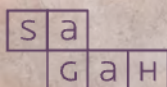


# ADMINISTRAÇÃO DE BANCO DE DADOS

Maurício de Oliveira Saraiva



SOLUÇÕES  
EDUCACIONAIS  
INTEGRADAS



# Backup de banco de dados

## Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Conceituar *backup* (ou cópias de segurança em um banco de dados).
- Caracterizar os *backups*.
- Aplicar *backups* via *script* (*shell* e linhas de comandos).

## Introdução

A evolução da tecnologia da informação e comunicação tem tornado as empresas cada vez mais competitivas no mercado de trabalho. No cenário atual, poucas empresas não possuem sistemas automatizados que armazenem suas informações em bancos de dados. Contudo, essa modernização produz dependências. Assim, ter bancos de dados indisponíveis pode ser catastrófico para muitas organizações. Para evitar a perda ou a indisponibilidade de dados, as rotinas de *backup* surgiram como uma boa solução, pois permitem que cópias de segurança sejam realizadas para garantir que nenhuma ou pouca informação seja realmente perdida.

Neste capítulo, você vai estudar os conceitos de cópia de segurança e *backup* em banco de dados. Além disso, vai ver a aplicação desses conceitos por linhas de comando no MySQL.

## Cópia de segurança em banco de dados

Com o advento dos sistemas computacionais, as informações passaram a ser armazenadas em meios digitais. A ideia por trás disso é registrá-las e acessá-las de modo mais rápido e fácil. Assim, é possível, por exemplo, cadastrar clientes e fornecedores, registrar dados sobre pedidos, consultar históricos de vendas, etc.

Como você sabe, existem diversas formas de armazenar informações em meio digital, como planilhas, arquivos em formato de texto e, principalmente, bancos de dados. Por serem mais seguros e eficientes, os bancos de dados

normalmente são os mais utilizados, especialmente por empresas que gerenciam grandes volumes de dados. No entanto, para muitas organizações, a informação é o seu maior valor, uma vez que pode concentrar dados importantes e sigilosos sobre o seu próprio negócio ou sobre o negócio de outras instituições. Assim, ter as informações disponíveis quando necessário pode representar a diferença entre o sucesso e o fracasso de suas operações.

Por isso, mais do que permitir o armazenamento e o acesso a informações, os bancos de dados precisam ser protegidos contra eventos adversos que podem corromper ou tornar as informações indisponíveis. Entre os principais problemas que podem ocorrer com bancos de dados, destacam-se desastres e sinistros, defeitos em equipamentos, falhas em mídias de armazenamento, dados inconsistentes lançados por usuários e até instruções executadas acidentalmente por analistas e desenvolvedores. O Quadro 1 mostra alguns dos principais eventos e ameaças que podem ocorrer em bancos de dados.

**Quadro 1.** Principais problemas em bancos de dados

<b>Evento/ameaça</b>	<b>Descrição</b>
Desastres/sinistros	Catástrofes ou fatalidades podem ocorrer nos prédios em que os equipamentos estão armazenados, por exemplo, terremotos, incêndios e alagamentos.
Defeitos em equipamentos e dispositivos	Equipamentos como servidores de bancos de dados podem ficar indisponíveis devido a algum problema técnico, como indisponibilidade de rede, problemas na placa-mãe, memória, etc., impedindo o acesso às bases de dados.
Falhas em mídias de armazenamento	Discos que armazenam os bancos de dados podem sofrer falhas físicas e corromper os dados, causando a perda de informações.
Dados inconsistentes	Usuários podem lançar dados indevidos no sistema, como ao registrar informações incorretas, importar registros em duplicidade, excluir dados erroneamente, etc.
Instruções acidentais	Analistas e desenvolvedores podem excluir tabelas ou registros acidentalmente por meio de instruções que são executadas diretamente nas bases de dados.

