

Arquiteturas de Sistema de Banco de Dados

Eduardo Ogasawara

eduardo.ogasawara@cefet-rj.br
<https://eic.cefet-rj.br/~eogasawara>

Sistemas Centralizados

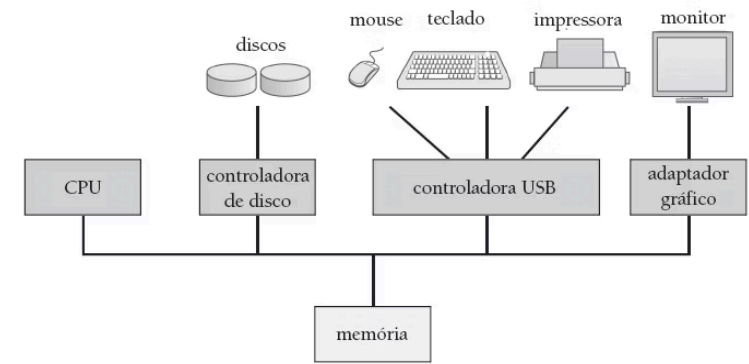
Os sistemas centralizados representam a arquitetura mais tradicional de bancos de dados, onde todo o processamento ocorre em uma única máquina. Estes sistemas executam em um único computador e não interagem com outros sistemas computacionais externos, mantendo todos os recursos e dados consolidados.

Características Fundamentais

Um sistema de computador de uso geral possui uma ou mais CPUs e uma série de controladoras de dispositivo que estão conectadas por meio de um barramento comum, oferecendo acesso compartilhado à memória. Esta arquitetura pode ser configurada tanto como sistema de único usuário quanto como sistema multiusuário.

Sistemas Multiusuário

Os sistemas multiusuário são equipados com várias CPUs e sistema operacional especializado, atendendo a diversos usuários simultaneamente que se conectam por meio de terminais. Estes sistemas são normalmente chamados de sistemas servidores e formam a base para ambientes corporativos que exigem acesso concorrente aos dados.



Sistemas Cliente-Servidor

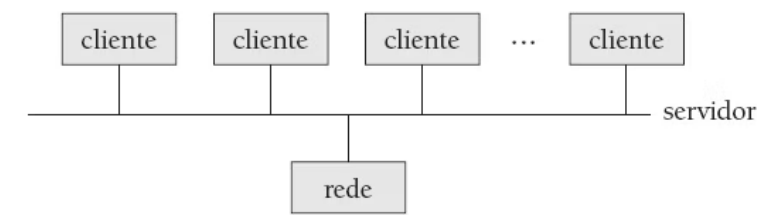
A arquitetura cliente-servidor revolucionou o design de sistemas de banco de dados ao separar as responsabilidades de processamento e interface, permitindo maior flexibilidade e escalabilidade. Esta separação fundamental dividiu a funcionalidade em dois componentes principais que trabalham de forma integrada.

Back-end
Responsável pelo controle das estruturas de acesso, avaliação e otimização de consultas, além do controle de concorrência e mecanismos de recuperação de dados.

Front-end
Consiste em ferramentas como formulários, criadores de relatórios e facilidades de interface gráfica que permitem ao usuário interagir com o sistema.

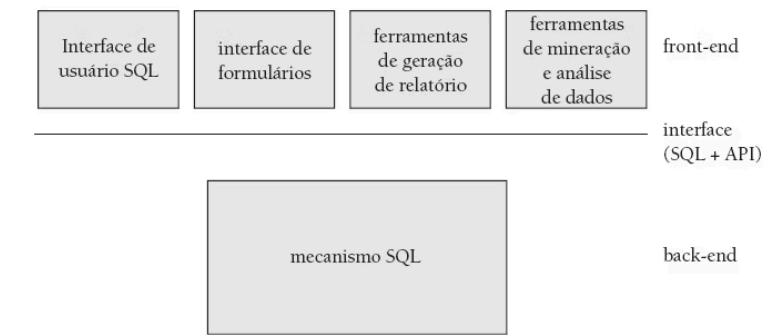
Servidores de Transação

Muito utilizados em sistemas OLTP (Online Transaction Processing), processando múltiplas transações concorrentes com foco em integridade e consistência dos dados.



