TEMA 4: ELABORACIÓN DEL DISEÑO LÓGICO. MODELO RELACIONAL. 2ª PARTE

Reglas de integridad

OBJETIVOS:

- Conocer los fundamentos del modelo de datos relacional.
- Comprender las ventajas del modelo relacional que derivan del alto grado de independencia de los datos que proporciona, y de la simplicidad y la uniformidad del modelo.
- Conocer las operaciones del álgebra relacional.
- Saber utilizar las operaciones del álgebra relacional para consultar una base de datos.

REGLAS DE INTEGRIDAD

Regla de integridad de unicidad de la clave primaria

Regla de integridad de entidad de la clave primaria

Regla de integridad referencial

Restricción

Actualización en cascada

Anulación

Selección de la política de mantenimiento de la integridad referencial

Regla de integridad de dominio

3. Reglas de integridad

Una base de datos contiene unos datos que, en cada momento, deben reflejar la realidad o, más concretamente, la situación de una porción del mundo real. En el caso de las bases de datos relacionales, esto significa que la extensión de las relaciones (es decir, las tuplas que contienen las relaciones) deben tener valores que reflejen la realidad correctamente.

Puede ocurrir que determinadas configuraciones de valores para las tuplas de las relaciones no tengan sentido, porque no representan ninguna situación posible del mundo real.

> Un sueldo negativo
>
> En la relación de esquema EMPLEADOS(DNI, nombre, apellido, sueldo), una tupla que tiene un valor de –1.000 para el sueldo probablemente no tiene sentido, porque los sueldos no pueden ser negativos.

Denominamos integridad la propiedad de los datos de corresponder a representaciones plausibles del mundo real.

Como es evidente, para que los datos sean íntegros, es preciso que cumplan varias condiciones.

> El hecho de que los sueldos no puedan ser negativos es una condición que se debería cumplir en la relación EMPLEADOS.

En general, las condiciones que garantizan la integridad de los datos pueden ser de dos tipos:

1) **Las restricciones de integridad de usuario** son condiciones específicas de una base de datos concreta; es decir, son las que se deben cumplir en una base de datos particular con unos usuarios concretos, pero que no son necesariamente relevantes en otra base de datos.

> Restricción de integridad de usuario en EMPLEADOS
>
> Éste sería el caso de la condición anterior, según la cual los sueldos no podían ser negativos. Observad que esta condición era necesaria en la base de datos concreta de este ejemplo porque aparecía el atributo sueldo, al que se quería dar un significado; sin embargo, podría no ser necesaria en otra base de datos diferente donde, por ejemplo, no hubiese sueldos.

2) **Las reglas de integridad de modelo**, en cambio, son condiciones más generales, propias de un modelo de datos, y se deben cumplir en toda base de datos que siga dicho modelo.

> Ejemplo de regla de integridad del modelo de datos relacional
>
> En el caso del modelo de datos relacional, habrá una regla de integridad para garantizar que los valores de una clave primaria de una relación no se repitan en tuplas diferentes de la relación. Toda base de datos relacional debe cumplir esta regla que, por lo tanto, es una regla de integridad del modelo.

Los SGBD deben proporcionar la forma de definir las restricciones de integridad de usuario de una base de datos; una vez definidas, deben velar por su cumplimiento.

Las reglas de integridad del modelo, en cambio, no se deben definir para cada base de datos concreta, porque se consideran preestablecidas para todas las base de datos de un modelo. Un SGBD de un modelo determinado debe velar por el cumplimiento de las reglas de integridad preestablecidas por su modelo.

A continuación estudiaremos con detalle las reglas de integridad del modelo relacional, reglas que todo SGBD relacional debe obligar a cumplir.

3.1. Regla de integridad de unicidad de la clave primaria

La regla de integridad de unicidad está relacionada con la definición de clave primaria. Concretamente, establece que toda clave primaria que se elija para una relación no debe tener valores repetidos.

Ejemplo:

Tenemos la siguiente relación:

edificio	*número*	*superficie*
Marina	120	10
Marina	122	15
Marina	230	20
Diagonal	120	10

En esta relación, dado que la clave primaria está formada por edificio y número, no hay ningún despacho que repita tanto edificio como número de otro despacho. Sin embargo, sí se repiten valores de edificio (por ejemplo, Marina); y también se repiten valores de número (120). A pesar de ello, el edificio y el número no se repiten nunca al mismo tiempo.

