Temas Generales para la preparación de la Oposición al Cuerpo Superior de Sistemas y Tecnologías de la Información de la Administración del Estado.

Cuerpo Superior de Estadísticos del Estado
Especialidad de Estadística-Ciencia de Datos.

Almacenamiento y modelos de datos

Tema 2. El modelo relacional

AUTOR: **Leonor Torres Moreno**

Asociación Profesional de Cuerpos Superiores de Sistemas y Tecnologías de la Información de las Administraciones Públicas

Creación: Agosto 2021

ÍNDICE

1 Introducción

Una base de datos es una recopilación organizada de información o datos estructurados relevante para una organización, que se almacena de forma electrónica en un sistema informático. Normalmente, una base de datos está controlada por un sistema de gestión de bases de datos (SGBD o DBMS) que consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a dichos datos. En conjunto, los datos y el SGBD, junto con las aplicaciones asociadas a ellos, reciben el nombre de sistema de bases de datos. El objetivo principal de un SGBD es proporcionar una forma de almacenar y recuperarla información de una base de datos de manera que sea tanto práctica como eficiente.

Los sistemas de bases de datos se diseñan para gestionar grandes cantidades de información. La gestión de los datos implica tanto la definición de estructuras para almacenar la información como la provisión de mecanismos para la manipulación de la información. Además, los sistemas de bases de datos deben proporcionar la fiabilidad de la información almacenada, a pesar de las caídas del sistema o los intentos de acceso no autorizados. Si los datos se van a compartir entre diversos usuarios el sistema debe evitar resultados anómalos.

Antes de la llegada de los SGBD, las organizaciones almacenaban la información en sistemas de ficheros, pero mantener la organización en estos sistemas tenía una serie de inconvenientes importantes como: redundancia e inconsistencia de datos, aislamiento, concurrencia, seguridad, etc.

Uno de los propósitos principales de un SGBD es proporcionar a los usuarios una visión abstracta de los datos. Es decir, el sistema oculta ciertos detalles de cómo se almacenan y mantienen los datos. Por esta razón la arquitectura de un SGBD, generalmente, se basa en la arquitectura de tres niveles (externo, conceptual e interno) ANSI-SPARC (Ilustración 1).

Se trata de separar la forma en que los usuarios ven los datos, de los detalles de almacenamiento físico de los mismos. Este principio de **independencia de datos** hace posible que el administrador de la base de datos cambie la estructura física (nivel interno) sin que la manera en la cual los diferentes usuarios ven los datos (nivel externo) se afecte. El nivel interno describe la forma como los datos se almacenan en la base de datos (i.e. estructuras de datos, espacios de almacenamiento, índices, formato de registros), trata con los mecanismos de almacenamiento físico que el sistema operativo utiliza (dispositivos físicos). El nivel conceptual, representado en la arquitectura, corresponde a la descripción de los datos y de las relaciones entre éstos. A este nivel, la base de datos se ve como la integración de todas las vistas de los usuarios de la base de datos. En el nivel externo se representa cada una las partes de la base de datos que es relevante para cada uno de los diferentes usuarios.

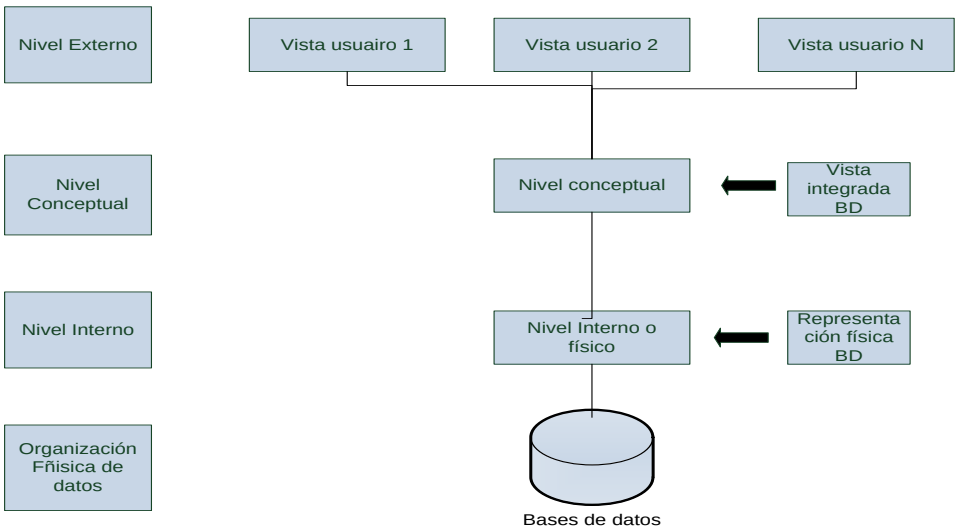


Ilustración 1 Arquitectura ANSI-SPARC

Los modelos de bases de datos han evolucionado drásticamente desde su inicio a principios de la década de 1960. Empezaron desde el modelo de la base de datos jerárquica (que se basaba en un modelo de árbol y permitía una relación de uno a muchos), más tarde el de red (un modelo más flexible

que permitía relaciones múltiples). Aunque eran sencillos, estos primeros sistemas eran inflexibles. Cada aplicación almacenaba datos en su propia estructura única. Cuando los desarrolladores querían crear aplicaciones para usar esos datos, tenían que conocer muy bien esa estructura de datos concreta a fin de encontrar los datos que necesitaban. Esas estructuras de datos eran poco eficaces, el mantenimiento era complicado y era difícil optimizarlas para ofrecer un buen rendimiento en las aplicaciones. El modelo de base de datos relacional se diseñó para resolver el problema causado por estructuras de datos múltiples y arbitrarias, basándose en la lógica de predicados y en la teoría de conjuntos.

En 1970, Edgar Frank Codd propone el modelo relacional, que no tardó en consolidarse como un nuevo paradigma en los modelos de base de datos. Donde la estructura de los datos se basa en relaciones o tablas, formadas por atributos (o columnas) de datos y cada tupla (o fila) contiene un valor por atributo.

El modelo relacional proporcionó una forma estándar de representar y consultar datos que podía utilizarse en cualquier aplicación. Desde el principio, los desarrolladores se dieron cuenta de que la virtud principal del modelo de base de datos relacional era el uso de tablas, ya que era una forma intuitiva, eficiente y flexible de almacenar y acceder a información estructurada.

En la década de 1980 es cuando se hicieron populares las bases de datos relacionales, y siguen siendo actualmente el principal modelo de datos. Ha conseguido mantener esta posición y ser superior a enfoques tradicionales, debido a su simplicidad, que facilita el trabajo de desarrolladores en comparación con otros modelos, por ejemplo, mantener la uniformidad de los datos en todas las aplicaciones y copias de la base de datos (llamadas instancias).

Otra de las ventajas del modelo es el uso del lenguaje de consulta estructurado (SQL) para escribir y hacer consultas en una base de datos: Durante muchos años, el SQL se ha utilizado como el lenguaje para realizar consultas en bases de datos. Se basa en el álgebra relacional y proporciona un lenguaje matemático de uniformidad interna que facilita la mejora del rendimiento de todas las consultas en bases de datos.

El modelo relacional es un modelo vigente, flexible, potente y que sirve de complemento a otros enfoques contemporáneos como bases de datos orientada a objetos, las bases de datos No-SQL o una base de datos EAV(modelo Entidad- Atributo- Valor).

2 Historia del modelo relacional

El modelo relacional fue propuesto por primera vez por Edgard Frank Codd en su artículo *"A relational model of data for large shared data banks"* (Codd, 1970). Este documento causó gran impacto y se acepta ahora como un hito en los sistemas de bases de datos, aunque ya se había propuesto un modelo orientado a conjuntos por el matemático holandés David L. Childs, que estudiaba los problemas de rendimiento en las estructuras de almacenamiento en la informática. En agosto de 1968, publicó el documento *"Feasibility of a set-theoretical data structure*", de creación propia, sin referencia a ningún otro. Childs afirma que se puede expresar cualquier pregunta con solo tres funciones básicas: la selección, la relación y el agrupamiento.

Codd tuvo una idea sobresaliente, al pensar que, gracias al almacenamiento de datos organizados en relaciones, la teoría de Childs puede implementarse en un lenguaje universal que denominó: "universal data sub-lenguaje".

