una relación 1-N

Siguiendo con el análisis de diseños, veamos que este conjunto de tablas guarda similitud con el presentado en la relación 1-1. Observe que la entidad usuario permanece autónoma y mantiene la descripción que venimos utilizando. Sin embargo, la tabla *bank_account* posee una clave foránea *id_user* que no tiene ninguna restricción aplicada. Por lo tanto, si usted tuviera este modelo implementado con datos precargados, una posible salida sería:

user		
id	*username*	*password*
1	mgarcia	t63r6
2	gfernandez	$c#1jk

bank_account			
id	*id_user*	*cbu*	*money*
1	1	01702046600000087865	1500,80
2	1	01402044600000089911	3700,75
3	1	01602023600000086402	12450,33
4	2	01200202360000005140	600,50
5	2	01200202360000004440	21500,70

Usted puede leer, en función de esta salida hipotética, que el usuario *mgarcia* tiene 3 cuentas bancarias y el usuario *gfernandez* 2 cuentas bancarias. A su vez, una cuenta bancaria pertenece *a un y solo un usuario*.

En relación al estudio de relaciones 1-1, nótese que en un momento se ha mencionado en el análisis previo que si no se presentaba una restricción *UNIQUE* sobre la clave foránea *id_user*, se podría presentar una relación 1-N adversa, contraria a la que se buscaba con anterioridad.

Siguiendo nuestro estilo de análisis, proponemos un segundo diseño alternativo para mostrar otra manera de modelar la relación 1-N.

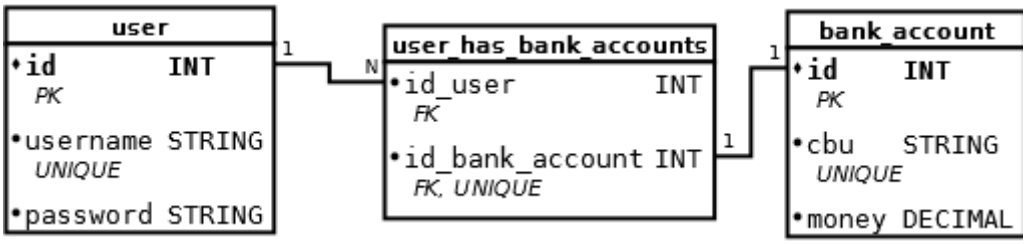

Figura 4. Diseño simplificado para el modelado de una relación 1-N

Observe que, de igual manera, el estilo aquí es otra vez delegar en una entidad auxiliar llamada *user_has_bank_accounts*. Si nos situamos en ella, vemos que la clave foránea *id_bank_account* tiene la restricción *UNIQUE*. Tal vez parezca poco intuitivo que si queremos modelar que UN usuario tiene N cuentas bancarias, que el carácter de único esté en *id_bank_account*. Sin embargo, tiene sentido. Piense en el rol que está teniendo esta tabla auxiliar. Lo que está haciendo es guardar las relaciones entre *user* y *bank_account*. El usuario debe tener N relaciones a las cuentas bancarias, por lo tanto, el *id_user* debe poder repetirse aquí.

Para visualizarlo con mayor claridad:

user			**user_has_bank_accounts**		**bank_account**		
id	*username*	*password*	*id_user*	*id_bank_account*	*id*	*cbu*	*money*
1	mgarcia	t63r6	1	1	1	01702046600000087865	1500,80
2	gfernandez	$c#1jk	1	2	2	01402044600000089911	3700,75
			1	3	3	01602023600000086402	12450,33
			2	4	4	01200202360000005140	600,50
			2	5	5	01200202360000004440	21500,70

Relación N-N

“Varias instancias de una entidad, pueden estar vinculadas a una o más instancias de otra o igual entidad”.

Este tipo de relación es conocida como de “*muchos* a *muchos*”. Tenga en consideración el ejemplo anterior, *“*un usuario tiene varias cuentas bancarias*”*, y que “una cuenta bancaria pertenece *a un y solo un* usuario”. Si tomamos otro ejemplo diferente “*varios usuarios tienen varias aulas virtuales*” y que “*un aula virtual puede pertenecer a varios usuarios*”. Vemos cómo se extiende la multiplicidad de la relación. Puede emplear analogías similares como estrategia para determinar si usted está en presencia o no de algún tipo de conectividad N-N.

Una posible implementación de este caso podría ser:

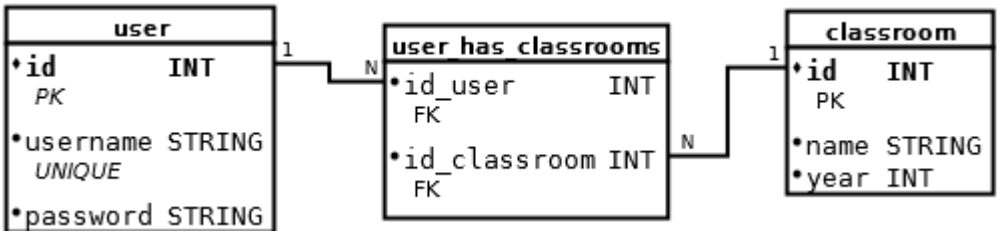

Figura 5. Diseño posible para el modelado de una relación N-N

A este nivel de la lectura, tomando en base las explicaciones anteriores podría dilucidar que ahora la relajación de la restricción UNIQUE que se dió de manera progresiva sobre las claves foráneas de la entidad auxiliar intermedia, ahora posibilitan todo el espectro de combinaciones posibles.

Queda a cargo del lector representar una posible representación de datos para visualizar un ejemplo concreto del conjunto de datos.

Tal vez a esta altura podría preguntarse también si existe alguna otra manera de implementar esta relación N-N mediante los elementos que provee una base de datos relacional.

Esta es una pregunta que podría animarse a contestar y poner a prueba en sus prácticas de base de datos. Puede pensar en alguna propuesta y llevarla a cabo y determinar si la misma se comporta acorde a lo esperado.