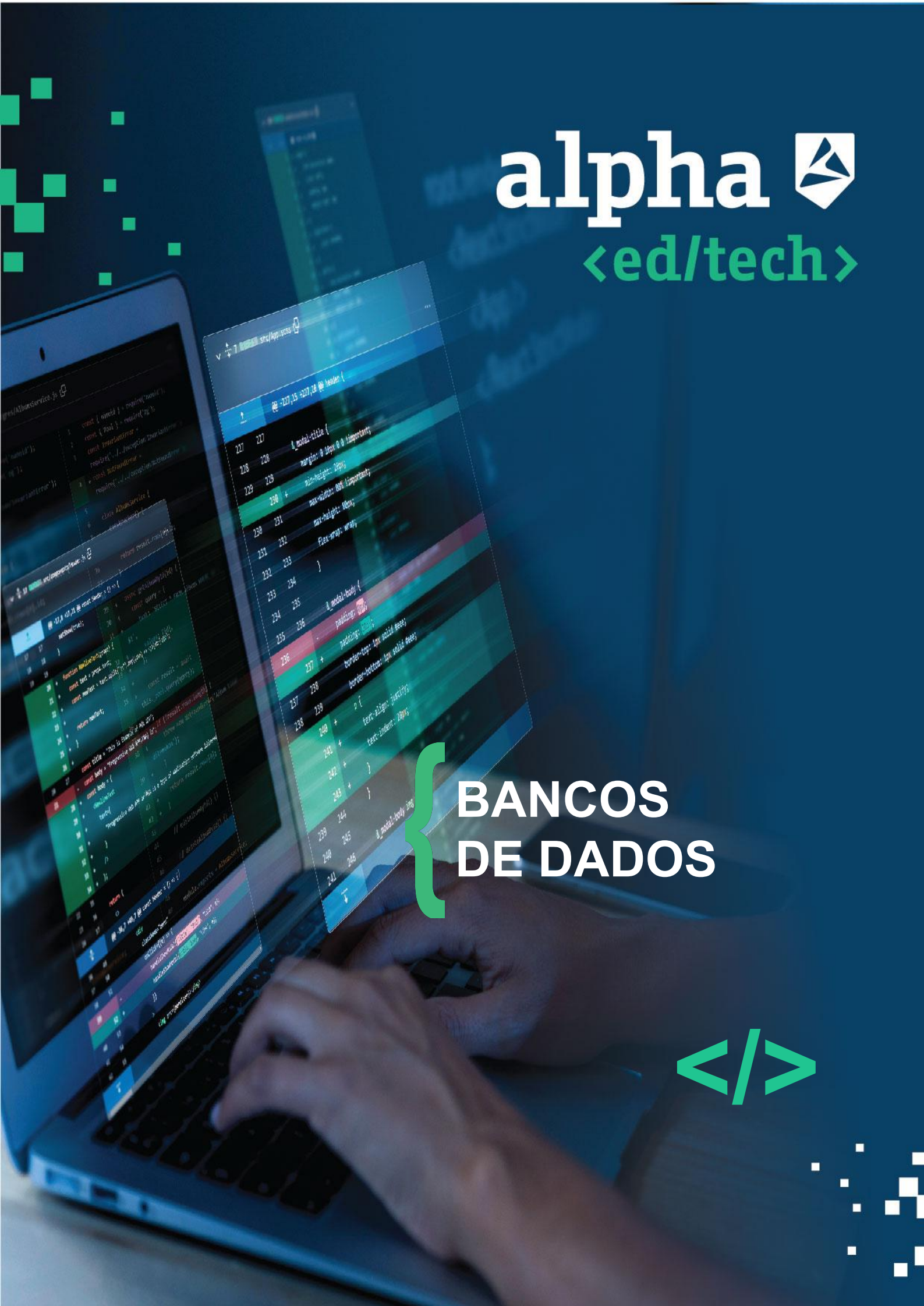
alpha

<ed/tech>



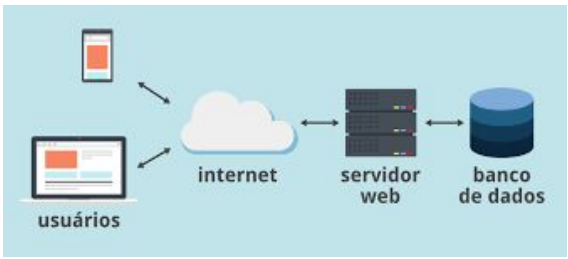
BANCOS DE DADOS

</>

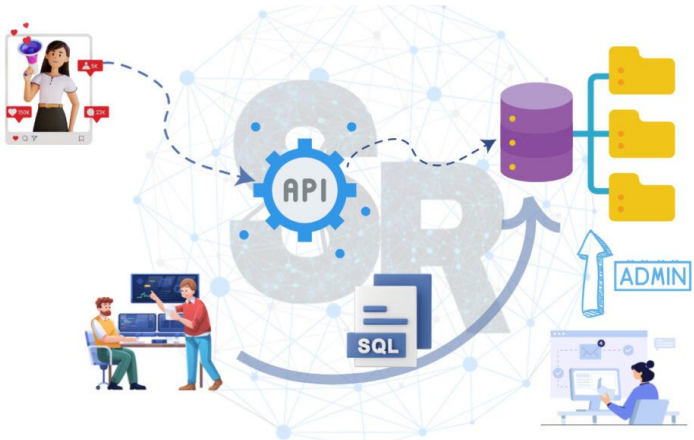
Banco de dados

O que é?

Um banco de dados é um conceito central na ciência da computação e na gestão da informação. Em essência, é um sistema organizado e estruturado para armazenar, gerenciar e recuperar dados de forma eficiente. Desde os primeiros sistemas computacionais até as mais avançadas aplicações de tecnologia da informação, os bancos de dados desempenham um papel crucial em praticamente todas as áreas da vida moderna, da gestão empresarial à pesquisa científica.



O banco de dados é a fonte verdadeira de toda informação dentro de uma aplicação.



Com ele é possível persistir, otimizar, acesso e gerenciar as informações.

A importância dos bancos de dados na sociedade moderna não pode ser subestimada.

Aqui estão algumas das razões pelas quais eles são fundamentais:

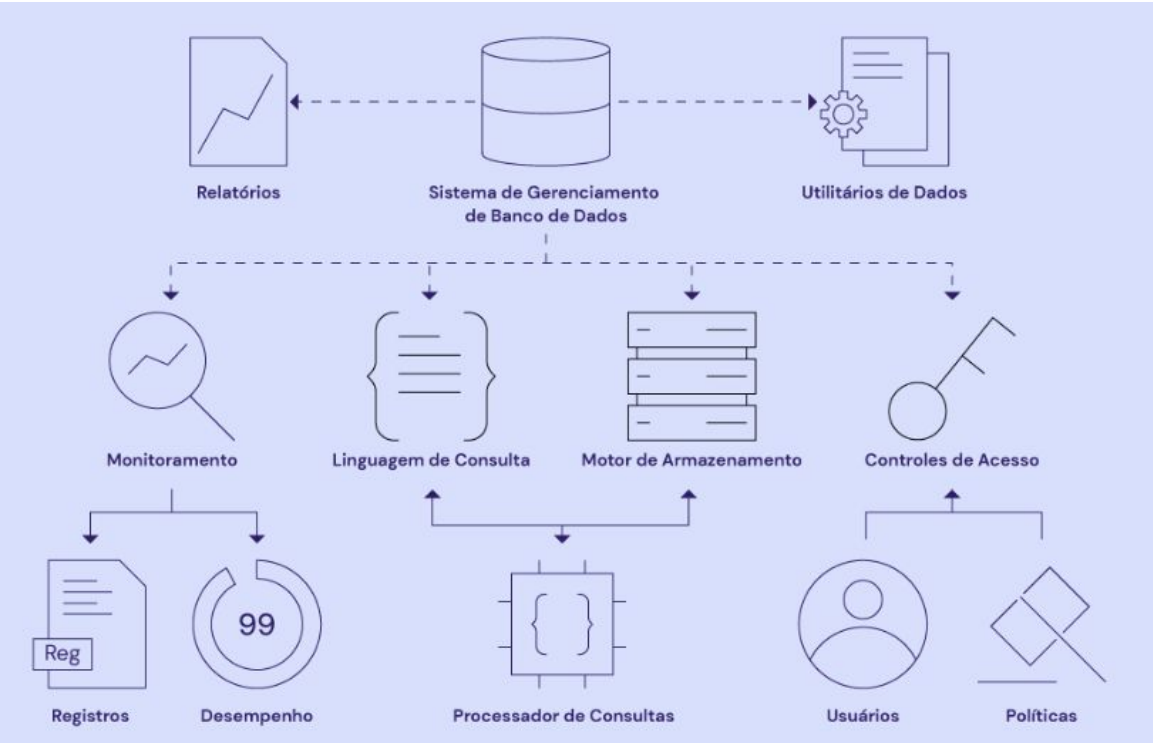
- Organização da Informação: Os bancos de dados permitem que grandes volumes de informações sejam organizados de maneira estruturada e acessados de forma eficiente.
- Recuperação de Informações: Eles facilitam a recuperação rápida e precisa de informações específicas, permitindo que os usuários encontrem e manipulem os dados conforme necessário.
- Consistência e Integridade dos Dados: Os bancos de dados garantem que os dados sejam consistentes e íntegros, implementando restrições e validações para evitar inconsistências e erros.
- Compartilhamento de Dados: Eles facilitam o compartilhamento de informações entre diferentes usuários e sistemas, permitindo colaboração e integração entre diferentes partes de uma organização ou entre organizações diferentes.
- Segurança dos Dados: Os bancos de dados oferecem recursos avançados de segurança, como controle de acesso, criptografia e auditoria, para proteger os dados contra acesso não autorizado e garantir conformidade com regulamentações de privacidade e segurança.