Lo que Codd intentaba fundamentalmente era evitar que los de la base de datos tuvieran que verse obligadas a aprender los entresijos internos del sistema, como pasaba con el modelo dominante en aquel momento, el modelo en red, que era bastante físico. Su enfoque fue revolucionario al evitar conceptos del mundo de la computación en su modelo.

Los objetivos del modelo relacional se especificaron de la siguiente manera:

- Permitir un alto grado de independencia de los datos. Los programas de aplicación no deben verse afectados por las modificaciones de la representación interna de los datos.
- Proporcionar una base sustancial para tratar los problemas de semántica, consistencia y redundancia de los datos. En particular, el documento de Codd introdujo el concepto de relaciones normalizadas, es decir, relaciones que no tienen grupos repetidos.
- Permitir la expansión de los lenguajes de manipulación de datos orientados a conjuntos.

Aunque el interés por el modelo relacional procedía de varias direcciones, las investigaciones más importantes pueden atribuirse a tres proyectos con perspectivas bastante diferentes:

- En el Laboratorio de Investigación de San José de IBM, en California, fue el prototipo de DBMS relacional System R. Este proyecto fue diseñado para demostrar la viabilidad del modelo relacional proporcionando una implementación de sus estructuras de datos y operaciones.
- Proyecto INGRES (Interactive Graphics Retrieval System) de la Universidad de California en Berkeley.
- Proyecto Peterlee Relational Test Vehicle en el IBM UK Scientific Centre de Peterlee (Todd, 1976). Este proyecto tenía una orientación más teórica que los proyectos System R e INGRES y fue importante, principalmente para la investigación de cuestiones como el procesamiento de consultas y la optimización, así como la extensión funcional.

En 1974, dos informáticos de IBM D.D. Chamberlin y R.F Boyce publican "Sequel: a estructured english query lenguaje". Surgen las primeras palabras del lenguaje SQL, traduciendo las tres funciones de conjuntos: selección por SELECT/WHERE, relación por FROM y agrupamiento por GROUP by.

En 1974, D.D. Chamberlin se asocia con JN Gray e IL Traiger y publica "Views, authorization and looking in a relational base data system. Esta publicación supuso una innovación porque introduce todos los elementos del futuro lenguaje SQL: la creación de tablas, INSERT, UPDATE, DELETE, GRANT, REVOKE, VIEW. Los conceptos de cerradura y combinación interna hacen su aparición, permitiendo la creación del primer motor de SQL.

En 1976, Chamberlin lo resume todo en el libro: "Sequel 2: A unified Approach to Data Definition"

En 1978 IBM cuenta con todas las ventajas para crear la primera base de datos relacional, los diseñadores Codd y Chamberlin y un prototipo llamado System R. Sin embargo, estaba centrado en el mercado de los mainframes y no se apresuró a desarrollarlas comercialmente. Fueron otras empresas las que implementaros sus teorías, en especial Larry Elison quien aprovechó la oportunidad y comercializa la primera base de datos relacional llamada Oracle e implementa el lenguaje "Sequel" que pasan a denomina Structured Query languaje.

3 Reglas de Codd

La definición práctica que hace Edgar F. Codd en 1970 resultaba poco clara de lo que era y no era un Sistema de Base de Datos Relacional (SGBDR). Así en la década de los 80's comenzaron a aparecer numerosos SGBDR que se anunciaban como relacionales. Sin embargo, no lo eran exactamente y carecían de muchas características que se consideran importantes en un sistema relacional, perdiendo muchas de las ventajas del modelo. Por ejemplo, había sistemas que simplemente utilizaban tablas para almacenar la información, no disponiendo de elementos como claves primarias, etc.

Por ello el propio Dr. Codd escribió en 1985 un artículo estableciendo doce reglas a seguir por cualquier base de datos que fuera "relacional". Estas reglas fueron precedidas de la "Regla de Fundación" o "Regla Cero"(por lo que realmente son 13 reglas). Y se han convertido en la definición teórica de una base de datos relacional. Estas se derivan del trabajo teórico de Codd sobre el modelo relacional y representan realmente más un objetivo ideal que una definición de una base de datos relacional. En la práctica algunas de estas reglas son difíciles de implementar, así que un sistema podrá considerarse más relacional cuanto más siga estas reglas.

0. **REGLA 0: REGLA DE FUNDACIÓN.** *Para que un sistema se denomine Sistema de Gestión de Bases de Datos Relacionales, este sistema debe usar exclusivamente sus capacidades relacionales para gestionar la base de datos.*

1. **REGLA 1: REGLA DE LA INFORMACIÓN.** *Toda la información en una base de datos relacional se representa explícitamente en el nivel lógico mediante tablas y sólo mediante tablas.*

Por tanto, los metadatos (diccionario, catálogo) se representan y se manipulan exactamente igual que los datos de usuario, usando el mismo lenguaje. No hay información que no esté en tablas.

2. **REGLA 2: REGLA DEL ACCESO GARANTIZADO.** *Todos los datos son accesibles sin ambigüedad. Para todos y cada uno de los datos (valores atómicos) de una base de datos relacional se garantiza que son accesibles a nivel lógico utilizando una combinación de nombre de tabla, valor de clave primaria y nombre de columna*.

Cualquier dato almacenado en una base de datos relacional tiene que poder ser direccionado unívocamente. Para ello hay que indicar en qué tabla está, cuál es la columna y cuál es la fila (mediante la clave primaria). Si a un dato no podemos acceder de esta forma, no estamos usando un modelo relacional

3. **REGLA 3: TRATAMIENTO SISTEMÁTICO DE VALORES NULOS.** *Se debe disponer de valores nulos, distintos de la cadena vacía, blancos, 0, etc., para representar información desconocida o no aplicable de manera sistemática, independientemente del tipo de datos.*

Se reconoce la necesidad de la existencia del valor nulo, el cual podría servir para representar, o bien, una información desconocida, o bien una información que no aplica.

Hay problemas para soportar los valores nulos en las operaciones relacionales, especialmente en las operaciones lógicas, para lo cual se considera una lógica trivaluada, con tres (no dos) valores de verdad: Verdadero, Falso y null.

4. **REGLA 4: CATÁLOGO DINÁMICO EN LÍNEA BASADO EN EL MODELO RELACIONAL.** *La descripción de la base de datos se representa a nivel lógico de la misma manera que los datos ordinarios, de modo que los usuarios autorizados pueden aplicar el mismo lenguaje relacional a su consulta, igual que lo aplican a los datos ordinarios.*

Los metadatos se almacenan y se manejan usando el modelo relacional con todas las consecuencias.

5. **REGLA 5: REGLA DEL SUBLENGUAJE DE DATOS COMPLETO**. *Un sistema relacional debe soportar varios lenguajes y varios modos de uso de terminal (ejemplo: rellenar formularios, etc.). Sin embargo, debe existir al menos un lenguaje cuyas sentencias sean expresables, mediante una sintaxis bien definida, como cadenas de caracteres y que sea completo, soportando:*

- *Definición de datos.*
- *Definición de vistas.*
- *Manipulación de datos (interactiva y por programa).*
- *Restricciones de integridad.*
- *Restricciones de transacciones (begin, commit, rollback).*
- *Autorizaciones*

Además de poder tener interfaces más amigables para hacer consultas, etc. siempre debe haber una manera de hacerlo todo de manera textual, que es tanto como decir que pueda ser incorporada en un programa tradicional. Un lenguaje que cumple esto en gran medida es SQL.

6. **REGLA 6: REGLA DE ACTUALIZACIÓN DE VISTAS** *Todas las vistas que son teóricamente actualizables se pueden actualizar también por el sistema.*

El problema es determinar cuáles son las vistas teóricamente actualizables, ya que no está muy claro.

Cada sistema puede hacer unas suposiciones particulares sobre las vistas que son actualizables.

7. **REGLA 7: INSERCIÓN, ACTUALIZACIÓN Y BORRADO DE ALTO NIVEL.** *La capacidad de manejar una relación base o derivada como un solo operando se aplica no sólo a la recuperación de los datos (consultas), sino también a la inserción, actualización y borrado de datos.*

Esto es, el lenguaje de manejo de datos también debe ser de alto nivel (de conjuntos). Algunos sistemas de bases de datos inicialmente sólo podían modificar las filas de una tabla de una en una (un registro cada vez).