Servidores de Dados

Otimizados para sistemas OLAP (Online Analytical Processing), focados em consultas analíticas complexas e processamento de grandes volumes de dados.



❏ A interface entre o front-end e o back-end é estabelecida por meio da linguagem SQL ou através de uma interface de programa de aplicação (API), garantindo comunicação padronizada e eficiente.

Servidores de Transação

Os servidores de transação, também conhecidos como sistemas servidores de consulta ou sistemas servidores SQL, representam uma evolução importante na arquitetura cliente-servidor. Nesta configuração, os clientes enviam solicitações ao sistema servidor onde as transações são executadas de forma centralizada, e os resultados são entregues de volta ao cliente de maneira eficiente.

Mecanismo de Comunicação

As solicitações são especificadas em SQL e comunicadas ao servidor por meio de um mecanismo de Remote Procedure Call (RPC), que permite a execução remota de procedimentos. A RPC transacional é particularmente importante, pois permite que várias chamadas de RPC formem coletivamente uma única transação, garantindo atomicidade e consistência das operações distribuídas.

ODBC

Open Database Connectivity é uma interface de programa de aplicação em linguagem C, desenvolvida pela Microsoft, que estabelece um padrão para conectar aplicações a servidores de banco de dados, enviando solicitações SQL e recebendo resultados de forma padronizada e independente de plataforma.

JDBC

Java Database Connectivity é o padrão equivalente ao ODBC para a linguagem Java, oferecendo funcionalidades similares mas otimizado para o ecossistema Java, amplamente utilizado em aplicações corporativas modernas.

Estrutura do Processo Servidor de Transação

A arquitetura interna de um servidor de transação típico é complexa e altamente otimizada para lidar com múltiplas requisições simultâneas. O sistema consiste em vários processos especializados que acessam dados mantidos em memória compartilhada, garantindo eficiência e coordenação adequada.



Processos Servidores

Estes processos recebem consultas dos usuários (transações), executam-nas e enviam os resultados de volta. Os processos podem ser multithreaded, permitindo que um único processo execute múltiplas consultas de usuários simultaneamente, otimizando o uso de recursos. Normalmente, o sistema mantém vários processos servidores multithreaded ativos para balancear a carga de trabalho.



Processo Gerenciador de Bloqueio

Responsável por coordenar o acesso concorrente aos dados, garantindo que as transações não interfiram umas com as outras. Este processo mantém e gerencia as estruturas de bloqueio necessárias para o controle de concorrência.



Processo Escritor de Banco de Dados

Opera continuamente enviando blocos de buffer modificados para os discos de forma assíncrona, garantindo que as alterações sejam persistidas adequadamente e melhorando o desempenho geral do sistema ao separar operações de I/O do processamento de transações.

Processos do Sistema de Transação

A memória compartilhada é o coração da arquitetura de servidores de transação, contendo todas as estruturas de dados críticas que devem ser acessadas por múltiplos processos simultaneamente. Esta organização permite comunicação eficiente e coordenação entre os diversos componentes do sistema.

Estruturas na Memória Compartilhada

- **Pool de Buffer:** Armazena páginas de dados frequentemente acessadas, reduzindo drasticamente a necessidade de acessos a disco
- **Tabela de Bloqueio:** Mantém informações sobre todos os bloqueios ativos no sistema
- **Buffer de Log:** Armazena registros de log antes de serem escritos em disco permanente
- **Planos de Consulta em Cache:** Mantém planos de execução otimizados que podem ser reutilizados se a mesma consulta for submetida novamente

Mecanismos de Sincronização

Todos os processos do banco de dados podem acessar a memória compartilhada, mas é fundamental garantir que dois processos não estejam acessando a mesma estrutura de dados simultaneamente, evitando condições de corrida e inconsistências.

