

UNIDAD DIDÁCTICA 3

INTERPRETACIÓN DEL DISEÑO LÓGICO. MODELO RELACIONAL

MODELO DE DATOS	2
Propiedades de un Modelo de Datos	2
Estática.....	2
Dinámica.....	3
Modelo Relacional. Terminología y Estructura.	4
Estructura del Modelo Relacional.....	5
Relación o Tabla.....	5
Dominio y Atributo	6
Claves.....	7
Restricciones	9
Restricciones inherentes son las que impone el propio modelo:	9
Restricciones semánticas:	10
Reglas de la Integridad Referencial	11
Opción RESTRICT (NO ACTION)	12
Opción SET NULL	13
Opción SET DEFAULT	13
Opción CASCADE	14
Regla INSERT	14
Regla DELETE	15
Regla UPDATE.....	16
Restricciones de rechazo	16
LOS VALORES NULOS EN EL MODELO RELACIONAL	17
Concepto de valor nulo	17

OBJETIVOS

- Interpretar el diseño lógico basado en el modelo relaaiconal.
- Identificar la terminología propia del modelo relacional.
- Identificar la estructura de una base de datos relacional.
- Reconocer las restricciones del modelo relacional.

MODELO DE DATOS

Un **modelo** es una representación de cualquier aspecto o tema extraído del mundo real, en una base de datos esta representación es gráfica.

Un **modelo de datos** es un conjunto de conceptos que nos permiten describir los datos, las relaciones que existen entre ellos, la semántica y las restricciones de consistencia.

Propiedades de un Modelo de Datos

Las propiedades del modelo de datos son de dos tipos:

- **Estáticas o relativamente invariantes** en el tiempo, son las que responden a lo que se suele entender como estructuras.
- **Dinámicas** que son las operaciones que se aplican a los datos o valores almacenados en las estructuras, los cuales varían en el transcurso del tiempo al aplicarles dichas operaciones.

Estática

La estática de un modelo de datos está compuesta por:

- **Elementos permitidos**, no son los mismos para todos los modelos de datos, varían generalmente en la terminología, en general son:
 - **Objetos**: entidades, relaciones, registros, etc.
 - **Asociaciones** entre objetos: interrelaciones, set, etc.
 - **Propiedades** o características de los objetos o de las asociaciones: atributos, campos, elementos de datos, etc.
 - **Dominios** conjuntos nominales de valores sobre los que se definen propiedades.

La representación de estos elementos depende de cada modelo de datos, en el modelo jerárquico o Codasyl se utilizan los **grafos**, en el modelo relacional las **tablas**.

- **Elementos no permitidos o restricciones**. No todos los valores o cambios de valor o estructuras están permitidos en el mundo real, por ejemplo, un niño de 3 años no puede estar casado, ni trabajar, una persona no puede pasar de soltera a viuda directamente. De la misma forma el modelo de datos impone limitaciones a las estructuras que admite, por ejemplo, el modelo relacional no admite 2 filas iguales en una tabla.

Estas limitaciones que pueden venir impuestas por el propio modelo de datos o por el propio discurso que queremos modelar se llaman **restricciones**. Las restricciones impuestas por el propio modelo se llaman **restricciones inherentes** y las que responden al deseo de que el sistema de información sea un reflejo lo más fiel posible del mundo real son **las restricciones de integridad o semánticas**

Las restricciones inherentes no son determinadas por los usuarios sino que son propias del modelo y, por tanto varían de un modelo a otro; imponen rigideces a la hora de modelar, ya que no permiten describir ciertas estructuras.

Por ejemplo en el modelo relacional:

- No puede haber dos tuplas iguales
- El orden de las tuplas no importa
- El orden de los atributos no importa
- Cada atributo sólo puede tomar un valor en el dominio en el que está inscrito, es decir, no puede haber valores multivaluados.

Las restricciones de integridad son facilidades que se ofrecen al diseñador para que pueda representar los más fielmente posible la semántica de los datos.

Por ejemplo en el modelo relacional:

- **Clave primaria.** Hace que los atributos marcados como clave primaria no puedan repetir valores.
- **Unicidad.** Impide que los valores de los atributos marcados de esa forma, puedan repetirse.
- **Obligatoriedad.** Prohíbe que el atributo marcado de esta forma no tenga ningún valor
- **Integridad referencial.** Prohíbe colocar valores en una clave externa que no estén reflejados en la tabla donde ese atributo es clave primaria.
- **Regla de validación.** Condición que debe de cumplir un dato concreto para que sea actualizado.

Dinámica

Los valores que toman los distintos objetos de un esquema en un momento determinado t_i se llaman **ocurrencia** del esquema o **estado de la base de datos** en el tiempo t_i (BD_i); en otro momento t_j la ocurrencia del esquema será (BD_j). Si entre t_i y t_j se ha producido un cambio en algún valor de la base de datos (alta, baja o modificación), tanto los valores BD_i y BD_j deben ser ocurrencias válidas de la base de datos, es decir, ambas deben cumplir las restricciones de integridad así como las posibles restricciones asociadas al cambio de estado.