8. **REGLA 8: INDEPENDENCIA FÍSICA DE DATOS** *Los programas de aplicación y actividades del terminal permanecen inalterados a nivel lógico cualesquiera sean los cambios efectuados, tanto en la representación del almacenamiento, como en los métodos de acceso*.

El modelo relacional es un modelo lógico de datos, y oculta las características de su representación física.

9. **REGLA 9: INDEPENDENCIA LÓGICA DE DATOS.** *Los programas de aplicación y actividades del terminal permanecen inalterados a nivel lógico cualesquiera sean los cambios que se realicen a las tablas base que preserven la información.*

Cuando se modifica el esquema lógico preservando información (no valdría, por ejemplo, eliminar un atributo) no es necesario modificar nada en niveles superiores.

Ejemplos de cambios que preservan la información:

- Añadir un atributo a una tabla base.
- Sustituir dos tablas base por la unión de las mismas.

10. **REGLA 10: INDEPENDENCIA DE INTEGRIDAD** *Las restricciones de integridad específicas para una determinada base de datos relacional deben poder ser definidos en el sublenguaje de datos relacional, y almacenables en el catálogo, no en los programas de aplicación*.

El objetivo de las bases de datos no es sólo almacenar los datos, sino también sus relaciones y evitar que estas restricciones se codifiquen en los programas. Por tanto en una base de datos relacional se deben poder definir restricciones de integridad.

Cada vez se van ampliando más los tipos de restricciones de integridad que se pueden utilizar en los Sistemas de Gestión de Bases de Datos Relacionales, aunque hasta hace poco eran muy escasos.

Como parte de las restricciones inherentes al modelo relacional (forman parte de su definición) están:

- Integridad de Entidad: Toda tabla debe tener una clave primaria.
- Integridad de Dominio: Toda columna de una tabla contendrá valores exclusivamente de un determinado dominio (conjunto de valores válidos)
- Integridad Referencial: Toda clave foránea no nula debe existir en la relación donde es clave primaria.

11. **REGLA 11: INDEPENDENCIA DE DISTRIBUCIÓN** *Una Base de Datos Relacional es independiente de la distribución.*

Las mismas órdenes y programas se ejecutan igual en una base de datos centralizada que en una distribuida.

Las bases de datos son fácilmente distribuibles.

Esta regla es responsable de tres tipos de transparencia de distribución:

- Transparencia de Localización. El usuario tiene la impresión de que trabaja con una base de datos local. (Regla de Independencia Física)
- Transparencia de Fragmentación: El usuario no se da cuenta de que la relación con que trabaja está fragmentada. (Regla de Independencia Lógica).
- Transparencia de Replicación: El usuario no se da cuenta de que pueden existir copias (réplicas) de una misma relación en diferentes lugares.

12. **REGLA 12: REGLA DE LA NO SUBVERSIÓN** *Si un sistema relacional tiene un lenguaje de bajo nivel (un registro a la vez), ese bajo nivel no puede ser usado para subvertir (saltarse) las reglas de integridad y las restricciones expresadas en los lenguajes relacionales de más alto nivel (una relación a la cada vez).*

Algunos problemas no se pueden solucionar directamente con el lenguaje de alto nivel.

Normalmente se usa SQL incorporado en un lenguaje anfitrión para solucionar estos problemas. Se utiliza el concepto de cursor para tratar individualmente las filas de una tabla. En cualquier caso no debe ser posible saltarse las restricciones de integridad impuestos al tratar las filas a ese nivel.

4 Fundamentos del modelo relacional

4.1 Características

El modelo relacional está basado en la lógica de predicados y la teoría matemática de conjuntos y su objetivo fundamental es mantener la independencia de la estructura lógica respecto al modo de almacenamiento y otras características de tipo físico. El modelo relacional presenta las siguientes ventajas:

- Flexibilidad, ya que cada usuario puede definir los datos de la forma que más le convenga.
- Uniformidad, pues todos los datos se representan de la misma forma en la base de datos.
- Sencillez, tanto para definir la base de datos como para manipularla o realizar consultas.

La idea fundamental es el uso de relaciones. Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados tuplas. Pese a que esta es la teoría de las bases de datos relacionales creadas por Codd, la mayoría de las veces se conceptualiza de una manera más fácil de imaginar, pensando en cada relación como si fuese una tabla que está compuesta por registros (cada fila de la tabla sería un registro o "tupla") y columnas (también llamadas “atributos” o "campos").

Una base de datos relacional se compone de:

- Metadatos (también llamados catálogo o diccionario de datos): Son los elementos en la base de datos que describen la propia estructura de esta.
- Objetos de la base de datos: Tablas, relaciones entre ellas y cualquier otro objeto que se almacena en la base de datos: vistas, índices, secuencias, etc.
- Reglas de integridad: que ayudan a definir el dominio de conocimiento almacenado.

4.2 Conceptos

El modelo relacional se basa en el concepto matemático de **relación**, que normalmente se representa como una tabla.

El esquema de una base de datos relacional se compone de uno o más esquemas de relación y de un conjunto de restricciones de integridad.

Relación: se define con un nombre seguido de los nombres de los atributos: Nombre_Relación (Atributo1, atributo2,…Atributo N) y es una tabla bidimensional con filas y columnas. El nombre de la tabla tiene que ser único.

Una definición más completa de relación: “Dados los dominios D1, D2,… Dn, no necesariamente distintos, R es una relación entre estos n-conjuntos, si es un conjunto de n-tuplas (d1,d2,…dn) tal que d1 pertenece a D1 .. dn pertenece a DN.

Los usuarios perciben una base de datos como un conjunto de tablas, sin embargo, esta percepción aplica a la estructura lógica, no a la física que puse ser implementada con otro tipo de estructuras.

Atributo: es una columna de la relación y le corresponde un nombre único dentro de la relación. También se denomina campo o columna.

En el modelo relacional, las relaciones se usan para almacenar la información de los objetos que se representa en la base de datos. Una relación se representa en una tabla bidimensional en la que las filas corresponden a registros y las columnas a atributos.

Los atributos pueden aparecer en cualquier orden, la relación será la misma y, por tanto, tendrá el mismo significado.

Dominio: es un conjunto de posibles valores para uno o más atributos. Se trata de una característica extremadamente potente en el modelo relacional. Cada atributo se define con relación a un dominio. Los dominios pueden ser diferentes para cada atributo, o dos o más atributos pueden definirse en el mismo dominio.

El concepto de dominio es importante porque permite al usuario definir el significado y el origen de los valores que pueden tener los atributos. De este modo, el sistema dispone de más información cuando ejecuta una operación relacional y puede evitar operaciones semánticamente incorrectas. Por ejemplo, no es sensato comparar un nombre de calle con un número de teléfono, aunque las definiciones de dominio de ambos atributos sean cadenas de caracteres.

Una tupla es una fila en una relación. Las tuplas pueden aparecer en cualquier orden y la relación seguirá siendo la misma y, por tanto, tendrá el mismo significado. También se denominan registros o filas.