Os sistemas implementam exclusão mútua usando semáforos do sistema operacional ou instruções indivisíveis como testar-e-definir (test-and-set), garantindo acesso coordenado aos recursos compartilhados.

Sistemas Paralelos

Os sistemas de banco de dados paralelos representam um avanço significativo em termos de capacidade de processamento, consistindo em múltiplos processadores e vários discos conectados por uma rede de interconexão de alta velocidade. Esta arquitetura permite processar grandes volumes de dados e consultas complexas de forma muito mais eficiente que sistemas sequenciais.

Classificação por Granularidade

Os sistemas paralelos podem ser classificados de acordo com o número e poder dos processadores utilizados:

Grão Grosso (Coarse-grain)

Consiste em um pequeno número de processadores extremamente poderosos, tipicamente utilizados em aplicações que requerem alto poder computacional por núcleo.

Grão Fino (Fine-grain)

Também chamadas de máquinas maciçamente paralelas, utilizam milhares de processadores menores trabalhando coordenadamente.

Medidas de Desempenho

Vazão (Throughput)

Representa o número de tarefas que podem ser completadas em determinado intervalo de tempo, medindo a capacidade total do sistema.

Tempo de Resposta

Mede o tempo gasto para completar uma única tarefa desde o momento em que ela é submetida até sua conclusão.

Ganho de Velocidade e Ganho de Escala

Duas métricas fundamentais avaliam a efetividade de sistemas paralelos: o ganho de velocidade e o ganho de escala. Estas medidas determinam quão bem um sistema aproveita recursos adicionais para melhorar o desempenho.



Ganho de Velocidade (Speedup)

Mede o quanto mais rápido uma tarefa pode ser executada quando recursos adicionais (processadores) são adicionados ao sistema. Em um cenário ideal, dobrar o número de processadores deveria reduzir o tempo de execução pela metade.

$\text{\$ganho de velocidade} = \frac{\text{"tempo gasto no sistema pequeno"}}{\text{"tempo gasto no sistema grande"}}$



Ganho de Escala (Scaleup)

Avalia a capacidade do sistema de manter o mesmo nível de desempenho quando tanto os recursos quanto o tamanho do problema aumentam proporcionalmente. Um sistema com bom ganho de escala consegue processar o dobro de dados em tempo similar quando o dobro de recursos é adicionado.

$\text{\$ganho de escala} = \frac{\text{"tempo gasto com problema pequeno no sistema pequeno"}}{\text{"tempo gasto com problema grande no sistema grande"}}$

📌 Na prática, tanto o ganho de velocidade quanto o ganho de escala raramente são lineares devido a diversos fatores limitantes que serão explorados no próximo slide. Compreender estas limitações é crucial para projetar sistemas paralelos eficientes.

Fatores Limitando o Ganho de Velocidade e Ganho de Escala

Apesar das vantagens teóricas do paralelismo, o ganho de velocidade e o ganho de escala normalmente são sublineares na prática. Três fatores principais contribuem para esta limitação, reduzindo a eficiência esperada quando recursos adicionais são incorporados ao sistema.

Custos de Partida (Startup Costs)

O custo computacional para iniciar e coordenar vários processos pode dominar o tempo total da computação, especialmente quando o grau de paralelismo é muito alto. Isso inclui o overhead de criação de processos, alocação de recursos, e sincronização inicial. Em tarefas de curta duração, este overhead pode superar os benefícios do paralelismo.

Interferência (Interference)

Os processos acessando recursos compartilhados como barramento do sistema, discos ou estruturas de bloqueio competem entre si constantemente. Esta competição faz com que os processos gastem tempo significativo esperando por recursos ou por outros processos, em vez de realizar trabalho útil, reduzindo drasticamente a eficiência geral do sistema paralelo.

