Taller 7

Juan Pablo Guerrero Camacho - 02210131026

Ingeniería De Sistemas

Universidad de Santander UDES

Cúcuta, Norte de Santander

2023

Descargar MySQL en Linux

Paso 1: Actualiza los repositorios del sistema:

Antes de descargar cualquier software en Linux, es una buena práctica actualizar los repositorios del sistema. Puedes hacerlo ejecutando el siguiente comando en la terminal:

sudo apt-get update

Paso 2: Descarga e instala MySQL:

MySQL está disponible en los repositorios de software predeterminados de la mayoría de las distribuciones de Linux, por lo que puedes descargar e instalar MySQL utilizando el siguiente comando en la terminal:

sudo apt-get install mysql-server

Durante la instalación, se te pedirá que establezcas una contraseña para la cuenta de root de MySQL. Asegúrate de establecer una contraseña segura y recuerda guardarla en un lugar seguro.

Paso 3: Verifica que MySQL se ha instalado correctamente:

Para verificar que MySQL se ha instalado correctamente, ejecuta el siguiente comando en la terminal:

```
systemctl status mysql
```

Paso 4: Configura MySQL:

Después de instalar MySQL, es una buena práctica configurar algunos aspectos importantes de MySQL. Para hacerlo, puedes ejecutar el siguiente comando en la terminal:

```
sudo mysql_secure_installation
```

Este comando te guiará a través de una serie de preguntas y configuraciones de seguridad para MySQL. Asegúrate de seguir las instrucciones cuidadosamente para garantizar la seguridad de tu servidor de MySQL.

Paso 5: Comprueba la versión de MySQL instalada:

Para verificar la versión de MySQL instalada en tu sistema, puedes ejecutar el siguiente comando en la terminal:

mysql –versión

Este comando te mostrará la versión de MySQL que está instalada en tu sistema.

Descargar PostgreSQL en Linux

Paso 1: Actualiza los repositorios del sistema:

Antes de descargar cualquier software en Linux, es una buena práctica actualizar los repositorios del sistema. Puedes hacerlo ejecutando el siguiente comando en la terminal:

sudo apt-get update

Paso 2: Descarga e instala PostgreSQL:

PostgreSQL está disponible en los repositorios de software predeterminados de la mayoría de las distribuciones de Linux, por lo que puedes descargar e instalar PostgreSQL utilizando el siguiente comando en la terminal:

sudo apt-get install postgresql postgresql-contrib

Durante la instalación, se te pedirá que establezcas una contraseña para la cuenta de postgres de PostgreSQL. Asegúrate de establecer una contraseña segura y recuerda guardarla en un lugar seguro.

Paso 3: Verifica que PostgreSQL se ha instalado correctamente:

Para verificar que PostgreSQL se ha instalado correctamente, ejecuta el siguiente comando en la terminal:

```
systemctl status postgresql
```

Si PostgreSQL se ha instalado correctamente, deberías ver un mensaje que indica que el servicio está activo.

Paso 4: Configura PostgreSQL:

Después de instalar PostgreSQL, es una buena práctica configurar algunos aspectos importantes de PostgreSQL. Para hacerlo, puedes ejecutar el siguiente comando en la terminal:

```
sudo nano /etc/postgresql/13/main/pg_hba.conf
```

Este comando abrirá el archivo de configuración de PostgreSQL en el editor de texto nano. Busca las líneas que contienen "IPv4 local connections" y "IPv6 local connections", y cambia "peer" a "md5" en ambas líneas. Deberían verse así:

```
# IPv4 local connections:
host    all             all             127.0.0.1/32            md5
# IPv6 local connections:
host    all             all             ::1/128                 md5
```

Guarda los cambios y cierra el archivo.

Paso 5: Comprueba la versión de PostgreSQL instalada:

Para verificar la versión de PostgreSQL instalada en tu sistema, puedes ejecutar el siguiente comando en la terminal:

```
psql –version
```

Este comando te mostrará la versión de PostgreSQL que está instalada en tu sistema.

CREATE VIEW en MySQL

Sintaxis básica de esta función

CREATE [OR REPLACE] VIEW nombre_vista [column_list]

AS consulta_SELECT

Explicación del Código:

OR REPLACE: Reemplaza una vista existente en caso de coincidir en nombre.

nombre_vista: Nombre de la vista a crear.

column_list: Listado de columnas a crear.

consulta_SELECT: Consulta SELECT que queremos realizar para obtener la información que contendrá la vista.

Ventajas de usar vistas en MySQL

- Privacidad de la información: Mostramos a los usuarios con acceso a la vista únicamente la información que creamos conveniente. De esta manera no se tiene acceso a la tabla original con todas sus filas y columnas.
- Optimización del rendimiento de la base de datos: Podemos crear de querys sobre vistas complejas, es decir, vistas cuya información ha sido extraída y creada a través de unas

SELECT complejas. De esta manera nos ahorramos estar ejecutando queys pesadas y atacamos directamente al resultado de dichas querys.

