UniFil
EDUCAÇÃO A DISTÂNCIA

Unidade 3

Banco de Dados Distribuídos, conceitos e aplicação

Abertura

Olá, amigo(a) discente! Seja bem-vindo(a)!

Com a existência de um crescimento na demanda por organizar os dados nas grandes empresas, se fez necessário estruturar de forma diferente os bancos de dados. Na aula 01 vamos conhecer como surgiu e alguns conceitos sobre banco de dados distribuídos, como eles auxiliam na organização e acesso dos dados em grandes organizações.

Na aula 02 vamos entrar em detalhes sobre o funcionamento de bancos de dados distribuídos e sua aplicação na prática, os impactos que eles geram nas organizações e assim conseguir identificar quando é necessário possuir uma base de dados distribuída ou não.

Objetivos

- Conhecer os conceitos de banco de dados distribuídos;
- Conhecer como aplicar na prática os conceitos de banco de dados distribuídos.

Conteúdo Programático

Aula 01 – Banco de Dados Distribuídos seus conceitos

Aula 02 – Banco de Dados Distribuídos sua necessidade e aplicação

Banco de Dados Distribuídos seus conceitos

Olá, estudante, como você está? Vamos começar agora um estudo muito interessante sobre banco de dados. Vamos conhecer como funciona o banco de dados distribuído e sua aplicação no cotidiano.

De forma usual, um banco de dados comum relacional está presente em um único servidor onde nele se tem um banco de dados com todas as suas tabelas, *views*, procedures etc.

Figura 1. Estrutura centralizada

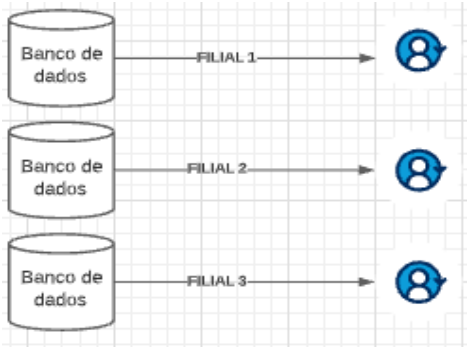
Fonte: O autor, 2021.

A figura 1 ilustra como é o funcionamento de um banco de dados centralizado, onde todos os dados estão localizados na mesma base de dados e os usuários acessam sempre a mesma base de informações.

Porém, quando falamos de softwares locais, onde esses não acessam a internet, acaba causando um prejuízo já que o banco de dados precisa estar no mesmo servidor que o software para ser acessado. Analisando questões como esta é que surgiu a arquitetura de banco de dados distribuídos.

Imagine uma grande empresa, com diversas filiais, cada uma com seu banco de dados, dessa forma, nenhuma filial consegue acessar os dados de outra filial, causando no cotidiano alguns problemas. Com o banco de dados distribuídos esses problemas podem ser mitigados.

Figura 2. Empresa com arquitetura centralizada



Fonte: O autor, 2021.

Analisando a figura 2, é possível descrever como se atribui o banco de dados em uma empresa com softwares locais sem a centralização da base de dados na internet, utilizando infraestrutura local.

Cada filial possui seu banco de dados, dessa forma, uma filial não acessa a informações de outra filial e os dados não são compartilhados, com essa arquitetura é possível que um cliente possua cadastro nas 3 filiais com as mesmas informações.

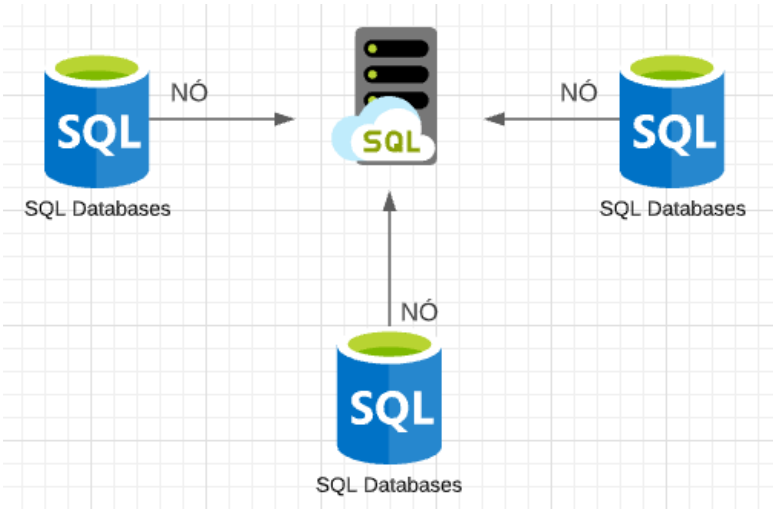
Videoaula 1

Utilize o QR Code para assistir!