Distorção (Skew)

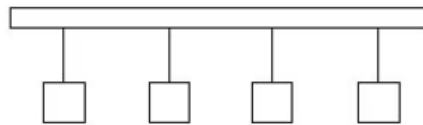
Aumentar o grau de paralelismo aumenta inevitavelmente a variância nos tempos de serviço das tarefas executando em paralelo. Como o tempo de execução geral é determinado pela tarefa mais lenta (gargalo), mesmo que a maioria das tarefas termine rapidamente, o sistema ainda precisa esperar pela conclusão da tarefa mais demorada.

Arquiteturas de Rede de Interconexão

A topologia da rede de interconexão é crucial para determinar o desempenho e a escalabilidade de sistemas paralelos. Diferentes arquiteturas oferecem trade-offs entre custo, desempenho, complexidade e tolerância a falhas.

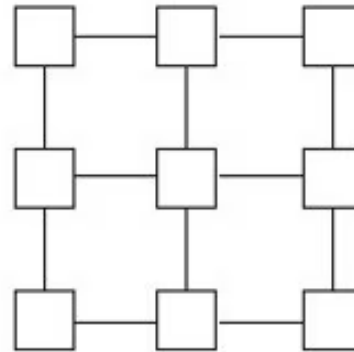
Barramento (Bus)

Todos os componentes do sistema enviam e recebem dados através de um único barramento de comunicação compartilhado. Esta é a topologia mais simples e econômica, mas torna-se um gargalo significativo com muitos processadores, limitando a escalabilidade.



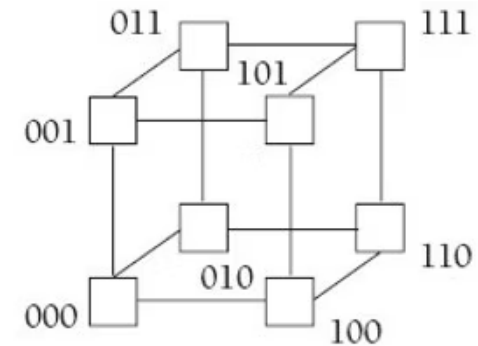
Malha (Mesh)

Os componentes são organizados como nós em uma estrutura de grade bidimensional, onde cada componente é conectado diretamente a todos os seus vizinhos adjacentes. Esta topologia oferece múltiplos caminhos de comunicação e melhor escalabilidade que o barramento.



Hipercubo (Hypercube)

Os componentes são numerados em representação binária e conectados entre si se suas representações binárias diferirem em exatamente um bit. Esta topologia oferece excelentes propriedades de roteamento e diâmetro logarítmico, permitindo comunicação eficiente mesmo em sistemas grandes.

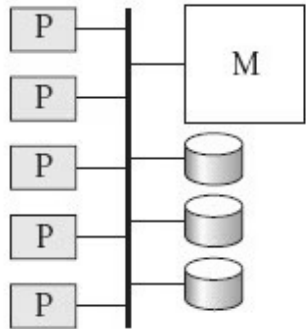


Arquiteturas de Bancos de Dados Paralelas

As arquiteturas de bancos de dados paralelos podem ser classificadas em quatro categorias principais, cada uma com características distintas de compartilhamento de recursos e escalabilidade. A escolha da arquitetura adequada depende dos requisitos específicos de desempenho, custo e escalabilidade da aplicação.