La componente dinámica del modelo consta de un conjunto de operadores que se definen sobre la estructura del correspondiente modelo de datos, ya que no todas las estructuras admiten el mismo tipo de operaciones. La aplicación de una operación a una ocurrencia de un esquema transforma a ésta en otra ocurrencia.

Una operación tiene dos componentes:

- **Localización** o enfoque o selección consiste en localizar una ocurrencia de u objeto indicando un camino, o bien un conjunto de ocurrencias especificando una condición.
- **Acción** que se realiza sobre la(s) ocurrencia(s) pedidamente localizada(s) mediante una operación de localización, y puede consistir en una recuperación o en una actualización (inserción, borrado o modificación).

La distinción entre localización y acción es de tipo formal, por ejemplo en lenguajes como LMD de Codasyl, tienen dos órdenes distintas, mientras que en SQL se reúnen ambas operaciones en un único operador.

CODASIL	SQL
LOCALIZACIÓN <condición>	SELECT
ACCIÓN <OBJETIVO>	FROM
	WHERE

MODELO RELACIONAL. TERMINOLOGÍA Y ESTRUCTURA.

A finales de los años 70, Codd introdujo la teoría matemática de las relaciones en el campo de las bases de datos, lo que supuso un importante paso en la investigación de los SGBD, aportando un fundamento teórico para el desarrollo de los nuevos sistemas.

Codd propuso un modelo de datos basado en la teoría de las relaciones, en donde los datos se estructuran lógicamente en forma de relaciones, "tablas", siendo el objetivo del modelo mantener la independencia de esta estructura lógica respecto al modo de almacenamiento y a otras características de tipo físico.

El trabajo de Codd presenta un modelo de datos, cuyos objetivos principales son:

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

Para conseguir estos objetivos Codd introduce el concepto de **relación (tabla)** como estructura básica del modelo. Todos los datos de una base de datos se representan en forma de relaciones cuyo contenido varía en el tiempo. Una relación, en terminología relacional, es un conjunto de **filas, tuplas**, con unas determinadas características.

Con respecto a la parte dinámica del modelo, se proponen un conjunto de **operadores** que se aplican a las relaciones. Algunos de estos operadores son clásicos de la teoría de conjuntos, intersección, unión, mientras que otros fueron introducidos por el modelo relacional. Todos forman el **álgebra relacional**, que fue definida por Codd en 1972.

La **teoría de la normalización**, elimina las dependencias entre atributos que originan anomalías en la actualización de la base de datos y proporciona una estructura más

regular en la representación de las relaciones, se utiliza para el diseño de las bases de datos.

Estructura del Modelo Relacional

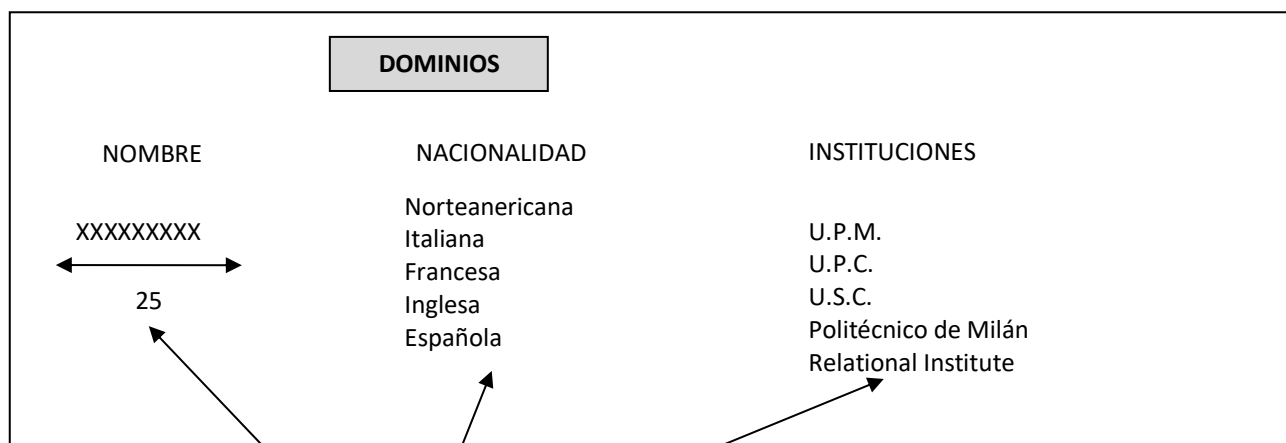
Relación o Tabla

La relación es el elemento básico del modelo relacional y se puede representar como una tabla.

