

7 – Introdução

- Muitos desenvolvedores costumam não dar a devida importância ao backup, visto que ele não será usado de imediato, mas somente no caso de desastres (problemas de software ou hardware).
- Backup consiste em gerar cópias de segurança do banco de dados (estrutura e dados), com o objetivo de garantir que este não seja perdido no caso de problemas com o hardware (danificação do disco) ou com o próprio sistema operacional.
- Muitos desenvolvedores imaginam que o processo de backup consiste em, basicamente, apertar um botão ou emitir um comando, mas infelizmente, nem sempre é tão simples assim.
- No momento de se planejar uma rotina de backup, é preciso analisar a frequência com que deve ser gerado o arquivo de backup, o tamanho da base (visto que base de dados muito grandes exigem técnicas especiais de backup), em quais mídias serão salvos os dados do backup, importância dos dados, entre outros aspectos.

7 – Introdução

- O processo de restauração (restore) consiste em restaurar os dados do BD caso estes sejam perdidos por decorrência de desastres.
- Se tudo correr bem, o desenvolvedor nunca precisará restaurar um backup. Mas quando for necessário, não basta termos o melhor sistema gerador de backup do mundo, será necessário o uso de uma boa ferramenta de recuperação de dados, que seja capaz de carregar o arquivo de backup e restaurar os dados com total consistência.
- O fato do backup ser feito diariamente acaba naturalmente chamando mais a atenção dos desenvolvedores, visto que o processo de recuperação (restore) é usado com bem menos frequência.
- Os backups não são feitos sob pressão, mas a recuperação geralmente é feita em momentos complicados, onde existe o risco de perda dos dados.

7 - Introdução

- Um profissional pode planejar e desenvolver toda a rotina de backup do banco de dados. Entretanto, este profissional pode não estar presente no momento que uma recuperação precisa ser feita. Por isso é importante ter mais de um profissional habilitado a recuperar dados.
- O responsável por gerar backup do banco de dados deve sempre testar a consistência do mesmo, principalmente se o volume de dados for muito grande. **Backup só é backup após o teste de consistência.**
- Uma boa prática consiste em ter um banco de dados para teste, onde seja possível recuperar os arquivos de backup e testar sua consistência.

Observação: Backup e replicação não são sinônimos, sendo necessário a implementação de rotinas de backup mesmo para banco de dados replicados. Como exemplo, vamos imaginar que um usuário execute o comando DROP DATABASE, esta será replicado para os servidores escravos.

7.1 – Backup Online e Offline

- O ideal para a realização de um backup seguro é deixar o servidor de banco de dados offline, desta forma, nenhuma operação de escrita será feita, reduzindo o risco de inconsistências nos arquivos de backup.
- Entretanto, nem sempre será possível deixar o servidor offline para realização do backup, pois muitos bancos de dados possuem dados críticos, e geram grandes problemas quando ficam inacessíveis, mesmo que por pouco tempo (Ex: hospitais, usinas nucleares, planos de saúde, etc.)
- As ferramentas de backup atuais possuem mecanismos para fazer backup no banco de dados, mesmo que ele esteja totalmente online, mas o risco de problemas é bem maior, visto que o servidor ainda pode ter dados na memória e nos buffers do BD que ainda não foram gravados no disco.

7.1 – Backup Online e Offline

- Os SGBD's atuais trabalham com sistemas de bloqueio de tabelas no momento de geração de backup, para evitar que possíveis usuários façam modificações na tabela durante o processo.
- Em situações de banco de dados com grande carga de trabalho e um alto volume de dados, uma outra alternativa muito usada por desenvolvedores é a geração do backup a partir de um banco de dados escravo de replicação.
- Considerando que a geração de backup requer uma fatia considerável de recursos, podendo gerar lentidão no banco de dados, fazer esse processo em um escravo de replicação não irá comprometer o uso do BD pelos usuários.