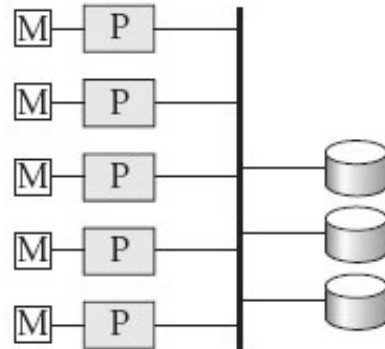
Memória Compartilhada

Processadores compartilham acesso a uma memória comum



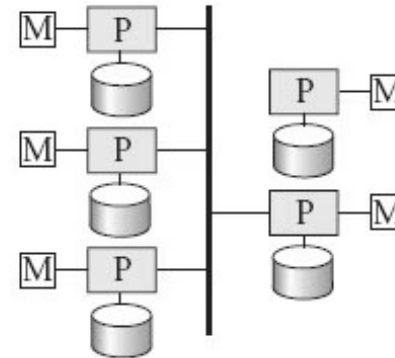
Disco Compartilhado

Processadores compartilham acesso aos mesmos discos



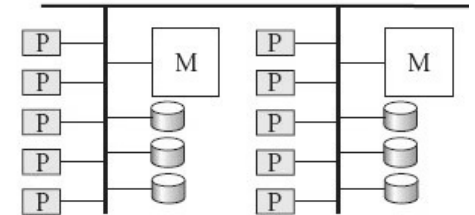
Nada Compartilhado

Processadores não compartilham memória nem discos



Hierárquica

Combinação híbrida das arquiteturas anteriores



Memória Compartilhada

Na arquitetura de memória compartilhada, todos os processadores e discos têm acesso direto a uma memória comum, normalmente conectados por meio de um barramento de alta velocidade ou por uma rede de interconexão especializada. Esta configuração permite comunicação extremamente eficiente entre os processadores.

Vantagens

A principal vantagem desta arquitetura é a comunicação extremamente eficiente entre os processadores. Os dados mantidos na memória compartilhada podem ser acessados por qualquer processador sem a necessidade de movê-los explicitamente usando software, eliminando overhead de comunicação e simplificando significativamente a programação do sistema.

Limitações

A arquitetura não é facilmente expansível além de 32 ou 64 processadores devido a limitações fundamentais. O barramento ou a rede de interconexão eventualmente se torna um gargalo severo quando muitos processadores competem simultaneamente pelo acesso à memória compartilhada.

Disco Compartilhado

A arquitetura de disco compartilhado representa um meio-termo interessante entre memória compartilhada e nada compartilhado. Nesta configuração, todos os processadores podem acessar diretamente todos os discos por meio de uma rede de interconexão, porém cada processador mantém sua própria memória privada local.



Memórias Privadas

Como o barramento da memória não é compartilhado, ele não se torna um gargalo como na arquitetura de memória compartilhada, permitindo melhor escalabilidade em termos de número de processadores.



Tolerância a Falhas

A arquitetura oferece um grau natural de tolerância a falhas: se um processador falhar, os outros processadores podem assumir suas tarefas automaticamente, pois o banco de dados é residente nos discos que são acessíveis por todos os processadores do sistema.

Características de Escalabilidade

Os sistemas de disco compartilhado podem se expandir até um número consideravelmente maior de processadores que sistemas de memória compartilhada, embora a comunicação entre os processadores seja mais lenta e complexa. O gargalo agora ocorre na interconexão com o subsistema de disco, especialmente quando múltiplos processadores tentam acessar os mesmos dados simultaneamente.

Nada Compartilhado

A arquitetura de nada compartilhado (shared-nothing) representa o paradigma mais escalável para sistemas de banco de dados paralelos. Cada nó consiste em um processador completo, memória local dedicada e um ou mais discos exclusivos, formando uma unidade independente de processamento.

Arquitetura e Comunicação

Os processadores em um nó se comunicam com processadores em outros nós exclusivamente através de uma rede de interconexão, utilizando protocolos de mensagens. Cada nó funciona como servidor autônomo para os dados armazenados nos discos que possui, garantindo localidade de dados e minimizando contenção.

Minimização de Interferência

Os dados acessados a partir de discos locais e as operações de acesso à memória local não passam pela rede de interconexão, minimizando drasticamente a interferência causada pelo compartilhamento de recursos entre processadores.