- Tablas de prueba: Para los desarrolladores que no tengan entornos de preproducción es muy útil usar las vistas para no tener miedo a perder información.

CREATE VIEW en PostgreSQL

La sintaxis de definición de una vista en SQL es:

```
CREATE [OR REPLACE] VIEW nombre_de_vista

AS sentencia_SELECT
```

Explicación del Código:

```
CREATE [OR REPLACE] VIEW <nombre_de_vista>  AS

< SELECT campos1 [, campo2, ... , campoN ]

FROM tabla1 [, tabla2, ... , tablaN ]

[ WHERE condiciones_de_consulta ]

[ ORDER BY lista_de_campos ]

[ GROUP BY lista_de_campos ] >
```

Ejemplo sencillo:

```
CREATE VIEW cliente_apellido AS

SELECT* FROM clientes

WHERE ap_paterno LIKE 'A%'
```

Ventajas y Desventajas de las Vistas

Ventajas

- SEGURIDAD: Las vistas pueden proporcionar un nivel adicional de seguridad. Por ejemplo, en la tabla de empleados, cada responsable de departamento sólo tendrá acceso a la información de sus empleados.
- SIMPLICIDAD: Las vistas permiten ocultar la complejidad de los datos. Una base de datos se compone de muchas tablas. La información de dos o más tablas puede recuperarse utilizando una combinación de dos o más tablas (relacional), y estas combinaciones pueden llegar a ser muy confusas. Creando una vista como resultado de la combinación se puede ocultar la complejidad al usuario.
- ORGANIZACION: Las vistas ayudan a mantener unos nombres razonables en la base de datos para acceder a consultas complejas.
- EXACTITUD EN LOS DATOS SOLICITADOS: Permiten acceder a un subconjunto de datos específicos, omitiendo datos e información innecesaria e irrelevante para el usuario.
- AMPLIA PERSPECTIVAS DE LA BASE DE DATOS: Proporciona diversos modelos de información basados en los mismos datos, enfocándolos hacia distintos usuarios con necesidades especificas. El mostrar la información desde distintos ángulos nos ayuda a crear ambientes de trabajo y operación acordes a los objetivos de la empresa. Debe evaluarse el perfil y requerimientos de información de los usuarios destino de la vista.
- TRANSPARENCIA EN LAS MODIFICACIONES: El usuario final no se verá afectado por el diseño o alteraciones que se realicen en el esquema conceptual de la base de datos. Si el sistema requiere una modificación en su funcionamiento interno, podrán afectarse diversas estructuras que proveen el desempeño de este; se pretende que los usuarios finales no adviertan tales alteraciones.

Desventajas

- NO SON ACTUALIZABLES : Las vistas en Postgre no son actualizables, es decir, si bien es cierto, son tratadas como tablas, no es posible hacer INSERT, DELETE ni UPDATE sobre las vistas, esta desventaja es una característica particular en Postgre dado que esta cualidad si esta disponible en otros motores de bases de datos como ORACLE, Informix y SQL Server, sin embargo cabe notar que Postgre cubre esta falencia en las vistas con la creación de reglas (CREATE RULE) que permite llenar el vacío dejado por la vista.

CREATE TRIGGER en MySQL

Los triggers o disparadores de MySQL son usa serie de reglas predefinidas que están asociadas a una tabla. Estas reglas permiten la ejecución de una serie de instrucciones cuando se producen ciertos eventos como pueden ser la inserción de un nuevo registro, la actualización o el borrado de los datos de una tabla.

Técnicamente, un trigger es un objeto de una base de datos que está asociado a una tabla, y que será activado cuando la acción que tiene asociada tiene lugar. El trigger se puede ejecutar cuando tiene lugar una acción INSERT, UPDATE o DELETE, siendo posible su ejecución tanto antes como después del evento.

Vamos a crear un trigger que actualice automáticamente el precio de los productos de la tabla cada vez que se actualice su coste. Le llamaremos actualizarPrecioProducto.

11

```
DELIMITER $$
CREATE TRIGGER 'actualizarPrecioProducto'
BEFORE UPDATE ON 'productos'
FOR EACH ROW
BEGIN
  IF NEW.coste <> OLD.coste
    THEN
      SET NEW.precio = NEW.coste * 2;
  END IF ;
END$$
DELIMITER ;
```

Lo que hacemos es crear un trigger que se ejecute antes de la actualización del registro, algo que indicamos con la sentencia «BEFORE UPDATE ON«. Luego comprobamos si el coste antiguo del producto difiere del nuevo y, si es así, actualizamos el precio con el doble del valor de su nuevo coste.