Banco de dados

O que é?

Os bancos de dados são compostos por vários componentes, incluindo:

- Dados: São as informações que são armazenadas no banco de dados. Podem variar desde simples valores numéricos até textos complexos, imagens, vídeos e outros tipos de mídia.
- Sistema de Gerenciamento de Banco de Dados (SGBD): É o software responsável por gerenciar o acesso aos dados, garantindo sua integridade, segurança e eficiência. Exemplos de SGBDs incluem o PostgreSQL, MySQL, Oracle Database, SQL Server, MongoDB, entre outros.
- Usuários e Aplicativos: São as pessoas ou sistemas que interagem com o banco de dados, realizando consultas, inserções, atualizações e exclusões de dados conforme necessário. Isso pode incluir desde usuários humanos acessando um sistema de gerenciamento de banco de dados via interface gráfica até aplicativos web ou móveis que se comunicam com o banco de dados por meio de APIs.
- Metadados: São dados sobre os dados. Eles descrevem a estrutura dos dados armazenados no banco de dados, incluindo informações sobre tabelas, colunas, tipos de dados, restrições, relacionamentos e outras características importantes.



Em resumo, os bancos de dados são a espinha dorsal da infraestrutura de informação moderna, capacitando organizações e indivíduos a armazenar, gerenciar e aproveitar o vasto volume de dados que impulsiona o nosso mundo digital.

Bancos de Dados Relacionais

Os bancos de dados relacionais constituem uma das abordagens mais tradicionais e amplamente adotadas para o armazenamento e gerenciamento de dados. Este modelo, introduzido por Edgar F. Codd na década de 1970, revolucionou a forma como as informações são estruturadas e manipuladas em sistemas computacionais. Nesta página, exploraremos os principais conceitos e características dos bancos de dados relacionais.

Exemplo	A	B	C	D	E
1	Id	Nome	Raça	Cor	Idade
2	1	Caramelo	Vira lata	Amarela	8
3	4	Xuxa	Dálmata	Branca	3
4	3	Lola	Golden Retriever	Amarela	4
5	2	Bolinha	Pinscher	Preta	12

Um exemplo de tabela sobre cachorros, onde as linhas representam um elemento dessa lista e cada coluna suas propriedades.