7.2 – Backup Lógicos e Físico

- O backup lógico (também chamado de dump) consiste em gerar um arquivo (geralmente formato SQL) com todos os dados (estrutura e dados) presentes no banco de dados. Existem diversas ferramentas que fazem backup por meio de geração de arquivos.
- As vantagens do método lógico é que são arquivos padronizados, que podem ser visualizados e editados por meio de editores. São fáceis de restaurar, e nos possibilitam filtrar os dados que desejamos salvar (Ex: Salvar dados de somente uma tabela, ou somente dados de clientes de uma determinada cidade.
- Entretanto, existem desvantagens para o backup lógico, como alto custo de recursos durante o processo, risco do arquivo gerado estar corrompido, perda de precisão para dados com ponto flutuante, entre outros. Mas a grande desvantagem é o alto custo (memória e CPU).

7.2 – Backup Lógicos e Físico

- O backup físico consiste em copiar os dados fisicamente do diretório onde o banco de dados armazena seus arquivos. No linux (debian) o MySQL armazena sua pasta de dados no diretório **/var/lib/mysql/**. Já no Windows, o diretório de dados fica em **C:/Arquivos de programas/MySQL/MySQL Server <versão>/data**.
- Cada database criado no banco de dados irá possuir uma pasta com o mesmo nome no diretório data do MySQL, bastando copiar esta pasta para o backup ser feito. E caso necessário, para fazer o restore, basta colar a pasta no mesmo diretório. Este processo deve ser feito com o banco de dados parado, para evitar inconsistências.
- Entretanto, este método não costuma ser muito confiável, pois muitos problemas podem surgir no momento do restore, principalmente para tabelas com a engine **InnoDB**. Já para a engine **MyISAM** esse tipo de backup é mais eficiente.

7.3 – Backup Completo

- A estratégia mais simples de backup consiste em salvar os dados e a estrutura (tabelas) do banco de dados. Entretanto, para recuperar totalmente um servidor em produção, pode ser necessário um backup mais completo.
- Portanto, no momento de se montar a estratégia de backup, considerar os seguintes itens:

Categoria	Descrição
Dados não óbvios	Logs binários, logs de transação, bibliotecas externas, etc.
Códigos	Triggers, stored procedures, stored functions, cursores, views, etc.
Configuração de Replicação	Caso o banco de dados faça parte de uma rede replicada, deve-se salvar seus arquivos de configuração, para futura sincronia.
Configuração do servidor	Salvar as configurações específicas para o servidor, como número máximo de conexões, reserva de memória, etc. (Ex: my.cnf)

7.4 – Replicação

- Usar a replicação como estratégia de backup é considerada uma boa estratégia, visto que teremos diversas máquinas com os mesmos dados,.
- Desta forma, além de equilibrar a carga, caso um dos servidores apresente problemas, os demais poderão suprir a necessidade, assegurando que os dados não sejam perdidos.
- Outra vantagem do uso da replicação é a possibilidade de fazer o backup no servidor escravo, poupando o mestre da alta carga de recursos que este processo ocupa.
- Outra estratégia é deixar o escravo com uma diferença de sincronia com o mestre. No caso de problemas na base do mestre, o escravo não herdará o mesmo problema imediatamente, possibilitando recuperar os dados.
- Replicação não deve ser a única estratégia de backup, visto que todos os servidores replicados podem apresentar problemas simultaneamente (um raio queimar todos os servidores, por exemplo). Sempre é

7.5 – Backup de Arquivos Delimitados

- Como visto anteriormente, o backup lógico consiste na geração de um arquivo com toda a estrutura e dados do banco de dados. Esses arquivos são chamados de Dump.
- O MySQL possui diversas ferramentas que geram dumps, como o mysqldump, que é uma das mais utilizadas para geração de backup lógico.
- O dump contém toda a estrutura e todos os dados do banco em comandos SQL, geralmente gerados em um único arquivo. Estes comandos deverão ser executados no momento da restauração.
- Entretanto, existem algumas desvantagens no fato de gerar um backup em um arquivo monolítico (estes problemas não são específicos do mysqldump, mas das ferramentas de geração de backups SQL lógicos monolíticos em geral)

7.5 – Backup de Arquivos Delimitados

- Os principais problemas em se gerar dumps (com o banco em forma de estrutura SQL) como forma de backup são:

1 – Estrutura e dados armazenados juntos: Embora pareça prático ter o banco inteiro em um único arquivo, isso pode ser ruim caso o objetivo seja a restauração de apenas uma tabela.