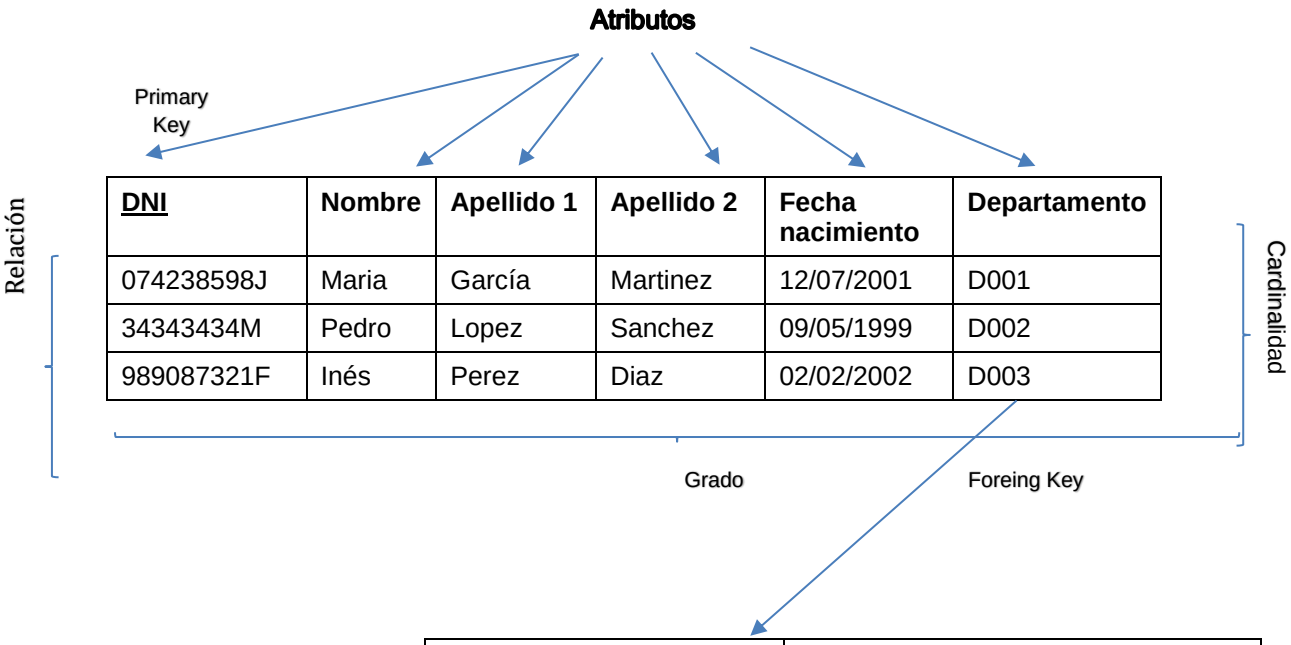
La estructura de una relación, junto con una especificación de los dominios y cualquier otra restricción de los valores posibles, se llama a veces su **intensión** que suele ser fija, a menos que se modifique el significado de una relación para incluir atributos adicionales. Las tuplas se denominan extensión (o estado) de una relación, que cambia con el tiempo.

El grado de una relación es el número de atributos que contiene. Una relación con un solo atributo tendría un grado uno y se llamaría unario. Una relación con dos atributos se denomina binaria y una con tres atributos se denomina ternaria y después el término n -ario se suele utilizar. El grado de una relación es una propiedad de la intensión de la relación.

La **cardinalidad** de una relación es el número de tuplas que contiene.

El número de tuplas se denomina cardinalidad de la relación y ésta cambia cuando se añaden o eliminan tuplas. La cardinalidad es una propiedad de la extensión de la relación y se determina a partir de la instancia particular de la relación en un momento dado.

Ejemplo de la relación Alumnos:



DNI	**Nombre**	**Apellido 1**	**Apellido 2**	**Fecha nacimiento**	**Departamento**
074238598J	Maria	García	Martinez	12/07/2001	D001
34343434M	Pedro	Lopez	Sanchez	09/05/1999	D002
989087321F	Inés	Perez	Diaz	02/02/2002	D003

Codigo Departamento	**Descripción Departamento**
D001	Departamento Matemáticas
D002	Departamento Física
D003	Departamento

Ilustración 2 Relación de Alumnos

Una b**ase de datos relacional** es una colección de relaciones normalizadas con distintos nombres de relación. Se compone de relaciones que están estructuradas adecuadamente. Esta adecuación se denomina normalización.

Cuando se habla de base de datos el **esquema de la base de datos** es el diseño de la misma, mientras que el **ejemplar de la base de datos** es una instantánea de los datos de la misma en un momento dado.

Propiedades de las relaciones

Una relación tiene las siguientes propiedades:

- La relación tiene un nombre distinto de todos los demás nombres de relaciones en el esquema relacional.
- Cada relación debe contener un solo tipo de fila., El formato de cada fila queda definido por el esquema de la relación. Es decir, todas las filas tienen las mismas columnas y formato.
- Cada fila o tupla tiene que ser única; no hay tuplas duplicadas.
- El orden de las tuplas es indiferente, en teoría. Sin embargo, en la práctica, el orden puede afectar a la eficiencia del acceso a las tuplas.
- Cada atributo debe estar especificada por un nombre específico.
- El orden de los atributos o columnas dentro de una tabla es indiferente.
- Los valores de un atributo son todos del mismo dominio.
- Cada celda de la relación contiene exactamente un valor atómico (único).

4.2.1 Claves relacionales

En el modelo relacional, como ya se ha dicho, no hay tuplas duplicadas dentro de una relación. Por lo tanto, necesitamos poder identificar uno o más atributos (llamados **claves relacionales**) que identifiquen de forma única cada tupla de una relación. A continuación, se explica cada uno de los conceptos:

- **Superclave:** Es un atributo, o conjunto de atributos, que identifica de forma exclusiva una tupla dentro de una relación. Sin embargo, puede contener atributos adicionales que no son necesarios para la identificación única, y lo que interesa es identificar superclaves que contengan sólo el número mínimo de atributos necesarios para la identificación única.
- **Clave Candidata:** Es una superclave en la que **no hay ningún subconjunto superclave**. Es decir, son superclaves mínimas, cuyos valores son suficientes para distinguir a dos tuplas cualesquiera de una relación. Una clave candidata, K para una relación R tiene dos propiedades:
  - Unicidad - en cada tupla de R los valores de K identifican de forma única esa tupla;
  - Irreductibilidad, no existe ningún otro subconjunto de K que cumpla la regla de unicidad. no existe ningún subconjunto de K que pueda tener claves candidatas.
  - Cuando una clave está formada por más de un atributo, la llamamos **clave compuesta**.
- **Clave Principal** (Primary Key) es la clave candidata que se selecciona para identificar las tuplas de forma única dentro de la relación. Como una relación no tiene tuplas duplicadas, siempre es posible identificar cada fila de forma única. Esto significa que una relación siempre tiene una **clave primaria**. En el peor de los casos, todo el conjunto de atributos podría servir como clave primaria, pero normalmente basta con un subconjunto más pequeño para distinguir las tuplas. Las claves candidatas que no se seleccionan para ser la clave primaria se denominan **claves alternativas**.
- **Clave Ajena** (Foreing Key) es el atributo, o conjunto de atributos, de una relación que coincide con la clave candidata de otra relación (posiblemente la misma). Cuando un atributo aparece en más de una relación, su aparición suele representar una relación entre tuplas de las dos relaciones.
- **Clave alternativa,** es cualquier clave candidata que no es elegida como primaria. Hay que tenerlas en cuenta para, al menos, aplicar las restricciones correspondientes.

4.2.2 Restricciones del modelo Relacional- Reglas de Integridad

Un modelo de datos tiene otras dos partes: una parte de manipulación de datos, que define los tipos de operaciones permitidas sobre los datos, y un conjunto de restricciones de integridad, que garantizan la exactitud de los datos.

Antes de avanzar es las restricciones de integridad, es necesario entender el concepto del valor nulo.

Valor Nulo: Cuando en una tupla un atributo es desconocido, se dice que es nulo. El nulo implica la ausencia de información, bien porque se desconozca el valor del atributo, o bien porque no aplica y ese

atributo no tiene sentido para esa tupla. Un valor nulo no representa el valor cero, ni la cadena vacía, estos valores tienen significado.

Ya que los nulos no son valores, deben de tratarse de forma diferente, lo que causa problemas en a la implementación, debido a que el modelo relacional se basa en el cálculo de predicados de primer orden, que es una lógica de dos valores o booleana: los únicos valores permitidos son verdadero o falso. Permitir nulos significa que tenemos que trabajar con una lógica de mayor valor, como la de tres. De hecho, hay SGBDR que no soportan los nulos

Al definir cada atributo sobre un dominio se impone una restricción sobre el conjunto de valores permitidos para cada atributo. A este tipo de restricciones se les denomina **restricciones de dominio**.

Restricción de Clave: Una relación está definida cono un conjunto de tuplas. Por definición todos los elementos del conjunto son distintos y, por lo tanto, todas las tuplas de una relación deben serlo.