A continuación explicamos esta regla de forma más precisa.

La regla de integridad de unicidad de la clave primaria establece que si el conjunto de atributos CP es la clave primaria de una relación R, entonces la extensión de R no puede tener en ningún momento dos tuplas con la misma combinación de valores para los atributos de CP.

Es preciso destacar que el mismo concepto de clave primaria implica esta condición. Un SGBD relacional deberá garantizar el cumplimiento de esta regla de integridad en todas las **inserciones**, así como en todas las **modificaciones** que afecten a atributos que pertenecen a la clave primaria de la relación.

Ejemplo:

Tenemos la siguiente relación:

edificio	*número*	*superficie*
Marina	120	10
Marina	122	15
Marina	230	20
Diagonal	120	10

En esta relación no se debería poder insertar la tupla <Diagonal, 120, 30>, ni modificar la tupla <Marina, 122, 15>, de modo que pasara a ser <Marina, 120, 15>.

3.2. Regla de integridad de entidad de la clave primaria

La regla de integridad de entidad de la clave primaria dispone que los atributos de la clave primaria de una relación no puedan tener valores nulos.

Ejemplo

Tenemos la siguiente relación:

Edificio	*número*	*superficie*
Marina	120	10
Marina	122	15
Marina	230	20
Diagonal	120	10

En esta relación, puesto que la clave primaria está formada por edificio y número, no hay ningún despacho que tenga un valor nulo para edificio, ni tampoco para número.

Esta regla es necesaria para que los valores de las claves primarias puedan identificar las tuplas individuales de las relaciones. Si las claves primarias tuviesen valores nulos, es posible que algunas tuplas no se pudieran distinguir.

Ejemplo de clave primaria incorrecta con valores nulos

En el ejemplo anterior, si un despacho tuviese un valor nulo para edificio porque en un momento dado el nombre de este edificio no se conoce, por ejemplo <NULO, 120, 30>, la clave primaria no nos permitiría distinguirlo del despacho <Marina, 120, 10> ni del despacho <Diagonal, 120,10>. No podríamos estar seguros de que el valor desconocido de edificio no es ni Marina ni Diagonal.

A continuación definimos esta regla de forma más precisa.

La regla de integridad de entidad de la clave primaria establece que si el conjunto de atributos CP es la clave primaria de una relación R, la extensión de R no puede tener ninguna tupla con algún valor nulo para alguno de los atributos de CP.

Un SGBD relacional tendrá que garantizar el cumplimiento de esta regla de integridad en todas las **inserciones** y, también, en todas las **modificaciones** que afecten a atributos que pertenecen a la clave primaria de la relación.

Ejemplo:

En la relación DESPACHOS anterior, no se debería insertar la tupla <Diagonal, NULO, 15>. Tampoco debería ser posible modificar la tupla <Marina, 120, 10> de modo que pasara a ser <NULO, 120, 10>.

3.3. Regla de integridad referencial

La regla de integridad referencial está relacionada con el concepto de clave foránea. Concretamente, determina que todos los valores que toma una clave foránea deben ser valores nulos o valores que existen en la clave primaria que referencia.

Ejemplo

Si tenemos las siguientes relaciones:

•Relación DESPACHOS:

edificio	*número*	*superficie*
Marina	120	10
Marina	122	15
Marina	230	20
Diagonal	120	10

•Relación EMPLEADOS:

DNI	*nombre*	*apellido*	*edificiodesp*	*númerodesp*
40.444.255	Juan	García	Marina	120
33.567.711	Marta	Roca	Marina	120
55.898.425	Carlos	Buendía	Diagonal	120
77.232.144	Elena	Pla	NULO	NULO

donde edificiodesp y númerodesp de la relación EMPLEADOS forman una clave foránea que referencia la relación DESPACHOS. Debe ocurrir que los valores no nulos de edificiodesp y númerodesp de la relación EMPLEADOS estén en la relación DESPACHOS como valores de edificio y número. Por ejemplo, el empleado <40.444.255, Juan García, Marina, 120> tiene el valor Marina para edificiodesp, y el valor 120 para númerodesp, de modo que en la relación DESPACHOS hay un despacho con valor Marina para edificio y con valor 120 para número.

La necesidad de la regla de integridad relacional proviene del hecho de que las claves foráneas tienen por objetivo establecer una conexión con la clave primaria que referencian. Si un valor de una clave foránea no estuviese presente en la clave primaria correspondiente, representaría una referencia o una conexión incorrecta.