**Fonte:** Adaptado de Loney e Bryla (2005).

Como você viu, há diversas situações que podem excluir, corromper ou tornar os bancos de dados indisponíveis. Para mitigar essas ameaças, a melhor alternativa é realizar uma cópia de segurança dos bancos de dados, ou seja, um *backup*. De acordo com Faria (2017, p. 1), o *backup* consiste em “[...] gerar dados redundantes com o propósito específico de recuperação no caso de perda dos originais”. Em outras palavras, fazer um *backup* de um banco de dados significa fazer uma cópia de segurança dele para que seja possível restaurá-lo futuramente.

A cópia de segurança deve ser realizada, preferencialmente, em outro local onde o banco de dados esteja armazenado. Ela pode envolver diversos aspectos, como o método de realização e o tipo de *backup* a ser executado. Esses aspectos fazem parte da estratégia de *backup* a ser implantada.



### Fique atento

O principal objetivo de fazer uma cópia de segurança de um banco de dados é justamente possibilitar a sua recuperação quando necessário. A partir dessa premissa, é fundamental elaborar uma estratégia de *backup* que esteja de acordo com os interesses da organização.

A estratégia de realizar cópias de segurança de um banco de dados MySQL, por exemplo, pode levar em consideração diversos fatores. Entre eles, considere os destacados a seguir.

- O tipo de *backup* que será executado:
  - *backups on-line e off-line*;
  - *backups* físico e lógico;
  - *backups* completo e incremental;
  - *backups* local e remoto.
- O método de *backup* que será implantado:
  - *backup* dinâmico;
  - *backup table files*;
  - *backup* de arquivos de texto delimitados;
  - *backup* com replicação em *slave*;
  - *backup file system*.

- Os agendamentos realizados:
  - os dias e os horários em que os *backups* serão realizados;
  - os locais/mídias em que os *backups* serão armazenados.
- As responsabilidades na equipe:
  - como os *backups* serão executados (rotinas automáticas ou manuais) e quem os executará;
  - quem serão os responsáveis pelo monitoramento da execução dos *backups*;
  - quem está autorizado a solicitar a restauração de um banco de dados;
  - quem pode executar a restauração de um banco de dados.

## **Backup aplicado a banco de dados**

Realizar *backups* de bancos de dados é uma atividade muito importante que requer a definição de uma boa estratégia. Assim, os dados podem ser copiados e recuperados com integridade e segurança. Essa estratégia de *backup* envolve uma variedade de características que podem ser definidas de acordo com os objetivos de cada organização. A seguir, você vai ver alguns dos principais métodos de *backup* com base nas especificações do banco de dados MySQL (ORACLE, c2019a).

### **Backups on-line e off-line**

Um dos principais objetivos de um banco de dados é estar disponível para receber ou disponibilizar informações. Quando um banco de dados está rodado, diz-se que ele está *on-line*; quando ele está parado, diz-se que ele está *off-line*. Dessa forma, um *backup on-line* é caracterizado pela execução de uma cópia de segurança quando o banco de dados está rodando — *backup* quente. De outro modo, entende-se que o *backup* é *off-line* quando o banco de dados está parado — *backup* frio.

O *backup on-line* permite que os usuários continuem acessando o banco de dados durante a realização da cópia de segurança. Nesse tipo de *backup*, é possível ocorrer modificação de dados durante a execução, o que pode comprometer a integridade do *backup*. Nesse caso, pode ser necessário realizar bloqueios de gravação, permitindo apenas a leitura de dados durante a execução do *backup* — *backup* consistente.

Se os bloqueios não forem executados durante a realização de um *backup* quente, o *backup* poderá conter dados inconsistentes, pois armazenará infor-



mações desatualizadas. Nesse caso, os arquivos de *logs* de transações poderão ser usados no caso de uma restauração de banco de dados.