Además, hay dos reglas de integridad muy importantes, que son restricciones que se deben cumplir en todas las bases de datos relacionales y en todos sus estados, y se aplican a todas las instancias de la base de datos. Estas dos reglas principales del modelo relacional se conocen como **integridad de la entidad e integridad referencial**. Existen otros tipos de restricciones de integridad como son la multiplicidad de las restricciones generales.

.

- **Regla de integridad de la entidad**. En una relación base, ningún atributo de una clave primaria puede ser nulo. Por definición, una clave primaria es un identificador mínimo que se utiliza para identificar tuplas de forma única. Esto significa que ningún subconjunto de la clave primaria es suficiente para proporcionar una identificación única de las tuplas. Si permitimos un nulo para cualquier parte de una clave primaria, estamos implicando suponiendo que no se necesitan todos los atributos para distinguir entre tuplas, lo que contradice la definición de la clave primaria.
- **Integridad referencial**: Está especificada entre dos relaciones y se utilizan para mantener la consistencia entre tuplas de dos relaciones. Si existe una **clave ajena** en una relación, el valor de la clave ajena debe coincidir con un valor de clave candidata de alguna tupla en su relación de origen o el valor de la clave externa debe ser totalmente nulo.
- **Restricciones generales** Reglas adicionales especificadas por los usuarios o administradores de la base de datos de una base de datos que definen o restringen algún aspecto de la organización. En SQL se usan mecanismos como triggers o asertions.
    - **Regla de validación (check).** Es una restricción que impone una condición lógica que deberá de cumplir una o más columnas cuando se la añadan o modifiquen los datos. Por ejemplo, podríamos restringir la columna llamada sueldo para que siempre acepte valores mayores de 1000; no se permitiría entonces indicar sueldos menores de 1000 en dicha columna.

      A veces las reglas implican a varias columnas, como por ejemplo que la fecha de inicio sea mayor que la fecha final.
    - **Disparadores o triggers.** Son restricciones más complejas, presentes en sistemas avanzados de bases de datos. Se trata de código almacenado en la base de datos, cuyas instrucciones se ejecutan automáticamente cuando ocurre un determinado evento.

Otras restricciones:

- **Restricción de unicidad (unique**). Impide que los valores de los atributos marcados de esa forma puedan repetirse en distintas filas. Es decir, en esa columna los valores deben ser distintos para cada fila, o bien quedar vacíos.
- **Restricción de obligatoriedad (not null).** Prohíbe que el atributo marcado de esta forma quede vacío (es decir impide que pueda contener el valor nulo, null) en alguna fila. También se debe de indicar en las columnas que son clave alternativa.

Políticas de actualización y eliminación

La restricción de integridad referencial causa problemas en las operaciones de borrado y modificación de registros, ya que si se ejecutan esas operaciones sobre la tabla principal quedarán filas en la tabla secundaria con la clave externa haciendo referencia a un valor que ya no existe, y eso la propia restricción no lo permite.

Para solventar esta situación se utilizar políticas especiales cuando creemos la restricción en la clave secundaria:

- Prohibir la operación (no action). Esto no permitiría hacer en las claves principales ninguna operación si hay claves secundarias relacionadas con ellas. Suele ser la opción por defecto. Es muy rígida.
- Transmitir la operación en cascada (cascade). Es decir, si se modifica o borra un departamento, también se modificarán o borrarán los profesores relacionados con él.
- Colocar nulos (set null). Si modificamos o borramos un departamento, en la relación profesores se marcarán el antiguo código de ese departamento como nulo.
- Usar el valor por defecto (default). Se colocan un valor por defecto en las claves externas relacionadas cuando se borre o modifique la clave principal relacionada. Este valor por defecto se indica al crear la tabla (opción default).

No todas las bases de datos admiten todas estas posibles políticas a aplicar en operaciones de actualizar o eliminar. Además, podemos indicar una política al actualizar y otra al eliminar. Es decir, podremos, por ejemplo, indicar nulos ante cambios en las claves principales y actuar en cascada ante eliminaciones en las claves.

4.3 Vistas

Aunque en la arquitectura ANSI-SPARC se describe una vista externa como la estructura de la base de datos tal y como la ve un usuario concreto. En el modelo relacional, una relación base es una relación que corresponde a una entidad del esquema conceptual y cuyas tuplas se almacenan físicamente en la base de datos, y una vista es una relación que no existe por sí misma, sino que deriva de una o varias relaciones base.

Una vista es el resultado dinámico de una o varias operaciones relacionales sobre unas relaciones base para producir otra relación. Una vista es una relación virtual que no existe necesariamente en la base de datos pero que puede producirse a petición de un usuario concreto y puede ser manipulada como si fuera una relación de base. (aunque su definición se almacena en el catálogo del sistema).

Las vistas son dinámicas, lo que significa que los cambios realizados en las relaciones base que afectan a la vista se reflejan inmediatamente en la vista. Cuando los usuarios realizan cambios permitidos en la vista, estos cambios se realizan a las relaciones subyacentes.

El mecanismo de las vistas proporciona las siguientes ventajas:

- Un mecanismo de seguridad potente y flexible al ocultar partes de la base de datos de ciertos usuarios. Los usuarios no conocen la existencia de atributos o tuplas que faltan en la vista.

- Permite que los usuarios accedan a los datos de forma personalizada a sus necesidades.

- Puede simplificar las operaciones complejas sobre las relaciones base. Por ejemplo, si una vista se define como una combinación (join) de dos relaciones los usuarios pueden realizar operaciones más sencillas en la vista, que serán traducidas por el SGBDR en operaciones equivalentes sobre la unión.

Una vista debe estar diseñada para soportar el modelo externo que el usuario encuentra familiar.

4.3.1 Actualización de las vistas

Todas las actualizaciones de una relación base deben reflejarse inmediatamente en todas las vistas que hacen referencia a esa relación base. Del mismo modo, si se actualiza una vista, la relación base subyacente debe reflejar el cambio. Sin embargo, existen restricciones en cuanto a los tipos de modificación que pueden realizarse a través de las vistas. Resumimos aquí las condiciones bajo las cuales la mayoría de los sistemas determinan si se permite una actualización a través de una vista:

- Las actualizaciones se permiten a través de una vista definida mediante una consulta simple que implique una única relación base y que contenga la clave primaria o una clave candidata de la relación base.

- Las actualizaciones no se permiten a través de vistas que incluyan múltiples relaciones base.

- No se permiten actualizaciones a través de vistas que incluyan operaciones de agregación o agrupación.

Se han definido clases de vistas teóricamente no actualizables, teóricamente actualizables y parcialmente actualizables.

5 Normalización

El objetivo principal cuando se diseña una Base de Datos es crear una representación exacta de los datos, las relaciones entre ellos y las restricciones pertinentes para esa organización. Las características de un conjunto adecuado de relaciones son las siguientes:

- el número mínimo de atributos necesarios para satisfacer las necesidades de datos de la empresa;
- los atributos con una relación lógica estrecha (descrita como dependencia funcional) se encuentran en la misma relación;
- redundancia mínima, con cada atributo representado sólo una vez, con la importante excepción de los atributos que forman parte o la totalidad de las claves externas.

La normalización es una técnica formal para analizar las relaciones en función de su clave primaria (o claves candidatas) y las dependencias funcionales. Esta técnica implica una serie de reglas que se pueden utilizar para probar las relaciones individuales, de modo que una base de datos se puede normalizar en cualquier grado. Cuando no se cumple un requisito, la relación que viola el requisito debe descomponerse en relaciones que cumplan individualmente los requisitos de normalización.

Inicialmente se propusieron tres formas normales denominadas Primera Forma Normal (1FN) Segunda forma normal (2FN) y Tercera forma normal (3FN). Posteriormente, R. Boyce y E. F. Codd introdujeron una definición más fuerte de la tercera forma normal llamada Forma Normal Boyce-Codd (BCNF) (Codd, 1974). Con la excepción de 1FN, todas estas formas normales se basan en dependencias funcionales entre los atributos de una relación (Maier, 1983). Más adelante se introdujeron formas normales superiores que van más allá de la FNBC se introdujeron posteriormente, como la cuarta forma normal (4FN), la quinta forma normal (5FN) (Fagin, 1977, 1979) y la sexta forma normal (6FN) Sin embargo, estas formas normales posteriores se ocupan de situaciones que son muy raras.

