RECUPERACION

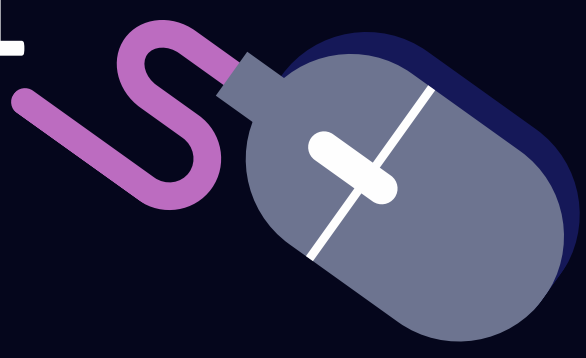
Administracion de bases de datos

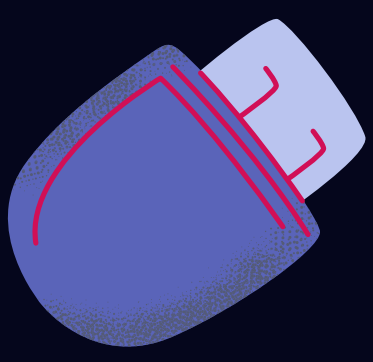
Kevin Hernandez

Administracion de bases de datos

Sentencias DML

Las sentencias DML son un subconjunto del lenguaje SQL (Structured Query Language) que se utilizan para manipular los datos almacenados en una base de datos. A través de estas sentencias, puedes realizar operaciones sobre las tablas, como insertar, actualizar, eliminar y consultar datos.





Principales sentencias DML:

- SELECT: Recupera datos de una o más tablas.
- INSERT: Inserta nuevos registros en una tabla.
- UPDATE: Modifica registros existentes en una tabla.
- DELETE: Elimina registros de una tabla.

Propósito:
Las sentencias DML sirven para interactuar directamente con los datos de una base de datos, realizando operaciones como consultas, inserciones, actualizaciones y eliminaciones.

Sentencias DDL

Las sentencias DDL son un subconjunto de SQL que se utilizan para definir y modificar la estructura de la base de datos, como la creación, modificación y eliminación de tablas, vistas, índices, entre otros objetos.

Principales sentencias DDL:

- CREATE: Crea una nueva tabla, base de datos, vista, etc.
- ALTER: Modifica la estructura de una tabla o base de datos existente.
- DROP: Elimina una tabla, base de datos, índice, etc.
- TRUNCATE: Elimina todos los registros de una tabla (sin eliminar la tabla misma).

Propósito:
Las sentencias DDL permiten definir y cambiar la estructura de la base de datos, sin modificar los datos en sí. Son fundamentales para establecer la arquitectura de la base de datos.

Tipos de JOIN en MySQL

En MySQL, un JOIN se utiliza para combinar filas de dos o más tablas basadas en una condición de relación. Los tipos de JOIN más comunes son:

- INNER JOIN: Devuelve las filas cuando hay una coincidencia en ambas tablas. Si no hay coincidencia, la fila no será incluida.
- LEFT JOIN (o LEFT OUTER JOIN): Devuelve todas las filas de la tabla de la izquierda y las filas coincidentes de la tabla de la derecha. Si no hay coincidencia, se completan con valores nulos.
- RIGHT JOIN (o RIGHT OUTER JOIN): Devuelve todas las filas de la tabla de la derecha y las filas coincidentes de la tabla de la izquierda. Si no hay coincidencia, se completan con valores nulos.
- FULL JOIN (o FULL OUTER JOIN): Devuelve todas las filas cuando hay una coincidencia en una de las tablas. Si no hay coincidencia, las filas de la tabla sin coincidencia se completan con valores nulos. (Nota: MySQL no soporta FULL JOIN de forma nativa, pero se puede simular con UNION).
- CROSS JOIN: Devuelve el producto cartesiano de las dos tablas, es decir, todas las combinaciones posibles de filas de las dos tablas.

Propósito:
Los JOIN permiten combinar datos de diferentes tablas basándose en relaciones entre columnas, de manera que se pueda obtener una vista más completa de los datos.