Já o *backup off-line* é realizado com o banco de dados parado, impedindo qualquer acesso aos dados durante a realização. Essa opção deve levar em consideração a execução de *backups* em horários em que os usuários não precisam acessar o banco de dados. No entanto, pode ser uma boa opção quando os dados já são replicados em um servidor *slave* e o *backup* frio é realizado nesse servidor, pois assim as informações ficam disponíveis por meio do servidor principal.

## Backups físico e lógico

O *backup* físico ou bruto é caracterizado pela cópia dos arquivos e pastas do banco de dados a partir de sua localização física no servidor onde ele está armazenado. Esse tipo de *backup* é adequado para grandes bancos de dados ou para os casos em que um banco de dados precisa ser restaurado rapidamente.

O nível de compactação dos dados copiados e a velocidade de execução e/ou restauração do *backup* físico são bastante superiores aos do *backup* lógico. Além disso, o *backup* bruto pode conter também arquivos de *logs* binários — *logs* de transações e de configuração de banco de dados. Já um *backup* lógico salva a estrutura do banco de dados e as informações nele armazenadas por meio de instruções SQL gravadas em arquivos. Esse modo de *backup* é normalmente indicado para pequenas e médias bases de dados, pois a gravação da cópia de segurança e a sua recuperação podem ser mais demoradas.

Por meio de instruções específicas, o *backup* lógico permite que apenas determinados bancos de dados ou partes de algum banco de dados sejam copiadas de maneira independente, podendo selecionar determinadas tabelas individualmente. No entanto, o *backup* lógico não permite realizar a cópia de arquivos de *logs* de transações e de configuração.



### Saiba mais

Tabelas *Memory* do MySQL não são armazenadas fisicamente no banco de dados. Portanto, normalmente não são copiadas pelo *backup* físico, pois essas tabelas são armazenadas na memória RAM do servidor. Existem ferramentas do MySQL que permitem a sua cópia em *backups* físicos.

## Backups completo e incremental

O *backup* completo (*full*) é responsável por realizar uma cópia total do banco de dados de uma organização. Executar o *backup full* pode parecer uma boa solução. No entanto, produzir diariamente essas cópias de segurança pode requerer muito espaço em disco, bem como levar bastante tempo. Por isso, pode não ser possível executar *backups* completos diariamente.

Os *backups* incrementais permitem copiar apenas os dados modificados desde a realização do último *backup* completo executado. Para isso, utilizam-se os *logs* de transações dos bancos de dados para identificar quais modificações ocorreram, como registros incluídos, alterados e excluídos, estruturas de tabelas editadas, etc.

Devido ao pequeno tamanho dos arquivos de *logs* de transações em relação ao *backup* completo, é possível fazer *backup* diariamente desses arquivos, criando uma espécie de cadeia de *backups* incrementais que, se adicionados ao último *backup full*, restauram um banco de dados ao estado atual.

Os *backups* completo e incremental são uma boa alternativa de cópia de segurança de banco de dados, pois unem os benefícios dos *backups* físico e lógico em uma única solução. Essa é uma das alternativas de *backup* mais utilizadas pelas grandes organizações.



### Fique atento

Uma orientação importante é manter o banco de dados e os *backups* em dispositivos separados. Caso isso não aconteça, se o dispositivo que contém o banco de dados falhar, os *backups* vão ficar indisponíveis (RAY *et al.*, 2018).

## Arquivos de log de transações — logs binários do MySQL

Os *logs* de transações são um conjunto de arquivos individuais numerados que contêm as informações sobre as alterações que foram realizadas — eventos de bancos de dados. Esses *logs* são importantes para os *backups* porque possuem as modificações que foram realizadas no banco de dados desde o último *backup* completo executado.

No banco de dados MySQL, os arquivos de *log* de transações podem ser armazenados em três formatos, que podem ser definidos pelo administrador do banco de dados (*DataBase Administrator* [DBA]) por meio de instruções específicas. A opção padrão do banco de dados depende da versão do MySQL que estiver rodando no servidor (ORACLE, c2019b). Veja a seguir.