Assista ao vídeo para entender melhor o assunto.

Para melhor compreender a estrutura de banco de dados distribuídos é necessário visualizar que cada banco de dados é uma ponta da estrutura, ou como alguns autores vão relatar, cada BD é um NÓ dentro da arquitetura distribuída.

Figura 3. Banco de Dados distribuídos (Nó)



Fonte: O autor, 2021.

Pode-se observar na figura 3 um exemplo da arquitetura de banco de dados distribuído, onde cada SQL *Database* está presente em um servidor diferente, sendo

cada um desses um NÓ dentro da arquitetura, no exemplo disposto, esses servidores são gerenciados por um Sistema de Gerenciamento de Banco de Dados Distribuído (DDBMS).

Indicação de Vídeo

Banco de Dados Distribuídos: https://youtu.be/3WO3cy22dQ0

Sistema de Gerenciamento de Banco de Dados Distribuídos (DDBMS)

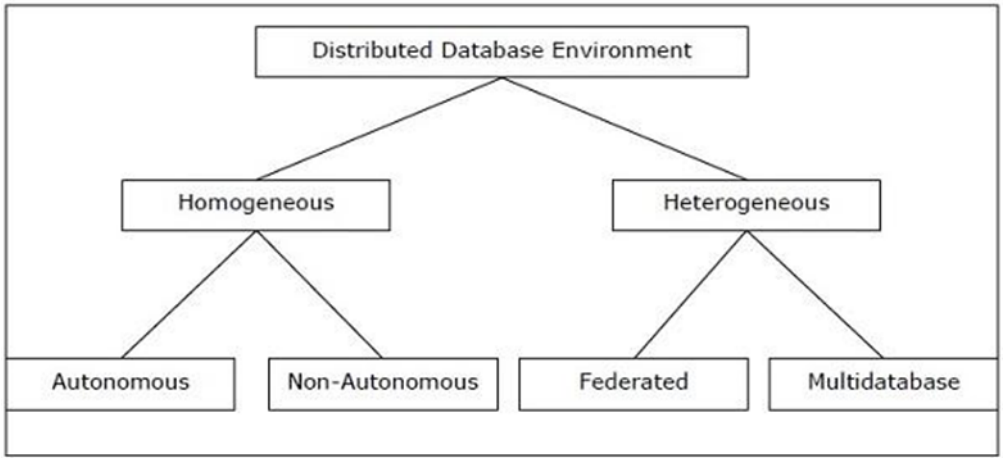
Um DDBMS serve para centralizar e integrar os dados logicamente dos bancos de dados presentes nesta estrutura, faz com que o usuário dele sinta que os dados estão sendo armazenados em um único local, mesmos estes estando em servidores diferentes.

Para que isso ocorra, o DDBMS sempre está sincronizando os dados para assim garantir que todas as operações de inserção, atualização e exclusão sejam executadas nos dados que estão em um ou outro local.

Normalmente, utilizam da réplica de BDs para manter a segurança dos dados, caso seja necessário realizar a recuperação de uma informação ou operação, por conta disto um DDBMS mantém consigo uma ou diversas réplicas de cada server BD.

O banco de dados distribuídos, podem ser classificados de duas formas:

Figura 4. Banco de Dados distribuídos divisão



Tutorialspoint, imagem retirada da internet.

Analisando a figura 4, visualiza-se como é dividido os tipos e submodelos dos bancos de dados distribuídos, podendo ser esse Homogêneo ou Heterogêneo onde cada um deles tem seu submodelo e suas vantagens e desvantagens.

As arquiteturas DDBMS estão sempre envolvidas e dependentes de três parâmetros, são eles:

- **Distribuição**: indicando como é realizada a distribuição dos dados em diferentes servidores.
- **Autonomia**: indica qual o grau de controle de cada servidor de banco de dados, demonstrando o quanto um é independente do outro.
- **Heterogeneidade**: indica a dissimilaridade ou uniformidade dos modelos de dados, esquemas e componentes.

