

Nombre de la práctica	Motores de Base de Datos de MySQL 8			No.	1
Asignatura:	Administración de Base de Datos	Carrera:	Ingeniería en sistemas computacionales	Duración de la práctica (Hrs)	

Alumno: Orlando Urbano Trejo
Competencia(s) específica(s)

II. Lugar de realización de la práctica (laboratorio, taller, aula u otro):

- Salon de clases.
- Casa.

III. Material empleado:

- Navicat.
- MariaDB.
- Laptop.

IV. Desarrollo de la práctica:

1. Motor de Almacenamiento MEMORY:

El motor de almacenamiento **MEMORY** (también conocido como HEAP) en MySQL o MariaDB se utiliza para crear tablas que se almacenan completamente en la memoria RAM del servidor en lugar de en el disco.

Ventajas:

- **Alta velocidad de acceso:** Almacenar tablas completamente en la memoria RAM proporciona un acceso extremadamente rapido a los datos.
- **Ideal para datos temporales:** Son útiles para almacenar datos temporales que no necesitan persistencia más allá de la duración de una sesión o de una ejecución de consulta.
- **Bajo consumo de recursos:** Este tipo de tablas suele requerir menos recursos del sistema, lo que las hace adecuadas para entornos con restricciones de recursos.

Desventajas:

- **Volatilidad de datos:** Los datos almacenados en estas tablas se pierden en caso de un reinicio del servidor o un fallo del sistema.
- **Bloqueo de nivel de tabla:** Este motor utiliza un mecanismo de bloqueo de nivel de tabla en lugar de un bloqueo de nivel de fila, porque las operaciones de escritura en tablas **MEMORY** resultan en bloqueos mas grandes y afectan el rendimiento.

Demostración Motor de Almacenamiento MEMORY:

1. Crear la base de datos para este ejemplo.

```
MariaDB [(none)]> CREATE DATABASE memory;  
Query OK, 1 row affected (0.009 sec)
```

2. Creamos un usuario para la base de datos.

```
MariaDB [(none)]> CREATE USER 'memory'@'%' IDENTIFIED BY 'orlando';  
Query OK, 0 rows affected (0.015 sec)
```

3. Damos los permisos al usuario para poder acceder a la base de datos.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON memory.* TO 'memory'@'%';  
Query OK, 0 rows affected (0.004 sec)
```

4. Guardamos cambios.

```
MariaDB [(none)]> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0.001 sec)
```

5. Accedemos otra vez con ese nuevo usuario y contraseña:

```
starlord@fedora:~/Documents/Sexto_Semestre/Administracion_Base_De_Datos/SQL$ mysql -u memory -p  
Enter password: 
```

Hemos accedido correctamente.

```
starlord@fedora:~/Documents/Sexto_Semestre/Administracion_Base_De_Datos/SQL$ mysql -u memory -p  
Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 6  
Server version: 10.5.23-MariaDB MariaDB Server  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]>
```

6. Creamos una tabla con algunos campos pero al final de la inserción especificamos que nuestra tabla tendra el motor de **MEMORY**.

```
MariaDB [memory]> CREATE TABLE Alumno (  
-> folio_alumno VARCHAR(5),  
-> nombre_alumno VARCHAR(50),  
-> apellido_paterno VARCHAR(50),  
-> apellido_materno VARCHAR(50),  
-> matricula VARCHAR(9),  
-> PRIMARY KEY (folio_alumno)  
-> ) ENGINE = MEMORY;  
Query OK, 0 rows affected (0.013 sec)
```

7. Ahora hacemos una insercion a esa tabla.

```
MariaDB [memory]> INSERT INTO Alumno VALUES ('12345','Orlando','Urbano','Trejo','202123423');  
Query OK, 1 row affected (0.000 sec)
```

8. Realizamos un **SELECT** sobre la tabla para ver que la inserción se realizo.

