DIGITALHOUSE

Revisión de MySQL

Certified Tech Developer
The Ultimate Degree

Índice

01 Consultas

Recordando consultas

Recordemos una vez más, algunos comandos SQL.

- **INSERT**

  Para insertar datos en columnas específicas, indicamos las columnas entre paréntesis:

```sql
INSERT INTO artistas (nombre, rating)
VALUES ('Maluma', 1.0);
```

- **SELECT**

  Se utiliza para realizar consultas sobre una o varias tablas. Para especificar en qué tabla queremos realizar esta consulta, usamos la palabra FROM seguida del nombre de la tabla.

  .

```sql
SELECT id, titulo, rating
FROM peliculas;
```

Recordando consultas

- **DELETE**

  Elimina información de una tabla. Es importante especificar la condición en el WHERE. Si no se especifica, se eliminan todos los datos de la tabla.

```sql
DELETE FROM artistas WHERE id = 4;
```

- **UPDATE**

  Modifica registros en una tabla. Es importante especificar la condición en el WHERE, para que solo se afecte elo los registros indicados.

```sql
UPDATE artistas
SET nombre = 'Charly Garcia',
rating = 1.0
WHERE id = 1;
```

Recordando consultas

- **WHERE**

  Le permite especificar una condición de búsqueda para los registros devueltos.

```sql
SELECT nombre, apellido
FROM clientes
WHERE pais = 'Brasil';
```

- **ORDER BY**

  Ordena los resultados de una consulta según el valor de la columna especificada. De forma predeterminada, el orden es ascendente (ASC), pero se puede ordenar de forma descendente (DESC) especificándolo en la consulta.

```sql
SELECT nombre, rating
FROM artistas
WHERE rating > 1.0
ORDER BY nombre DESC;
```

Recordando consultas

- **GROUP BY**

  Agrupar los registros de la tabla resultantes de una consulta por una o más columnas

```sql
SELECT id, marca
FROM carro
GROUP BY id,marca;
```

- **HAVING**

  Cumple una función similar que WHERE, la diferencia es que HAVING permite funciones de agregación en las condiciones de selección de datos.

```sql
SELECT pais, COUNT(clienteId)
FROM clientes
GROUP BY pais
HAVING COUNT(clienteId)>=3;
```

02 Procedimientos / Funciones

Stored procedures

Las SPs (stored procedures) son un conjunto de instrucciones en formato SQL que se almacenan, compilan y ejecutan en el lado del servidor de la base de datos.

Puede incluir parámetros de entrada, salida o entrada y salida, devolver resultados tabulares o escalares, mensajes al cliente e invocar instrucciones DDL y DML

SQL

```sql
DELIMITER $$

CREATE PROCEDURE sp_salario(INOUT aumento FLOAT)
BEGIN
     SET aumento = aumento + 10000;
END $$
```

Funciones almacenadas

Una función almacenada en MySQL es una rutina creada para recibir uno o más parámetros, realizar alguna operación y devolver los resultados.

Suelen utilizarse para realizar cálculos sobre los datos, obteniendo así lo que llamamos datos derivados. Esto reduce la necesidad de codificar esta lógica en los programas del cliente.

SQL

```sql
CREATE FUNCTION adicionar_IVA(precio_sin_impuestos DOUBLE(10,12))
RETURNS DOUBLE(10,12)
BEGIN
    DECLARE IVA INT DEFAULT 21;
    RETURN ((precio_sin_impuestos * IVA) / 100) +
precio_sin_impuestos;
END
```

03 Funciones MySQL

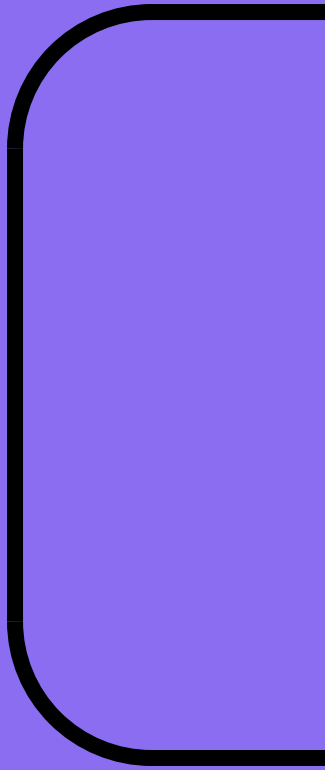
Funciones MySQL

Las funciones son fragmentos de código que realizan operaciones y devuelven un resultado. Algunas funciones aceptan parámetros mientras que otras no.