La normalización se ejecuta a menudo como una serie de pasos. Cada paso corresponde a una forma normal específica que tiene propiedades conocidas. A medida que avanza la normalización, las relaciones se vuelven progresivamente más restringidas (más fuertes) en su formato y también menos vulnerables a las anomalías de actualización. Para el modelo de datos relacional, es importante reconocer que sólo la primera forma normal (1FN) es fundamental para crear relaciones; todas las formas normales posteriores son opcionales. Sin embargo, para evitar anomalías de actualización se recomienda generalmente que se proceda hasta al menos la tercera forma normal (3FN), aunque los diseñadores de bases de datos aspiran a llegar a la 5FN siempre que sea posible y razonable.

A veces los diseñadores de bases de datos escogen un esquema que tiene información redundante; es decir, que no está normalizada. Utilizan la redundancia para mejorar el rendimiento para aplicaciones concretas. La penalización sufrida por no emplear un esquema normalizado es el trabajo extra (en términos de tiempo de codificación y de tiempo de ejecución) de mantener consistentes los datos redundantes. El proceso de tomar un esquema normalizado y hacerlo no normalizado se denomina **desnormalización,** y los diseñadores lo utilizan para ajustar el rendimiento de los sistemas para dar soporte a las operaciones críticas en el tiempo.

5.1 Conceptos

Dependencia Funcional: Describe la relación entre los atributos de una relación. En el ejemplo, si A y B son atributos de la relación R, B es funcionalmente dependiente de A, si cada valor de A está asociado con exactamente un valor de B(A y B pueden estar formados por uno o más más atributos). Cuando existe una dependencia funcional, el atributo o grupo de atributos en el lado izquierdo de la flecha se denominan **determinante**. En el siguiente ejemplo se tienen la relación Alumnos, en la que Nombre y fecha de nacimiento dependen de DNI. En cambio, Curso no, porque hay más de un valor posible para un mismo DNI.

DNI	Nombre	Fecha nacimiento	Curso
074238598J	Maria García Martinez	12/07/2001	1º
074238598J	Maria García Martinez	12/07/2001	2º

34343434M	Pedro Lopez Sanchez	09/05/1999	3º
989087321F	Inés Perez Diaz	02/02/2002	2º
989087321F	Inés Perez Diaz	02/02/2002	1º

Dependencia funcional completa. Indica que, si A y B son atributos de una relación, B es totalmente funcionalmente dependiente de A, si B es funcionalmente dependiente de A, pero no de ningún subconjunto propio de A. Una dependencia parcial es si hay algún atributo que puede ser eliminado de A y, sin embargo, la dependencia se mantiene. Por ejemplo, en la siguiente relación:

DNI	**Curso**	**Asignatura**	**Nota**
074238598J	1º	Programación	6
074238598J	2º	Redes de computadores	7
34343434M	3º	Inteligencia Artificial	5
989087321F	2º	Programación	4
989087321F	1º	Redes de computadores	8

El atributo Nota depende completamente de la Clave: (DNI, Curso, Asignatura)

Dependencia transitiva, Una condición en la que A, B y C son atributos de una relación tal que si A →B y B → C, entonces C depende transitivamente de A a través de B, siempre que A no sea funcionalmente dependiente de B o C. En la siguiente relación:

DNI	**Nombre**	**Cod Provincia**	**Nombre Provincia**
074238598J	Maria García Martinez	03	Alicante
34343434M	Pedro Lopez Sanchez	28	Madrid
989087321F	Inés Perez Diaz	47	Valladolid

El Cod Provincia depende de DNI y Nombre Provincia depende de Código de provincia, en cambio el nombre de provincia no depende del DNI.

Dependencia multivaluada La posible existencia de dependencias multivaluadas en una relación se debe a la 1FN que impide que un atributo de una tupla tenga un conjunto de valores. Por ejemplo, si tenemos dos atributos multivaluados en una relación, tenemos que repetir cada valor de uno de los atributos con cada valor del otro atributo, para garantizar que las tuplas de la relación sean coherentes. Este tipo de restricción se denomina dependencia multivaluada y da lugar a una redundancia de datos.

Representamos una MVD entre los atributos A, B y C en una relación utilizando la siguiente

notación: A ->> B y A ->> C.

Por ejemplo, tenemos la relación con empresa, empleado y responsable:

Empresa	**Empleado**	**Responsable**
1001	Maria Lopez	Juan Sanchez
1001	Fernando García	Juan Sanchez
1001	Laura Gomez	Javier Rodriguez
1002	Juan Diaz	Pablo Martinez

Cada vez que añadimos a un empleado tenemos que incluir al responsable.

5.1.1 Primera Forma normal

Se dice que una relación está en primera forma normal (1FN) si no contiene grupos repetitivos, es decir, cada atributo de una tupla tiene a lo sumo un valor.

Por ejemplo, partimos de una tabla no normalizada de alumnos:

DNI	Nombre	Fecha nacimiento	Curso	Asignatura
074238598J	Maria García Martinez	12/07/2001	1º 2º	Calculo Física Algebra Programación Redes de computadores
34343434M	Pedro Lopez Sanchez	09/05/1999	3º	Inteligencia Artificial Seguridad
989087321F	Inés Perez Diaz	02/02/2002	2º 1º	Arquitectura Bases de datos Algebra

Las columnas Curso y Asignatura para cada tupla tienen más de un valor, para que quede en 1FN sería:

DNI	Nombre	Fecha nacimiento	Curso	Asignatura
074238598J	Maria García Martinez	12/07/2001	1º	Calculo
074238598J	Maria García Martinez	12/07/2001	1º	Algebra
074238598J	Maria García Martinez	12/07/2001	1º	Programación
074238598J	Maria García Martinez	12/07/2001	2º	Redes de computadores
34343434M	Pedro Lopez Sanchez	09/05/1999	3º	Inteligencia Artificial
34343434M	Pedro Lopez Sanchez	09/05/1999	3º	Seguridad
989087321F	Inés Perez Diaz	02/02/2002	2º	Arquitectura
989087321F	Inés Perez Diaz	02/02/2002	2º	Bases de datos
989087321F	Inés Perez Diaz	02/02/2002	1º	Algebra

5.1.2 Segunda Forma normal

Se dice que una relación está en Segunda Forma Normal (2FN) si está en 1FN y cada atributo que no pertenezca a la clave primaria tiene una dependencia funcional completa de la clave.

La segunda forma normal (2FN) se basa en el concepto de dependencia funcional completa que describimos anteriormente. La 2FN se aplica a las relaciones con claves compuestas, es decir, relaciones con una clave primaria compuesta por dos o más atributos. Una relación con una clave primaria de un solo atributo está automáticamente en al menos 2FN.

La normalización de las relaciones 1FN a 2FN implica la eliminación de las dependencias parciales. Si existe una dependencia parcial, se eliminan los atributos parcialmente dependientes de la relación, colocándolos en una nueva relación junto con una copia de su determinante.

Por ejemplo, siendo la relación Alumnos con (DNI,Curso) de clave primaria:

DNI	Curso	Nombre
074238598J	1º	Maria García Martinez
074238598J	2º	Maria García Martinez
34343434M	3º	Pedro Lopez Sanchez
989087321F	2º	Inés Perez Diaz
989087321F	1º	Inés Perez Diaz

El atributo Nombre sólo depende del DNI y no de Curso. Por lo que habría que descomponerla en dos relaciones:

La relación de Alumnos en Curso con clave primaria (DNI,Curso):

DNI	Curso
074238598J	1º
074238598J	2º
34343434M	3º
989087321F	2º
989087321F	1º

Y la relación Alumnos con clave primaría DNI:

DNI	Nombre
074238598J	Maria García Martinez
34343434M	Pedro Lopez Sanchez
989087321F	Inés Perez Diaz

5.1.3 Tercera Forma normal

Una relación está en tercera forma normal (3FN) si está en segunda forma normal y cada atributo que no pertenezca a la clave primaria no depende transitivamente de dicha clave.

La normalización de las relaciones 2FN a 3FN implica la eliminación de las dependencias transitivas. Si existe una dependencia transitiva, eliminamos de la relación el atributo o atributos que dependen transitivamente de la relación colocando los atributos en una nueva relación junto con una copia del determinante. Por ejemplo:

La relación de ganadores de torneos, con clave primaria: año y torneo.