```
MariaDB [memory]> SELECT * FROM Alumno;  
+-----+-----+-----+-----+-----+  
| folio_alumno | nombre_alumno | apellido_paterno | apellido_materno | matricula |  
+-----+-----+-----+-----+-----+  
| 12345        | Orlando       | Urbano           | Trejo            | 202123423 |  
+-----+-----+-----+-----+-----+  
1 row in set (0.000 sec)
```

9. Reiniciamos nuestra computadora.

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ init 6
```

10. Una vez reiniciada la computadora volvemos a iniciar mariadb con el usuario y la contraseña correspondiente, y realizamos en **SELECT** sobre la misma tabla.

```
MariaDB [memory]> SELECT * FROM Alumno;  
Empty set (0.000 sec)
```

Con esto nos damos cuenta de que nuestro registro que se habia realizado previamente se elimino y esto se debe a que al crear la tabla especificamos con el **MEMORY** que los datos iban a ser temporales es decir que se ibana a borrar cuando la computadora se apagara o se reiniciara el sistema operativo.

2. Motor de Almacenamiento CSV:

El motor de almacenamiento **CSV** permite crear tablas que almacenan datos en archivos **CSV** (Valores Separados por Comas) en el sistema de archivos del servidor.

Ventajas:

- **Portabilidad de datos:** Los archivos CSV son faciles de mover y de compartir entre diferentes sistemas y plataformas, lo que puede ser útil para transferir datos entre entornos de desarrollo, pruebas y producción.
- **Facilidad de lectura y edición:** Los archivos CSV son legibles por humanos y se pueden editar facilmente con programas de hojas de cálculo o editores de texto simples, lo que facilita la manipulación de datos almacenados en tablas CSV.
- **Estructura simple y ligera:** Los archivos CSV no tienen una estructura compleja y no requieren un procesamiento intensivo para almacenar o recuperar datos, lo que puede resultar en un rendimiento más rápido en comparación con otros motores de almacenamiento mas complejos.

Desventajas:

- **Ausencia de soporte para índices y claves foráneas:** El motor de almacenamiento CSV no admite la creación de índices ni claves foráneas, lo que puede limitar la eficiencia de las consultas y las operaciones que requieren búsquedas rápidas o relaciones entre tablas.
- **Pérdida de tipos de datos y metadatos:** Los archivos CSV no conservan la información sobre los tipos de datos de las columnas ni otros metadatos de la tabla.

Demostración Motor de Almacenamiento CSV:

1. Crear la base de datos para este ejemplo.

```
MariaDB [(none)]> CREATE DATABASE csv;  
Query OK, 1 row affected (0.001 sec)
```

2. Creamos un usuario para la base de datos.

```
MariaDB [(none)]> CREATE USER 'csv'@'%' IDENTIFIED BY 'orlando';  
Query OK, 0 rows affected (0.025 sec)
```

3. Damos los permisos al usuario para poder acceder a la base de datos.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON csv.* TO 'csv'@'%';  
Query OK, 0 rows affected (0.012 sec)
```

4. Guardamos cambios.

```
MariaDB [(none)]> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0.001 sec)
```

5. Accedemos otra vez con ese nuevo usuario y contraseña:

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u csv -p
Enter password:
```

Hemos accedido correctamente.

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u csv -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 4
Server version: 10.5.23-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

6. Creamos una tabla con algunos campos pero al final de la inserción especificamos que nuestra tabla tendra el motor de CSV.

```
MariaDB [csv]> CREATE TABLE Producto (
-> folio_producto VARCHAR(5) NOT NULL,
-> nombre VARCHAR(50) NOT NULL,
-> precio INT NOT NULL,
-> descripcion VARCHAR(100) NOT NULL
-> ) ENGINE = CSV;
Query OK, 0 rows affected (0.008 sec)
```

7. Ahora hacemos algunas inserciones sobre la tabla.

```
MariaDB [csv]> INSERT INTO Producto VALUES ('12345','Jamon',12,'Jamon de pavo');
Query OK, 1 row affected (0.008 sec)

MariaDB [csv]> INSERT INTO Producto VALUES ('54380','Queso',25,'Queso fresco de vaca');
Query OK, 1 row affected (0.000 sec)
```

8. Realizamos un **SELECT** sobre la tabla para ver que la inserción se realizo.

```
MariaDB [csv]> SELECT * FROM Producto;
+-----+-----+-----+-----+
| folio_producto | nombre | precio | descripcion |
+-----+-----+-----+-----+
| 12345         | Jamon  | 12     | Jamon de pavo |
| 54380         | Queso  | 25     | Queso fresco de vaca |
+-----+-----+-----+-----+
2 rows in set (0.001 sec)
```