Administracion de bases de datos

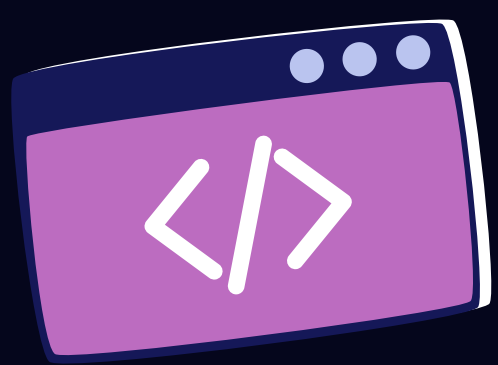
Constraints en SQL

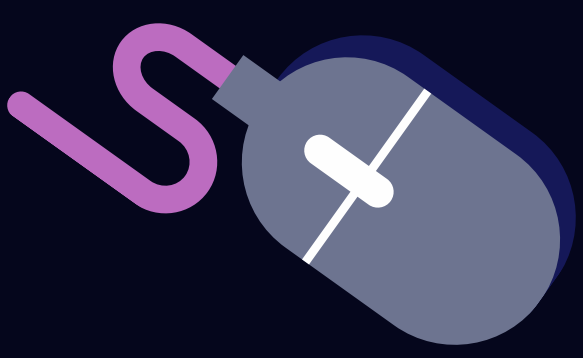
Las constraints (o restricciones) en SQL son reglas que se aplican a las columnas de una tabla para garantizar la integridad de los datos, asegurando que los datos cumplan ciertas condiciones.

Tipos de constraints:

- PRIMARY KEY: Identifica de manera única cada fila en una tabla. No puede haber valores duplicados ni nulos en la columna o columnas que forman la clave primaria.
- FOREIGN KEY: Establece una relación entre dos tablas, asegurando que los valores de una columna (en la tabla secundaria) coincidan con los valores de una columna clave primaria en otra tabla.
- UNIQUE: Asegura que todos los valores de una columna (o un conjunto de columnas) sean únicos.
- NOT NULL: Asegura que una columna no puede tener valores nulos.
- CHECK: Asegura que los valores en una columna cumplan una condición específica.
- DEFAULT: Asigna un valor predeterminado a una columna cuando no se proporciona uno.

Propósito:
Las constraints se utilizan para mantener la integridad referencial y la calidad de los datos dentro de la base de datos.





Diferencia entre una llave primaria (PRIMARY KEY) y una constraint UNIQUE

- PRIMARY KEY:
  - Es una restricción que garantiza que cada fila en una tabla tiene un valor único y no nulo en una columna o combinación de columnas.
  - Solo puede haber una PRIMARY KEY por tabla.
  - No permite valores nulos.
- UNIQUE:
  - También garantiza que los valores en una columna o conjunto de columnas sean únicos.
  - Puede haber múltiples restricciones UNIQUE en una tabla.
  - Permite valores nulos, pero cada valor nulo se considera único.

Diferencia clave:

- PRIMARY KEY: No permite valores nulos y es única en toda la tabla.
- UNIQUE: Permite valores nulos (uno o varios) y puede aplicarse varias veces en una tabla.

Cardinalidad en una base de datos

La cardinalidad se refiere al número de elementos en una relación entre dos tablas.
Existen tres tipos comunes de cardinalidad en bases de datos:

- Cardinalidad uno a uno (1:1): Cada fila en una tabla está relacionada con solo una fila en otra tabla.
- Cardinalidad uno a muchos (1): Una fila en una tabla está relacionada con muchas filas en otra tabla.
- Cardinalidad muchos a muchos (N): Muchas filas en una tabla están relacionadas con muchas filas en otra tabla.

Propósito:
La cardinalidad describe la naturaleza de las relaciones entre las tablas, lo cual es fundamental para el diseño y la normalización de bases de datos.

Diagramas utilizados en la documentación de una base de datos

Para documentar una base de datos, se utilizan varios tipos de diagramas que representan las estructuras y relaciones entre las tablas.

- Diagrama Entidad-Relación (ERD - Entity-Relationship Diagram): Representa las entidades (tablas) y las relaciones entre ellas. Es uno de los diagramas más utilizados para diseñar bases de datos.
- Diagrama de Clases: Utilizado en programación orientada a objetos, pero también se usa en bases de datos para representar las clases (o tablas) y sus relaciones.
- Diagrama de Flujo de Datos (DFD - Data Flow Diagram): Representa el flujo de datos en el sistema, aunque no es específico de bases de datos, a veces se usa para entender cómo interactúan los datos en el sistema.
- Diagrama de Dependencias: Muestra cómo las tablas o elementos dependen de otros en la base de datos.

Propósito:
Los diagramas proporcionan una representación visual de la estructura de la base de datos y sus relaciones, lo cual facilita la comprensión, el diseño y el mantenimiento de la base de datos.