REGLAS DE CODD DEL MODELO RELACIONAL

En la década de los 80's comenzaron a aparecer numerosos Sistemas de Gestión de Bases de Datos que se anunciaban como *relacionales*. Sin embargo estos sistemas carecían de muchas características que se consideran importantes en un sistema relacional, perdiendo muchas ventajas del modelo relacional. Como ejemplo extremo de esto *sistemas relacionales* eran simplemente sistemas que utilizaban tablas para almacenar la información, no disponiendo de elementos como claves primarias, etc.

En 1984 Edgar F. Codd, creador de del Modelo Relacional publicó las **12 Reglas** que un verdadero Sistema Relacional de Bases de Datos debería cumplir. En la práctica algunas de estas reglas son difíciles de implementar, así que un sistema podrá considerarse *más relacional* cuanto más siga estas reglas.

REGLA 0

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.

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 quizás el mismo lenguaje (ejemplo SQL)

REGLA 2: REGLA DEL ACCESO GARANTIZADO

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).

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 (ejemplo, no se sabe la dirección de un empleado), o bien una información que no aplica (a un empleado soltero no se le puede asignar un nombre de esposa). Así mismo, consideremos el caso de un alumno que obtiene 0 puntos en una prueba y el de un alumno que no presentó la prueba.

- 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*. Se crean tablas de verdad para las operaciones lógicas:

  null AND *null* = *null*
  Verdadero AND *null* = *null*
  Falso AND *null* = Falso
  Verdadero OR *null* = Verdadero, etc.

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 normales, de modo que los usuarios autorizados pueden aplicar el mismo lenguaje relacional a su consulta, igual que lo aplican a los datos normales.

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

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*).

- 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.

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.

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 de cada vez).

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.

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. Usando vistas de la unión se pueden recrear las tablas anteriores...

REGLA 10: INDEPENDENCIA DE INTEGRIDAD

Los 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.

REGLA 11: INDEPENDENCIA DE DISTRIBUCIÓN

Una Base de Datos Relacional es independencia 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.

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.