O modelo relacional organiza os dados em tabelas, também conhecidas como relações. Cada tabela é composta por linhas e colunas, onde as colunas representam os atributos dos dados e as linhas representam as entradas individuais ou registros. Mais um exemplo, em um banco de dados de uma biblioteca, poderíamos ter uma tabela "Livros" com colunas como "ID do Livro", "Título", "Autor" e "Ano de Publicação".

Bancos de Dados Relacionais

Chave Primária e Chave Estrangeira

Em um banco de dados relacional, cada tabela tem uma chave primária, que é um conjunto de um ou mais atributos que identificam exclusivamente cada registro na tabela. Isso garante a unicidade dos dados e facilita a indexação e recuperação eficiente. Além disso, as chaves estrangeiras são utilizadas para estabelecer relações entre diferentes tabelas, permitindo a integridade referencial e a manutenção da consistência dos dados.

Chave Primária (Primary Key):

A chave primária é um conceito fundamental em bancos de dados relacionais. Ela é um ou mais atributos que identificam exclusivamente cada registro em uma tabela.

A chave primária deve ser única para cada registro na tabela, o que significa que nenhum outro registro pode ter o mesmo valor para a chave primária.

A chave primária também deve ser não nula, garantindo que cada registro na tabela tenha uma identificação válida.

Exemplos de chaves primárias incluem números de identificação exclusivos (ID), códigos alfanuméricos únicos ou qualquer combinação de atributos que satisfaça os requisitos de unicidade e não nulidade.

O uso de chaves primárias facilita a indexação e recuperação eficiente de dados, pois permite que o banco de dados localize rapidamente registros específicos com base em suas chaves primárias.

Chave Estrangeira (Foreign Key):

A chave estrangeira é um atributo em uma tabela que estabelece uma relação com a chave primária de outra tabela.

Ela é usada para garantir a integridade referencial entre tabelas, permitindo que o banco de dados mantenha relacionamentos entre os dados e mantenha a consistência dos dados.

Ao definir uma chave estrangeira em uma tabela, estamos essencialmente indicando que os valores nessa coluna devem corresponder a valores existentes na chave primária de outra tabela.

Por exemplo, em um banco de dados de uma biblioteca, poderíamos ter uma tabela "Empréstimos" que contém uma chave estrangeira "ID do Livro", que se relaciona com a chave primária "ID do Livro" na tabela "Livros". Isso garante que cada empréstimo esteja associado a um livro existente na biblioteca.

As chaves estrangeiras são uma ferramenta poderosa para garantir a integridade dos dados e manter a consistência do banco de dados, evitando a inserção de dados inválidos ou órfãos.

É importante observar que as operações de atualização e exclusão de registros em uma tabela com chaves estrangeiras podem ter impacto nas tabelas relacionadas. Por exemplo, ao excluir um registro em uma tabela pai, é possível configurar o banco de dados para automaticamente excluir ou atualizar registros relacionados em tabelas filhas (chamado de ação de cascata).

Bancos de Dados Relacionais

Linguagem SQL

A Linguagem de Consulta Estruturada, mais conhecida como SQL (Structured Query Language), é uma linguagem de programação específica para gerenciamento de dados em bancos de dados relacionais. Criada originalmente pela IBM nos anos 1970, o SQL tornou-se um padrão amplamente adotado na indústria de tecnologia da informação, sendo utilizado por sistemas de bancos de dados como Oracle, MySQL, SQL Server, PostgreSQL, entre outros.
Exploraremos os principais aspectos da Linguagem SQL e sua importância na manipulação e consulta de dados.

Comandos SQL Básicos:

A SQL é uma linguagem declarativa, o que significa que os usuários especificam quais resultados desejam, sem precisar se preocupar com os detalhes de como obtê-los. Os comandos básicos do SQL incluem mas não se limitam a:

- SELECT: Utilizado para recuperar dados de uma ou mais tabelas. É o comando principal para consulta de dados.

  Exemplo:

```
SELECT nome, idade FROM clientes
WHERE cidade = 'São Paulo';
```

- INSERT: Utilizado para adicionar novos registros a uma tabela.

  Exemplo:

```
INSERT INTO produtos (nome, preco)
VALUES ('Camiseta', 29.99);
```

- UPDATE: Utilizado para modificar os valores de um ou mais registros em uma tabela.