2 – Grandes Expressões SQL: É muito custoso para o servidor de banco de dados ler e executar todos os comandos SQL de um arquivo dump.

3 – Apenas um único grande arquivo: A maioria dos editores de texto não pode editar arquivos muito extensos. Arquivos grandes podem se corromper.

4 – Backups Lógicos são caros: O backup lógico exige uma grande quantia de recursos do MySQL, comprometendo o uso do banco para os demais usuários.

7.5 – Backup de Arquivos Delimitados

- O exemplo abaixo mostra os comandos necessários para gerar um backup a partir de uma consulta SQL.
- Será feito uma consulta na tabela de clientes, e será gerado um arquivo com extensão “csv” contendo todos os clientes que possuem o campo “id_cliente” menor que 5.

```
SELECT * FROM cliente WHERE id_cliente<=4  
  
INTO OUTFILE '/tmp/cliente.csv'  
  
FIELDS TERMINATED BY ','  
  
ENCLOSED BY ''''  
  
LINES TERMINATED BY '\n';
```

7.5 – Backup de Arquivos Delimitados

- Abrindo o arquivo gerado pelo comando anterior (no caso do linux “cat /tmp/cliente.csv), temos o seguinte resultado:

```
paulo@paulo:~$ cat /tmp/cliente.csv;  
"1","Paulo Soares","Rua A, 48","1","4499577872","paulo@paulo.com.br"  
"2","Edmundo da Silva","Rua B, 23","5","4199545876","ed@edmundo.com"  
"3","Paulo Nunes","Rua P, 78","2","2399763456","paulo@nunes.com"
```

7.5 – Backup de Arquivos Delimitados

- O arquivo resultante de uma consulta delimitada é mais compacto e mais fácil de manipular em ferramentas de linha de comando do que os arquivos de dump.
- Mas a maior vantagem desta abordagem é a velocidade para fazer e restaurar os backups, além da portabilidade do arquivo entre diversos sistemas gerenciadores de banco de dados, visto que o format “CSV” é padrão internacional.
- O comando abaixo ilustra como é possível carregar os dados de um arquivo delimitado para o banco de dados MySQL:

```
LOAD DATA INFILE '/tmp/cliente.csv'  
INTO TABLE cliente  
FIELDS TERMINATED BY ','  
OPTIONALLY ENCLOSED BY ''''  
LINES TERMINATED BY '\n';
```

7.5 – Backup de Arquivos Delimitados

➤ Abaixo são apresentados os respectivos efeitos de cada uma das instruções utilizadas:

1. **INTO OUTFILE:** Especifica o caminho absoluto do arquivo onde o resultado da consulta será escrito.
2. **FIELDS TERMINATED BY:** Delimitador a ser utilizado entre os valores de cada coluna.
3. **ENCLOSED BY:** Caractere usado para delimitar o valor de um campo (geralmente aspas duplas).
4. **LINES TERMINATED BY:** Delimitador a ser utilizado entre cada linha do resultado.

7.5 – Backup de Arquivos Delimitados

➤ Entretanto, existem algumas desvantagens em se utilizar a técnica de consultas delimitadas, são elas:

1. É possível gerar o arquivo apenas das máquinas cujo o servidor MySQL está rodando, não sendo possível em máquinas clientes.
2. O MySQL deve ter permissão de escrita no diretório onde o arquivo será gerado (O MySQL e não o usuário que está executando o comando, pois quem escreve é o banco de dados)
3. Não é possível sobrescrever um arquivo existente (por questões de segurança.
4. É possível gerar arquivos csv apenas dos dados e não da estrutura do banco de dados (tabelas, restrições, etc.)

7.6 – Backup de Log Binário

- O backup consistente do banco de dados é de extrema importância para que possamos manter a integridade dos dados caso haja uma falha do sistema, hardware ou até mesmo para corrigir eventuais falhas de usuários, como por exemplo, a remoção acidental de um banco de dados.
- Para isto, é importante a adoção de uma política consistente de backup (diariamente), bem como conhecer as possíveis técnicas para fazê-lo.
- Ao realizar o procedimento de backup cria-se uma imagem dos seus dados no momento da execução da rotina de backup. Quando houver problemas no banco de dados que necessite do backup, basta utilizar o último backup feito. Entretanto, os dados alterados e inseridos entre o último backup e a falha, serão perdidos,
- Para resolver este problema, o MySQL permite que se configure logs binários, que armazenam todas as alterações (inserções, deleções e alterações) que são feitas no banco de dados.