NOMBRE

Atributo 1	Atributo 2	Atributo n	
XXXX	XXXX	XXXX	→ Tupla 1
XXXX	XXXX	XXXX	→ Tupla 2
.....
XXXX	XXXX		XXXX	→ Tupla n

En la tabla se puede distinguir su **nombre**, un conjunto de columnas, llamadas **atributos**, que representan propiedades de la tabla y que también están caracterizadas por su nombre, y un conjunto de filas llamadas **tuplas** que contienen los valores que toma cada uno de los atributos para cada elemento de la relación.



Autores			Atributos	Cardinalidad
Nombre	Nacionalidad	Institucion		
Date, C. J.	Norteamericana	Relational Institue	T	4
Codd, E.F.	Norteamericana	Relational Institue	U	
Cerl, S.	Italiana	Politécnico de Milán	P	
Saltor, F.	Española	U.P.C.	L	
*			A	
			S	

Grado 4

En la figura se representa una relación o tabla, en la que podemos ver:

- El **nombre** de la relación, **Autores**;

- Los **atributos**, Nombre, Nacionalidad, Institución;
- Los **dominios**, de donde los atributos toman sus valores, hay que tener en cuenta que varios atributos pueden tomar valores del mismo dominio,
- Las **tuplas** cada una de ellas contiene los valores que toman los atributos Nombre, Nacionalidad, e Institución para un determinado autor.
- El **grado** es el número de atributos
- La **cardinalidad** es el número de tuplas.

Una relación se puede representar en forma de tabla, tiene las siguientes características:

- No se admiten filas duplicadas.
- Las filas y las columnas no están ordenadas.
- Es plana, es decir, que en el cruce de una fila y de una columna sólo puede haber un valor, no se admiten atributos multivaluados.

Comparación de la terminología relacional con las tablas y ficheros

RELACIÓN	TABLA	FICHERO
TUPLA	FILA	REGISTRO
ATRIBUTO	COLUMNA	CAMPO
GRADO	Nº DE COLUMNAS	Nº DE CAMPOS
CARDINALIDAD	Nº DE FILAS	Nº DE REGISTROS

Dominio y Atributo

Un **dominio** D es un conjunto **finito** de valores homogéneos y atómicos V_1, V_2, \dots, V_n , caracterizado por un **nombre**, es **homogéneo** porque tienen que ser todos del mismo tipo, y **atómicos** porque son indivisibles en lo que al modelo se refiere, si se descompusiese perderían la semántica a ellos asociada.

Por ejemplo: el dominio de Nacionalidad tiene valores: Española, norteamericana, inglesa, etc., que son todas del mismo tipo y no se pueden dividir sin que se pierda su semántica, si descomponemos el valor "Española" en las letras "E", "S", "P", etc. Se perdería la semántica ya que las letras aisladamente no tienen el significado que tiene "Española" como un valor de nacionalidad.

Los dominios pueden definirse por **intensión** o por **extensión**.

Por ejemplo, el dominio de las edades de las personas activas se puede definir por **intensión** como entero de longitud dos comprendido entre 18 y 65, mientras que la definición del dominio nacionalidades por intensión sería pobre semánticamente, ya que permitiría la combinación de 10 letras aun cuando no formasen un nombre válido de nacionalidad; por ello, sería preferible definir este dominio por **extensión** con los nombres de las distintas nacionalidades que admitiésemos en nuestra base de datos.

Un **dominio compuesto** se puede definir como una combinación de dominios simples a la que se pueden aplicar ciertas restricciones de integridad. Por ejemplo, podemos

necesitar manejar de forma independiente los dominios Día, Mes y Año, pero además también necesitar un dominio compuesto por ellos, llamado fecha, al que se le podrían aplicar las adecuadas restricciones de integridad, para que no apareciesen valores no válidos de fecha: Otro ejemplo, podría ser el del nombre y los apellidos, que según qué tipo de aplicación necesitemos, se puede tratar de forma conjunta o separada.

Claves

Una clave candidata de una relación es un conjunto de atributos que identifican unívoca y mínimamente cada tupla de la relación. Por lo menos tiene que existir una clave candidata, ya que al ser una relación un conjunto no pueden existir dos tuplas iguales y, por tanto, el conjunto de todos los atributos siempre tiene que identificarse unívocamente a cada tupla.

Por ejemplo en la relación AUTORES, tiene una clave candidata Nombre, ya que suponemos que no existen dos autores con el mismo nombre.

Una relación puede tener más de una clave candidata, entre las cuales hay que distinguir:

- **Clave primaria:** es aquella clave candidata que el usuario escogerá, para identificar las tuplas de la relación. Cuando sólo existe una clave candidata, está será la clave primaria. En la relación AUTORES, será Nombre, ya que es la única clave candidata.
- **Claves alternativas:** son aquellas claves candidatas que no han sido escogidas como clave primaria. En la relación AUTORES, no existen claves alternativas.