Videoaula 2

Utilize o QR Code para assistir!

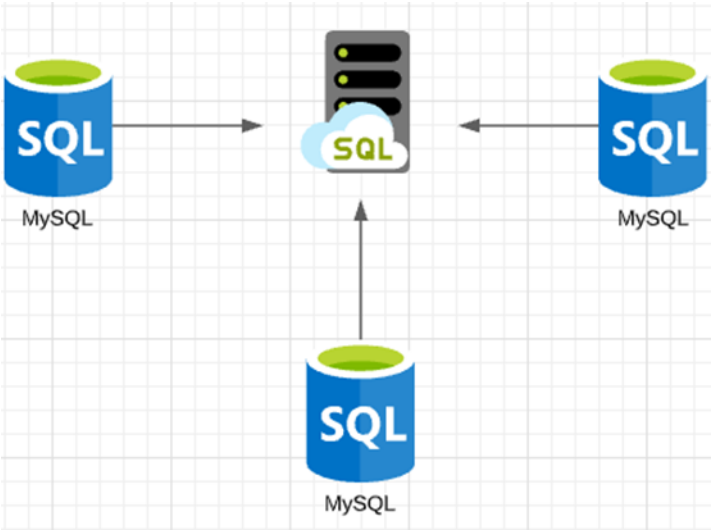
Assista ao vídeo para entender melhor o assunto.

Homogêneos

Nesta classificação, o banco de dados é distribuído em sua arquitetura física ou na nuvem, porém para o usuário e programador ele acessa os dados como se fosse somente um único banco de dados.

Para que isso ocorra cada **NÓ** precisa ser **idêntico**, sendo assim, caso seja um BD MySQL todos os nós precisam ser um servidor MySQL com as configurações muito parecidas.

Figura 5. Banco de Dados Homogêneos



Fonte: O autor, 2021.

A figura 5 relata como funciona a arquitetura de um Banco de Dados Distribuído de forma Homogênea, normalmente utilizado quando é necessário distribuir as tabelas em diversos servidores do mesmo tipo.

Os servidores de BD cooperam entre si para processar as operações solicitadas pelo usuário, desse modo, utilizando de um **DDBMS** para gerenciar todo esse fluxo e até realizar operações direto no banco de dados quando necessário.

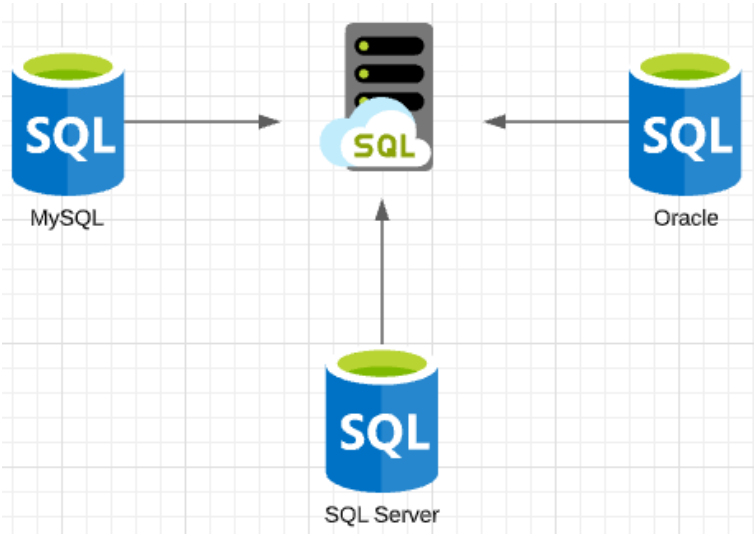
Um banco de dados distribuído de forma homogênea pode ser de dois modelos, sendo eles:

- **Autônomo**: cada servidor do banco de dados (nó) funciona de modo independente, porém utilizam de um aplicativo de controle e utilizam da passagem de mensagem para compartilhar as transações entre si.
- **Não autônomo**: nesse caso é necessário um DDBMS que centralize as operações e transações coordenando as operações dentro dos servidores.

Heterogêneo

Nesse tipo os servidores de bancos de dados podem possuir modelos e configurações diferentes e tipos de BD diferentes, o que favorece quando na empresa se processa tipos de dados e operações diferentes, pois concede a liberdade para melhor aproveitar cada dado.

Figura 6. Banco de Dados Heterogêneos



Fonte: O autor, 2021.