Vantagem

Escalabilidade excepcional - sistemas podem expandir até milhares de processadores sem interferência significativa



Desvantagem

Custo aumentado de comunicação e acesso a dados não locais, requerendo interação de software em ambas as pontas

Hierárquica

A arquitetura hierárquica combina inteligentemente as características das arquiteturas de memória compartilhada, disco compartilhado e nada compartilhado em uma estrutura multinível. Esta abordagem híbrida busca aproveitar os benefícios de cada arquitetura enquanto minimiza suas desvantagens individuais.

Estrutura em Múltiplos Níveis

No nível superior, a arquitetura utiliza o paradigma nada compartilhado, onde os nós estão conectados por uma rede de interconexão e não compartilham discos ou memória entre si. Esta camada garante escalabilidade massiva similar à arquitetura nada compartilhado pura.

Nós de Memória Compartilhada

Cada nó do sistema pode ser internamente um sistema de memória compartilhada com alguns processadores, aproveitando a comunicação eficiente desta arquitetura para processamento local. Esta configuração é ideal para operações que requerem acesso intensivo a dados compartilhados dentro do nó.

Nós de Disco Compartilhado

Alternativamente, cada nó pode ser configurado como um sistema de disco compartilhado, onde múltiplos sistemas que compartilham um conjunto de discos podem ser internamente sistemas de memória compartilhada, criando uma hierarquia de três níveis.

- ❏ Esta arquitetura reduz significativamente a complexidade de programação através do uso de arquiteturas de memória virtual distribuída, também chamada de arquitetura de memória não uniforme (NUMA), que fornece uma abstração de memória compartilhada sobre hardware fisicamente distribuído.

Sistemas Distribuídos

Os sistemas distribuídos representam um paradigma fundamental em computação moderna, onde dados e processamento são espalhados por múltiplas máquinas fisicamente separadas. Estas máquinas, também chamadas de sites ou nós, são interligadas por redes de comunicação que permitem coordenação e compartilhamento de recursos.

Distribuição Geográfica

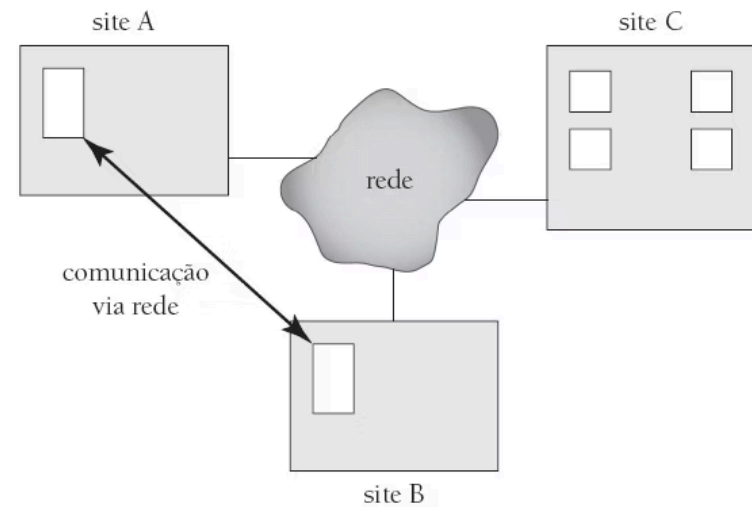
Os dados são espalhados estrategicamente por várias máquinas que podem estar localizadas em diferentes localizações geográficas, desde salas adjacentes até continentes diferentes.

Interconexão de Rede

Uma rede de comunicação robusta interliga todas as máquinas do sistema, permitindo troca de mensagens, sincronização e coordenação entre os componentes distribuídos.

Compartilhamento de Dados

Os dados são compartilhados e acessíveis por usuários conectados a diferentes máquinas no sistema, proporcionando uma visão unificada dos recursos disponíveis.



Bancos de Dados Distribuídos

Os bancos de dados distribuídos podem ser classificados em duas categorias principais baseadas no grau de homogeneidade entre os sites participantes. Esta classificação é fundamental para entender os desafios e benefícios de cada abordagem arquitetural.