Nombre del campo	Tipo de datos	
Codigo_Usuario	Texto	Identificación del usuario
Nombre	Texto	Nombre del usuario
Apellidos	Texto	Apellidos del usuario
DNI	Texto	DNI del usuario de la biblioteca
Domicilio	Texto	Domicilio del usuario
CodigoPostal	Texto	Codigo Postal del usuario
Ciudad	Texto	Ciudad donde vive el usuario
Telefono	Texto	Teléfono del usuario

[Propiedades del campo](#)

General		Búsqueda
Tamaño del campo	5	
Formato		
Máscara de entrada	00000	
Título		
Valor predeterminado		
Regla de validación		
Texto de validación		
Requerido	No	
Permitir longitud cero	No	
Indexado	Sí (Sin duplicados)	
Compresión Unicode	Sí	
Modo IME	Sin Controles	
Modo de oraciones IME	Nada	
Etiquetas inteligentes		

En este ejemplo el atributo DNI, es una clave alternativa.

- **Clave ajena** de una relación R2 a un conjunto no vacío de atributos cuyos valores han de coincidir con los valores de la clave candidata de una relación R1. La clave

ajena y la correspondiente clave candidata han de estar definidas sobre el mismo dominio.

Prestamos		
Nombre del campo	Tipo de datos	Descripción
IdPrestamo	Autonumérico	Número del préstamo
Codigo_libro	Texto	Código del libro prestado
Codigo_Usuario	Texto	Código del usuario al que se prestó el libro
Cantidad	Número	Cantidad de libros prestada
Fechaprestamo	Fecha/Hora	Fecha en que se hizo el prestamo
Fechadevolucion	Fecha/Hora	Fecha en que se tiene que devolver el libro
Estado	Sí/No	Libro devuelto (SI/No)
Propiedades del campo		
General	Búsqueda	
Tamaño del campo	Entero largo	
Nuevos valores	Incrementalmente	
Formato		
Título		
Indexado	No	
Etiquetas inteligentes		
Alineación del texto	General	

En este ejemplo los campos Codigo_libro es una clave ajena, es un atributo de la tabla Libros, además tiene otra clave ajena que es Codigo_Usuario, que a su vez es un atributo de la tabla Usuarios.

Teniendo en cuenta los distintos tipos de claves una tabla o relación se puede clasificar como:

- **Tabla referenciada:** su clave primaria es referenciada por una o más claves foráneas de la misma o diferente tabla.

Ejemplos: en la base de datos de la Biblioteca, las tablas Usuarios y Libros son referenciadas, porque la tabla Préstamos hace referencia a ellas.

- **Tabla dependiente:** tiene al menos una clave foránea.

Ejemplos: en la base de datos de la Biblioteca, la tabla Prestamos es dependiente de Usuarios y Libros porque tiene atributos que referencia a esas tablas.

- **Tabla independiente:** no tiene ninguna clave foránea.

Ejemplos: en la base de datos de la Biblioteca, la tabla Usuarios es independiente de sus atributos no hacen referencia a ninguna tabla.

- **Tabla autorreferenciada:** su clave primaria está referenciada por una clave foránea definida en esta misma tabla.

Ejemplos: en tabla siguiente DEPARTAMENTOS, que tiene como atributo clave De_codigo, hay otra columna De_Depar que representa el código del departamento del que depende, es decir, que hace referencia al atributo clave de la propia tabla. En el modelo E/R se expresaba como una relación reflexiva.

Departamentos		
Nombre del campo	Tipo de datos	Descripción
De_codigo	Número	Código del Departamento
De_codigocen	Número	Código del Centro al que está asignado el Departamento
De_Tipodir	Texto	Tipo del director Funciones o Propiedad
De_Presupuesto	Número	Presupuesto del departamento
De_Depar	Número	Código del departamento del que depende
De_Nombre	Texto	Nombre del departamento
De_director	Número	Código Empleado que es Director del departamento
Propiedades del campo		
General		
Tamaño del campo	Entero largo	
Formato		
Lugares decimales	Automático	
Máscara de entrada		
Título		
Valor predeterminado		
Regla de validación		
Texto de validación		
Requerido	No	
Indexado	No	
Etiquetas inteligentes		

Restricciones

En el modelo relacional, existen restricciones, es decir, estructuras u ocurrencias no permitidas, hay que distinguir entre restricciones inherentes y restricciones semánticas (de usuario). Los datos almacenados en la base de datos han de adaptarse a las estructuras impuestas por el modelo y han de cumplir las restricciones de usuario a fin de constituir una ocurrencia válida del esquema.

Restricciones inherentes son las que impone el propio modelo:

- No hay dos tuplas iguales (obligatoriedad de la clave primaria)
- El orden de las tuplas no es significativo.
- El orden de los atributos no es significativo.
- Cada atributo sólo puede tomar un único valor del dominio sobre el que está definido, no admitiéndose por tanto los grupos repetitivos.
- Restricción de integridad de entidad. Ningún atributo que forme parte de la clave primaria de una relación puede tomar un valor nulo.
- Restricción de integridad de dominio: un atributo no puede tomar valores que excedan el rango del dominio al que están asociados. Sólo serán comparables entre sí aquellos atributos asociados al mismo dominio. Una clave ajena debe pertenecer al mismo dominio que la clave primaria a la que referencia.

