

Capítulo 3

Arquiteturas de Sistemas de BD

Sistema Centralizado

Um sistema centralizado é aquele que corre numa só máquina e não interage com outros computadores.

Composto por 1 host e vários terminais que possuem pouco poder de processamento que ficam conectados a um grande PC (*mainframe*) que possui a capacidade de processar e armazenar dados. Esta arquitetura tem como característica a necessidade de grandes investimentos tanto na aquisição como manutenção.

Vantagens	Desvantagens
<ul style="list-style-type: none">- Os hosts fornecem alto grau de segurança, concorrência e controlo de cópias de segurança e recuperação.- Não há necessidade de um diretório distribuído, já que todos os dados são localizados num único host.- Não existe a necessidade de junções distribuídas, já que todos os dados são localizados num único host.	<ul style="list-style-type: none">- Todos os acessos aos dados realizados por outro que não seja o host onde a BD está, gera alto custo de comunicação.- O host em que a BD está localizada pode criar um “engarrafamento” dependendo da quantidade de acessos simultâneos- Pode acontecer problemas de disponibilidade dos dados se o host onde os dados estão armazenados ficar offline

Existem dois modelos de sistemas centralizados:

1) Single-user System (Utilizador Único) - sistema operativo associado a um único utilizador

2) Multi-user System (Multi Utilizador) - sistema operativo de múltiplos utilizadores, que estão conectados ao sistema através de terminais.

Sistema Cliente-Servidor

O funcionamento deste sistema pode ser dividido em:

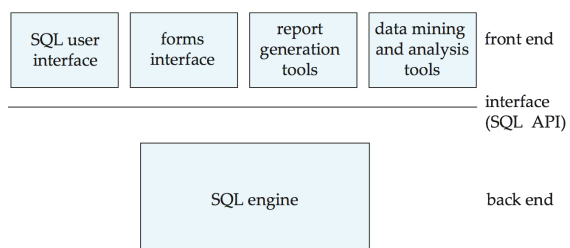
1) Back-end (Servidor) - Gere estruturas de acesso, avaliação de consultas e otimização, controlo de concorrência e recuperação.

2) Front-End (Cliente) - Consiste em ferramentas tais como formulários, relatórios e interface gráfica de utilizador.

A interface entre o front-end e o back-end é através de SQL ou através de uma interface de um programa.

Vantagens de substituição de *mainframes* com redes de estações de trabalho ou computadores pessoais ligados a máquinas de servidores back-end:

- Melhor funcionalidade para o custo
- Flexibilidade na localização de recursos e expansão de instalações
- Melhores interfaces de utilizador
- Manutenção mais fácil



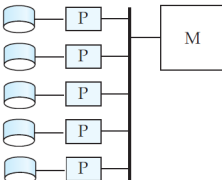
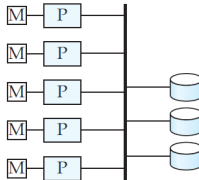
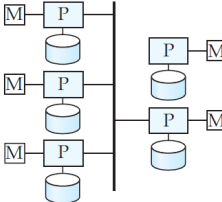
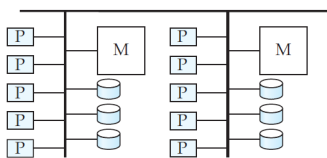
Sistema Paralelo

Num sistema paralelo é utilizado o paralelismo de vários servidores que trabalham simultaneamente sobre as suas respectivas base de dados para a obtenção de um rápido resultado. No paralelismo, os servidores são alinhados e cada um é responsável pelas sub-tarefas de uma base de dados.

