Perfeito! 😎 Vamos organizar um **backup automático de banco de dados** de forma segura e prática. Vou focar em MySQL/MariaDB e Laravel, mas também darei opções gerais.

**1️⃣ Usando Laravel Artisan + agendamento (scheduler)**

Se você usa Laravel, pode criar um comando Artisan para fazer backup:

**Passo 1: Criar um comando**

php artisan make:command BackupDatabase

**Passo 2: Editar o comando (app/Console/Commands/BackupDatabase.php)**

namespace App\Console\Commands;

use Illuminate\Console\Command;

use Illuminate\Support\Facades\Storage;

class BackupDatabase extends Command

{

protected $signature = 'db:backup';

protected $description = 'Faz backup do banco de dados';

public function handle()

{

$dbHost = env('DB\_HOST');

$dbName = env('DB\_DATABASE');

$dbUser = env('DB\_USERNAME');

$dbPass = env('DB\_PASSWORD');

$backupPath = storage\_path('backups');

$fileName = 'backup\_' . date('Y-m-d\_H-i-s') . '.sql';

if (!file\_exists($backupPath)) {

mkdir($backupPath, 0755, true);

}

$command = "mysqldump -h {$dbHost} -u {$dbUser} -p{$dbPass} {$dbName} > {$backupPath}/{$fileName}";

$returnVar = null;

$output = null;

exec($command, $output, $returnVar);

if ($returnVar === 0) {

$this->info("Backup criado com sucesso: {$fileName}");

} else {

$this->error('Erro ao criar backup!');

}

}

}

**Passo 3: Agendar no Kernel.php**

No app/Console/Kernel.php:

protected function schedule(Schedule $schedule)

{

$schedule->command('db:backup')->dailyAt('02:00'); // todo dia às 2h

}

Assim o Laravel faz backup automaticamente todos os dias.

**2️⃣ Usando script direto no servidor (Linux/Mac)**

Crie um arquivo backup.sh:

#!/bin/bash

DB\_NAME="meubanco"

DB\_USER="usuario"

DB\_PASS="senha"

BACKUP\_DIR="/caminho/para/backups"

DATE=$(date +"%Y-%m-%d\_%H-%M-%S")

FILENAME="backup\_$DATE.sql"

mysqldump -u $DB\_USER -p$DB\_PASS $DB\_NAME > $BACKUP\_DIR/$FILENAME

Torne executável:

chmod +x backup.sh

Agende no cron para rodar diariamente:

crontab -e

E adicione:

0 2 \* \* \* /caminho/para/backup.sh

Roda todo dia às 2h da manhã.

**3️⃣ Usando Windows (batch + agendamento)**

Crie um arquivo backup.bat:

@echo off

set DB\_NAME=meubanco

set DB\_USER=usuario

set DB\_PASS=senha

set BACKUP\_DIR=C:\backups

set DATE=%DATE:~6,4%-%DATE:~3,2%-%DATE:~0,2%\_%TIME:~0,2%-%TIME:~3,2%

set FILENAME=backup\_%DATE%.sql

mysqldump -u %DB\_USER% -p%DB\_PASS% %DB\_NAME% > %BACKUP\_DIR%\%FILENAME%

Depois agende no **Agendador de Tarefas do Windows** para rodar diariamente.

**4️⃣ Boas práticas**

1. Armazenar backups **fora do servidor principal**, ex.: S3, Google Drive ou outro servidor.
2. Manter backups **compactados**: mysqldump ... | gzip > backup.sql.gz.
3. Apagar backups antigos automaticamente (ex.: manter 30 dias).
4. Testar a restauração regularmente:

mysql -u usuario -p meuBanco < backup.sql

Perfeito! 😎 Vou criar para você **uma solução completa de backup automático de banco de dados no Laravel**, pronta para rodar via scheduler e mantendo apenas os últimos 30 backups.

**1️⃣ Criar comando Artisan**

No terminal do seu projeto Laravel:

php artisan make:command BackupDatabase

Isso vai gerar app/Console/Commands/BackupDatabase.php.

**2️⃣ Código do comando**

Substitua o conteúdo do comando por:

<?php

namespace App\Console\Commands;

use Illuminate\Console\Command;

use Illuminate\Support\Facades\Storage;

class BackupDatabase extends Command

{

protected $signature = 'db:backup';

protected $description = 'Faz backup do banco de dados automaticamente';

public function handle()

{

$dbHost = env('DB\_HOST');

$dbName = env('DB\_DATABASE');

$dbUser = env('DB\_USERNAME');

$dbPass = env('DB\_PASSWORD');

$backupDir = storage\_path('app/backups'); // Local de armazenamento

if (!file\_exists($backupDir)) {

mkdir($backupDir, 0755, true);

}

$date = date('Y-m-d\_H-i-s');

$fileName = "backup\_{$date}.sql.gz";

$filePath = $backupDir . DIRECTORY\_SEPARATOR . $fileName;

// Comando mysqldump com compressão gzip

$command = "mysqldump -h {$dbHost} -u {$dbUser} -p{$dbPass} {$dbName} | gzip > {$filePath}";

$returnVar = null;

$output = null;

exec($command, $output, $returnVar);

if ($returnVar === 0) {

$this->info("Backup criado com sucesso: {$fileName}");

// Limitar backups: mantém apenas os 30 mais recentes

$files = collect(glob($backupDir . DIRECTORY\_SEPARATOR . '\*.sql.gz'))

->sortByDesc(function ($file) {

return filemtime($file);

})->values();

if ($files->count() > 30) {

$files->slice(30)->each(function ($file) {

@unlink($file);

});

}

} else {

$this->error('Erro ao criar backup!');

}

}

}

**3️⃣ Agendar o backup**

No app/Console/Kernel.php, adicione no método schedule():

protected function schedule(\Illuminate\Console\Scheduling\Schedule $schedule)

{

// Executa todo dia às 2h da manhã

$schedule->command('db:backup')->dailyAt('02:00');

}

O Laravel Scheduler precisa estar rodando via cron no servidor Linux:

\* \* \* \* \* cd /caminho/do/projeto && php artisan schedule:run >> /dev/null 2>&1

Isso verifica a cada minuto se há comandos agendados para executar.

**4️⃣ Observações importantes**

* O backup é compactado em .gz para economizar espaço.
* Mantemos **apenas os últimos 30 arquivos**, os mais antigos são deletados automaticamente.
* O diretório de backup é storage/app/backups. Pode ser alterado para S3 ou outro storage se quiser.
* Teste manualmente com:

php artisan db:backup

Se funcionar, o scheduler fará automaticamente todo dia.

Se você quiser, posso **adaptar esse comando para enviar os backups automaticamente para o Google Drive ou S3**, para ter uma cópia off-site segura.