Na figura 6, demonstra-se a arquitetura de um banco de dados distribuído heterogêneo, onde cada nó pode possuir servidor e configurações de banco de dados diferentes.

Neste caso, operações de consultas e transações são mais complexas, pois cada servidor possui um esquema diferente de BD. Um servidor não conhece os outros, desse modo, a cooperação é limitada no processamento de informações para cada solicitação dos usuários.

Um banco de dados distribuído de forma heterogênea pode ser de dois modelos, sendo eles:

- **Não federado:** possui um módulo de coordenação central que realiza o intermédio de qual banco de dados será acessado.
- **Federado:** Os servidores de banco de dados são independentes entre si, porém são integrados para que funcionem como um único sistema de banco de dados.

Videoaula 3

Utilize o QR Code para assistir!

Assista ao vídeo para entender melhor o assunto.

Estratégia para Banco de Dados Distribuídos

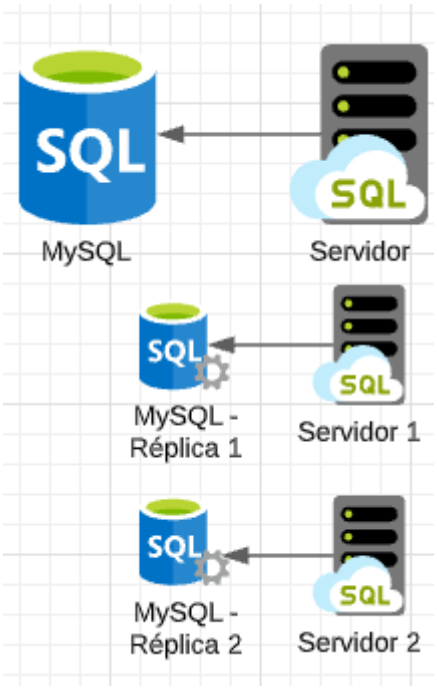
Para se utilizar de banco de dados distribuídos é preciso se atentar para alguns pontos e estratégias que devem ser construídas. Em sua maioria, as estratégias utilizadas são de replicação dos dados e a fragmentação.

Porém, pode-se utilizar das duas estratégias em conjunto, dependendo do projeto e como se organizará a estrutura onde esse projeto será utilizado.

Replicação dos Dados

Essa estratégia consiste em armazenar uma réplica do banco de dados em dois ou mais servidores, essa técnica é muito popular pois mitiga falhas em bancos de dados distribuídos.

Figura 7. Réplica dos Dados



Fonte: O autor, 2021.

Observa-se na figura 7 um modelo de réplica dos dados onde os servidores 1 e 2 armazenam réplicas do banco de dados MySQL. Essa aplicação resulta em maior segurança e mitigação de falhas no banco de dados como a inconsistência dos dados.

A utilização da técnica de replicação resulta em alguns benefícios são eles:

- **Confiabilidade/Disponibilidade:** caso ocorra uma falha no servidor o sistema de banco de dados continua disponível, pois existem cópias em outros servidores que seguem funcionando normalmente.
- **Redução na carga de rede:** com os dados disponibilizados em mais de um local, o processamento de consultas pode ser realizado com o uso reduzido da rede.
- **Resposta mais rápida:** como o dado está presente em vários locais, a resposta para uma busca é mais rápida.

Porém existem desvantagens na utilização dessa técnica, são eles:

- **Maior requisito de armazenamento:** realizar a manutenção de diversas cópias de um banco de dados resulta em maiores custos no armazenamento.
- **Custo maior:** manter réplicas geram um custo monetário bem maior do que possuir somente um servidor.
- **Complexidade na manutenção dos dados:** como as réplicas precisam ser idênticas é necessário possuir um mecanismo de atualização dos dados bem implementado para que não haja inconsistência das informações.

Fragmentação

A técnica da fragmentação consiste em dividir uma tabela em um conjunto de tabelas menores, esses conjuntos são denominados fragmentos, esse processo pode ser realizado de três tipos:

- Fragmentação horizontal
- Fragmentação vertical
- Fragmentação híbrida (horizontal + vertical)

A fragmentação precisa ser realizada corretamente, para que caso se faça necessário a reconstrução da tabela original, isso seja possível a partir de seus fragmentos, isso é um requisito no processo de fragmentação e é chamado de reconstrutividade. Toda tabela que é fragmentada deve atender ao requisito de reconstrutividade.