• Duas principais medidas de desempenho:

- rendimento - número de tarefas que podem ser concluídas num determinado intervalo de tempo
- tempo de resposta - quantidade de tempo que leva para completar uma única tarefa a partir do momento em que são fornecidas

Arquiteturas das Base de Dados Paralelas

Memória Partilhada	Disco Partilhado
<p>- Processadores e discos acedem a uma memória comum, através de uma rede de interconexão</p> <p>- Comunicação extremamente eficiente entre processadores - dados na memória compartilhada podem ser acedidos por qualquer processador sem precisar movê-lo, utilizando o software.</p> <p>- <i>Desvantagem</i>: Arquitetura não-escalável</p> 	<p>- Todos os processadores podem aceder diretamente a todos os discos através de uma rede de interconexão, mas os processadores têm memórias privadas</p> <p>- Arquitetura tolerante a falhas - se um processador falhar, os outros processadores podem assumir as suas funções, desde que a base de dados seja residente num dos discos que estejam acessíveis aos processadores</p> <p>- <i>Desvantagem</i> - Ocorre engarrafamento na interligação ao subsistema de disco</p> <p>- Podem ser dimensionado para um número maior de processadores, mas a comunicação entre processadores é mais lenta.</p> 
Nenhuma Partilha	Hierárquica
<p>- Nó é composto por um processador, memória e um ou mais discos. Os processadores num nó comunicam-se com outro processador em outro nó, usando uma rede de interconexão. Um nó funciona como um servidor para os dados no disco ou discos que possui o nó.</p> <p>- Escalável até milhares de processadores sem interferência.</p> <p>- <i>Desvantagem</i> - custo de comunicação e acesso ao disco não-local; o envio de informação envolve a interação de software em ambos os pontos.</p> 	<p>- Combina as características das arquiteturas de memória partilhada e de disco partilhado</p> <p>- Nível superior é atingido numa arquitetura sem partilha, onde os nós estão conectados numa rede de interconexão, e não partilham disco e memória com o outro.</p> <p>- Cada nó do sistema podia ser um sistema de memória partilhada com alguns processadores.</p> <p>- Alternativamente, cada nó pode ser um sistema de disco partilhado, e cada um dos sistemas de partilha de um conjunto de discos pode ser um sistema de memória partilhada.</p> 

Base de Dados Distribuídas

Consiste em múltiplos servidores, cada um responsável pelos seus dados. Estes dados podem ser acedidos utilizando uma rede, e apesar de estarem fisicamente distribuídos, devem apresentar-se ao utilizador logicamente integrados, como se fossem um único servidor.

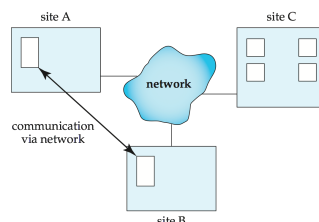
Um sistema de bases de dados distribuídas consiste em múltiplos servidores, cada um responsável pelos seus dados. Estes dados podem ser acedidos utilizando uma rede e, apesar de estarem fisicamente distribuídos, devem apresentar-se ao utilizador logicamente integrados, como se estivessem num único servidor.

A **principal razão** para utilizar base de dados distribuídos é o facto do problema em análise ter natureza distribuída. Por exemplo: Uma organização tem sede em Coimbra, uma fábrica A em Braga, um armazém B em Aveiro e uma fábrica C em Évora, pode fazer sentido utilizar uma BDD, pois espelha a estrutura da organização.

Não se devem utilizar quando não faz sentido ter um servidor em vários locais. Por exemplo: Uma empresa em sede na zona alta de Coimbra e três lojas na baixa, não faz sentido ter um servidor em cada local. Portanto se o problema não for de natureza distribuída, não se deve utilizar uma BDD. A solução mais adequada aqui era cliente/servidor.

Cada servidor que participa na BDD tem uma autonomia local, ou seja, é administrado separado e independentemente dos outros servidores. Esta autonomia local tem várias vantagens:

- Os nós dos sistemas podem espelhar mais facilmente a estrutura lógica das organizações
- Os dados locais são controlados por um administrador local, pelo que o domínio de administração é menor e mais maneável
- A recuperação de falhar pode, em muitos casos, ser efetuada numa base estritamente local
- Os nós do sistema podem ter a dimensão mais adequada a cada problema local e crescer independentemente.