  Exemplo:

```
UPDATE produtos SET preco = 34.99 WHERE id = 101;
```

- DELETE: Utilizado para excluir registros de uma tabela.

  Exemplo:

```
DELETE FROM clientes WHERE idade < 18;
```

Bancos de Dados Relacionais

Linguagem SQL

Cláusulas SQL:

Além dos comandos básicos, a SQL também possui várias cláusulas que permitem refinar consultas e operações de manipulação de dados. Algumas das cláusulas mais comuns incluem:

- WHERE: Utilizada para filtrar os resultados de uma consulta com base em uma condição específica.

  Exemplo:

```
SELECT * FROM pedidos WHERE valor_total > 1000;
```

- ORDER BY: Utilizada para ordenar os resultados de uma consulta com base em uma ou mais colunas.

  Exemplo:

```
SELECT * FROM produtos ORDER BY preco DESC;
```

- GROUP BY: Utilizada para agrupar os resultados de uma consulta com base em uma ou mais colunas.

  Exemplo:

```
SELECT cidade, COUNT(*) AS total_clientes
FROM clientes GROUP BY cidade;
```

- HAVING: Utilizada em conjunto com a cláusula GROUP BY para filtrar grupos de resultados com base em condições específicas.

  Exemplo:

```
SELECT cidade, COUNT(*) AS total clientes
FROM clientes GROUP BY cidade HAVING COUNT(*) > 5;
```

Bancos de Dados Relacionais

Linguagem SQL

Funções SQL:

A SQL também possui uma variedade de funções embutidas que podem ser utilizadas para realizar cálculos, manipular strings, datas e muito mais. Algumas das funções mais comuns incluem:

- SUM: Calcula a soma dos valores de uma coluna.
- AVG: Calcula a média dos valores de uma coluna.
- COUNT: Conta o número de registros em um conjunto de resultados.
- MAX: Retorna o maior valor de uma coluna.
- MIN: Retorna o menor valor de uma coluna.
- UPPER: Converte uma string para letras maiúsculas.
- LOWER: Converte uma string para letras minúsculas.
- DATE_FORMAT: Formata uma data de acordo com um padrão específico.

Transações SQL:

Além de consultas e manipulações simples de dados, a SQL também suporta transações, que são sequências de operações que devem ser executadas como uma unidade indivisível. Isso garante a consistência e a integridade dos dados em ambientes multiusuários. As principais operações de controle de transações incluem:

- BEGIN TRANSACTION: Inicia uma nova transação.
- COMMIT: Confirma uma transação, efetivando as mudanças realizadas.
- ROLLBACK: Cancela uma transação, revertendo todas as mudanças realizadas desde o início da transação.

Conclusão:

A Linguagem SQL é uma ferramenta poderosa e versátil para manipulação e consulta de dados em sistemas de bancos de dados relacionais. Com sua sintaxe simples e poderosa, ela permite aos desenvolvedores e administradores de bancos de dados realizar uma ampla gama de operações, desde consultas simples até transações complexas e controle de acesso seguro. Dominar o SQL é essencial para qualquer profissional envolvido no desenvolvimento de software e na administração de sistemas de banco de dados.

Bancos de Dados Relacionais

Vantagens dos Bancos de Dados Relacionais

Vantagens dos Bancos de Dados Relacionais

Os bancos de dados relacionais têm sido a espinha dorsal da gestão de dados em muitas organizações ao redor do mundo há décadas. Sua popularidade e ampla adoção são impulsionadas por uma série de vantagens distintas que oferecem sobre outros modelos de banco de dados. Exploraremos algumas das principais vantagens dos bancos de dados relacionais.

1. **Estruturação de Dados:**

   Um dos principais benefícios dos bancos de dados relacionais é sua capacidade de estruturar os dados de maneira organizada e lógica. Os dados são armazenados em tabelas, onde cada tabela representa uma entidade ou conceito específico e as colunas representam os atributos dessas entidades. Essa estruturação facilita a compreensão e o gerenciamento dos dados, tornando-os mais acessíveis e fáceis de manipular.