Ejemplo

AUTOR1

Nombre	Nacionalidad	Institución	Idiomas
Codd, E. F.	Norteamericana	Relational Institute	Inglés, Español
Date, C. J.	Norteamericana	Relational Institute	Inglés
Ceri, S.	Italiana	Politécnico de Milán	Italiano, Inglés
Saltor, F.	Española	U.P.C.	Español, Catalán

No cumple con la regla de que cada atributo debe tomar un solo valor, en algunos casos el atributo idiomas, se refiere a los idiomas que escribe un autor tiene dos valores. Solución

AUTOR1

Nombre	Nacionalidad	Institución	Idiomas
Codd, E. F.	Norteamericana	Relational Institute	Inglés,
Codd, E. F.	Norteamericana	Relational Institute	Español
Date, C. J.	Norteamericana	Relational Institute	Inglés
Ceri, S.	Italiana	Politécnico de Milán	Italiano,
Ceri, S.	Italiana	Politécnico de Milán	Inglés
Saltor, F.	Española	U.P.C.	Español,
Saltor, F.	Española	U.P.C.	Catalán

Si creásemos una tabla AUTOR1, la clave deberían ser Nombre + Idiomas.

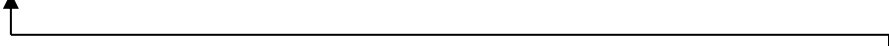
Restricciones semánticas:

Son restricciones de usuario, que permiten que el modelo pueda reflejar en el esquema, lo más fielmente posible, la semántica del mundo real. Las principales son:

- **Clave primaria** (PRIMARY KEY). Permite declarar un atributo o un conjunto de atributos como clave primaria de una relación, por lo que sus valores no se podrán repetir ni se admitirán los nulos (o valores ausentes). La obligatoriedad de la clave primaria es una restricción inherente; pero la declaración de un atributo como clave primaria de una relación es una restricción semántica que corresponde a la necesidad del usuario de imponer que los valores del conjunto de atributos que constituyen la clave primaria no se repitan en la relación, ni tomen valores nulos.
- **Unicidad** (UNIQUE): mediante la cual se indica que los valores de un conjunto de atributos (uno o más) no pueden repetirse en una relación. Estas restricciones permiten la definición de claves alternativas.
- **Obligatoriedad** (NOT NULL) de uno o más atributos, con lo que se indica que el conjunto de atributos no admite valores nulos.
- **Integridad referencial** (FOREIGN KEY) se define como el conjunto de mecanismos para conservar y garantizar de forma automática la integridad de todos los datos de los SGBD. Si una relación R2 (relación que referencia) tiene un atributo descriptor que es una clave candidata de la relación R1 (relación referenciada), todo valor de dicho descriptor debe concordar con un valor de la clave candidata de R1 o bien ser nulo. El descriptor es, una clave ajena de la relación R2. Las relaciones R1 y R2 no son necesariamente distintas. Hay que destacar que la clave ajena puede ser también parte o la totalidad de la clave primaria de R2.

Ejemplo: Las Relaciones EDITORIAL, tiene como atributo clave NombreEditorial y además es una clave ajena de la relación LIBRO que tiene como clave primaria el código del libro, se puede apreciar como los valores del atributo Editorial de la relación LIBRO concuerdan con los de la clave primaria NombreEditorial de la relación EDITORIAL.

EDITORIAL			
NombreEditorial	Dirección	País	Ciudad
Universal Books	Brown Sq. 23	EEUU	Los Ángeles
Rama	Canillas, 144	España	Madrid
Mc Graw-Hill	Basauri, 17	España	Madrid
Paraninfo	Virtudes, 7	España	Madrid
Eni	Pº Ferrocarriles Catalanes	España	Cornellá de Llobregat



LIBRO			
Codigo	Titulo	Editorial
00345D7	Fundamentos BBDD	Rama
1022305	UML 2	Eni
4939H2	Programación MySQL	Anaya
0045307	Domine SQL Server	Nulo
01123J3	Organización basada en Procesos	Rama

En el modelo relacional, se establecen las restricciones referenciales en la etapa de diseño conceptual. Etapa que es muy importante, ya que de ella depende el correcto funcionamiento y la integridad de la aplicación. Esto conlleva un mayor consumo de recursos del sistema, ya que el SGBD debe realizar gran cantidad de validaciones para garantizar el cumplimiento de las restricciones referenciadas establecidas.

Si una relación tiene un atributo que referencia a un atributo de otra relación, cualquier valor que tome el atributo de la primera relación debe existir en la segunda relación.