BDD Homogéneas	BDD Heterogéneas
<ul style="list-style-type: none">- Todos os sites têm um software igual- Estão conscientes uns dos outros e acordam em cooperar no processamento dos pedidos dos utilizadores- Cada site aceita perder parte da sua autonomia em termos de direitos de mudança de esquema e software- <u>Objetivo</u>: Dar a noção ao utilizador que o sistema é único	<ul style="list-style-type: none">- Sites diferentes podem usar esquemas e softwares diferentes<ul style="list-style-type: none">- diferenças nos esquemas constituem problemas complicados no processamento de consultas- diferenças no software constituem problemas complicados no processamento de transações- Os sites podem não estar conscientes uns dos outros e poderão fornecer apenas capacidades limitadas para a cooperação no processamento de transações- <u>Objetivo</u>: Integrar base de dados existentes para fornecer funcionalidade útil

- Diferença entre transações locais e globais:
 - Transações Local - Uma transação local acede a dados no local onde a transação foi iniciada
 - Transações Global - Uma transação global pode aceder a dados em localizações diferentes daquelas onde a operação foi iniciada

Vantagens BDD	Desvantagens BDD
<ul style="list-style-type: none"> - Partilha de dados - os utilizadores podem aceder a dados residentes num site diferente - Autonomia - cada site pode manter um nível de controlo sobre os dados guardados localmente - Maior disponibilidade do sistema através da redundância - como há replicação dos dados, se um site falha o sistema continua ativo 	<ul style="list-style-type: none"> - A complexidade aumenta pois é necessário assegurar uma coordenação correta entre os sites, o que resulta em: <ul style="list-style-type: none"> - maior custo no desenvolvimento do software - maior possibilidade de bugs - aumenta a carga do processamento
<p align="center">Vantagens da autonomia de cada servidor da BDD</p> <p>As BDDs estão divididas por diversas sub-redes, cada sub-rede contém um servidor que funciona de modo autónomo, isto é, cada servidor pode executar tarefas independentemente dos outros servidores, mas o resultado final vai ser obtido juntando a informação que cada servidor contém.</p>	

Soluções implementadas

- Atomicidade dos dados em dois ou mais nós
- Protocolo de duas fases:
 - Cada transação é executada e só aceite por um coordenador
 - Cada localização segue a decisão de um coordenador

Tipos de Rede

- **Local-area networks (LANs)**
 - Constituída por processadores que estão distribuídos em pequenas áreas geográficas, como um único edifício ou alguns edifícios adjacentes.
 - Maior largura de banda e Baixa Latência
- **Wide-area networks (WANs)**
 - Constituída por processadores distribuídos numa grande área geográfica.
 - Baixa largura de banda e Maior latência

Indicadores de Desempenho

- Capacidade: O número de tarefas que são concluídas num determinado intervalo de tempo
- Tempo de resposta: A quantidade de tempo que leva para completar uma única tarefa a partir do momento em que são fornecidas

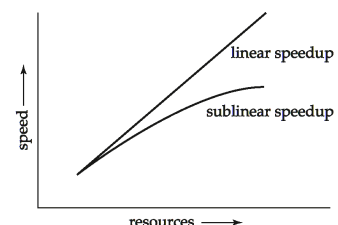
Acelerar e Aumentar (Speed-Up e Scale-Up)

• Acelerar - Speed-Up

- Um problema é executado num sistema pequeno e é recolhido por um sistema N vezes maior.

Medido por: $\text{speedup} = \frac{\text{tempo decorrido no sistema menor}}{\text{tempo decorrido no sistema maior}}$

- Speed-Up é linear se a equação for igual a N.

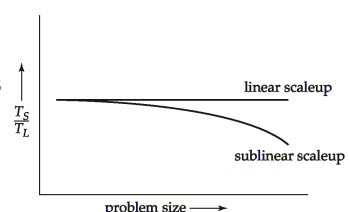


• Aumentar - Scale-Up

- Aumenta tanto o tamanho do problema como o de sistema
- Sistema N vezes maior para realizar operações N vezes maiores

Medido por: $\text{scaleup} = \frac{\text{tempo decorrido no problema menor}}{\text{tempo decorrido no sistema maior}}$

- Scale-Up é linear se a equação for igual a 1.