Referencia incorrecta

Supongamos que en el ejemplo anterior hubiese un empleado con los valores <56.666.789, Pedro, López, Valencia, 325>. Ya que no hay un despacho con los valores Valencia y 325 para edificio y número, la tupla de este empleado hace una referencia incorrecta; es decir, indica un despacho para el empleado que, de hecho, no existe.

A continuación explicamos la regla de modo más preciso.

La regla de integridad referencial establece que si el conjunto de atributos CF es una clave foránea de una relación R que referencia una relación S (no necesariamente diferente de R), que tiene por clave primaria CP, entonces, para toda tupla t de la extensión de R, los valores para el conjunto de atributos CF de t son valores nulos, o bien valores que coinciden con los valores para CP de alguna tupla s de S.

En el caso de que una tupla t de la extensión de R tenga valores para CF que coincidan con los valores para CP de una tupla s de S, decimos que t es una tupla que referencia s y que s es una tupla que tiene una clave primaria referenciada por t.

Un SGBD relacional tendrá que hacer cumplir esta regla de integridad. Deberá efectuar comprobaciones cuando se produzcan las siguientes operaciones:

a) **Inserciones** en una relación que tenga una clave foránea.

b) **Modificaciones** que afecten a atributos que pertenecen a la clave foránea de una relación.

c) **Borrados** en relaciones referenciadas por otras relaciones.

d) **Modificaciones** que afecten a atributos que pertenecen a la **clave primaria** de una relación referenciada por otra relación.

Ejemplo

Retomamos el ejemplo anterior, donde edificiodesp y númerodesp de la relación EMPLEADOS forman una clave foránea que referencia la relación DESPACHOS:

Las siguientes operaciones provocarían el incumplimiento de la regla de integridad referencial:

- Inserción de <12.764.411, Jorge, Puig, Diagonal, 220> en EMPLEADOS.
- Modificación de <40.444.255, Juan, García, Marina, 120> de EMPLEADOS por <40.444.255, Juan, García, Marina, 400>.
- Borrado de <Marina, 120, 10> de DESPACHOS.
- Modificación de <Diagonal, 120, 10> de DESPACHOS por <París, 120, 10>.

Un SGBD relacional debe procurar que se cumplan las reglas de integridad del modelo. Una forma habitual de mantener estas reglas consiste en **rechazar toda operación de actualización** que deje la base de datos en un estado en el que alguna regla no se cumpla. En algunos casos, sin embargo, el SGBD tiene la posibilidad de **aceptar la operación y efectuar acciones adicionales** compensatorias, de modo que el estado que se obtenga satisfaga las reglas de integridad, a pesar de haber ejecutado la operación.

Esta última política se puede aplicar en las siguientes operaciones de actualización que violarían la regla de integridad:

a) Borrado de una tupla que tiene una clave primaria referenciada por otras.

b) Modificación de los valores de los atributos de la clave primaria de una tupla que tiene una clave primaria referenciada por otras.

En los casos anteriores, algunas de las **políticas** que se podrán aplicar serán las siguientes: **restricción, actualización en cascada y anulación**. A continuación explicamos el significado de las tres posibilidades mencionadas.

3.3.1. Restricción

La **política de restricción** consiste en no aceptar la operación de actualización.

Más concretamente, la restricción en caso de borrado, consiste en no permitir borrar una tupla si tiene una clave primaria referenciada por alguna clave foránea.

De forma similar, la restricción en caso de modificación consiste en no permitir modificar ningún valor de atributo de la clave primaria de una tupla si la clave primaria está referenciada por alguna clave foránea.

Ejemplo de aplicación de la restricción

Supongamos que tenemos las siguientes relaciones:

- Relación CLIENTES:

Numcliente
10
15
18

- Relación PEDIDOS_PENDIENTES:

numped	*...*	*numcliente*
1.234	–	10
1.235	–	10
1.236	–	15

a) Si aplicamos la restricción en caso de borrado y, por ejemplo, queremos borrar al cliente número 10, no podremos hacerlo porque tiene pedidos pendientes que lo referencian.

b) Si aplicamos la restricción en caso de modificación y queremos modificar el número del cliente 15, no será posible hacerlo porque también tiene pedidos pendientes que lo referencian.