- *Log* baseado em instrução: registra no arquivo de *log* as instruções SQL que foram executadas.
- *Log* baseado em linha: indica qual linha da tabela foi modificada no arquivo de *log*.
- *Log* de base mista: utiliza por padrão o *log* baseado em instrução, porém se modifica automaticamente para o modo baseado em linha em algumas operações.

## **Backup com replicação em *slave***

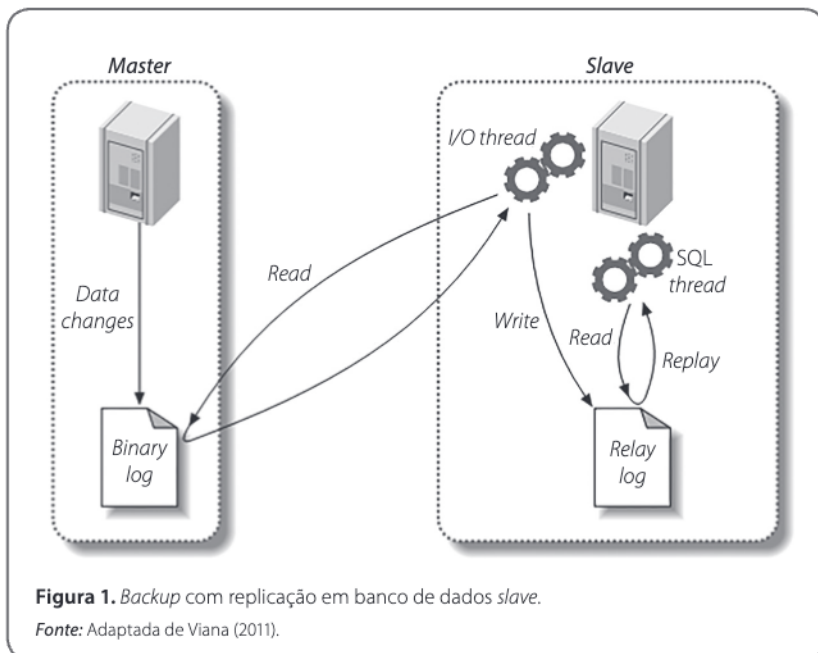
Manter a disponibilidade de seus bancos de dados pode ser um requisito necessário em determinadas organizações. Uma forma de garantir a disponibilidade é manter um servidor de banco de dados *slave* que contém os dados do banco de dados principal.

O banco de dados *slave* normalmente fica armazenado em outro servidor. Conforme a política de replicação implementada pelo DBA, ele receberá os arquivos de *logs* de transações para atualizar todas as modificações que foram realizadas no banco de dados principal.

No caso de uma pane no servidor principal do banco de dados, o banco de dados *slave* assume a continuidade dos serviços com os dados atualizados desde a última replicação dos *logs* de transações. Além disso, o banco de dados *slave* permite realizar *backups on-line* sem interromper as operações de gravação dos usuários. Nesse caso, ele suspenderá apenas as operações de atualização dos *logs*, pois os usuários continuarão a usar o servidor principal.

A Figura 1 mostra um modelo de replicação com servidor *slave* a partir de *logs* de transações com base nos arquivos binários de *log* do MySQL.





Os arquivos de *log* de transações do servidor principal são copiados para o servidor do banco de dados *slave* em intervalos predeterminados. O servidor *slave* atualiza o seu banco de dados e registra no servidor principal os *logs* que já foram atualizados.

## **Backup em banco de dados por linhas de comando**

Para que você veja como funciona a execução de *scripts* de *backup*, será utilizado aqui o banco de dados MySQL. Esse banco de dados pode ser baixado gratuitamente.



