Sistemas Distribuidos e Tolerância a Falhas

Gonçalo Amador
Ricardo Alexandre

Departamento de Informática
Universidade da Beira Interior

Bases de Dados Distribuídas

1. **Modelos de Bases de Dados**
2. **Conceitos em Bases de Dados Distríbuidas (BDDs)**
3. **12 Regras das BDD**
4. **Sistema de Gestão de Bases de Dados Distrubuídas (SGBDD)**
   - Características dos SGBDD
   - Funções de um SGBDD
5. **Quando se devem utilizar ou não BDDs?**
6. **Vantagens da autonomia de cada servidor da BDD**
7. **Vantagens e Desvantagens das BDDs**
8. **Noções importantes em BDDs**
   - Objectivo
   - Noções no desenho de uma BDDs
9. **Arquitecturas**
   - BDDs homogéneas
   - BDDs heterogéneas

- Dificuldades da integração total em SGBDs heterogéneos
- Arquitectura Cliente/Servidor

10 **Transacções**
- Tipos de transacções e arquitectura
- Transacções distribuídas

11 **Protocolos de COMMIT**
- Protocolo two-phase commit (2PC)
- Protocolo three-phase commit (3PC)

12 **Transparência**
- Transparência tipos
- Transparência de transacção
- Transparência de Distribuição
- Transparência de nomeação
- Transparência de Desempenho

13 **Armazenamento de dados distribuídos**

Modelos de Bases de Dados

- Modelo Hierárquico e de Rede (1ª geração);
- Modelo Relacional (2ª geração);
- Modelo Relacional Estendido, Modelo Lógico/Dedutivo e Orientado a Objectos (3ª geração).

Exemplos de casos de uso:

1. Sistemas de Informação Geográfica;
2. Aplicações médicas;
3. Aplicações científicas;
4. Sistemas CAD/CAM;
5. Sistemas Multimédia.

Modelos de Bases de Dados (Cont.)

Bases de dados Relacionais (BDRs) VS Bases de dados Orientadas a Objectos (BDOOs)

- **As BDRs e as BDOOs possuem características distintas mas servem o mesmo propósito:**
  - Persistência dos dados necessários para a manutenção de um negócio para o qual são aplicados;
  - Possibilitam a recuperação, comparação e tratamento desses dados a fim de produzir resultados tangíveis.

Modelos de Bases de Dados (Cont.)

Bases de dados Relacionais (BDRs) VS Bases de dados Orientadas a Objectos (BDOOs)

- Nas BDRs, têm-se uma colecção de tabelas, todas com nomes únicos, podendo estas estar relacionadas com uma ou mais tabelas. Conceitos como integridade referencial de dados e chaves primárias, garantem que um conjunto de informações possa ser representado de maneira consistente, independentemente da forma de acesso.
- **Nas BDOOs existem três pilares principais:** herança, polimorfismo e encapsulamento. Este modelo apresenta maior flexibilidade na manipulação do seu contéudo, manipulando por meio de identificadores de objectos os dados de forma consistente.

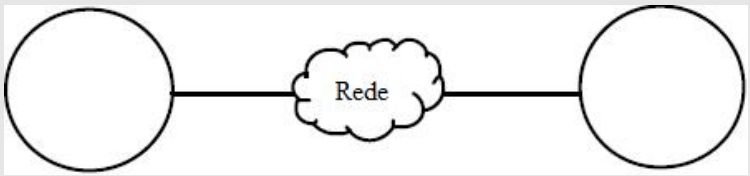
Modelos de Bases de Dados (Cont.)

Modelo Orientado a Objectos

- **O modelo OO, combina:**
  - Bases de Dados (segurança, integridade, etc);
  - Modelos de dados semânticos (generalização, agregação, etc);
  - Linguagens de programação OO (objectos, classes, métodos, encapsulamento, etc).

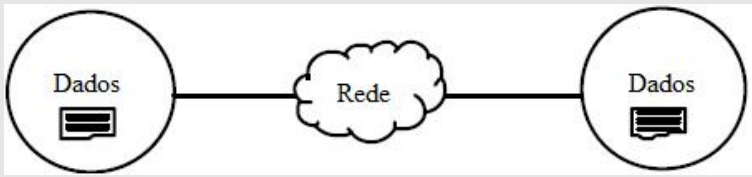
Bases de Dados Distríbuidas (BDDs) Conceitos

Sistema Distríbuido (SD):



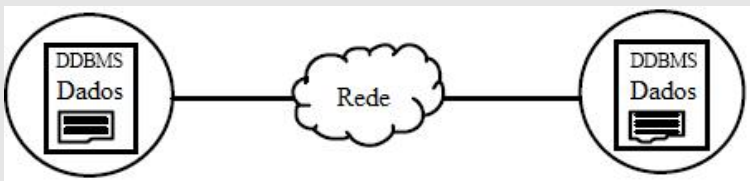
Nós (locais) interconectados através de uma rede de computadores.