Bancos de Dados Homogêneos

Utilizam o mesmo software de gerenciamento de banco de dados e esquema de dados em todos os sites participantes. Os dados podem estar particionados entre os sites de acordo com estratégias de distribuição específicas. O objetivo principal é oferecer aos usuários uma visão transparente de um único banco de dados unificado, ocultando completamente os detalhes da distribuição física dos dados.

Bancos de Dados Heterogêneos

Caracterizam-se por utilizar software de gerenciamento e esquemas de dados diferentes em sites distintos. Esta configuração é comum quando há necessidade de integrar bancos de dados existentes e legados. O objetivo é integrar estes sistemas previamente independentes para oferecer funcionalidade útil e coordenada, mantendo a autonomia individual de cada sistema.

Classificação de Transações

Transações Locais

Acessam dados exclusivamente no site onde a transação foi iniciada, sem necessidade de comunicação com outros sites.

Transações Globais

Acessam dados em site diferente daquele onde foram iniciadas, ou coordenam acesso a dados em múltiplos sites simultaneamente.

Escolhas nos Sistemas Distribuídos

O design de sistemas de banco de dados distribuídos envolve balancear cuidadosamente três objetivos fundamentais que frequentemente apresentam trade-offs entre si. Cada escolha arquitetural tem impactos significativos no desempenho, confiabilidade e complexidade do sistema.



Compartilhamento de Dados

Usuários em um site devem ser capazes de acessar dados residindo em outros sites remotos de forma transparente, como se todos os dados estivessem localmente disponíveis.



Autonomia

Cada site deve ser capaz de reter um grau significativo de controle sobre os dados armazenados localmente, incluindo decisões sobre replicação, particionamento e políticas de acesso.



Redundância

Os dados podem ser estrategicamente replicados em sites remotos, permitindo que o sistema continue funcionando mesmo quando um ou mais sites falhem, aumentando disponibilidade e tolerância a falhas.

Desvantagens e Complexidades

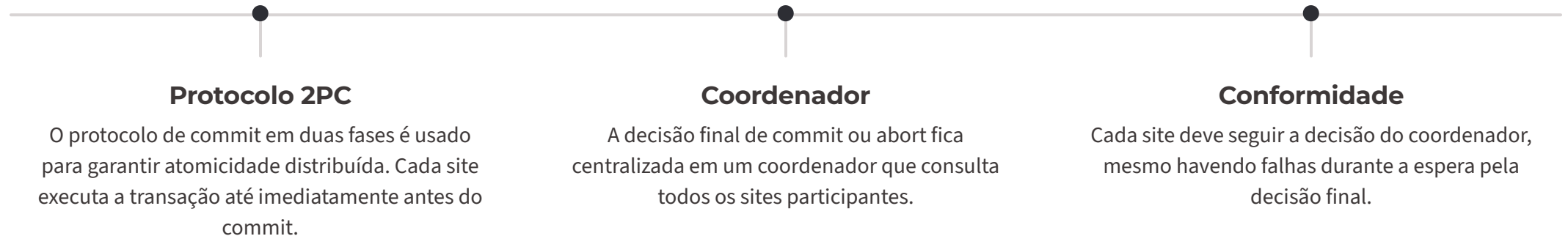
Estes benefícios vêm com custos significativos. A maior complexidade exigida para garantir coordenação apropriada entre os sites se manifesta em diversos aspectos: aumento substancial no custo de desenvolvimento de software, maior potencial para bugs difíceis de detectar e reproduzir, e considerável sobrecarga no processamento para manter consistência e coordenação entre os sites distribuídos.

Questões de Implementação para Bancos de Dados Distribuídos

A implementação de bancos de dados distribuídos apresenta desafios técnicos complexos que não existem em sistemas centralizados. A garantia de propriedades ACID em ambiente distribuído requer protocolos sofisticados e coordenação cuidadosa entre múltiplos sites.