Alguns benefícios da fragmentação:

- **Eficiência:** como os dados são armazenados perto do local de uso, a eficiência do banco de dados aumenta.
- **Otimização das consultas:** como as tabelas possuem dados menores, é possível pesquisar valores únicos de forma simples.
- **Aumento da segurança:** como os dados estão espalhados em diversas tabelas, a privacidade da tabela original é mantida.

Desvantagens da aplicação da fragmentação:

- **Velocidade de acesso:** quando é necessário retornar um conjunto de dados próximo da tabela original, pode-se afetar a velocidade de resposta do banco de dados.
- **Reconstrução complexa:** quando se utiliza de fragmentações recursivas.
- **Não réplica:** sem uma cópia do banco de dados em outros servidores, pode ser um problema caso ocorra alguma falha no servidor.

Banco de Dados Distribuídos necessidade e aplicação

Olá, aluno, tudo bem? Nesta aula vamos conhecer melhor os conceitos de Banco de Dados Distribuídos, vamos descobrir juntos como funciona a sua aplicação e como essa prática vem mudando a forma de armazenar os dados.

Fragmentação Vertical

Esse processo de fragmentação ocorre com cada coluna ou um conjunto de colunas resultam em novas tabelas ou “fragmentos” para atender ao requisito de reconstrução da tabela original, todos os fragmentos devem possuir uma referência da chave primária da tabela original.

Figura 8. Tabela Produto

	id_produto [PK] integer	nome character varying (255)	valor numeric (10,2)	fabricante character varyi	endereço character varying (255)	modelo character var	custo numeric (10,2)
1	1	Notebook Dell Intel Octa Core	5200.00	Dell	Av. São Paulo - Londrina	Vostro	3000.00
2	2	Notebook Acer AMD	4100.00	Acer	Rua Pernambuco - Londrina	Aspire 5	2700.00

Fonte: O autor, 2021.

Na figura 8, encontra-se uma tabela de origem para ser fragmentada verticalmente, sendo ela a tabela Produto, com suas colunas: id_produto; nome; valor; fabricante; endereço; modelo; custo.

Aplicando a técnica de fragmentação na tabela produto, pode-se encontrar o resultado de dois fragmentos. Sendo o fragmento 1:

Figura 9. Tabela Fragmento 1

	id_produto [PK] integer	nome character varying (255)	valor numeric (10,2)
1	1	Notebook Dell Intel Octa Core	5200.00
2	2	Notebook Acer AMD	4100.00

Fonte: O autor, 2021.

Na figura 9, visualiza-se as colunas alocadas no fragmento 1 da tabela produto, onde se encontram os dados nome e valor juntamente com a chave do produto denominado id_produto.

Figura 10. Tabela Fragmento 2

	id_produto [PK] integer	fabricante character varying (255)	endereco character varying (255)	modelo character varying (255)	custo numeric (10,2)
1	1	Dell	Av. São Paulo - Londrina	Vostro	3000.00
2	2	Acer	Rua Pernambuco - Londrina	Aspire 5	2700.00

Fonte: O autor, 2021.

Na figura 10, encontra-se descrito o fragmento 2 da tabela produto com as colunas: fabricante; endereço; modelo; custo. Juntamente com chave da tabela de origem o id_produto.

Esse é um exemplo de fragmentação bem simples, respeitando a regra de reconstrução da tabela de origem caso seja necessário.

Fragmentação Horizontal

A fragmentação horizontal ocorre por conta das **tuplas (linhas)**, sendo assim, os fragmentos são derivados da tabela de origem, porém com todas as colunas, porque os fragmentos possuem os dados e a fragmentação ocorreu pelo agrupamento de dados com valores semelhantes.

Figura 11. Tabela de origem Produto

	id_produto [PK] integer	nome character varying (255)	valor numeric (10,2)	fabricante character varying (255)	endereco character varying (255)	modelo character varying (255)	custo numeric (10,2)
1	1	Notebook Dell Intel Octa Core	5200.00	Dell	Av. São Paulo - Londrina	Vostro	3000.00
2	3	Notebook Acer INTEL	3900.00	Acer	Rua Pernambuco - Londrina	Aspire 4	2100.00
3	4	Notebook Acer INTEL 1TB	3850.00	Acer	Rua Pernambuco - Londrina	Aspire 4	2085.00
4	2	Notebook Acer AMD	4100.00	Acer	Rua Pernambuco - Londrina	Aspire 5	2700.00

Fonte: O autor, 2021.