2. **Integridade dos Dados:**

   Os bancos de dados relacionais oferecem mecanismos poderosos para garantir a integridade dos dados armazenados. Restrições de integridade, como chaves primárias, chaves estrangeiras e restrições de integridade referencial, ajudam a manter a consistência e a validade dos dados, evitando a inserção de informações inválidas ou inconsistentes.

3. **Flexibilidade:**

   Os bancos de dados relacionais são altamente flexíveis e podem se adaptar facilmente a mudanças nos requisitos do sistema. Permitindo adicionar, modificar e excluir dados e estruturas sem comprometer a integridade dos dados existentes. Isso oferece uma grande vantagem em ambientes onde os requisitos estão sujeitos a mudanças frequentes.

4. **Suporte a Transações:**

   Os bancos de dados relacionais oferecem suporte robusto para transações, garantindo que operações complexas possam ser executadas de forma consistente e confiável. As propriedades ACID (Atomicidade, Consistência, Isolamento, Durabilidade) garantem que as transações sejam tratadas de forma segura, mesmo em ambientes multi-usuários e concorrentes.

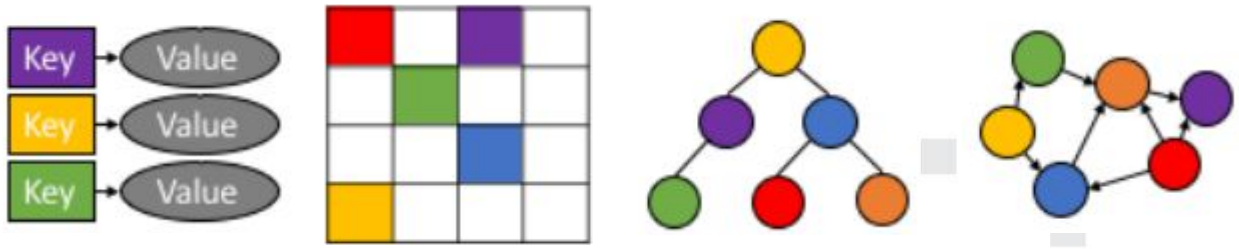
5. **Desempenho:**

   Em geral, os bancos de dados relacionais têm um desempenho muito bom para consultas que envolvem a recuperação de conjuntos de dados pequenos a médios. Além disso, a capacidade de indexação eficiente e otimização de consultas permite melhorar ainda mais o desempenho em cenários de alto volume de dados.

Em resumo, os bancos de dados relacionais oferecem uma série de vantagens significativas em termos de estruturação de dados, integridade, flexibilidade, desempenho e segurança. Sua capacidade de lidar com uma variedade de requisitos e cenários torna-os uma escolha popular para uma ampla gama de aplicações, desde sistemas de gestão empresarial até aplicações web e móveis.

Bancos de Dados Não-Relacionais

Os bancos de dados não-relacionais, também conhecidos como bancos de dados NoSQL (Not Only SQL), representam uma categoria de sistemas de armazenamento de dados que diferem significativamente dos tradicionais bancos de dados relacionais. Enquanto os bancos de dados relacionais seguem o modelo tabular, os bancos de dados não-relacionais adotam modelos de dados mais flexíveis e escaláveis, adequados para cenários em que a estrutura dos dados é variável ou desconhecida. Exploraremos os principais aspectos dos bancos de dados não-relacionais e suas vantagens em relação aos sistemas relacionais.



Exemplos de bancos de dados não-relacionais.

Os bancos de dados não-relacionais oferecem uma série de vantagens em relação aos sistemas relacionais, especialmente em cenários onde a escalabilidade, a flexibilidade e o desempenho são prioridades:

- **Flexibilidade de Esquema:**
  Os modelos de dados não-relacionais permitem uma flexibilidade significativamente maior em termos de estrutura de dados. Não é necessário definir um esquema rígido de antemão, permitindo que os dados sejam modelados de forma mais natural e orgânica.

- **Escalabilidade Horizontal:**
  Os bancos de dados não-relacionais são altamente escaláveis e podem lidar com grandes volumes de dados distribuídos em vários servidores. Isso permite escalar horizontalmente o sistema, adicionando mais servidores conforme necessário para lidar com o aumento da carga de dados e usuários.