Atomicidade em Ambiente Distribuído

A atomicidade é essencial mesmo para transações que atualizam dados em vários sites geograficamente separados. Uma transação não pode ser confirmada (committed) em um site e abortada em outro, pois isso violaria a consistência do banco de dados distribuído e poderia levar a estados inconsistentes irreversíveis.



O protocolo 2PC nem sempre é apropriado para todos os cenários. Outros modelos de transação baseados em mensagens persistentes e fluxos de transações também são utilizados em contextos específicos. O controle de concorrência distribuído e a detecção de impasses (deadlocks) distribuídos são necessários e significativamente mais complexos que em sistemas centralizados.

📄 A replicação de itens de dados é frequentemente necessária para melhorar a disponibilidade, mas introduz desafios adicionais de manutenção de consistência entre réplicas.

Tipos de Redes

A infraestrutura de rede que conecta os componentes de um sistema distribuído tem impacto fundamental nas características de desempenho, latência, confiabilidade e custo do sistema. As redes podem ser classificadas de acordo com sua escala geográfica e padrão de conectividade.

Redes Locais (LANs)

Compostas de processadores distribuídos por pequenas áreas geográficas, como um único prédio ou alguns prédios adjacentes dentro de um campus. Caracterizam-se por alta largura de banda, baixa latência e alta confiabilidade. Tipicamente utilizam tecnologias como Ethernet ou Wi-Fi com velocidades que podem variar de 100 Mbps a 100 Gbps ou mais.

Redes Remotas (WANs)

Compostas de processadores distribuídos por uma área geográfica ampla, potencialmente abrangendo cidades, países ou continentes. Apresentam latências maiores e menor confiabilidade que LANs, mas permitem conectividade global. A Internet é o exemplo mais proeminente de uma WAN pública.

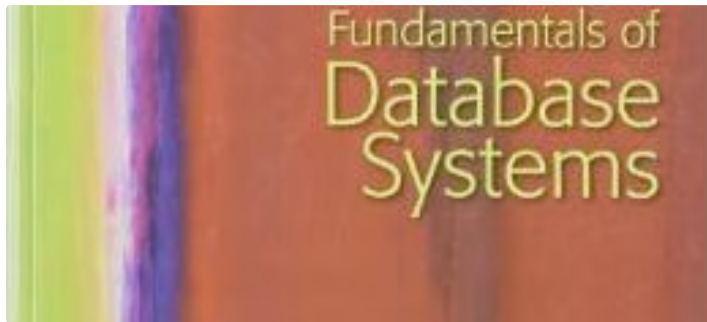
Conexão Descontínua

Refere-se a WANs que estão conectadas apenas parte do tempo, como aquelas baseadas em discagem periódica. Um exemplo histórico é o UUCP (Unix-to-Unix Copy Program), onde sistemas se conectavam periodicamente para trocar dados e mensagens. Este modelo ainda é relevante em cenários de baixa conectividade ou para aplicações que toleram latências altas.

Conexão Contínua

WANs onde os hosts estão conectados à rede continuamente, 24 horas por dia. A Internet moderna exemplifica este modelo, onde dispositivos mantêm conectividade persistente. Esta conectividade constante permite transações em tempo real e acesso imediato aos dados, mas requer infraestrutura mais robusta e custos operacionais mais altos.

Referências



Elmasri & Navathe

Fundamentals of Database Systems

Pearson, 2016

Referência abrangente sobre fundamentos de sistemas de bancos de dados, cobrindo aspectos teóricos e práticos.



Korth, Sudarshan & Silberschatz

Database System Concepts

McGraw-Hill, 2019

Texto fundamental que serviu como base para a maioria dos exemplos apresentados nesta apresentação.



Özsu & Valduriez

Principles of Distributed Database Systems

Springer Nature, 2019

Obra especializada em sistemas de bancos de dados distribuídos, essencial para compreensão avançada.

❏ **Nota:** Os conceitos e exemplos apresentados baseiam-se principalmente na literatura clássica de sistemas de bancos de dados, em especial *Database System Concepts* e *Fundamentals of Database Systems*.