Na figura 11, encontra-se a tabela de origem denominada **Produto**, com seus dados preenchidos. Para construir a fragmentação horizontal, deve-se executar o script:

Figura 12. Script criação fragmentação horizontal

```
1  CREATE TABLE fragmentacao.fragmentacao_horizontal (
2      CHECK (fabricante = 'Acer'))
3      INHERITS (fragmentacao.produto);
```

Fonte: O autor, 2021.

A figura 12 demonstra o script de criação de uma tabela com fragmento horizontal, onde todos os produtos da marca Acer irão ficar presentes na nova tabela denominada **fragmentacao_horizontal**. Que possui os dados:

Figura 13. Fragmentação horizontal da tabela Produto

	id_produto [PK] integer	nome character varying (255)	valor numeric (10,2)	fabricante character varying (255)	endereco character varying (255)	modelo character varying (255)	custo numeric (10,2)
1	3	Notebook Acer INTEL	3900.00	Acer	Rua Pernambuco - Londrina	Aspire 4	2100.00
2	4	Notebook Acer INTEL 1TB	3850.00	Acer	Rua Pernambuco - Londrina	Aspire 4	2085.00
3	2	Notebook Acer AMD	4100.00	Acer	Rua Pernambuco - Londrina	Aspire 5	2700.00

Fonte: O autor, 2021.

Analisando a figura 13, encontram-se todos os dados da tabela Produto, porém fragmentados por tuplas onde todas possuem um fabricante em comum. Alguns autores defenderão que a fragmentação horizontal favorece o processo de reconstrução da tabela de origem caso seja necessário.

Outra grande vantagem da fragmentação é a otimização de consultas rápidas no BD, isso ocorre devido aos dados estarem em locais separados e assim fragmentados. Mas, caso a consulta precise ser mais complexa, a fragmentação pode acabar se tornando um grande problema.

Existe também a possibilidade da fragmentação híbrida, leva este nome pois é a combinação das duas técnicas de fragmentação, onde pode ocorrer de duas formas:

- Gerar um conjunto de fragmentos horizontais, em seguida gerar fragmentos verticalizados do fragmento horizontal.
- Gerar um conjunto de fragmentos verticais, em seguida gerar fragmentos horizontais do fragmento vertical.

A fragmentação é uma técnica fortemente utilizada em banco de dados distribuídos, a decisão para a utilização dessa técnica e qual tipo dela será utilizada demanda uma análise por parte do time de desenvolvimento em conjunto com infraestrutura e gestores de banco de dados.

Videoaula 1

Utilize o QR Code para assistir!

Assista ao vídeo para entender melhor o assunto.

Transparência

A transparência é uma das propriedades mais trabalhadas em bancos de dados distribuídos, visto que com essa propriedade é que são tratados detalhes internos do BD e sua distribuição de dados entre os usuários.

De acordo com o DDBMS, a sua estrutura pode optar por realizar de diferentes técnicas: fragmentação, replicação e assim armazenar em locais diferentes. Dessa forma, a transparência em banco de dados distribuídos pode ser atribuída da seguinte maneira:

- Transparência local: o usuário pode consultar qualquer tabela ou fragmento como se este estivesse alocado localmente em seu servidor. Nesse caso, os dados estarem sendo armazenados em um local remoto não deve afetar o usuário, até porque os dados de acesso remoto devem ficar ocultos para ele.
- Transparência de fragmentação: permite ao usuário consultar qualquer tabela como se ela fosse desfragmentada. Desse modo, sendo invisível para o usuário que ele está consultando um fragmento ou um conjunto de fragmentos.
- Transparência de replicação: nesses casos, a réplica da tabela ou banco de dados não é visível para o usuário, sendo assim, para ele é como se existisse somente uma cópia da tabela ao qual está consultando. Essas situações são utilizadas quando ocorre uma atualização também, pois o usuário atualiza somente um local e o DDMBS se responsabiliza em atualizar as réplicas.

Pode ocorrer a combinação de diversas transparências em um banco de dados distribuído, porém quem está arquitetando esse BD deve garantir que as transparências definidas sejam mantidas em toda a base de dados.