Ahora para poder comprobar que nuestros registros se estan ingresando en un archivo CSV hay que ingresar a la siguiente dirección **/var/lib/mysql/**

9. Ingresamos a la ruta con **cd**

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ cd /var/lib/mysql/
```

10. Hacemos un ls sobre esa carpeta.

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql$ ls
aria_log.00000001  aria_log_control  crud_python  ib_buffer_pool  ib_logfile0  memory  mysql  mysql_upgrade_info  zoologico_orlando
crud_php          csv              ibdata1      ibtmp1          multi-master.info  mysql.sock  performance_schema
```

11. Ingresamos como usuario root con el comando **sudo su**

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql$ sudo su
[sudo] password for starlord:
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql#
```

12. Damos permisos sobre la base de datos para poder acceder a la carpeta donde se encuentran los registros con el comando **chmod 777 csv/**

```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql# chmod 777 csv/
```

13. Ingresamos a la carpeta con un **cd csv/** e ingresamos un **ls** sobre esa misma carpeta.

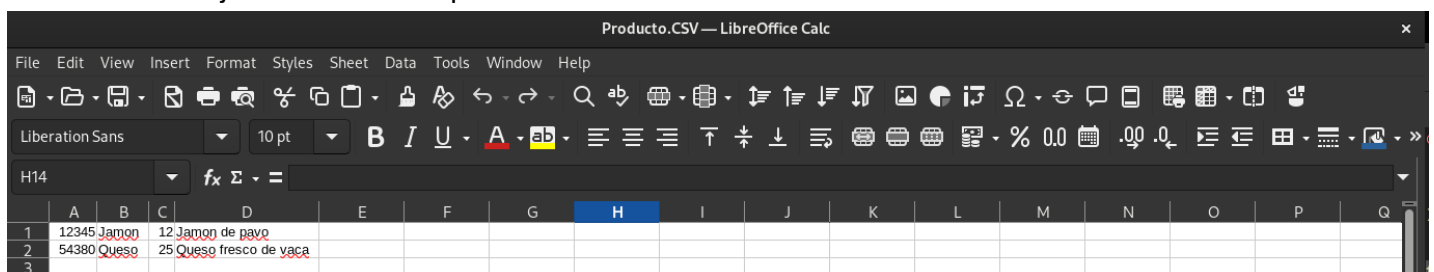
```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql# cd csv/
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/csv# ls
db.opt  Producto.CSM  Producto.CSV  Producto.frm
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/csv#
```

Ubicamos un archivo llamado **Producto.CSV** y procedemos a abrirlo y ver que informacion contiene.

14. Pero antes hay que dar permisos de lectura, escritura y ejecución y esto es con el siguiente comando **chmod 777 Producto.CSV**.

```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/csv# chmod 777 Producto.CSV
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/csv#
```

Procedemos a ejecutar el archivo para ver su contenido.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	12345	Jamon	12	Jamon de pavo													
2	54380	Queso	25	Queso fresco de vaca													
3																	

Con esto comprobamos que los registros que estamos haciendo se estan registrando en un archivo CVS y es mas sencillo de poder analizar este tipo información que desde la terminal o desde el gestor en el que esten trabajando.

3. Motor de Almacenamiento ARCHIVE:

Es un motor especializado diseñado para la comprensión y el almacenamiento eficiente de grandes cantidades de datos históricos o de solo lectura.

Ventajas:

- **Eficiencia en el almacenamiento de grandes volúmenes de datos:** ARCHIVE es ideal para almacenar grandes cantidades de datos de forma eficiente gracias a su compresión de datos.
- **Ahorro de espacio en disco:** Al reducir el tamaño de los datos almacenados en disco mediante la compresión puede ayudar a ahorrar espacio en disco.
- **Rendimiento mejorado en operación de solo lectura:** ARCHIVE está optimizado para operaciones de lectura eficientes, lo que puede mejorar el rendimiento en consultas que acceden a datos almacenados en tablas.

Desventajas:

- **No es apto para operaciones de escritura frecuentes:** No es adecuado para operaciones de escritura frecuentes o actualizaciones de datos. Ya que la compresión de datos y descompresión pueden llegar a ser costosas.
- **Recuperación de datos más lenta:** Debido a la compresión de datos la recuperación de datos de tablas ARCHIVE puede ser más lenta que en otros motores de almacenamiento.

Demostración del motor de Almacenamiento ARCHIVE:

1. Crear la base de datos para este ejemplo.