AÑO	TORNEO	Nombre	Nacionalidad
2021	Abierto de Australia	Novak Djokovic	Serbia
2021	Roland Garros	Novak Djokovic	Serbia
2020	Roland Garros	Rafael Nadal	Española
2020	Abierto de EEUU	Dominic Thiem	Austriaca
2018	Abierto de Australia	Roger Federer	Suiza

La nacionalidad no depende de la clave primaria, por lo que se descompone en las relaciones:

La relación de ganadores de torneo con clave "año y torneo".

AÑO	TORNEO	Nombre
2021	Abierto de Australia	Novak Djokovic
2021	Roland Garros	Novak Djokovic
2020	Roland Garros	Rafael Nadal
2020	Abierto de EEUU	Dominic Thiem
2018	Abierto de Australia	Roger Federer

La relación de ganadores con clave primaria "Nombre".

Nombre	Nacionalidad
Novak Djokovic	Serbia
Rafael Nadal	Española
Dominic Thiem	Austriaca
Roger Federer	Suiza

5.1.4 Forma normal de Boyce-Codd (FNBC)

Una relación está en la forma normal de Boyce y Codd (FNBC) si y sólo si cada determinante es una clave candidata.

Se trata de una forma normal parecida a la 3FN pero más restrictiva, se basa en dependencias funcionales que tienen en cuenta todas las claves candidatas en una relación. Para comprobar si una relación está en FNBC, identificamos todos los determinantes y los que son claves candidatas. Recordemos que un determinante es un atributo, o un grupo de atributos, del que depende funcionalmente algún otro atributo.

La diferencia entre 3NF y FNBC es que para una dependencia funcional A → B, 3FN permite esta dependencia en una relación si B es un atributo de clave primaria y A no es una clave candidata, mientras que FNBC insiste en que para que esta dependencia permanezca en una relación, A debe ser una clave candidata. Por lo tanto, BCNF es una forma más fuerte de 3NF, de modo que toda relación en FNBC está también en 3FN. Sin embargo, una relación en 3FN no está necesariamente en FNBC.

Por ejemplo:

Tenemos la relación de prestamos con clave primaria Id_prestamo.

Id prestamo	Sucursal	Id- Cliente	Importe
340001	1001	67477738N	10.000
340001	1001	24999098Y	10.000
350877	2003	05546683J	12.000
238767	2003	22997799M	25.000
128433	3001	90909090H	50.000

Se observa que, al haber varios nombres de clientes asociados a un préstamo, es obligatorio repetir el nombre de la sucursal y el importe por cada cliente.

Por lo tanto, se tendría que descomponer en dos relaciones de la siguiente forma:

Id prestamo	Id- Cliente
340001	67477738N

340001	24999098Y
350877	05546683J
238767	22997799M
128433	90909090H

Id prestamo	**Sucursal**	**Importe**
340001	1001	10.000
350877	2003	12.000
238767	2003	25.000
128433	3001	50.000

5.1.5 Cuarta Forma normal

Aunque la BCNF elimina cualquier anomalía debida a las dependencias funcionales, se identificaron otro tipo de dependencia llamada Dependencia Multi-Valor o Multivaluadas. (DMV), que también puede causar redundancia de datos (Fagin, 1977).

Una relación está en 4FN si y sólo si está en 3FN y además toda dependencia multivaluada no trivial está determinada por una clave candidata.

La cuarta forma normal (4FN) impide que una relación contenga una DMV no trivial sin que el determinante asociado sea una clave candidata de la relación. Cuando se viola la regla 4FN, existe la posibilidad de que haya redundancia de datos, como se muestra a continuación. La normalización de una relación que rompe la regla 4FN requiere la eliminación del DMV infractora de la relación colocando el atributo o atributos multivaluados en una nueva relación junto con una copia del determinante.

Por ejemplo, en la relación de Profesores de asignaturas por curso:

Asignatura	**Profesor/a**	**Curso**
Matemáticas	Yolanda García	1º
Matemáticas	Yolanda García	2º
Matemáticas	Pedro Gomez	1º
Matemáticas	Pedro Gomez	2º
Física	Pedro Gomez	1º
Física	Pedro Gomez	2º
Física	Sonia Perez	1º
Física	Sonia Perez	2º

Se puede observar que hay dos dependencias multivaluadas. El atributo *Asignatura* multidetermina al atributo *Profesor/a* y también al atributo *Curso*. Dado que los atributos *Profesor* y *Curso* no tienen ninguna dependencia entre ellos (no existe ninguna relación entre los profesores que imparten una asignatura y los cursos donde dicha asignatura se imparte) se está introduciendo una redundancia de

información en la tabla. La 4FN soluciona este problema dividiendo la tabla anterior en dos, como se muestra a continuación, de manera que las dependencias multivaluadas se almacenan de manera correcta evitando redundancias:

Asignatura	**Curso**
Matemáticas	1º
Matemáticas	2º
Física	1º
Física	2º

Profesor/a	**Asignatura**
Yolanda García	Matemáticas
Pedro Gomez	Matemáticas
Pedro Gomez	Física
Sonia Perez	Física

5.1.6 Quinta Forma normal

La quinta forma normal (5FN), también conocida como forma normal de proyección-unión (PJ/NF), es un nivel de normalización de bases de datos diseñado para reducir redundancia en las bases de datos relacionales que guardan hechos multi-valores aislando semánticamente relaciones múltiples relacionadas. Una tabla se dice que está en 5NF si y sólo si está en 4NF y cada dependencia de unión (join) en ella es implicada por las claves candidatas.

Una relación está en quinta forma normal (5FN) si y sólo si para cada relación de unión (R1, R2, . . . Rn) en una relación R, cada proyección incluye una clave candidata de la relación original.

La quinta forma normal (5FN) impide que una relación contenga una dependencia de unión no trivial (JD) sin que la proyección asociada incluya una clave candidata de la relación original (Fagin, 1977). Las JD no triviales que no están asociadas a claves candidatas son muy raras, por lo que las relaciones 4FN suelen estar también en 5FN.

Por ejemplo, la relación Profesor/a, asignatura y grado. Tenemos que un/a Profesor/a imparte ciertas asignaturas en ciertos grados y no en todos. Además, cada grado comprende una serie de asignaturas.

Profesor/a	**Asignatura**	**Grado**
Yolanda García	Matemáticas	Informática
Yolanda García	Física	Informática
Pedro Gomez	Matemáticas	Telecomunicaciones
Pedro Gomez	Matemáticas	Informática
Pedro Gomez	Física	Física

Sonia Perez	Física	Informática
Sonia Perez	Física	Telecomunicaciones
Sonia Perez	Física	Física
Sonia Perez	Matemáticas	Física

Para poder implementar estas restricciones sin redundancias, se necesita descomponer en tres relaciones:

Profesor/a	**Grado**
Yolanda García	Informática
Pedro Gomez	Telecomunicaciones
Pedro Gomez	Informática
Pedro Gomez	Física
Sonia Perez	Informática
Sonia Perez	Telecomunicaciones
Sonia Perez	Física

Profesor/a	**Asignatura**
Yolanda García	Matemáticas
Yolanda García	Física
Pedro Gomez	Matemáticas
Pedro Gomez	Física
Sonia Perez	Matemáticas
Sonia Perez	Física

Grado	**Asignatura**
Informática	Matemáticas
Informática	Física
Telecomunicaciones	Matemáticas
Telecomunicaciones	Física
Física	Matemáticas

Física	Física

5.1.7 La forma normal del dominio-clave

La forma normal de dominio-clave (DKNF) fue propuesta por Ronald Fagin en 1981. A diferencia de las otras formas normales, que se refieren a dependencias funcionales, multivaluadas o joins, en cambio, esta forma normal se define en términos de dominios y claves.

Una relación está en DKNF si cada restricción de la relación es una consecuencia lógica de la definición de sus claves y dominios.

5.1.8 Sexta Forma normal

Sexta Forma Normal (6FN). Fue propuesta por C.J. DATE en 2003, después de varios años de investigación, y es relativo a las llamadas bases de datos temporales. Se trata de bases de datos que contienen información histórica y actual. Las bases de datos temporales pueden considerarse las precursoras de los datawharehouse.