- Batch Scale-Up

- Tarefa grande, típico na maioria das consultas de apoio à decisão e simulação científica.
- Necessário usar um computador N vezes maior em problemas N vezes maiores

- Transaction Scale-Up

- Consultas submetidas por utilizadores independentes a uma base de dados partilhada
- Adequado para execução paralela

• Fatores Limitantes

Speed-Up e Scale-Up são frequentemente sublinear devido a:

- Custos de iniciação: custo de iniciar vários processos pode dominar o tempo de computação, se o grau de paralelismo for alto.
- Interferência: Processos que acedem a recursos compartilhados
- Inclinação: Aumentar o grau de paralelismo aumenta a variância em tempos de serviço de tarefas em execução paralelamente. Tempo total de execução é determinado pela mais demorada das tarefas paralelas em execução.

Estrutura de Base de Dados Distribuídas

Problema: Tomar decisões sobre a localização de dados e programas nos nós de uma rede de computadores, assim como possivelmente projetando a rede em si.

Sistema Gestão Base Dados Distribuídas (SGBDD)

Nos SGBDD, a localização das aplicações engloba:

- Localização do software SGBDD
- Localização das aplicações que executam sobre a base de dados

Distribuição dos Dados

• **Nível de partilha:**

- sem partilha: Cada aplicação e os seus dados são executados em um site e não há comunicação com qualquer outro programa ou acesso aos dados dos outros sites
- partilha de dados: Todos os programas são replicados em todos os sites, mas os dados não. Deste modo as solicitações do utilizador são manipuladas no site de onde são originárias e os dados são movidos pela rede
- partilha de dados + programa: Ambos os dados e o programa podem ser partilhados independentemente do site.

• **Padrões de Acesso:**

- estático: os padrões de acesso de pedidos de utilizador não mudam ao longo do tempo
- dinâmico: os padrões de acesso de pedidos de utilizador podem mudar ao longo do tempo

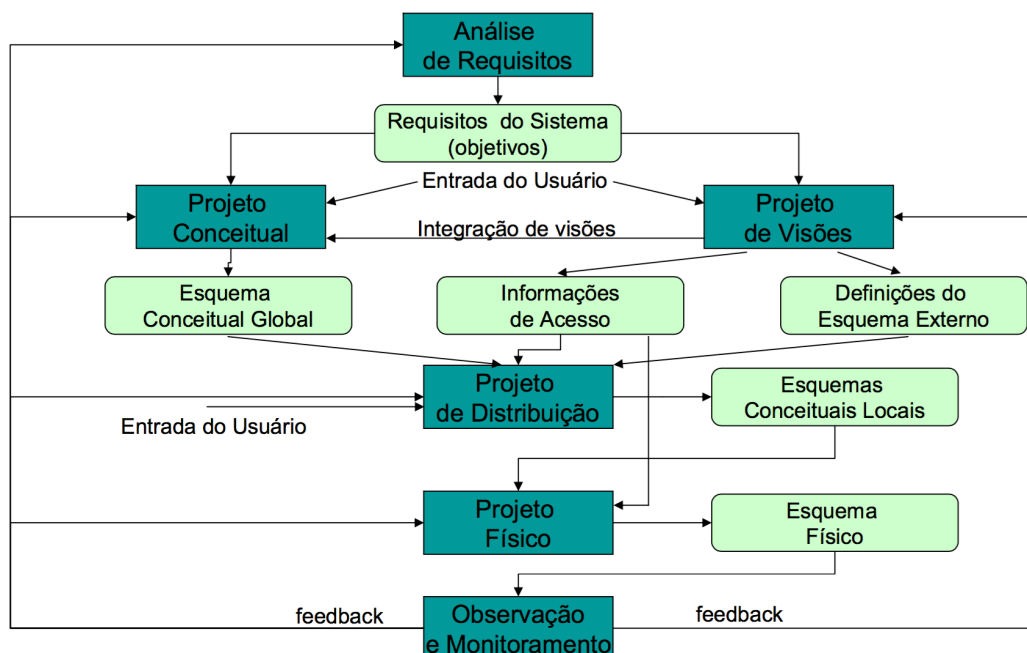
• **Nível de Conhecimento dos Padrões de Acesso:**