Las funciones integradas se pueden clasificar básicamente en las siguientes categorías más utilizadas:

- Funciones de string o texto
  Ejemplo: UPPER, LOWER, REPLACE
- Funciones numéricas
  Ejemplo: POW, ROUND, MOD
- Funciones de data
  Ejemplo: CURDATE, DATEDIFF, EXTRACT

04 Variables



Variables

Son elementos que almacenan datos que pueden cambiar durante la ejecución.

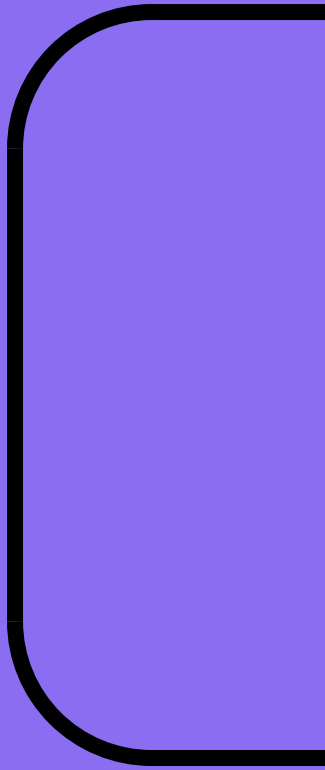
Estos son tipos de variables en el MySQL:

- **Variables definidas por el usuario**
  Se puede acceder a ellos sin necesidad de declararlos o inicializarlos. Se identifican con el símbolo @ utilizado como prefijo. Si no se inicializa, el valor predeterminado es NULL.

- **Variables locales**
  No necesitan el prefijo @ en sus nombres, pero deben declararse antes de usarse.

- **Variables del sistema**
  Se utilizan para almacenar valores que afectan a las conexiones de clientes individuales o que afectan a todo el funcionamiento del servidor.
  Son las variables SESSION o GLOBAL

05 Cursores



Cursores

Un cursor nos permite realizar una o más operaciones para cada uno de los registros de nuestra consulta.

El cursor es un espacio de trabajo temporal creado en la memoria del servidor.

También podemos insertar el resultado del escaneo del cursor en una tabla temporal o fija, según la necesidad.

- **Tablas Temporales**

  Una tabla temporal es un tipo de tabla que nos permite almacenar resultados temporalmente. Podemos acceder a estos datos tantas veces como queramos y siempre que estemos en la misma sesión.

  Estas tablas tienen una vida útil, los datos no se guardan en nuestra base de datos, sino que es temporal, por lo que cuando cerremos nuestra sesión, la tabla se eliminará automáticamente.

06 Triggers

Triggers

Estos son objetos de base de datos que se utilizan para ejecutar código SQL después de que se haya ejecutado una instrucción Insert, Update o Delete en una tabla específica.

- Un activador siempre está asociado a una tabla.
- No se pueden definir disparadores para tablas o vistas temporales (views).

- **Handlers**

  Handler es el controlador de errores. Por cada error que se ejecute, y si esta condición está definida en un handler, se ejecutará el código que definimos.

- **Conditions**

  Estos son nombres que podemos agregar a los errores para que nuestro código sea más legible.

07

Indexación / Optimización

Índices

Un índice es una estructura de datos. Su función es aumentar la velocidad de las consultas sobre una tabla.

Estos son tipos de índices en MySQL:

- **Índices agrupados**
  Identificado como PRIMARIO (PRIMARY). Se almacenan junto con los datos en la propia tabla y ordenan físicamente los registros.
  Por defecto, los campos definidos como Clave principal, Clave externa y Restricción ÚNICA (UNIQUE) son índices principales.