Una vez hayamos creado el trigger, no tendremos que hacer absolutamente nada para llamarlo, puesto que el motor de la base de datos lo invocará automáticamente cada vez que se actualice un registro de la tabla de productos.

Sin embargo, sí podemos comprobar el resultado del trigger actualizando un registro con una sentencia UPDATE como esta:

```
UPDATE productos SET coste = 5 WHERE id = 1;
SELECT * FROM productos;
```

Puedes eliminar una trigger haciendo uso de la sentencia DROP TRIGGER Por ejemplo, si quieres eliminar el trigger actualizarPrecioProducto del ejemplo anterior, tendrás que ejecutar esta sentencia:

```
DROP TRIGGER actualizarPrecioProducto;
```

CREATE TRIGGER en PostgreSQL

La sintaxis de la definición de un trigger es:

```
CREATE TRIGGER nombreDisparador { BEFORE | AFTER } { evento [ OR…] }
ON nombreTabla [FOR [EACH] {ROW | STATEMENT }  ] EXECUTE
PROCEDURE nombreFuncionDisparadora (argumentos)
```

Donde evento es la acción bajo la cual se ejecuta el disparador (sólo tiene sentido si es un disparador de fila) y puede tomar los valores: Insert, Update, Delete o Truncate. El parámetro argumentos convierte los argumentos especificados a una variable de tipo TG_ARGV[] accesible por la función disparadora.

¿Qué es Docker?

El término "Docker" se aplica a diferentes conceptos, entre los que se incluyen un proyecto de la comunidad open source y sus herramientas; Docker Inc., la principal empresa promotora del proyecto; y las herramientas que la empresa respalda formalmente. El hecho de que las tecnologías y la compañía compartan el mismo nombre puede ser confuso.

Esto es lo que debe saber:

- El sistema de software de TI llamado "Docker" es la tecnología de organización en contenedores que posibilita la creación y el uso de los contenedores de Linux.
- La comunidad open source Docker se encarga de mejorar estas tecnologías para beneficiar a todos los usuarios.
- La empresa, Docker Inc., se basa en el trabajo de la comunidad Docker para aumentar la seguridad de las herramientas y comparte los avances con el resto de la comunidad. Entonces, brinda soporte a las tecnologías mejoradas y reforzadas para los clientes empresariales.

Con Docker, puede utilizar los contenedores como máquinas virtuales muy livianas y modulares, y obtiene la flexibilidad necesaria para crearlos, implementarlos, copiarlos y trasladarlos de un entorno a otro, lo cual le permite optimizar las aplicaciones para la nube.

Software Visual Portainer

Portainer es un interfaz ligero de usuario para Gestión de contenedores docker. El software de código abierto centraliza y unifica la gestión de las infraestructuras de clústeres existentes. Para ello se utiliza una interfaz gráfica de usuario basada en la web. Además de la "Community Edition" (CE) gratuita, existe una versión de pago con soporte empresarial.

El objetivo principal de Portainer es unificar la gestión de los desarrollos existentes de Kubernetes. Portainer permite a los equipos de DevOps gestionar, configurar y asegurar de forma centralizada los entornos multiclúster. Los equipos de desarrollo también se benefician del software; por ejemplo, Portainer facilita el despliegue, la gestión y la resolución de problemas de las aplicaciones.

En este caso, Portainer no solo es adecuado como interfaz de gestión para Kubernetes; la gestión de clústeres y contenedores basada en Docker y Docker Swarm también forma parte del ámbito funcional. El software puede instalarse prácticamente en cualquier lugar. Por tanto,

la instalación funciona en entornos de nube, en dispositivos de vanguardia, así como en tu propia infraestructura informática local.

Bibliografía

- https://www.digitalocean.com/community/tutorials/crear-un-nuevo-usuario-y-otorgarle-permisos-en-mysql-es
- http://www.personal.fi.upm.es/~lmengual/GESTION_BD/GBD_GESTION_USUARIOS.pdf
- https://www.arteco-consulting.com/post/gestion-de-usuarios-en-postgresql
- https://www.anerbarrena.com/create-view-mysql-5101/
- https://silo.tips/download/vistas-en-postgresql
- https://www.neoguias.com/como-crear-y-utilizar-triggers-en-mysql/
- https://mappinggis.com/2016/06/crear-ejecutar-disparador-trigger-postgis/#:~:text=En%20Pl%2FPgsql%20una%20función,desde%20la%20sentencia%20Create%20Trigger
- https://www.redhat.com/es/topics/containers/what-is-docker
- https://www.ionos.es/digitalguide/servidores/configuracion/instalar-portainer-en-docker/