Además de definir las claves ajenas hay que determinar las consecuencias que pueden tener ciertas operaciones, inserción, borrado o modificación, realizadas sobre tuplas de la relación referenciada.

Reglas de la Integridad Referencial

Los SGBD siguen una serie de reglas para soportar la integridad de los datos mediante la integridad referencial:

- INSERT (Inserción de nuevas tuplas o filas)
- DELETE (borrado de tuplas o filas)
- UPDATE (modificación de algún atributo de una tupla)

Estas reglas se aplican al realizar las operaciones SQL (Sturcture Query Language) con el mismo nombre. Dentro de cada una de ellas existen unas opciones de integridad, impuestas por el SGBD en algunos casos:

- RESTRICT
- SET NULL
- SET DEFAULT
- CASCADE

Estas opciones se aplican en operaciones realizadas sobre claves primarias y foráneas.

Opción RESTRICT (NO ACTION)

Es la opción válida por defecto, es decir, si no se indica ninguna opción, el sistema tomará ésta. Con esta opción se impide que se propague la operación a otras tablas relacionadas con la afectada. Si la operación se aplica sobre:

- **La tabla referenciada:** esta opción sólo es aplicable a las operaciones (reglas de integridad) de actualización y borrado. No se permitirá actualizar o borrar una clave primaria que exista en la tabla dependiente como clave foránea.

Tabla Alumnos (Referenciada)

Cod_alumno	Nombre
01		
02		
03		
.....		

Tabla Matriculados (dependiente)

Cod_alumno	Cod_curso
01	100	
02	101	
03	100	
.....		

En la tabla referenciada Alumnos no podemos borrar, ni actualizar o modificar el alumno 01, porque existe en la tabla dependiente Matriculados

- **La tabla dependiente:** esta opción sólo es aplicable a las operaciones (reglas de integridad) de actualización e inserción. Sólo se permitirá insertar o modificar una clave foránea si existe previamente en la tabla referenciada como clave primaria.

Tabla Alumnos (Referenciada)

Cod_alumno	Nombre
01		
02		
03		
.....		

Tabla Matriculados (dependiente)

Cod_alumno	Cod_curso
01	100	
02	101	
03	100	
.....		

En la tabla dependiente Matriculados si el alumno que se ha matriculado del curso 101 hay que cambiarlo, el código nuevo debe existir en la tabla Alumnos. En el caso de un nuevo alumno matriculado su código debe existir en la tabla Alumnos por ejemplo El alumno 01 sería válido y el alumno 04 no se podría insertar.

Opción SET NULL

Se aplica siempre sobre la clave foránea. Con esta opción se ponen a valor nulo los valores de las claves foráneas que coincidan con la clave primaria que va a ser modificada o borrada.

Tabla Alumnos (Referenciada)

Cod_alumno	Nombre
01		
02		
03		
.....		

Tabla Matriculados (dependiente)

Cod_alumno	Cod_curso
01	100	
02	101	
03	100	
.....		

Si eliminamos el alumno 01 de la tabla de alumnos la tabla de matriculados quedaría

Cod_alumno	Cod_curso
	100	
02	101	
03	100	
.....		

Opción SET DEFAULT

Se aplica siempre sobre la clave foránea. Con esta opción se ponen al valor por defecto los valores de las claves foráneas que coincidan con la clave primaria que va a ser modificada o borrada. El valor por defecto habrá sido definido al crear la tabla correspondiente

Tabla Alumnos (Referenciada)

Cod_alumno	Nombre
01		
02		
03		
.....		

Tabla Matriculados (dependiente)

Cod_alumno	Cod_curso
01	100	
02	101	
03	100	
.....		

Si eliminamos el alumno 01 de la tabla de alumnos la tabla de matriculados quedaría, si en la tabla de matriculados el valor por defecto es 00

Cod_alumno	Cod_curso
00	100	
02	101	
03	100	
.....		

Opción CASCADE

Con esta opción se permite la propagación de la operación a las tablas relacionadas con la tabla afectada. Se aplica sobre las tablas dependientes nunca sobre las tablas referenciadas. Esta opción se aplica sobre las operaciones (reglas de integridad) de actualización y borrado.

Cuando se actualiza o borra una fila de la tabla referenciada, se actualizan o borran de forma automática todas las filas de la tabla dependiente cuya clave foránea coincida con el valor de la clave primaria de la fila sobre la que se realizó la operación.

Hay que tener cuidado con esta opción ya que puede provocar borrados masivos no deseados.

Tabla Alumnos (Referenciada)

Cod_alumno	Nombre
01		
02		
03		
.....		

Tabla Matriculados (dependiente)

Cod_alumno	Cod_curso
01	100	
02	101	
03	100	
.....		

Si eliminamos el alumno 01 de la tabla de alumnos también se eliminará de la tabla de matriculados que quedaría

Cod_alumno	Cod_curso
02	101	
03	100	
.....		