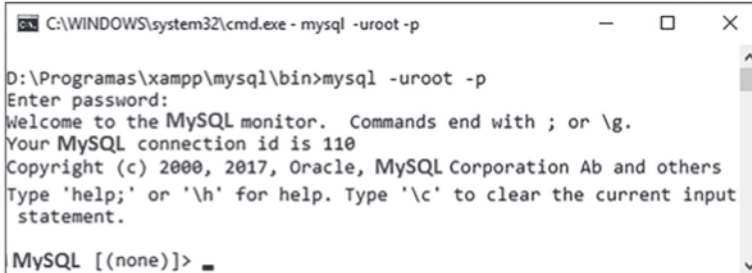
## Link

Acesse o *link* a seguir para fazer o *download* do MySQL. Clique na guia *Downloads* e, em seguida, na opção **MySQL Community Edition**. O *download* pode ser efetuado por meio da opção **MySQL Community Server**.

<https://qrgo.page.link/1dSq>

Após efetuar o *download* e realizar a instalação, acesse a subpasta *bin* em que o MySQL foi instalado por meio de uma janela de *prompt* de comando, como mostra a Figura 2. Digite a seguinte instrução para acessar o banco de dados:

```
mysql -uroot -p
```



```
C:\WINDOWS\system32\cmd.exe - mysql -uroot -p

D:\Programas\xampp\mysql\bin>mysql -uroot -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 110
Copyright (c) 2000, 2017, Oracle, MySQL Corporation Ab and others
Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.

MySQL [(none)]> _
```

Figura 2. Login no MySQL.

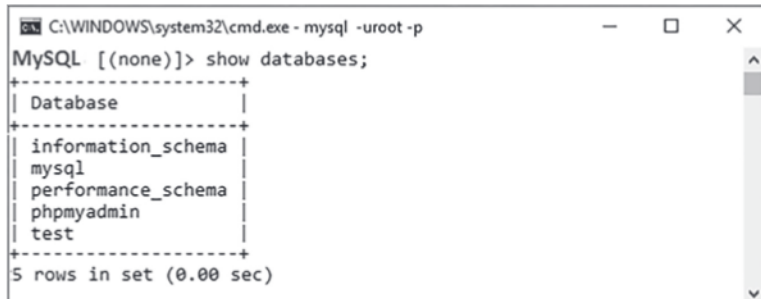
Após confirmar a senha definida na instalação, você vai ver a mensagem “Welcome to the MySQL monitor”.



### Fique atento

Algumas ferramentas integradas de desenvolvimento de *software*, como o XAMPP, distribuem o banco de dados MariaDB em vez do MySQL. No entanto, esses bancos de dados são compatíveis e compartilham as mesmas instruções.

A instrução `show databases` permite verificar quais bancos de dados estão instalados no servidor MySQL. Coloque ponto e vírgula no final para confirmar o encerramento do comando, como mostra a Figura 3.



```
C:\WINDOWS\system32\cmd.exe - mysql -uroot -p
MySQL [(none)]> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql       |
| performance_schema |
| phpmyadmin  |
| test        |
+-----+
5 rows in set (0.00 sec)
```

Figura 3. Listagem dos bancos de dados do servidor MySQL.

Entre com a instrução `exit;` para sair do MySQL e retornar ao *prompt* de comando, como mostra a Figura 4.



```
C:\WINDOWS\system32\cmd.exe
MySQL [(none)]> exit;
Bye
D:\Programas\xampp\mysql\bin>
```

Figura 4. Saída do MySQL e retorno ao *prompt* de comando.

## Backup lógico por mysqldump

Se você estiver no *prompt* de comando, pode realizar *backups* por meio de linhas de comando ou *scripts*. A execução de um *backup* lógico pode ser realizada com o auxílio do arquivo **mysqldump.exe**, localizado na pasta *bin* da instalação do MySQL.

As seguintes instruções apresentam algumas opções de *backup*. Porém, uma relação completa de opções pode ser obtida em MySQL (ORACLE, c2019c).