3.3.2. Actualización en cascada

La **política de actualización en cascada** consiste en permitir la operación de actualización de la tupla, y en efectuar operaciones compensatorias que propaguen en cascada la actualización a las tuplas que la referenciaban; se actúa de este modo para mantener la integridad referencial.

Más concretamente, la actualización en cascada en caso de borrado consiste en permitir el borrado de una tupla t que tiene una clave primaria referenciada, y borrar también todas las tuplas que referencian t.

De forma similar, la actualización en cascada en caso de modificación consiste en permitir la modificación del valor de atributos de la clave primaria de una tupla t que tiene una clave primaria referenciada, y modificar del mismo modo todas las tuplas que referencian t.

Ejemplo de aplicación de la actualización en cascada
Supongamos que tenemos las siguientes relaciones:

- Relación EDIFICIOS:

Nombreedificio	...
Marina	–
Diagonal	

- Relación DESPACHOS:

*edificio**	*número*	*superficie*
Marina	120	10
Marina	122	15
Marina	230	20
Diagonal	120	10

a) Si aplicamos la actualización en cascada en caso de borrado y, por ejemplo, queremos borrar el edificio Diagonal, se borrará también el despacho Diagonal 120 que hay en el edificio, y nos quedará:

- Relación EDIFICIOS:
-

Nombreedificio	...
Marina	–

- Relación DESPACHOS:

*edificio**	*número*	*superficie*
Marina	120	10
Marina	122	15
Marina	230	20

b) Si aplicamos la actualización en cascada en caso de modificación, y queremos modificar el nombre del edificio Marina por Mar, también se cambiará Marina por Mar en los despachos Marina 120, Marina 122 y Marina 230, y nos quedará:

- Relación EDIFICIOS:

Nombreedificio	*...*
Mar	–

- Relación DESPACHOS:

*edificio**	*número*	*superficie*
Mar	120	10
Mar	122	15
Mar	230	20

3.3.3. Anulación

Esta **política de anulación** consiste en permitir la operación de actualización de la tupla y en efectuar operaciones compensatorias que pongan valores nulos a los atributos de la clave foránea de las tuplas que la referencian; esta acción se lleva a cabo para mantener la integridad referencial.

Puesto que generalmente los SGBD relacionales permiten establecer que un determinado atributo de una relación no admite valores nulos, sólo se puede aplicar la política de anulación si los atributos de la clave foránea sí los admiten.

Más concretamente, la anulación en caso de borrado consiste en permitir el borrado de una tupla t que tiene una clave referenciada y, además, modificar todas las tuplas que referencian t, de modo que los atributos de la clave foránea correspondiente tomen valores nulos.

De forma similar, la anulación en caso de modificación consiste en permitir la modificación de atributos de la clave primaria de una tupla t que tiene una clave referenciada y, además, modificar todas las tuplas que referencian t, de modo que los atributos de la clave foránea correspondiente tomen valores nulos.

Ejemplo de aplicación de la anulación

El mejor modo de entender en qué consiste la anulación es mediante un ejemplo. Tenemos las siguientes relaciones:

- Relación VENDEDORES:

Numvendedor
1
2
3

- Relación CLIENTES:

Numcliente	...	*vendedorasig**
23	–	1
35	–	1
38	–	2
42	–	2
50	–	3

a) Si aplicamos la anulación en caso de borrado y, por ejemplo, queremos borrar al vendedor número 1, se modificarán todos los clientes que lo tenían asignado, y pasarán a tener un valor nulo en vendedorasig.

Nos quedará:

- Relación VENDEDORES:

Numcliente	...	*vendedorasig**
23	–	NULO
35	–	NULO
38	–	2
43	–	2
50	–	3

- Relación CLIENTES:

Numvendedor
2
3

b) Si aplicamos la anulación en caso de modificación, y ahora queremos cambiar el número del vendedor 2 por 5, se modificarán todos los clientes que lo tenían asignado y pasarán a tener un valor nulo en vendedorasig.

Nos quedará:

- Relación CLIENTES:

Numcliente	...	*vendedorasig**
23	–	nulo
35	–	nulo
38	–	nulo
44	–	nulo
50	–	3

- Relación VENDEDORES:

Numvendedor
5
3

3.3.4. Selección de la política de mantenimiento de la integridad referencial

Hemos visto que en caso de borrado o modificación de una clave primaria referenciada por alguna clave foránea hay varias políticas de mantenimiento de la regla de integridad referencial.

