**BASE DE DATOS RELACIONALES**

En el ámbito de la gestión de bases de datos, es fundamental comprender los principios que rigen la organización y estructuración de la información. En este informe, se abordarán tres temas clave: las 12 Reglas de Codd, la Normalización de Bases de Datos y el Modelo Relacional.

**1. Las 12 Reglas de Codd**

En el ámbito de la gestión de bases de datos, es fundamental comprender los principios que rigen la organización y estructuración de la información. En este informe, se abordarán tres temas clave: las 12 Reglas de Codd, la Normalización de Bases de Datos y el Modelo Relacional.

Al principio de las bases de datos relacionales no había una definición clara de lo que era y de lo que no era una base de datos relacional. Tuvo que llegar Edgar Codd con una serie de artículos para establecer una serie de reglas (las «Reglas de Codd» para determinar si una base de datos de podía llamar relacional

Desarrolladas en los 70’s por [Edgar Frank Codd](https://alchetron.com/Edgar-F-Codd) de IBM, son reglas que un verdadero sistema relacional debería tener. El documento principal de Codd es *A relational model of data for large shared data banks*, y en resumen las 12 reglas de Codd (que son 13) se pueden describir como:

* **Regla 0**: El sistema debe ser relacional, tanto la base de datos y administrador de sistema; es decir, un sistema de base de datos relacional debe utilizar sus facilidades relacionales (exclusivamente) para manejar la base de datos. Todo en una base de datos está guardado en un sistema relacional y cualquier elemento (usuario, tabla, índice, etc.) se guarda dentro de la misma base de datos.
* **Regla 1. Regla de la información**. Toda la información en la base de datos es representada unidireccionalmente, por valores en posiciones de las columnas dentro de filas de tablas. Toda la información en una base de datos relacional se representa explícitamente como valores en tablas**.**
* **Regla 2. Del acceso garantizado**. Todos los datos deben ser accesibles sin ambigüedad. Cada valor individual en la base de datos debe ser direccionable especificando el nombre de la tabla, la columna que lo contiene y la llave primaria. Eso significa que todo dato puede ser ubicado teniendo el nombre de la tabla, el nombre del campo y el registro del que se trate.
* **Regla 3. Tratamiento de valores nulos.** El sistema de gestión de base de datos debe permitir que haya campos nulos. Debe tener una representación de la «información que falta y de la información inaplicable», distinto de todos los valores regulares.
* **Regla 4. Catálogo basado en el modelo relacional.** El sistema debe soportar un catálogo en línea (la estructura misma de la base de datos). El catálogo relacional debe ser accesible a los usuarios autorizados. Es decir, los usuarios autorizados deben poder tener acceso a la estructura de la base de datos (catálogo). Esto es lo que en varios motores de base de datos se denomina **esquema**. Como todo en la base de datos está definido dentro de la misma base de datos, incluso la estructura está en tablas (hay una tabla con los nombres de las tablas, una tabla con los nombres de los campos existentes, una tabla con los nombres de los índices, etc.).
* **Regla 5. Regla comprensiva del sublenguaje de los datos**. El sistema debe soportar por lo menos un lenguaje relacional que:
  + Tenga una sintaxis lineal.
  + Pueda ser utilizado de manera interactiva.
  + Soporte operaciones de definición de datos, operaciones de manipulación de datos (actualización así como la recuperación), seguridad e integridad y operaciones de administración de transacciones.
* **Regla 6. Regla de actualización**. Todas las vistas que son teóricamente actualizables deben ser actualizables por el sistema, de manera transparente; es decir, que si en la base de datos se crea una vista de una tabla, se podría añadir un registro a la vista y eso significaría que se daría de alta el registro en la tabla original.
* **Regla 7**. Alto nivel de inserción, actualización y borrado, permitiendo el sistema realizar manipulación de datos de alto nivel, es decir, sobre conjuntos de registros. Esto significa que, además de leer muchos registros, se puede actualizar más de un registro a la vez, no sólo sobre registros individuales.
* **Regla 8. Independencia física de los datos**. Los clientes (aplicaciones, sistemas) permanecen inalterados a nivel lógico cuando se realicen cambios en las representaciones de almacenamiento o métodos de acceso. Ante cualquier cambio en la ubicación física de los datos, los querys escritos y probados no deben requerir modificaciones por dichos cambios en la ubicación física.
* **Regla 9. Independencia lógica de los datos**. Los cambios al nivel lógico (tablas, columnas, filas, etc.) no deben requerir un cambio a una solicitud basada en la estructura. La independencia de datos lógica es más difícil de lograr que la independencia física de datos, pero también debe ser posible que los querys que ya están escritos (si se modifica un tipo de dato, se añaden campos, se eliminan campos que no se requieren) no requieran cambios.
* **Regla 10. Independencia de la integridad**. Las reglas de integridad se deben especificar por separado de los programas o aplicaciones y se almacenan en la base de datos. Debe ser posible cambiar esas reglas sin afectar innecesariamente las aplicaciones existentes.
* **Regla 11. Independencia de la distribución**. La distribución de las porciones de la base de datos a las varias localizaciones debe ser invisible a los usuarios de la base de datos. Los usos existentes deben continuar funcionando con éxito:
  + Cuando una versión distribuida del DBMS se introduce por primera vez
  + Cuando los datos existentes se redistribuyen en todo el sistema.