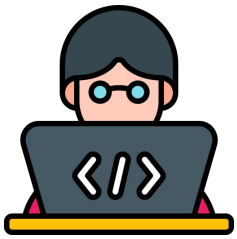
- **Desempenho:**
  Em muitos casos, os bancos de dados não-relacionais oferecem desempenho superior em comparação com os sistemas relacionais, especialmente em cenários de consultas complexas ou de alto volume de dados. Modelos de dados específicos, como o modelo de colunas, podem oferecer desempenho excepcional para determinados tipos de consultas.

- **Suporte a Dados Semi-Estruturados:**
  Os bancos de dados não-relacionais são ideais para lidar com dados semiestruturados, como documentos JSON, que são comuns em muitos aplicativos web modernos. Eles podem facilmente armazenar e consultar esses tipos de dados sem a necessidade de mapeamento para um esquema relacional.

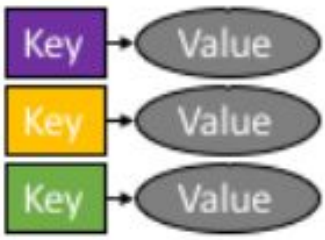
- **Adoção em Big Data e IoT:**
  Com o aumento do volume e da variedade de dados gerados por aplicativos de Big Data e Internet das Coisas (IoT), os bancos de dados não-relacionais tornaram-se uma escolha popular devido à sua capacidade de lidar com dados heterogêneos e de escala massiva.

Bancos de Dados Não-Relacionais

Modelos de Dados Não-Relacionais:



Os bancos de dados não-relacionais suportam uma variedade de modelos de dados, cada um projetado para atender a diferentes necessidades e requisitos de aplicação. Alguns dos modelos de dados mais comuns incluem, mas não estão limitados:

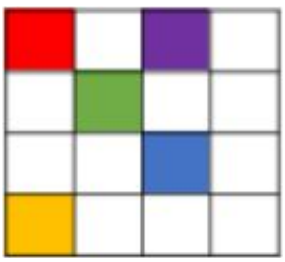


Bancos de Dados de Chave-Valor

Características Principais: Bancos de dados colunares armazenam dados em colunas em vez de linhas, o que os torna eficientes para consultas analíticas que envolvem a análise de um subconjunto de colunas. São ideais para cenários em que é necessário analisar grandes volumes de dados.

Exemplo de Uso:

- Armazenamento de Sessões de Usuários: Usado em sistemas web para armazenar sessões de usuário. Exemplo de software: Redis.
- Cache de Dados: Usado para armazenar em cache dados frequentemente acessados. Exemplo de software: Memcached.
- Gerenciamento de Configurações: Usado para armazenar configurações de aplicativos. Exemplo de software: etcd.



Bancos de Dados Colunar

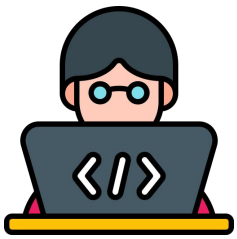
Características Principais: Nesse modelo, os dados são armazenados em colunas em vez de linhas. Isso permite uma compressão eficiente e a recuperação rápida de conjuntos de dados específicos, tornando-os ideais para análises e consultas complexas.

Exemplo de Uso:

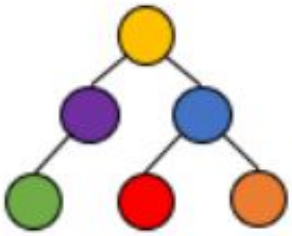
- Análise de Dados: Usado para análise de dados em cenários de business intelligence e data warehousing. Exemplo de software: Amazon Redshift.
- Armazenamento de Dados de Log: Usado para armazenar grandes volumes de dados de log para análise. Exemplo de software: Apache Kudu.
- Processamento Analítico Online (OLAP): Usado para consultas analíticas interativas em tempo real. Exemplo de software: ClickHouse.