```
MariaDB [(none)]> CREATE DATABASE archive;  
Query OK, 1 row affected (0.001 sec)
```

2. Creamos un usuario para la base de datos.

```
MariaDB [(none)]> CREATE USER 'archive'@'%' IDENTIFIED BY 'orlando';  
Query OK, 0 rows affected (0.011 sec)
```

3. Damos los permisos al usuario para poder acceder a la base de datos.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON archive.* TO 'archive'@'%';  
Query OK, 0 rows affected (0.012 sec)
```

4. Guardamos cambios.

```
MariaDB [(none)]> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0.001 sec)
```

5. Accedemos otra vez con ese nuevo usuario y contraseña:

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u archive -p  
Enter password: █
```

Hemos accedido correctamente.

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u archive -p  
Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 7  
Server version: 10.5.23-MariaDB MariaDB Server  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]>
```

6. Creamos una tabla con algunos campos pero al final de la inserción especificamos que nuestra tabla tendra el motor de CSV.

```
MariaDB [archive]> CREATE TABLE Cliente (  
-> folio_Cliente VARCHAR(5),  
-> nombre VARCHAR(50),  
-> apellido VARCHAR(50),  
-> email VARCHAR(30),  
-> telefono VARCHAR(10)  
-> ) ENGINE = ARCHIVE;  
Query OK, 0 rows affected (0.021 sec)
```

7. Hacemos una inserción sobre la base de datos.

```
MariaDB [archive]> INSERT INTO Cliente VALUES ('54032','Orlando','Urbano','urbanoorlando79@gmail.com','5614364142');  
Query OK, 1 row affected (0.006 sec)
```

8. Realizamos un **SELECT** sobre la tabla para ver el registro.

```
MariaDB [archive]> SELECT * FROM Cliente;  
+-----+-----+-----+-----+-----+  
| folio_Cliente | nombre | apellido | email | telefono |  
+-----+-----+-----+-----+-----+  
| 54032 | Orlando | Urbano | urbanoorlando79@gmail.com | 5614364142 |  
+-----+-----+-----+-----+-----+  
1 row in set (0.000 sec)
```

Ahora para poder comprobar que nuestros registros se encuentran en un archivo con terminación **ARZ** tenemos que ingresar a la siguiente ruta **/var/lib/mysql/**

9. Ingresamos a la ruta con **cd**

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ cd /var/lib/mysql/
```

10. Hacemos un **ls** sobre esa carpeta.

```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql# ls  
aria_log.00000001  crud_php  ibdata1  ibtmp1  multi-master.info  mysql.sock  performance_schema  
archive  aria_log_control  crud_python  ib_buffer_pool  ib_logfile0  memory  mysql  mysql_upgrade_info  zoologico_orlando  
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql#
```

11. Ingresamos como usuario root con el comando **sudo su**

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql$ sudo su  
[sudo] password for starlord:  
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql#
```

12. Damos permisos sobre la base de datos para poder acceder a la carpeta donde se encuentran los registros con el comando **chmod 777 archive/**

```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql# sudo chmod 777 archive/
```

13. Ingresamos a la carpeta con un **cd archive/** e ingresamos un **ls** sobre esa misma carpeta.

```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql# cd archive/  
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/archive# ls  
Cliente.ARZ  Cliente.frm  db.opt  
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/archive#
```

Ubicamos un archivo llamado **Cliente.ARZ** y revisamos la capacidad que tiene.

```
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/archive# du -h Cliente.ARZ
4.0K    Cliente.ARZ
root@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:/var/lib/mysql/archive#
```

Y una vez teniendo esta cantidad podemos ir haciendo inserciones u operaciones en las tablas y ver cual es el comportamiento del este tipo de motor de almacenamiento.

4. Motor de Almacenamiento MyISAM:

Es un motor de almacenamiento antiguo que requiere de un alto rendimiento en operacion de lectura y búsqueda de texto completo, pero puede no ser la mejor opción para aplicaciones que requieren transacciones o que tienen requisitos de escritura intensiva.

Ventajas:

- **Facilidad de mantenimiento y reparación:** MyISAM es relativamente fácil de mantener y reparar en comparación con otros motores de almacenamiento más complejos.
- **Soporte para índices de búsqueda de texto completo:** MyISAM es una opción popular para aplicaciones que requieren búsqueda de texto completo, como motores de búsqueda o sistemas de gestión de contenidos.

Desventajas:

- **Falta de soporte para transacciones:** MyISAM no admite transacciones ACID, lo que puede ser una limitación en aplicaciones que requieren operaciones atómicas o garantías de integridad de datos.
- **Bloqueo de nivel de tablas:** El bloqueo de nivel de tabla puede resultar en cuellos de botella en entornos con alta concurrencia, ya que las operaciones de escritura bloquean toda la tabla.

Demostración del motor de Almacenamiento MyISAM:

1. Crear la base de datos para este ejemplo.

```
MariaDB [(none)]> CREATE DATABASE myISAM;
Query OK, 1 row affected (0.000 sec)
```

2. Creamos un usuario para la base de datos.

```
MariaDB [(none)]> CREATE USER 'myISAM'@'%' IDENTIFIED BY 'orlando';
Query OK, 0 rows affected (0.013 sec)
```

3. Damos los permisos al usuario para poder acceder a la base de datos.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON myISAM.* TO 'myISAM'@'%;
Query OK, 0 rows affected (0.012 sec)
```

4. Guardamos cambios.

```
MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.001 sec)
```

5. Accedemos otra vez con ese nuevo usuario y contraseña:

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u myISAM -p
Enter password:
```


Hemos accedido correctamente.

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u myISAM -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 10
Server version: 10.5.23-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

6. Creamos una tabla con algunos campos pero al final de la inserción especificamos que nuestra tabla tendra el motor de MyISAM.

```
MariaDB [myISAM]> CREATE TABLE proveedor (
-> folio_proveedor VARCHAR(5),
-> nombre VARCHAR(50),
-> email VARCHAR(30),
-> telefono VARCHAR(10),
-> PRIMARY KEY(folio_proveedor)
-> ) ENGINE = MyISAM;
Query OK, 0 rows affected (0.006 sec)
```

7. Hacemos varias insercion a la tabla.

```
MariaDB [myISAM]> INSERT INTO proveedor VALUES ('43810','Gamesa','gamesa@gmail.com','6655774488');
Query OK, 1 row affected (0.012 sec)

MariaDB [myISAM]> INSERT INTO proveedor VALUES ('55431','Sabritas','sabritas@gmail.com','0908764333');
Query OK, 1 row affected (0.000 sec)

MariaDB [myISAM]> INSERT INTO proveedor VALUES ('23456','De la Rosa','rosa@gmail.com','8866774431');
Query OK, 1 row affected (0.000 sec)
```

8. Realizamos en **SELECT** sobre la tabla para ver el registro.

```
MariaDB [myISAM]> SELECT * FROM proveedor;
+-----+-----+-----+-----+
| folio_proveedor | nombre   | email                | telefono   |
+-----+-----+-----+-----+
| 43810           | Gamesa   | gamesa@gmail.com     | 6655774488 |
| 55431           | Sabritas | sabritas@gmail.com   | 0908764333 |
| 23456           | De la Rosa | rosa@gmail.com       | 8866774431 |
+-----+-----+-----+-----+
3 rows in set (0.001 sec)
```

Para revisar el comportamiento de este tipo de motor se tienen que realizar algunas operaciones las cuales van a facilitar identificar algun tipo de problema. Serán dos actualizaciones y dos eliminaciones..

9. Hacemos las actualizaciones a los registros.

```
MariaDB [myISAM]> UPDATE proveedor SET email= 'gamesa12@gmail.com' WHERE folio_proveedor = '43810';
Query OK, 1 row affected (0.023 sec)
Rows matched: 1 Changed: 1 Warnings: 0

MariaDB [myISAM]> UPDATE proveedor SET email= 'sabritas22@gmail.com' WHERE folio_proveedor = '55431';
Query OK, 1 row affected (0.000 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

10. Hacemos un **SELECT** para revisar los cambios realizados.

```
MariaDB [myISAM]> SELECT * FROM proveedor;
```

folio_proveedor	nombre	email	telefono
43810	Gamesa	gamesa12@gmail.com	6655774488
55431	Sabritas	sabritas22@gmail.com	0908764333
23456	De la Rosa	rosa@gmail.com	8866774431

```
3 rows in set (0.001 sec)
```

11. Por ultimo hacemos las eliminaciones.

```
MariaDB [myISAM]> DELETE FROM proveedor WHERE folio_proveedor = '43810';
Query OK, 1 row affected (0.000 sec)

MariaDB [myISAM]> DELETE FROM proveedor WHERE folio_proveedor = '55431';
Query OK, 1 row affected (0.001 sec)
```

12. Hacemos un **SELECT** para ver los cambios en nuestra base de datos.

```
MariaDB [myISAM]> SELECT * FROM proveedor;
```

folio_proveedor	nombre	email	telefono
23456	De la Rosa	rosa@gmail.com	8866774431

```
1 row in set (0.001 sec)
```

5. Motor de Almacenamiento MRG_MyISAM:

Es un tipo especial de motor de almacenamiento en MySQL y MariaDB que permite combinar varias tablas MyISAM en una sola tabla lógica para operaciones de consulta y manipulación de datos

Ventajas:

- **Simplifica consultas:** Permite realizar consultas y operaciones en un conjunto de datos que está distribuido en varias tablas MyISAM como si fuera una sola tabla.
- **Optimiza el rendimiento:** Puede mejorar el rendimiento al evitar la necesidad de unir tablas en tiempo de ejecución, especialmente para consultas que implican tablas que comparten la misma estructura.

Desventajas:

- **Limitado a tablas My_ISAM:** MRG_MYISAM solo puede fusionar tablas MyISAM, lo que puede ser una limitación si necesitas combinar tablas de otros motores de almacenamiento.
- **Restricciones en la estructura de las tablas subyacentes:** Las tablas subyacentes deben tener la misma estructura de columnas para que MRG_MYISAM pueda combinarlas. Cualquier diferencia en la estructura de las columnas puede causar errores o resultados inesperados.

Demostración del motor de Almacenamiento MRG_MYISAM:

1. Crear la base de datos para este ejemplo.

```
MariaDB [(none)]> CREATE DATABASE mrg_my_isam;  
Query OK, 1 row affected (0.001 sec)
```

2. Creamos un usuario para la base de datos.

```
MariaDB [(none)]> CREATE USER 'mrg_my_isam'@'%' IDENTIFIED BY 'orlando';  
Query OK, 0 rows affected (0.003 sec)
```

3. Damos los permisos al usuario para poder acceder a la base de datos.

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON mrg_my_isam.* TO 'mrg_my_isam'@'%;  
Query OK, 0 rows affected (0.003 sec)
```

4. Guardamos cambios.

```
MariaDB [(none)]> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0.001 sec)
```

5. Accedemos otra vez con ese nuevo usuario y contraseña:

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u mrg_my_isam -p  
Enter password: █
```

Hemos accedido correctamente.

```
starlord@2806-105e-0019-5a71-bc29-5d5b-2eab-ed84:~$ mysql -u mrg_my_isam -p  
Enter password:  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 12  
Server version: 10.5.23-MariaDB MariaDB Server  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]>
```

Para poder ver el funcionamiento de este motor de almacenamiento es necesario crear tablas con diferentes nombres pero con la misma estructura de columnas y al final unirlos

6. Creamos la primera tabla

```
MariaDB [mrg_my_isam]> CREATE TABLE cliente1 (  
-> id_cliente INT AUTO_INCREMENT,  
-> nombre VARCHAR(50) NOT NULL,  
-> email VARCHAR(30) NOT NULL,  
-> telefono VARCHAR(10) NOT NULL,  
-> PRIMARY KEY (id_cliente)  
-> ) ENGINE=MyISAM;  
Query OK, 0 rows affected (0.005 sec)
```

7. Hacemos inserciones a la tabla **cliente1**

```
MariaDB [mrg_my_isam]> INSERT INTO cliente1 (nombre, email, telefono) VALUES
->      ('Juan', 'juan@example.com', '1234567890'),
->      ('María', 'maria@example.com', '9876543210');
Query OK, 2 rows affected (0.001 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

8. Realizamos en **SELECT** sobre la tabla para ver el registro.

```
MariaDB [mrg_my_isam]> SELECT * FROM cliente1;
+-----+-----+-----+-----+
| id_cliente | nombre | email           | telefono |
+-----+-----+-----+-----+
|          1 | Juan   | juan@example.com | 1234567890 |
|          2 | María  | maria@example.com | 9876543210 |
+-----+-----+-----+-----+
2 rows in set (0.001 sec)
```

6. Creamos la segunda tabla

```
MariaDB [mrg_my_isam]> CREATE TABLE cliente2 (
->      id_cliente INT AUTO_INCREMENT,
->      nombre VARCHAR(50) NOT NULL,
->      email VARCHAR(30) NOT NULL,
->      telefono VARCHAR(10) NOT NULL,
->      PRIMARY KEY (id_cliente)
-> ) ENGINE=MyISAM;
Query OK, 0 rows affected (0.013 sec)
```

7. Hacemos inserciones a la tabla **cliente2**

```
MariaDB [mrg_my_isam]> INSERT INTO cliente2 (nombre, email, telefono) VALUES
->      ('Pedro', 'pedro@example.com', '5555555555'),
->      ('Ana', 'ana@example.com', '4444444444');
Query OK, 2 rows affected (0.001 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

8. Realizamos en **SELECT** sobre la tabla para ver el registro.

```
MariaDB [mrg_my_isam]> SELECT * FROM cliente2;
+-----+-----+-----+-----+
| id_cliente | nombre | email           | telefono |
+-----+-----+-----+-----+
|          1 | Pedro  | pedro@example.com | 5555555555 |
|          2 | Ana    | ana@example.com   | 4444444444 |
+-----+-----+-----+-----+
2 rows in set (0.000 sec)
```

9. Vamos a crear una ultima tabla donde vamos a combinar las dos tablas.

```
MariaDB [mrg_my_isam]> CREATE TABLE tabla_combinada (
->     id INT NOT NULL AUTO_INCREMENT,
->     nombre VARCHAR(50),
->     email VARCHAR(30),
->     telefono VARCHAR(10),
->     PRIMARY KEY (id)
-> ) ENGINE=MRG_MYISAM UNION=(cliente1, cliente2);
Query OK, 0 rows affected (0.006 sec)
```

10. Por ultimo damos un **SELECT** para ver que nuestros registros se combinaron en una misma tabla.

```
MariaDB [mrg_my_isam]> SELECT * FROM tabla_combinada;
+-----+-----+-----+-----+
| id | nombre | email          | telefono |
+-----+-----+-----+-----+
| 1 | Juan   | juan@example.com | 1234567890 |
| 2 | María  | maria@example.com | 9876543210 |
| 1 | Pedro  | pedro@example.com | 5555555555 |
| 2 | Ana    | ana@example.com   | 4444444444 |
| 1 | Luis   | luis@example.com  | 1111111111 |
| 2 | Elena  | elena@example.com | 2222222222 |
+-----+-----+-----+-----+
6 rows in set (0.001 sec)
```

V. Conclusiones:

Al explorar los diversos gestores de bases de datos, desde MySQL y MariaDB hasta sus diferentes motores de almacenamiento como MyISAM, CSV y ARCHIVE, así como las funcionalidades especiales como MRG_MYISAM, se revela un vasto mundo de opciones para el diseño y gestión de bases de datos. Cada gestor y motor tiene sus propias fortalezas y debilidades, ofreciendo un abanico de posibilidades para adaptarse a las necesidades específicas de cada proyecto. Desde la flexibilidad y robustez de MySQL y MariaDB hasta la optimización de almacenamiento de datos en formatos como CSV y ARCHIVE, estas herramientas proporcionan a los desarrolladores y administradores de bases de datos las herramientas necesarias para construir sistemas eficientes y escalables. Al comprender las características únicas de cada gestor y motor, los profesionales de la informática pueden tomar decisiones informadas para optimizar el rendimiento y la fiabilidad de sus sistemas de bases de datos.