Regla INSERT

Sólo se aplica en las filas que se insertan en una tabla dependiente. Se aplica automáticamente por lo que no se puede aplicar ninguna de las opciones de integridad. Cuando se inserta una fila en una tabla dependiente, la clave foránea puede ser:

- **Nula:** si está permitido en alguno de las propiedades que componen dicha clave. (Requerido).

- **Igual a** un valor de una clave primaria a la que referencia. Puede ser única o no serlo (Admite o no duplicados)

Tabla Alumnos (Referenciada)

Cod_alumno	Nombre
01		
02		
03		
.....		

Tabla Matriculados (dependiente)

Cod_alumno	Cod_curso
01	100	
02	101	
03	100	
.....		

Si insertamos el alumno 03 en la tabla matriculados no hay ningún problema ya que existe en la tabla alumnos.

Si insertamos el alumno 04 en la tabla matriculados no lo permitiría ya que no existe en la tabla alumnos.

Si en la tabla matriculados en el campo Cod_alumno no es requerido, es decir, que se permite la entrada en blanco, en este caso podríamos no escribir nada.

Regla DELETE

En esta regla son aplicables todas las opciones de integridad vistas anteriormente. Se aplica en las operaciones de borrado efectuadas sobre una tabla referenciada. Al intentar borrar una fila de una tabla referenciada, se comprobarán todas las condiciones impuestas por la opción de integridad definida, y sólo se efectuará el borrado si dichas condiciones se cumplen.

Las opciones que se aplican a esta regla son:

- **DELETE RESTRICT**

No se permitirá borrar una fila de una tabla referenciada si existen filas en una tabla dependiente de la anterior, cuya clave foránea coincida con el valor de la clave primaria de la fila que se quiere borrar.

Para poder realizar esta operación hay 2 opciones:

- Borrar las filas dependientes y después borrar la fila referenciada.
- Actualizar los valores de la clave foránea con otros diferentes a los de la clave primaria a borrar y después borrar la clave referenciada.

- **DELETE CASCADE**

Al borrar una fila de la tabla referenciada se produce un borrado en cadena en la tabla dependiente, es decir, se borran todas las filas dependientes (aquellas cuya clave foránea coincida con la clave primaria de la tabla a la que referencian).

Hay que tener cuidado con esta opción ya que puede provocar borrados masivos no deseados.

- **DELETE SET NULL**

Al borrar una fila de una tabla referenciada con una restricción de este tipo se producen las siguientes acciones:

- Borrado de la fila de la tabla referenciada.
- Se pone a nulo el valor de la clave foránea de las filas dependientes.

Una vez realizadas estas operaciones, las filas afectadas de la tabla dependiente no sufren validaciones de integridad hasta que el valor de su clave foránea se actualice con un valor que corresponda al de alguna clave primaria de la tabla referenciada.

Cuando una operación depende de varias restricciones referenciadas falla si no se cumple alguna de ellas. Las opciones de integridad CASCADE y SET NULL no provocan fallos, pero RESTRICT si que puede incumplirse. Por lo tanto, esta opción es la que decide si la operación se realiza o no.

Regla UPDATE

Se aplica en operaciones realizadas sobre tablas referenciadas y tablas dependientes. Si se trata de una tabla referenciada se aplica automáticamente la opción RESTRICT, y no se puede aplicar ninguna de las otras opciones de integridad. Sin embargo, si se trata de una tabla dependiente, se pueden aplicar todas las opciones de integridad definidas.

- **UPDATE RESTRICT**

No se permite actualizar el valor de la clave primaria de una tabla referenciada mientras existan claves foráneas con el mismo valor en tablas dependientes.

Para realizar esta operación hay 2 opciones:

- Borrar las filas dependientes con dicho valor de clave foránea.
- Actualizar el valor de las claves foráneas con un valor diferente al de la clave primaria y después actualizar el valor de dicha clave primaria.

- **UPDATE CASCADE**

Al actualizar el valor de una clave primaria de una tabla referenciada, se propaga el cambio a las claves foráneas de las tablas dependientes.

- **UPDATE SET NULL**

Al actualizar la clave primaria de una fila referenciada, el funcionamiento es el mismo que el de la regla DELETE con la opción SET NULL, es decir, que en la tabla dependiente el valor de la clave foránea se pone a nulo, excepto que en lugar de borrar la fila de la tabla referenciada, se actualiza la clave primaria con un nuevo valor.

Además de las restricciones anteriores existen en el modelo relacional otras restricciones que se llaman de **rechazo**.

Restricciones de rechazo

Las restricciones de rechazo son aquellas en las que el usuario pone una condición mediante un predicado definido sobre un conjunto de atributos, de tuplas o dominios, el

cual debe ser verificado por los correspondientes objetos en toda operación de actualización para que el nuevo estado sea una ocurrencia válida del esquema; en el caso de que la operación no cumpla la condición se impide que la operación se lleve a cabo, es decir, que se produce un rechazo.