Aplicar a transparência agrega segurança e mitiga falhas em um banco de dados distribuído, porém demanda um grande esforço.

Controle de acesso

A segurança aplicada no controle de acesso é uma tarefa normalmente realizada pelo Administrador de Banco de Dados (DBA), quando estruturado e organizado os acessos ao banco de dados, o usuário tem suas permissões, o que torna mais seguro a base de dados e mantém sua privacidade.

O controle de acesso pode ser definido em algumas dimensões:

Autenticação

Em banco de dados distribuídos, somente usuários nativos podem se autenticar e acessar os recursos, sendo assim, ao realizar o login no banco de dados o usuário só acessará os recursos que estão vinculados a sua autenticação.

Videoaula 2

Utilize o QR Code para assistir!

Assista ao vídeo para entender melhor o assunto.

Direitos de acesso

Considerando um ambiente distribuído existem diversas tabelas e também diversos usuários que precisam consultar e realizar outras operações dentro de uma base de dados.

Assim, o DDBMS define alguns papéis, sendo que cada papel possui seus privilégios de acesso que são relacionados aos cargos e assim, após tudo isso definido, os usuários são atribuídos aos cargos. Nesse caso se constitui uma hierarquia de acesso.

Figura 14. Exemplo comandos SQL permissões

```
CREATE ROLE ACCOUNTANT;
GRANT SELECT, INSERT, UPDATE ON EMP_SAL TO ACCOUNTANT;
GRANT INSERT, UPDATE, DELETE ON TENDER TO ACCOUNTANT;
GRANT INSERT, SELECT ON EXPENSE TO ACCOUNTANT;
COMMIT;
GRANT ACCOUNTANT TO ABC;
COMMIT;
```

Fonte: tutorialspoint, 2021.

A figura 14 demonstra um exemplo de comandos SQL para criação do cargo Accountant com seus privilégios e, em seguida, atribui este cargo ao usuário ABC. A definição desses privilégios pode ou não serem validados e verificados com as pessoas responsáveis pela segurança da informação.

Controle da integridade da entidade

Para falar de integridade da entidade, vale relembrar que durante a modelagem conceitual de um banco de dados se constrói um modelo de entidade e relacionamento, onde a entidade depois vem a ser uma tabela no BD.

Esse controle define que cada tupla (linha) da tabela possua uma definição que possa ser exclusivamente identificada de outras tuplas, por isso existe a chave primária que é o identificador único de uma linha.

O controle de integridade da entidade defende que cada linha deve ser única, ou seja, sua identificação não deve se repetir e a chave primária também não pode possuir um valor nulo. Essa regra de não possuir valores nulos e outras regras podem ser aplicadas a outros atributos da entidade.

Figura 15. Exemplo das regras no PostgreSQL

```
CREATE TABLE Alunos (
  ID_Aluno integer,
  Nome_Aluno text NOT NULL,
  Idade numeric CHECK (Idade >= 18)
);
```

Fonte: Boson Treinamentos, 2021.

Observa-se na figura 15 a aplicação de 2 regras na tabela Alunos, onde o atributo Nome_Aluno não pode receber um valor nulo e o atributo Idade deve ser maior igual à 18. São exemplos simples de regras que servem para manter a integridade e até mesmo aplicar conceitos de negócio ao banco de dados.

A aplicação dessas regras em uma tabela de banco de dados deve ocorrer na criação dela conforme demonstra a figura 15.

Essas regras são chamadas de ***Constraints*** (Restrições) e cada banco de dados apresenta um modelo e maneira de aplicar essas restrições que, em sua maioria, aumentam o controle de integridade de uma entidade no BD.

Indicação de Leitura

Constraints no PostgreSQL (Restrições). Disponível em: http://www.bosontreinamentos.com.br/postgresql-banco-dados/constraints-no-postgresql-restricoes/. Acesso em: 28 mar. 2022.

Restrição de integridade referencial

Esse tipo de restrição estabelece as normas para chaves estrangeiras, que são aquelas que servem de referência entre as entidades para estabelecer um relacionamento. Defende que essa chave deve existir na tabela referenciada ou possuir o valor nulo, dependendo do relacionamento ela pode se repetir em tuplas diferentes na mesma tabela.