- Determinado banco de dados: `mysqldump [database] > database.sql`
- Determinados bancos de dados: `mysqldump [db1] [db2] > databases.sql`
- Todos os bancos de dados do servidor, incluindo *triggers*, *stored procedures* e todos os registros: `mysqldump -all-databases > all_databases_completo.sql`
- Determinada tabela de um banco de dados: `mysqldump [database] [table] > table_database.sql`
- Apenas a estrutura de um banco de dados: `mysqldump [database] --no-data > database_estrutura.sql`
- *Backup* consistente com bloqueio de tabelas: `mysqldump [database] --lock-all-tables > database_consistente.sql`

Veja um exemplo de realização de *backup* lógico com bloqueio de tabelas do banco de dados *test*, incluindo as opções de cópia de *triggers* e *stored procedures* do MySQL:

```
mysqldump test -R -E -lock-all-tables > test.sql
```

Nenhuma mensagem é exibida na tela, o que significa que o *backup* foi realizado com sucesso. Como nenhuma pasta foi informada, o *backup* gerou o arquivo **test.sql** na mesma pasta em que o arquivo foi executado.



### Fique atento

É possível passar usuário e senha para o **mysqldump** realizar o *backup* em apenas uma linha de comando. Isso é útil quando o *backup* é realizado por *scripts*. No entanto, você precisa se certificar de que o usuário indicado possui privilégios suficientes no banco de dados.

## Backup completo e incremental

Existem ferramentas específicas do MySQL, como o MySQL Enterprise Backup, que auxiliam na execução de *backups* para diversas plataformas. No entanto, de forma mais simplificada, o *backup* completo pode ser realizado pela cópia e/ou compactação dos arquivos de banco de dados da pasta **mysql/data/nome\_banco\_de\_dados**. Observe a Figura 5.

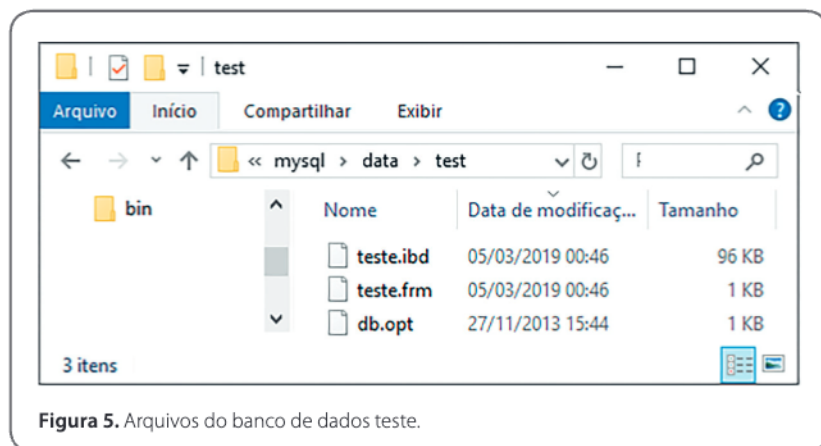


Figura 5. Arquivos do banco de dados teste.

Os seguintes comandos compactam e copiam o banco de dados para outro local, que pode ser uma unidade de fita ou outro servidor. Após, o arquivo compactado é excluído. Para copiar os arquivos, é preciso parar o serviço MySQL. Caso contrário, haverá erro de violação de compartilhamento para copiar os arquivos das tabelas do banco de dados. No exemplo a seguir, a cópia é realizada após a compactação por meio do utilitário **7z.exe**.



```
7z a -tzip backup_full_test.zip *.*
copy backup_full_test.zip \\servidor\compartilhamento
del backup_full_test.zip
```

Para utilizar o método de *backup* incremental do MySQL, é preciso habilitar o *log* de transações de arquivos binários. Para habilitar o *log*, acesse o arquivo **bin/mysql.ini** e retire o caractere # do início da linha **log-bin=mysql-bin**, como mostra a Figura 6.