Por ejemplo, tenemos la relación asignatura:

Código de Asignatura	**Denominación**	**Créditos**
C1001	Matemáticas	5
C1002	Física	10
C1003	Arquitectura computadores	7
C1004	Telecomunicaciones	7
C1005	Algoritmos	10
C1006	Estructuras de datos	12

Tanto la descripción como los créditos, puede cambiar en cualquier momento, por lo que sería necesario fechar cada una de las tuplas con la fecha efectiva del cambio.

Asignatura	**Denominación**	**Créditos**	**Fecha**
C1001	Matemáticas	5	2007
C1002	Física	4	2007
C1003	Arquitectura computadores	7	2009
C1004	Telecomunicaciones	7	2000
C1005	Algoritmos	3	2012
C1006	Estructuras de datos	4	2019

X RESUMEN ESQUEMÁTICO

Historia del modelo relacional

- Modelo relacional propuesto por primera vez por Edgard Frank Codd en 1970.
- En 1974, dos informáticos de IBM D.D. Chamberlin y R.F Boyce publican “Sequel: a estructured english query lenguaje”, predecesor de SQL.

Reglas de Codd un sistema podrá considerarse más relacional cuanto más siga estas reglas.

0. REGLA 0: REGLA DE FUNDACIÓN
1. REGLA 1: REGLA DE LA INFORMACIÓN
2. REGLA 2: REGLA DEL ACCESO GARANTIZADO
3. REGLA 3: TRATAMIENTO SISTEMÁTICO DE VALORES NULOS
4. REGLA 4: CATÁLOGO DINÁMICO EN LÍNEA BASADO EN EL MODELO RELACIONAL
5. REGLA 5: REGLA DEL SUBLENGUAJE DE DATOS COMPLETO
6. REGLA 6: REGLA DE ACTUALIZACIÓN DE VISTAS.
7. REGLA 7: INSERCIÓN, ACTUALIZACIÓN Y BORRADO DE ALTO NIVEL
8. REGLA 8: INDEPENDENCIA FÍSICA DE DATOS
9. REGLA 9: INDEPENDENCIA LÓGICA DE DATOS
10. REGLA 10: INDEPENDENCIA DE INTEGRIDAD
11. REGLA 11: INDEPENDENCIA DE DISTRIBUCIÓN
12. REGLA 12: REGLA DE LA NO SUBVERSIÓN

Conceptos:

- **Relación:** Tabla bidimensional con filas y columnas. El nombre de la tabla tiene que ser único
- **Atributo:** es una columna de la relación con nombre único dentro de la relación
- **Dominio**: es un conjunto de posibles valores para uno o más atributos
- **Tupla:** es una fila en una relación
- **Grado:** número de atributos que contiene.
- **Cardinalidad:** es el número de tuplas que contiene

Claves relacionales: que identifiquen de forma única cada tupla de una relación.

- **Superclave**: Es un atributo, o conjunto de atributos, que identifica de forma exclusiva una tupla dentro de una relación. Puede contener atributos adicionales e innecesarios.
- **Clave Candidata:** Es una superclave en la que no hay ningún subconjunto de superclave., Superclaves mínimas.
- **Clave Principal**: (Primary Key) es la clave candidata que se selecciona para identificar las tuplas de forma única dentro de la relación.
- **Clave Ajena**: (Foreing Key) es el atributo, o conjunto de atributos, de una relación que coincide con la clave candidata de otra relación (posiblemente la misma).
- **Clave alternativa**: es cualquier clave candidata que no es elegida como primaria. Hay que tenerlas en cuenta para, al menos, aplicar las restricciones correspondientes.

Restricciones del modelo Relacional- Reglas de Integridad

- **Valor Nulo**: Cuando en una tupla un atributo es desconocido, se dice que es nulo.
- **Regla de integridad de la entidad**: En una relación base, ningún atributo de una clave primaria puede ser nulo.
- **Integridad referencial**: Está especificada entre dos relaciones y se utilizan para mantener la consistencia entre tuplas de dos relaciones. Si existe una clave ajena en una relación, el valor de la clave ajena ¿?

Vistas:

Una vista es una relación virtual que no existe necesariamente en la base de datos pero que puede producirse a petición de un usuario concreto y puede ser manipulada como si fuera una relación de base.

Normalización:

- **Dependencia Funcional**: describe la relación entre los atributos de una relación.
- **Dependencia funcional completa:** indica que, si A y B son atributos de una relación, B es totalmente funcionalmente dependiente de A si B es funcionalmente dependiente de A, pero no de ningún subconjunto propio de A.
- **Dependencia transitiva**: una condición en la que A, B y C son atributos de una relación tal que si A →B y B → C, entonces C depende transitivamente de A a través de B.
- **Dependencia multivaluada**: si tenemos una relación en la que tenemos que repetir cada valor de uno de los atributos con cada valor del otro atributo, para garantizar que las tuplas de la relación sean coherentes. Este tipo de restricción se denomina dependencia multivaluada y da lugar a una redundancia de datos.
- **Formas Normales**:

    - **1FN,** Se dice que una entidad está en primera forma normal (1FN) si no contiene grupos repetitivos, es decir, todos los atributos dependen funcionalmente de la clave.
    - **2FN,** si está en 1FN y cada atributo que no pertenezca a la clave tiene una dependencia funcional completa de la clave.
    - **3FN** si está en segunda forma normal y cada atributo que no pertenezca a la clave no depende transitivamente de dicha clave.
    - **FNBC**, Se dice que una tabla está en FNBC si y solo si está en 3FN y cada dependencia funcional no trivial tiene una clave candidata como determinante. En términos menos formales, una tabla está en FNBC si está en 3FN y los únicos determinantes son claves candidatas. Y determinante el atributo del cual depende funcionalmente —por completo— algún otro atributo.
    - **4FN:** y sólo si está en 3FN, para cada una de sus dependencias múltiples no funcionales X->->Y, siendo X una super-clave que, X es o una clave candidata o un conjunto de claves primarias.
    - **5FN:** si y sólo si está en 4NF y cada dependencia de unión (join) en ella es implicada por las claves candidatas.
    - La **forma normal de dominio/clave (DKNF**) es una forma normal usada en normalización de bases de datos que requiere que la base de datos contenga restricciones de dominios y de claves.
    - Una restricción del dominio especifica los valores permitidos para un atributo dado, mientras que una restricción clave especifica los atributos que identifican únicamente una fila en una tabla dada.
    - **6FN**: La 6FN está pensada para gestionar información temporal en bases de datos, minimizando el número de operaciones necesario para mantener ese tipo de información.

La **desnormalización** es el proceso de procurar optimizar el desempeño de una base de datos por medio de agregar datos redundantes.

Y GLOSARIO

ANSI: American National Standards Institute (Instituto Nacional Americano de Estándares)
COM
DB2: No se corresponde con unas siglas, es el gestor de base de datos SQL de IBM.
DMV: Dependencia Multivaluada.
DKNF: Forma normal de dominio clave.
EAV: modelo Entidad- Atributo- Valor.
FN: Forma normal.
FNBC: Forma normal de Boyce y Codd.
JD: Dependencia de Unión.
IBM: Industrial Business Machine Corporation (Nombre comercial)
IEC: International Electrotechnical Commission (Comisión Electrotécnica Internacional)
ISO: International Organization for Standarization (Organización Internacional para la Estandarización)
JTC: Joint Technical Committee (Comité Técnico Conjunto)
SGBDR: Sistemas Gestores de Bases de datos Relacionales.
SQL: Structured Query Language (Lenguaje Estructurado de Consulta)
XML: Extensible Markup Language (Lenguaje Extensible de Marcado)

Z BIBLIOGRAFÍA BÁSICA

Database Systems. A Practical Approach to Design, Implementation, and Management. Sixth Edition .Thomas M. Connolly z Carolyn E. Begg. University of the west of Scotland. PEARSON

Fundamentos de Quinta edición. Abraham Silberschatz, Henry F. Korth, Bell Laboratories S. Sudarshan McGraw-Hill

Database Systems: A Pragmatic Approach.2º Edition. Elvis C. Foster with Shripad Godbole. Apress

www.oracle.com

www.Ibm.com