Figura 16. Exemplo de criação da tabela Student SQL

```
CREATE TABLE STUDENT (
   S_ROLL INTEGER PRIMARY KEY,
   S_NAME VARCHAR2(25) NOT NULL,
   S_COURSE VARCHAR2(10),
   S_HOSTEL VARCHAR2(5) REFERENCES HOSTEL
);
```

Fonte: tutorialspoint, 2021.

No exemplo demonstrado na figura 16, demonstra a criação de uma tabela utilizando do SQL nativo, onde na tabela estudante pode existir uma referência no campo SHOSTEL que é uma chave estrangeira da tabela HOSTEL.

Videoaula 3

Utilize o QR Code para assistir!

Assista ao vídeo para entender melhor o assunto.

Segurança

Com banco de dados distribuídos, os aspectos de segurança são diferentes, até porque há muitos usuários, diversos dados e em vários servidores. Dentro do processo de comunicação em um BD distribuído existem alguns preocupações e pontos a serem considerados como:

- O canal de comunicação deve estar livre de intrusos
- Os dados não podem ser corrompidos durante a transferência de dados.
- É preciso definir protocolos e algoritmos de segurança diariamente.

Em alguns casos, para manter a segurança na comunicação da rede de servidores de um bando de dados distribuídos, são utilizados dos recursos de:

- *Secure Socket Layer Protocol or Transport Layer Security Protocol.*
- *Virtual Private Networks (VPN).*

O que é mais valioso em um banco de dados são os próprios dados, dessa forma, existem algumas medidas que podem ser adotadas em um banco de dados distribuído para aumentar a segurança para com os dados, são elas:

- Autorização e autenticação: medida que valida para que somente usuários autênticos acessem e manipulem os dados, essa medida também já foi relatada nesta aula.
- Dados criptografados: podendo ser de 2 modelos:
    - Criptografia externa: onde os dados são criptografados pelo sistema e então salvos no banco de dados já criptografados, dessa forma, cada vez que for realizar uma operação o sistema que está interagindo deve criptografar o dado antes.
    - Criptografia interna: é quando dentro do próprio sistema de banco de dados distribuídos existem recursos de criptografia, nesse modelo, o usuário e o sistema que interage com o BD não sabe que a informação está sendo armazenada de forma criptografada.
- Entrada validada: Nessa medida a cada operação realizada a entrada dos comandos e informações são validadas antes de serem executadas.

Aplicando corretamente essas medidas o banco de dados fica mais seguro e consistente. Porém, mesmo assim pode ocorrer alguma falha na segurança e detectar essas falhas muitas vezes é difícil.

Para auxiliar na identificação e detecção das falhas de segurança. Uma medida essencial é a auditoria de dados. Isso ocorre por intermédio dos logs de auditoria de dados, esses logs devem possuir informações como:

- Data, hora, local de acesso que falhou;
- Detalhes dos acessos bem-sucedidos;
- Detalhes das modificações dentro do banco de dados;
- Registros de grandes transações de dados.

Essas informações devem ficar segregadas do BD e armazenadas de preferência em um outro servidor, para que no caso de invasão seja possível validar as informações e rastrear as modificações realizadas.

Encerramento

Chegamos ao final da nossa terceira unidade, onde você teve contato com os conceitos e ferramentas relacionadas ao Banco de Dados Distribuídos. Com certeza, muito do que foi mostrado aqui você já tinha uma pequena noção.

Na aula 1, conhecemos os conceitos de banco de dados distribuídos, ao qual a proposta se aplica em empresas com dados em escala para que assim qualquer empresa que deseja melhorar o seu desempenho possa utilizar.

Na aula 2, foi discutido e apresentado conceitos e boas práticas juntamente com parâmetros essenciais em BD distribuídos, além de a aplicação de alguma dessas práticas.

Juntando as duas aulas obtemos descobertas interessantes sobre como tratar e estabelecer minimamente um banco de dados distribuído. Caro aluno, pense um pouco em como os conhecimentos desta unidade podem te ajudar nas suas atividades e até mesmo no que você pode se aprofundar nesses conhecimentos. Até a próxima.

Referências

HEUSER, Carlos Alberto. **Projeto de banco de dados**: Volume 4 da Série Livros didáticos informática - UFRGS. Porto Alegre: Bookman, 2009.

TUTORIALSPOINT. Disponível em: https://www.tutorialspoint.com/. Acesso em: 28 mar. 2022.

Título da Aula 2

###


UniFil
EDUCAÇÃO A DISTÂNCIA
UNIFIL.BR