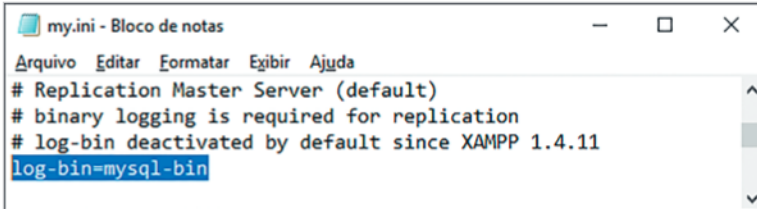


Figura 6. Habilitação dos arquivos de *logs* binários do MySQL.

Reinicie o banco de dados para que a alteração tenha efeito. A partir desse ponto, todas as operações realizadas no banco de dados serão gravadas em arquivos de *log*, numerados sequencialmente. Esses arquivos podem ser visualizados na pasta **mysql/data**, ou o arquivo de *log* atual pode ser identificado por meio da instrução **SHOW MASTER STATUS;** (Figura 7).

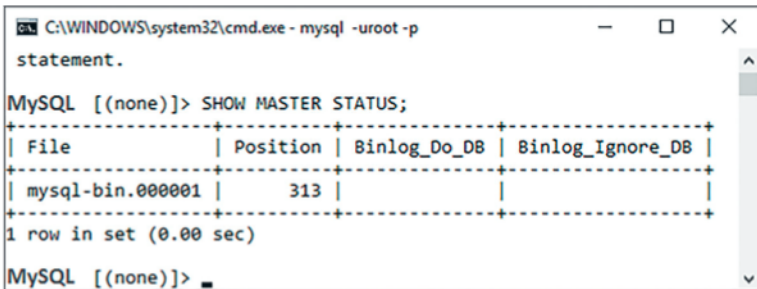


Figura 7. Arquivo de *log* binário atual do MySQL.

Assim que o serviço do MySQL for parado, o arquivo de *log* binário será fechado. Um novo arquivo será aberto quando o MySQL for iniciado novamente. A partir disso, o *backup* dos arquivos de *log* binário pode ser realizado da mesma maneira que são realizados os *backups* completos. A única diferença está na localização e no nome dos arquivos.



## Referências

FARIA, H. M. *Bacula: ferramenta livre de backup*. 3. ed. Rio de Janeiro: Brasport, 2017.

LONEY, K.; BRYLA, B. *Oracle 10g: o manual do DBA*. Rio de Janeiro: Elsevier, 2005.

ORACLE. *Database backup methods*. c2019a. Disponível em: <https://dev.mysql.com/doc/refman/5.6/en/backup-methods.html>. Acesso em: 9 abr. 2019.

ORACLE. *Mysqldump: a database backup program*. c2019c. Disponível em: <https://dev.mysql.com/doc/refman/8.0/en/mysqldump.html>. Acesso em: 9 abr. 2019.

ORACLE. *The binary log*. c2019b. Disponível em: <https://dev.mysql.com/doc/refman/5.6/en/binary-log.html>. Acesso em: 9 abr. 2019.

RAY, M. et al. *Fazer backup e restaurar bancos de dados do SQL Server*. 2018. Disponível em: <https://docs.microsoft.com/pt-br/sql/relational-databases/backup-restore/backup-and-restore-of-sql-server-databases?view=sql-server-2017>. Acesso em: 9 abr. 2019.

VIANA, A. L. S. *MySQL: replicação de dados*. 2011. Disponível em: <https://www.devmedia.com.br/mysql-replicacao-de-dados/22923>. Acesso em: 9 abr. 2019.

## Leitura recomendada

ORACLE. *MySQL*. c2019. Disponível em: <https://www.mysql.com/>. Acesso em: 9 abr. 2019.

Encerra aqui o trecho do livro disponibilizado para esta Unidade de Aprendizagem. Na Biblioteca Virtual da Instituição, você encontra a obra na íntegra.

Conteúdo:

