Backup Database MariaDB

1. Buka Terminal. Pastikan Anda memiliki akses ke terminal dan hak akses root atau pengguna dengan izin yang diperlukan.
2. Login ke MariaDB. Masuk ke MariaDB menggunakan perintah berikut dan masukkan password jika diminta: `sudo mysql -u root -p`
3. Pilih Database. Setelah masuk, Anda dapat melihat daftar database dengan perintah: `SHOW DATABASES;` Pilih database yang ingin Anda backup.
4. Backup Database dengan mysqldump
    a. Keluar dari MariaDB dengan perintah `exit` atau `quit`.
    b. Gunakan perintah `mysqldump` untuk membuat backup database. Gantilah your_database_name dengan nama database yang ingin di backup dan backup_file.sql dengan nama file backup yang diinginkan: `mysqldump -u root -p animal > backup_animal.sql`.
    c. Masukkan password root MariaDB jika diminta.
5. Verifikasi File Backup. Pastikan file backup (backup_file.sql) telah dibuat di direktori yang Anda tentukan.

Restore Database MariaDB

1. Buka Terminal. Pastikan Anda memiliki akses ke terminal dan hak akses root atau pengguna dengan izin yang diperlukan.
2. Login ke MariaDB. Masuk ke MariaDB menggunakan perintah berikut dan masukkan password jika diminta: `sudo mysql -u root -p`.
3. Buat Database Baru. Jika Anda ingin merestore database ke database baru, buat database baru dengan perintah berikut. Gantilah new_database_name dengan nama database yang diinginkan: `CREATE DATABASE animal;`. Jika Anda ingin merestore ke database yang sudah ada, Anda dapat melewati langkah ini.
4. Keluar dari MariaDB
5. Restore Database dengan mysql. Gunakan perintah mysql untuk merestore database dari file backup. Gantilah new_database_name dengan nama database yang ingin Anda restore dan backup_file.sql dengan nama file backup: `mysql -u root -p animal < backup_animal.sql`.
6. Verifikasi Restore. Masuk kembali ke MariaDB dan cek apakah database telah direstore dengan perintah: `USE animal; SHOW TABLES;`

DDL

```
CREATE TABLE animal (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    type VARCHAR(255) NOT NULL,
    created_at DATETIME,
    updated_at DATETIME
);
```

Create Stored Procedure

Create

```
DELIMITER //
CREATE PROCEDURE CreateSport(IN animal_type VARCHAR(255), IN animal_desc
VARCHAR(255))
BEGIN
INSERT INTO animal (name, type, created_at, updated_at)
VALUES (animal_type, animal_desc, NOW(), NOW());
END //

DELIMITER ;
CALL CreateSport('Mammals', 'Vivipar');
```

Read

```
DELIMITER //
CREATE PROCEDURE ReadSports()
BEGIN
SELECT * FROM animal;
END //

DELIMITER ;
CALL ReadSports();
```

Update

```
DELIMITER //
CREATE PROCEDURE UpdateSport(IN animal_id INT, IN animal_type
VARCHAR(255), IN animal_desc VARCHAR(255))
BEGIN
UPDATE animal
SET name = animal_type, type = animal_desc, updated_at = NOW()
WHERE id = animal_id;
END //

DELIMITER ;
CALL UpdateSport(1, 'Soccer', 'Outdoor');
```

Delete

```
DELIMITER //
CREATE PROCEDURE DeleteSport(IN animal_id INT)
BEGIN
DELETE FROM animal WHERE id = animal_id;
END //

DELIMITER ;
CALL DeleteSport(1);
```

Buat Tabel histories

```
CREATE TABLE histories (
    id INT AUTO_INCREMENT PRIMARY KEY,
    animal_id INT,
    action VARCHAR(50),
    old_name VARCHAR(255),
    new_name VARCHAR(255),
    old_type VARCHAR(255),
    new_type VARCHAR(255),
    changed_at DATETIME
);
```

Buat Trigger untuk INSERT :

```
DELIMITER //
CREATE TRIGGER after_animal_insert
AFTER INSERT ON animal
FOR EACH ROW
BEGIN
 INSERT INTO histories (animal_id, action, new_name, new_type, changed_at)
 VALUES (NEW.id, 'INSERT', NEW.name, NEW.type, NOW());
END //
DELIMITER ;
```

Buat Trigger untuk UPDATE

```
DELIMITER //
CREATE TRIGGER after_animal_update
AFTER UPDATE ON animal
FOR EACH ROW
BEGIN
 INSERT INTO histories (animal_id, action, old_name, new_name, old_type, new_type,
changed_at)
 VALUES (OLD.id, 'UPDATE', OLD.name, NEW.name, OLD.type, NEW.type, NOW());
END //
DELIMITER ;
```

Buat Trigger untuk DELETE

```
DELIMITER //
CREATE TRIGGER after_animal_delete
AFTER DELETE ON animal
FOR EACH ROW
BEGIN
 INSERT INTO histories (animal_id, action, old_name, old_type, changed_at)
 VALUES (OLD.id, 'DELETE', OLD.name, OLD.type, NOW());
END //
DELIMITER ;
Query dengan CTE:
WITH LatestHistory AS (
 SELECT h.animal_id, h.action, h.old_name, h.new_name, h.old_type, h.new_type,
h.changed_at,
 ROW_NUMBER() OVER (PARTITION BY h.animal_id ORDER BY
h.changed_at DESC) AS rn
 FROM histories h
)
SELECT s.id, s.name, s.type, s.created_at, s.updated_at,
 lh.action, lh.old_name, lh.new_name, lh.old_type, lh.new_type, lh.changed_at
FROM animal s
LEFT JOIN LatestHistory lh ON s.id = lh.animal_id AND lh.rn = 1
ORDER BY s.id;
```