7.6 – Backup de Log Binário

- Tendo os logs binários, quando o servidor apresentar problemas, basta voltar os dados do último backup e carregar os logs, que os dados serão recuperados.
- O quadro abaixo mostra as configurações para habilitar o log binário:

1 – crie um diretório para o log binário: Neste caso foi **“/home/paulo/mysql/logs”**

2 – Pare o serviço do MySQL: **/etc/init.d/mysql stop**

3 – Abra o arquivo de configuração: **mcedit /etc/mysql/my.cnf**

4 – No arquivo de configuração, insira as seguintes linhas:

#ativa o log-binário

log-bin = /home/paulo/mysql/logs/mysql-bin.log

#log binario ativado somente para o banco comercio

binlog-do-db=comercio

#identificador do servidor de MySQL.

server-id=1

7.6 – Backup de Log Binário

5 – É preciso dar permissão de escrita no diretório de log:

```
chown -R mysql:mysql /home/paulo/mysql  
chmod -R 775 /home/paulo/mysql
```

6 – Reinicie o serviço: **/etc/init.d/mysql start**

7 – A partir deste momento, todas as alterações no banco de dados “comercio” serão gravadas no log binário.

7.6 – Backup de Log Binário

- Deve ser feito cópias da pasta com os arquivos de log binário frequentemente, ou então deixá-las em um outro servidor, ou outra partição do disco.
- Caso seja necessário se efetuar a recuperação de um banco de dados, basta seguir os seguintes passos:

1 – Volte o último backup feito

2 – refaça as configurações do log binário

3 – Para voltar os dados perdidos de o seguinte comando:

mysqlbinlog <local onde os logs foram salvos> | **mysql -u** <usuário> **-p**<senha>

mysqlbinlog /home/paulo/Documentos/mysql/mysql-bin.000001 | **mysql -u root -p**

7.6 – Backup de Log Binário

- Caso o volume de transações no banco de dados seja muito grande, a quantidade de logs aumentará rapidamente.
- É possível excluir os logs mais antigos manualmente, mas não é aconselhado o uso desta técnica, pois ela irá dessincronizar o arquivo de índice do log (mysql-bin.index).
- Não faz sentido guardar logs que foram gerados antes do último backup, por isso é possível configurar o número de dias que o log se manterá no servidor.
- Para configurar, basta inserir no arquivo my.cnf do MySQL a variável **expire_logs_days** com o número de dias que o log deve ser guardado.

```
expire_logs_days = 10
```

7.6 – Backup de Log Binário

- O comando SHOW MASTER LOGS mostra todos os logs existentes no banco de dados com seus respectivos tamanhos, como mostra o exemplo abaixo:

SHOW MASTER LOGS

Log_name	File_size
mysql-bin.000001	126
mysql-bin.000002	2180
mysql-bin.000003	2318

7.7 – Mysqldump

- O Mysqldump é uma ferramenta para geração de backups lógicos, disponibilizada pelo MySQLAB, que acompanha o MySQL em sua instalação padrão.
- Muitas ferramentas de backup terceirizadas, que possuem interface gráfica e funções automatizadas de backup, usam em sua estrutura interna de funcionamento, o próprio Mysqldump.
- É possível utilizar o Mysqldump para gerar um arquivo de script do tipo SQL, que possui toda a estrutura do banco de dados com os seus respectivos dados, ou então parametrizar o backup, para que salve somente os dados, ou somente a estrutura do banco de dados.

```
mysqldump -u <usuario> -p<senha> <database> <diretorio>
```

7.7.1 – Mysqldump para um Database

- O código abaixo irá gerar um backup completo (estrutura do banco de dados + dados) de um database chamado “comercial” no endereço “/home/paulo/dumps/teste.sql”.
- Deve-se cuidar com a geração de backup em arquivos existentes, pois este será totalmente substituído pelo novo backup, perdendo os dados antecessores.
- Caso o diretório apontado no comando não tenha permissão de escrita, pode ser necessário criar o arquivo que será gerado o backup (no exemplo o “teste.sql”), para o servidor de banco de dados escreva os dados no arquivo.

```
mysqldump -u root -p comercial > /home/paulo/dumps/backup.sql;
```