- nenhuma informação: os designers não têm nenhuma informação de como os utilizadores irão aceder à base de dados (muito improvável de acontecer)
- informação completa: os designers têm a informação completa, ou seja, os padrões de acesso podem ser razoavelmente previstos e não se desviam significativamente destas previsões
- informação parcial: onde existem desvios em relações às previsões

Estratégias para a Estrutura da BDD

Top-Down

- É uma abordagem mais adequada para SGBDs integrados e distribuídos homogeneamente
- A atividade começa com uma análise de requisitos que define o ambiente do sistema e deduz as necessidades de dados e processamento de todos os utilizadores potenciais da BD
- O estudo de requisitos também especifica onde o sistema final deve ficar com respeito aos objetivos de um SGBDD. Estes objetivos são definidos tendo em conta a performance, fiabilidade, disponibilidade, economia e flexibilidade
- O documento de requisitos é a entrada para duas atividades paralelas:
 - vista do design - definir a interface para os utilizadores finais
 - design conceptual - processo pelo qual a empresa é examinada para determinar os tipos e as relações das entidades. Este processo por ser dividido em dois grupos de atividades relacionadas:
 - análise de entidades - determinação, atributos e relação das entidades
 - análise funcional - funções fundamentais
- O esquema concetual global e a informação do padrão de acesso recolhida como resultado da visão do design são entradas para a etapa design da distribuição
- O objetivo nesta fase, que é a fase deste capítulo, é desenhar o esquema concetual local distribuindo as entidades pelos sites do sistema distribuído
- As entidades correspondem a relações
- Em vez de distribuir relações, dividimos em sub-relações, a que se chama de fragmentos, que estes sim irão ser distribuídos
- Assim a atividade do design de distribuição consiste em dois passos:
 - Fragmentação
 - Alocação e Replicação
- O último passo é o design físico, que mapeia o esquema concetual local para os dispositivos de armazenamento físico disponíveis nos sites correspondentes
 - A entrada para este processo são:
 - Esquema concetual local e o padrão de acesso
- O resultado é feedback que pode resultar em fazer backup numa das etapas anteriores no design



Fragmentação

Processo de divisão de tuplos das tabelas em duas ou mais tabelas (fragmentos)

- **Informação necessária:**

- Informações da BD
- Informações da Aplicação
- Informações da Comunicação da Rede
- Informações do Sistema do Computador

- **Informação usada para decidir fragmentação:**

Quantitativa	Qualitativas
<ul style="list-style-type: none">- Localização do Site- Frequência de Consultas- Onde a consulta (query) é executada- Queries seletivas	<ul style="list-style-type: none">- Tipo de acesso a dados<ul style="list-style-type: none">- Ler, escrever

- **Porquê fragmentar?**

Vantagens	Desvantagens
<ul style="list-style-type: none">- Fiabilidade- Performance- Custos de Comunicação- Capacidade de Armazenamento e Custo- Segurança	<ul style="list-style-type: none">- Se os fragmentos não são mutuamente exclusivos- Degradação da performance nos <i>joins</i>

- **Fragmentos de relações como unidades de distribuição**

- Aplicações precisam de um sub-conjunto de relações
- Transações podem ocorrer ao mesmo tempo
- Execução paralela de uma única query
- Fragmentos pequenos - melhor desempenho
- Degradação do desempenho dos *joins*
- Controlo da semântica dos dados é mais difícil