En términos reales, el usuario final o el desarrollador no debe de preocuparse por la partición en la que estén los datos, el motor de base de datos debe saber dónde se encuentra cada dato y extraerlo cuando un query lo requiera.

* **Regla 12. Regla de la no subversión**. Si el sistema proporciona una interfaz de bajo nivel de registro, además de una interfaz relacional, que esa interfaz de bajo nivel no se pueda utilizar para subvertir el sistema (sin pasar por alto la seguridad relacional o limitación de integridad, por ejemplo).

Anteriormente existían sistemas anteriormente no relacionales que añadieron una interfaz relacional, pero con la interfaz nativa existía la posibilidad de trabajar no relacionalmente. Esto ya no debe ser posible.

Si bien algunas reglas ya parecen inútiles, es importante ubicar en su debido contexto estas Reglas de Codd, fueron los lineamientos básicos para la construcción de los nuevos motores de bases de datos relacionales.

**2. Normalización de Bases de Datos**

La normalización de bases de datos es un proceso de diseño que se utiliza para organizar la estructura de una base de datos relacional de manera eficiente y libre de redundancias, minimizando la posibilidad de anomalías en los datos como la actualización anómala, la inserción anómala o la eliminación anómala. El objetivo principal de la normalización es reducir la redundancia de datos y garantizar la integridad de la información almacenada en la base de datos.

El proceso de normalización se divide en diferentes formas normales (1NF, 2NF, 3NF, BCNF, 4NF, 5NF, entre otras), cada una de las cuales aborda un aspecto específico de la estructura de la base de datos. Aquí hay una descripción general de las tres primeras formas normales, que son las más comúnmente utilizadas:

**Primera Forma Normal (1NF)**: En esta forma normal, se eliminan las repeticiones de grupos de datos al organizar las tablas en un formato tabular. Esto implica que cada celda en una tabla debe contener un único valor atómico, y cada fila debe ser única y estar identificada por una clave primaria.

**Segunda Forma Normal (2NF):** Una tabla está en segunda forma normal si cumple con 1NF y todos los atributos no clave son totalmente dependientes de la clave primaria. En otras palabras, no debe haber dependencias parciales en los atributos no clave hacia la clave primaria. Si existen, esos atributos se deben mover a una nueva tabla.

**Tercera Forma Normal (3NF)**: Una tabla está en tercera forma normal si cumple con 2NF y no hay dependencias transitivas entre los atributos no clave. Esto significa que los atributos no clave deben depender solo de la clave primaria directamente y no a través de otro atributo no clave.

Además de estas formas normales, existen niveles adicionales de normalización (como BCNF, 4NF y 5NF) que pueden ser necesarios en casos específicos para garantizar una mayor reducción de la redundancia y una mayor integridad de los datos.

En resumen, la normalización de bases de datos es un proceso fundamental en el diseño de bases de datos relacionales que ayuda a mejorar la eficiencia, reducir la redundancia y garantizar la integridad de la información almacenada en la base de datos.

**3. El modelo relacional para el modelado y la gestión de bases de datos**

El modelo relacional es un enfoque para el diseño, organización y gestión de bases de datos que se basa en la teoría de conjuntos y álgebra relacional. Fue propuesto por Edgar F. Codd en 1970 y se ha convertido en uno de los modelos más ampliamente utilizados para el modelado de bases de datos.