7.7.2 – Mysqldump Múltiplos Databases

- Para gerar backup a partir da ferramenta Mysqldump de dois ou mais databases, deve-se utilizar o parâmetro “--databases” seguido do nome dos databases desejados.
- O código abaixo irá gerar um backup completo (estrutura do banco de dados + dados) de dois databases (rh e comercial):

```
mysqldump -u root -p --databases rh comercial > /home/paulo/dumps/backup.sql;
```

7.7.3 – Mysqldump de Tabelas Específicas

- Para gerar backup de apenas uma tabela de um database, deve-se informar o database seguido pelas tabelas que se deseja exportar.
- No exemplo abaixo, são exportadas as tabelas “cliente” e “produto” do database “comercial”.

```
mysqldump -u root -p comercial cliente produto > /home/paulo/dumps/backup.sql;
```

7.7.4 – Mysqldump de Estrutura

- O Mysqldump permite que seja feito um backup somente da estrutura do banco de dados, deixando de exportar os dados inseridos nas tabelas. Para isso,, utiliza-se o parâmetro “-d” no comando.
- O primeiro exemplo mostra o código para exportar apenas os comandos de criação das tabelas de cliente e produto do database comercial. O segundo exemplo mostra o código para geração de toda a estrutura do database comercial, sem os dados. O terceiro exemplo gera o código para geração de todos os databases, sem os dados:

```
mysqldump -u root -p -d comercial cliente produto > /home/paulo/dumps/backup.sql;
```

```
mysqldump -u root -p -d comercial > /home/paulo/dumps/backup.sql;
```

```
mysqldump -u root -p -d --all-databases > /home/paulo/dumps/backup.sql;
```

7.7.5 – Mysqldump Exclusivamente dos Dados

- O Mysqldump permite que seja feito um backup somente dos dados, sem exportar a estrutura de criação do banco de dados. Para isso, utiliza-se o parâmetro “**--no-create-info**” no comando.
- O primeiro exemplo mostra o código para exportar os dados apenas das tabelas de cliente e produto do database comercial. O segundo exemplo mostra os de todo o database comercial. O terceiro exemplo gera o backup dos dados de todos os databases:

```
mysqldump -u root -p --no-create-info comercial cliente produto >  
/home/paulo/dumps/teste.sql;
```

```
mysqldump -u root -p --no-create-info comercial > /home/paulo/dumps/teste.sql;
```

```
mysqldump -u root -p --no-create-info --all-databases > /home/paulo/dumps/teste.sql;
```

7.7.6 – Mysqldump – Mais Parâmetros

➤ A ferramenta Mysqldump possui uma série de parâmetros que permitem customizar o backup, deixando de acordo com as necessidades do desenvolvedor. Os principais parâmetros são:

1. **--complete-insert:** Por padrão, os comandos insert do backup são resumidos, mostrando apenas os valores e não os campos. Com esse parâmetro o comando insert fica com a sintaxe completa.
2. **--delayed-insert:** Utiliza o comando DELAYED para construir os comandos de insert.
3. **--disable-keys:** Faz com que os índices das tabelas sejam atualizados somente no final do processo e não a cada inserção, economizando recursos.
4. **--lock-all-tables:** Bloqueia as tabelas das bases que serão exportadas para garantir consistência.

7.7.7 – Recuperação de Dados

- Durante o processo de recuperação, seria muito recomendado deixar o servidor de banco de dados sem conexão, para que nenhum usuário possa acessar ou modificar informações durante o processo.
- Uma alternativa muito interessante é inserir no arquivo de configuração do MySQL o parâmetro “skip-networking”, para garantir que ele esteja indisponível durante o processo de recuperação.
- Por mais que as ferramentas atuais consigam, por meio de um moderno mecanismo de bloqueio, garantir a consistência dos dados durante o processo de recuperação, é sempre uma garantia deixar o banco de dados inacessível.