- **Índices no-agrupados**
  Estos se conocen como Secundarios y son creados por el usuario.
  En una tabla podemos tener varios índices secundarios.
  El tamaño de cada índice se suma al tamaño de la tabla.

Tipos de Índices No agrupados

Hay varios tipos de índices no agrupados. Vean algunos de ellos a continuación.

- **Unique**: los índices Unique se crean en columnas que tienen un valor único. No permite valores duplicados.
- **FullText**: índice utilizado para realizar búsquedas textuales con mayor precisión. Es adecuado para aplicaciones con una gran cantidad de texto y que necesitan realizar investigaciones basadas en la relevancia..
- **Index**: es un índice normal, no único y permite valores duplicados.

Índice FullText

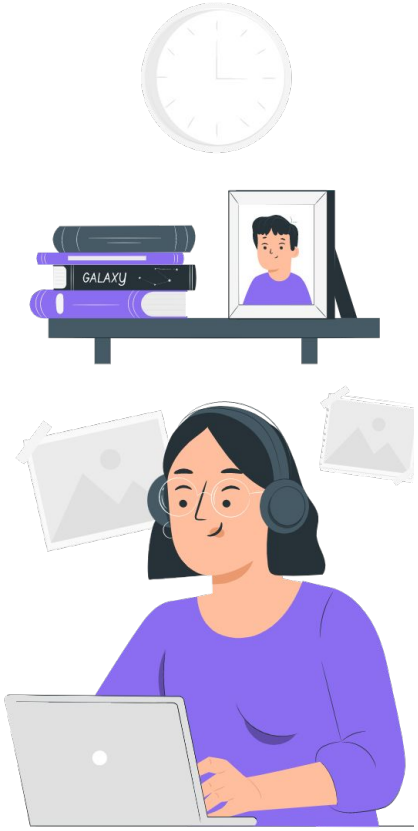
Les compartimos más información sobre el índice FullText.

- Para consultas con un índice de FullText, usamos la función MATCH AGAINST. Esta función recibe el nombre de los campos y el valor a buscar, respectivamente.
- Todas las columnas que componen el índice deben ser informadas en la consulta.
- Tipos de datos admitidos: VARCHAR, TEXT y CHAR.

Optimización

Optimizar una base de datos es crear las condiciones para que funcione de la mejor manera, con el mejor rendimiento.

Para ello, MySQL proporciona algunas herramientas que ayudan al desarrollador a monitorear y localizar posibles problemas que puedan causar “lentitud” en una aplicación.



Informes de optimización

Schema Inspector

Herramienta de MySQL Workbench que proporciona información sobre la base de datos. Por ejemplo: tamaño de la base de datos, número de tablas, columnas, índices y procedimientos creados. También proporciona opciones de mantenimiento como Analyze, Optimize, Check Table y Checksum.

Table Inspector

Proporciona información sobre una tabla específica, como el tamaño de la tabla, las columnas, los índices, los procedimientos creados y los permisos de usuario.

Informes de optimización

Herramientas de profiling

Estas herramientas se utilizan para monitorear y diagnosticar consultas realizadas en una base de datos.

Log

Tiene recursos para registrar las consultas que son lentas.
Se puede utilizar para analizar el consumo de recursos en la ejecución de la instrucción en la sesión actual.

Informes de optimización

Profile

Se utiliza para analizar el consumo de recursos en la ejecución de la instrucción en la sesión actual.

DBeaver

Agrega todas las opciones en un solo lugar. Con esta herramienta podemos consultar el Log, el plan de ejecución, consultar plazos en una base de datos, además de beneficiarnos de las opciones de visualización de resultados y utilizar modelos de consulta pre almacenados en la herramienta.

Informes de optimización

Planes de ejecución

Herramienta que muestra el orden de consulta definido por el optimizador y el performance de la consulta.

Comando Explain / Diagrama Visual Explain

Podemos utilizar el comando Explique o el recurso visual que proporciona el Workbench, que nos permite detectar posibles problemas de rendimiento y actuar con mayor rapidez para corregir la consulta.

DIGITALHOUSE

¡Muchas gracias!