En el modelo relacional, la información se representa en forma de tablas (también conocidas como relaciones), donde cada tabla contiene filas y columnas. Cada fila en una tabla representa una entidad única, y cada columna representa un atributo o propiedad de esa entidad. Las relaciones entre las tablas se establecen mediante claves primarias y claves foráneas.

Aquí hay algunos conceptos clave del modelo relacional:

**Tablas (Relaciones):** Las tablas son la unidad básica de almacenamiento en el modelo relacional. Cada tabla tiene un nombre único y está compuesta por filas y columnas. Cada fila representa una instancia única de la entidad que la tabla representa, y cada columna representa un atributo de esa entidad.

**Atributos:** Los atributos son las características o propiedades de las entidades que se almacenan en las columnas de las tablas. Cada atributo tiene un nombre único y un tipo de datos asociado que define el tipo de valores que puede contener.

**Claves Primarias:** Una clave primaria es un conjunto de uno o más atributos cuyos valores garantizan la unicidad de cada fila en una tabla. Se utiliza para identificar de manera única cada instancia de la entidad representada por la tabla.

**Claves Foráneas:** Una clave foránea es un atributo o conjunto de atributos en una tabla que establece una relación con la clave primaria de otra tabla. Se utiliza para representar relaciones entre entidades en la base de datos.

**Relaciones:** Las relaciones entre tablas se establecen mediante claves foráneas que hacen referencia a las claves primarias de otras tablas. Estas relaciones pueden ser uno a uno, uno a muchos o muchos a muchos, dependiendo de la naturaleza de la relación entre las entidades representadas por las tablas.

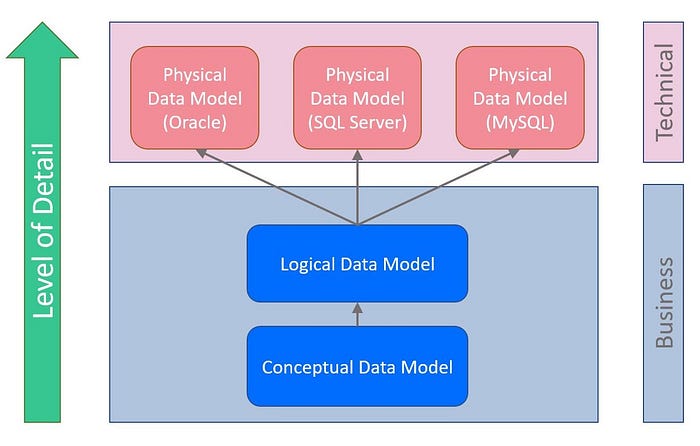
El modelo relacional proporciona un marco flexible y potente para el diseño y la gestión de bases de datos. Permite representar datos de manera estructurada y establecer relaciones entre ellos de una manera intuitiva y eficiente. Además, el modelo relacional es independiente del hardware y el software subyacente, lo que lo hace altamente portátil y adaptable a diferentes entornos de computación.

# Diseño e implementación de un modelo de datos relacional

El diseño del modelo relacional se puede dividir en tres niveles: el nivel conceptual, lógico y físico. Estos niveles representan diferentes perspectivas del modelo de datos relacional.

La separación en niveles del modelo de datos relacional proporciona claridad conceptual, flexibilidad, mantenibilidad, independencia de la plataforma y facilidad de mantenimiento y evolución a largo plazo. Permite a los diseñadores y administradores trabajar en diferentes niveles de detalle y enfoque según sus roles y responsabilidades, lo que conduce a una base de datos más robusta y adaptable.

1. **Modelo de datos conceptual**: En este modelo se define quecontiene el sistema**.** El propósito de crear un modelo de datos conceptual es establecer entidades, sus atributos y sus cardinalidades. Este modelo es independiente de cualquier consideración de implementación o detalles.
2. **Modelo de datos lógicos**: En este modelo se define cómo se debe implementar el sistema independientemente del DBMS (modelo genérico). El modelo de datos lógico, va un paso más adelante que el modelo de dato conceptual, definiendo tablas, claves primarias, claves foráneas y las restricciones de integridad referencial.
3. **Modelo de datos físico**s: En este modelo se define la implementación concreta utilizando un sistema DBMS. El modelo de datos físico adapta el modelo de datos lógico al DBMS, agregando particularidades propias del gestor, como lo pueden ser la nomenclatura, administración de memoria, almacenamiento en disco y otras definiciones.



**4. Línea de tiempo**