- **Tipos de Fragmentação**

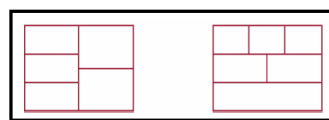
Horizontal	Vertical
Os tuplos de uma relação são divididos em sub-conjuntos e cada sub-conjunto é alocado a uma BD diferente (mantém as colunas)	Produz um conjunto de relações distintas a partir da mesma relação



(a) Horizontal Fragmentation



(b) Vertical Fragmentation



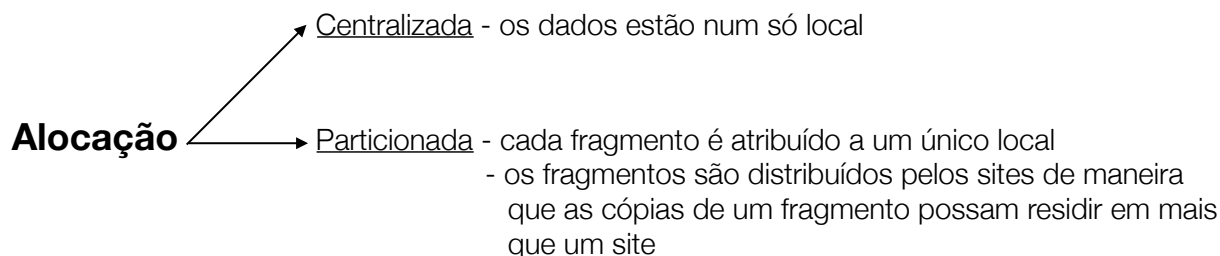
(c) Mixed Fragmentation

• Regras de Correção da Fragmentação / Regras de Semântica

Completeness	Decomposição de uma relação R em fragmentos R_1, R_2, \dots, R_n é completa se cada item dos dados em R pode ser encontrado em pelo menos um fragmento R_i
Reconstrução	Verifica se é possível reconstruir a tabela original a partir da fragmentação resultante
Disjunção	Se uma relação R é decomposta em fragmentos R_1, R_2, \dots, R_n , e o item de dados d_i está em R_j , então d_i não está em qualquer outro fragmento. Ou seja, não há dados repetidos (excepto nas PK da Fragmentação Vertical)

Alocação

- Assumindo que a base de dados está fragmentada corretamente, é necessário decidir sobre a alocação dos fragmentos nos vários sites da rede.
- Quando são alocados, podem ser replicados ou mantidos como cópia única.
- As razões para a replicação são:
 - fiabilidade
 - eficiência de queries "read-only"
- No entanto, a execução de "update queries" causam problemas, uma vez que o sistema tem de assegurar que todas as cópias dos dados são atualizadas corretamente.
- Uma base de dados não replicada contém fragmentos que são alocados nos sites e só existe uma cópia de cada fragmento na rede.
- Na execução da alocação deve-se ter em conta dois aspetos:
 - custo mínimo
 - performance - minimizar o tempo de resposta e maximizar o rendimento do sistema em cada site



Replicação

- O sistema mantém múltiplas cópias de dados, guardando em diferentes sites para recolha mais fácil e tolerância a falhas. O fragmento pode ser replicado ou único.
- Uma relação ou fragmento está replicado se está guardado redundantemente em dois ou mais sites

A replicação pode ser caracterizada em:

- replicação total: Os fragmentos estão disponíveis em todos os sites
- replicação parcial: Alguns fragmentos podem ser replicados enquanto outros não, ou seja, um fragmento disponível em alguns sites
- sem replicação: Um fragmento alocado em apenas um site

Vantagens	Desvantagens
<ul style="list-style-type: none"> - Disponibilidade (no caso de falhas) - Paralelismo - Redução de dados a transferir 	<ul style="list-style-type: none"> - Aumento do custo de atualizações - Aumento do controlo de concorrência (atualizações concorrentes em réplicas podem levar a inconsistência de dados)

Botton-Up

- Processo de integração de base de dados já existentes e independentes
- As BDs devem estar ligadas entre si para realizar tarefas entre si
- Razões:
 - Migração entre plataformas
 - Mudança organizacionais
 - Evolução da tecnologias