Bancos de Dados Não-Relacionais

Modelos de Dados Não-Relacionais:



Bancos de Dados de Documentos

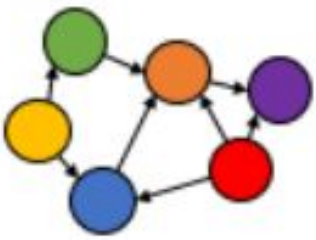


Características Principais: Bancos de dados de documentos armazenam dados em formato de documento, geralmente em JSON ou BSON. Cada documento é autocontido e pode conter campos de diferentes tipos. São flexíveis e escaláveis, adequados para uma ampla variedade de casos de uso.

Exemplo de Uso:

- Aplicações Web: Usado como banco de dados principal em muitas aplicações web modernas. Exemplo de software: MongoDB.
- Gestão de Conteúdo: Usado para gerenciar conteúdo de sites e blogs. Exemplo de software: Couchbase.
- Armazenamento de Perfil de Usuário: Usado para armazenar informações de perfil de usuário em sistemas web e aplicativos móveis. Exemplo de software: Firebase Firestore.

Bancos de Dados de Grafos



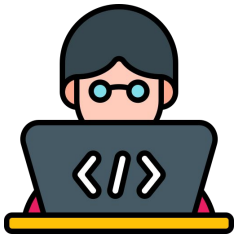
Características Principais: Bancos de dados de grafos são projetados para armazenar e consultar dados que estão estruturados como grafos, compostos por nós, arestas e propriedades. Eles são eficazes para modelar relacionamentos complexos e realizar consultas de grafos, como busca de caminhos e análises de redes.

Exemplo de Uso:

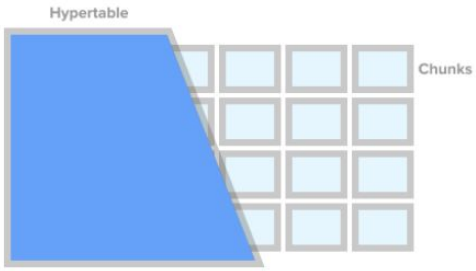
- Redes Sociais: Usado para modelar conexões entre usuários em redes sociais. Exemplo de software: Neo4j.
- Análise de Redes: Usado para analisar relacionamentos em redes complexas, como redes de transporte ou redes de infraestrutura. Exemplo de software: Amazon Neptune.
- Recomendações Personalizadas: Usado para sistemas de recomendação que dependem da análise de relações entre diferentes itens. Exemplo de software: ArangoDB.

Bancos de Dados Não-Relacionais

Outros Modelos de Dados Não-Relacionais:



Bancos de Dados de Séries Temporais



Características Principais: Bancos de dados de séries temporais são otimizados para armazenar e analisar dados que variam com o tempo, como dados de sensores, métricas de desempenho e registros de eventos. Eles fornecem funcionalidades específicas para agregação, interpolação e análise de séries temporais.

Exemplo de Uso:

- Monitoramento de Infraestrutura: Usado para armazenar métricas de desempenho de servidores, redes e aplicativos. Exemplo de software: InfluxDB.
- IoT (Internet das Coisas): Usado para armazenar dados de sensores e dispositivos IoT. Exemplo de software: TimescaleDB.
- Análise Financeira: Usado para análise de dados financeiros, como preços de ações e taxas de câmbio. Exemplo de software: Prometheus.

Cada tipo de banco de dados não-relacional tem suas próprias características e vantagens, e a escolha do modelo adequado depende dos requisitos específicos da aplicação, como flexibilidade na estrutura dos dados, escalabilidade, desempenho e tipos de consultas necessárias. Esses sistemas oferecem uma alternativa valiosa aos bancos de dados relacionais tradicionais, especialmente em cenários onde a estrutura dos dados é variável, ou onde o desempenho e a escalabilidade são críticos.

Espero que tenham compreendido os principais conceitos e diferenças entre os bancos de dados relacionais e não-relacionais. Como em qualquer área de estudo, a dedicação e a prática são essenciais para o aprofundamento do conhecimento.

Portanto, dediquem seu tempo para explorar mais sobre bancos de dados e experimentar com diferentes sistemas e modelos. Lembrem-se de que dominar os fundamentos dos bancos de dados é uma habilidade valiosa que abrirá portas para uma variedade de oportunidades profissionais no campo da tecnologia da informação.

Obrigado e até a próxima aula!