El diseñador **puede elegir para cada clave foránea qué política se aplicará en caso de borrado de la clave primaria referenciada, y cuál en caso de modificación de ésta**. El diseñador deberá tener en cuenta el significado de cada clave foránea concreta para poder elegir adecuadamente.

Aplicación de políticas diferentes :

Puede ocurrir que, para una determinada clave foránea, la política adecuada en caso de borrado sea diferente de la adecuada en caso de modificación. Por ejemplo, puede ser necesario aplicar la restricción en caso de borrado y la actualización en cascada en caso de modificación.

3.4. Regla de integridad de dominio

La regla de integridad de dominio está relacionada, como su nombre indica, con la noción de dominio. Esta regla establece dos condiciones.

- La primera condición consiste en que un valor no nulo de un atributo A_i debe pertenecer al dominio del atributo A_i; es decir, debe pertenecer a dominio(A_i).

Esta condición implica que todos los valores no nulos que contiene la base de datos para un determinado atributo deben ser del dominio declarado para dicho atributo.

Ejemplo

Si en la relación EMPLEADOS(DNI, nombre, apellido, edademp) hemos declarado que dominio(DNI) es el dominio predefinido de los enteros, entonces no podremos insertar, por ejemplo, ningún empleado que tenga por DNI el valor “Luis”, que no es un entero.

Recordemos que los dominios pueden ser de dos tipos: predefinidos o definidos por el usuario. Observad que los dominios definidos por el usuario resultan muy útiles, porque nos permiten determinar de forma más específica cuáles serán los valores admitidos por los atributos.

Ejemplo

Supongamos ahora que en la relación EMPLEADOS(DNI, nombre, apellido, edademp) hemos declarado que dominio(edademp) es el dominio definido por el usuario edad. Supongamos

también que el dominio edad se ha definido como el conjunto de los enteros que están entre 16 y 65. En este caso, por ejemplo, no será posible insertar un empleado con un valor de 90 para edademp.

- La segunda condición de la regla de integridad de dominio es más compleja, especialmente en el caso de dominios definidos por el usuario; los SGBD actuales no la soportan para estos últimos dominios. Por estos motivos sólo la presentaremos superficialmente.

Esta segunda condición sirve para establecer que los operadores que pueden aplicarse sobre los valores dependen de los dominios de estos valores; es decir, un operador determinado sólo se puede aplicar sobre valores que tengan dominios que le sean adecuados.

Analizaremos esta segunda condición de la regla de integridad de dominio con un ejemplo concreto. Si en la relación EMPLEADOS(DNI, nombre, apellido, edademp) se ha declarado que dominio(DNI) es el dominio predefinido de los enteros, entonces no se permitirá consultar todos aquellos empleados cuyo DNI sea igual a ‘Elena’ (DNI = ‘Elena’). El motivo es que no tiene sentido que el operador de comparación = se aplique entre un DNI que tiene por dominio los enteros, y el valor ‘Elena’, que es una serie de caracteres.
De este modo, el hecho de que los operadores que se pueden aplicar sobre los valores dependan del dominio de estos valores permite detectar errores que se podrían cometer cuando se consulta o se actualiza la base de datos. Los dominios definidos por el usuario son muy útiles, porque nos permitirán determinar de forma más específica cuáles serán los operadores que se podrán aplicar sobre los valores.

Ejemplo

Veamos otro ejemplo con dominios definidos por el usuario. Supongamos que en la conocida relación EMPLEADOS(DNI, nombre, apellido, edademp) se ha declarado que dominio(DNI) es el dominio definido por el usuario númerosDNI y que dominio(edademp) es el dominio definido por el usuario edad. Supongamos que númerosDNI corresponde a los enteros positivos y que edad corresponde a los enteros que están entre 16 y 65. En este caso, será incorrecto, por ejemplo, consultar los empleados que tienen el valor de DNI igual al valor de edademp. El motivo es que, aunque tanto los valores de DNI como los de edademp sean enteros, sus dominios son diferentes; por ello, según el significado que el usuario les da, no tiene sentido compararlos.

Sin embargo, los actuales SGBD relacionales no dan apoyo a la segunda condición de la regla de integridad de dominio para dominios definidos por el usuario. Si se quisiera hacer, sería necesario que el diseñador tuviese alguna forma de especificar, para cada operador que se desease utilizar, para qué combinaciones de dominios definidos por el usuario tiene sentido que se aplique.