- **CHECK** o restricción de Verificación

Comprueba, en todas las operaciones de actualización, si el predicado es cierto o falso y, en el segundo caso, rechaza la operación. La restricción de verificación se define sobre un único elemento (incluyéndose en la definición de dicho elemento) y puede no tener nombre. Por ejemplo: la edad tiene que estar comprendida entre 10 y 50 años.

- **ASSERTION** o restricción de aserción

Actúa de forma idéntica a la Verificación, pero se diferencia de ella en que puede afectar a varios elementos (por ejemplo, a dos relaciones distintas) y su definición; por tanto, no va unida a la de un determinado elemento, por lo que siempre ha de tener un nombre, ya que la aserción es un elemento más del esquema que tiene vida por sí mismo. Por ejemplo no hay ningún alumno matriculado en el curso de Access que pague más de 200€.

- **TRIGGER** o disparadores

Se utilizan cuando interesa especificar una acción distinta del rechazo cuando no se cumple una determinada restricción semántica. Los disparadores nos permiten además de indicar una condición, especificar la acción que queremos se lleve a cabo si la condición se hace verdadera. Los disparadores pueden interpretarse como reglas de tipo evento-condición-acción (ECA) que pueden interpretarse como reglas que especifican que cuando se produce un evento, si se cumple una condición, entonces se realiza una acción. Los disparadores no los soportan todos los SGBD. Por ejemplo se podría definir un disparador que “informará al administrador de la base de datos cuando haya algún alumno del curso de Word que pague más de 100€.

LOS VALORES NULOS EN EL MODELO RELACIONAL

Concepto de valor nulo

Un valor **Nulo** se puede definir como una señal utilizada para representar información desconocida, inaplicable, inexistente, no válida, no proporcionada, indefinida, etc. Codd en 1990 propuso abandonar el término valor nulo y sustituirlo por el de **marca** ya que:

- Los SGBDR no deberían tratar las marcas como si fueran cualquier otro valor.
- Al introducir la lógica **cuatrivaluada**, se desdobra el concepto de valor nulo.
- Algunos lenguajes anfitriones tratan objetos que denominan **nulos** pero son diferentes al significado de las **marcas**.
- Es más fácil decir **marcado** y **sin marcar** que **anulado** o **nulificado**, etc.

La necesidad de los valores nulos o marcas en las bases de datos es evidente por las siguientes razones:

- Crear tuplas o filas con ciertos atributos desconocidos en ese momento, por ejemplo, al año de edición de un libro.
- Añadir un nuevo atributo a una relación existente, atributo que en el momento de añadirse, no tendrá ningún valor para las tuplas de la relación.
- Atributos inaplicables a ciertas tuplas, por ejemplo, la editorial para un artículo, ya que un artículo no tiene editorial, o la profesión de un menor.

El tratamiento de valores nulos exige definir **Operaciones de comparación, aritméticas, algebraicas, Funciones de agregación** de forma específica para el caso de que alguno de los operandos tome valores nulos y obliga también a introducir nuevos operadores especiales.

En las operaciones de comparación se plantea el problema de saber si dos valores nulos son o no iguales. No se puede decir que es cierto que sean iguales puesto que estaríamos afirmando que no son tan desconocidos, tampoco que es falso que sean iguales, la única solución es decir que “**quizás**” sean iguales. Por ello surge la lógica trivaluada (L3V), diferente a la lógica habitual bivaluada (L2V)

Tablas de verdad de la lógica Trivaluada

A	B	A AND B	A OR B
V	V	V	V
V	F	F	V
V	Q	Q	V
F	V	F	V
F	F	F	F
F	Q	F	Q
Q	V	Q	V
Q	F	F	Q
Q	Q	Q	Q

A	NOT A
V	F
F	V
Q	Q

DATE en 1995 introduce dos nuevos **operadores especiales**:

- **ES NULO (IS NULL)** que toma el valor verdadero si el operando es nulo y falso en caso contrario.
- **SI NULO (IF NULL)** que se aplica a dos operandos y devuelve el valor del primero, salvo que sea nulo, en cuyo caso devuelve el segundo.

En cuanto a las **operaciones aritméticas** con valores nulos, se considera nulo el resultado de sumar, restar, multiplicar o dividir cuando alguno de los operandos toma el valor nulo.

Otro aspecto es el de la evaluación de la igualdad o desigualdad de dos tuplas; dos tuplas son iguales si atributo a atributo ambos son iguales y no nulos o ambos nulos.

Los valores nulos tienen incidencia en las funciones de agregación a la hora de calcular valores estadísticos, media, varianza, etc. En estos casos hay que establecer cómo

afectan los valores nulos, por ejemplo, si calculamos la media, como actúa el valor nulo, se tiene en cuenta como si fuese cero o no, en ambos casos daría resultados diferentes.