Bases de Dados Distríbuidas:



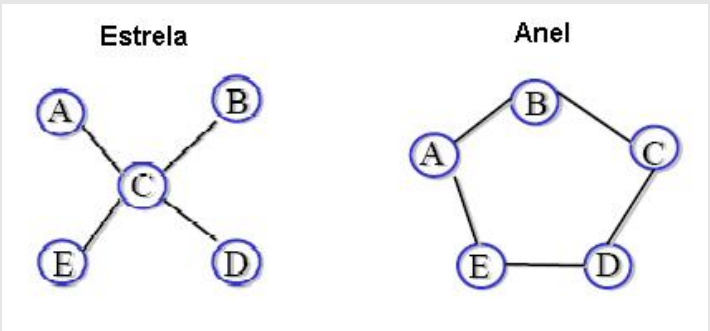
Colecção lógica interrelacionada de dados partilhados, distribuídos fisicamente por uma rede de computadores.

Sistema de Gestão de Bases de Dados Distríbuidas (SGBDD):



Permite a gestão de BDDs e faz com que a distribuição seja transparente para o utilizador.

Topologias de rede utilizadas



Dependem essencialmente de:

- Custos de instalação;
- Custos de comunicação;
- Fiabilidade;
- Disponibilidade.

12 Regras para ter uma BDD

Princípio fundamental: Para o utilizador, um SD deve parecer igual a um sistema centralizado.

1. Autonomia local;
2. Nenhuma referência a um local central;
3. Operação contínua;
4. Independência de localização;
5. Independência de fragmentação;
6. Independência de replicação;
7. Processamento distribuído de consultas;
8. Processamento distribuído de transacções;
9. Independência do hardware;
10. Independência do sistema operativo;
11. Independência da rede;
12. Independência da base de dados.

Características dos SGBDD

- **É um conjunto de dados partilhados logicamente relacionados:**
  - Os dados são a informação presente nas tabelas constituintes da base de dados.
- **Os dados são divididos em fragmentos:**
  - Fragmentos são partes, duma das diversas tabelas da BDD, distribuidos pelos nós da BDD. Nos nós em que se encontram são tabelas com existência Física.

Características dos SGBDD (Cont.)

- Os fragmentos podem ser replicados;
- Os fragmentos/réplicas podem ser alocados nos locais (zonas de armazenamento);
- Os locais estão ligados através de uma rede;
- Os dados em cada local estão sob o controlo de um SGBD;
- Os SGBD tratam de aplicações locais de forma autónoma;
- Gere as transacções distribuídas.

Características dos SGBDD (Cont.)

- Uma BDD para um utilizador aparece como uma única BD, mas é, na verdade, um conjunto de BDs armazenadas em vários computadores;
- Os dados sobre vários computadores podem ser acessados simultâneamente e modificados através de uma rede;
- Cada BDD é controlada por um SGBD local, e cada um colabora para manter a coerência global da BDD.

Características dos SGBDD (Cont.)

Clientes, servidores, e nós:

- Um SGBDD é o software de gestão da base de dados, um cliente é uma aplicação que solicita informações a partir de um servidor;
- Cada computador tem um sistema e um nó;
- Um nó num sistema de base de dados pode ser distribuído por um cliente, um servidor, ou por ambos.

Funções de um SGBDD

Além das funcionalidades de um SGBD um SGBDD tem ainda:

- Serviços de comunicação;
- Extensão do dicionário de dados;
- Processamento de consultas distribuídas;
- Controlo de Concorrência extendido;
- Serviços de recuperação extendidos.

Quando se devem utilizar ou não BDDs?

- **Devem-se utilizar** quando o problema em análise é de natureza distribuída. **Por exemplo:** Uma organização tem a 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 tem sede na zona alta de Coimbra e 3 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 base de dados distribuída.

Vantagens da autonomia de cada servidor da BDD

As BDDs estão divididas por diversas sub-redes, cada sub-rede contém um servidor que funciona de modo autonómo, 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.

- Os nós dos sistemas podem espelhar mais facilmente a estrutura lógica das organizações onde se inserem;
- Os dados locais são controlados por um administrador local (manutenção particionada);
- A recuperação de falhas pode, em muitos casos, ser efectuada numa base estrictamente local;
- Os nós do sistema podem ter uma dimensão adequada a cada problema local e crescer independentemente.

Vantagens e Desvantagens das BDDs

Vantagens

- Segurança;
- Partilha dos dados e autonomia local;
- Desempenho, face as BDs, melhorado. (Pesquisas por pequenas BDs + tempo comunicação VS Pesquisa global duma BD);
- Fiabilidade/disponibilidade melhoradas;
- É mais natural para representar varias organizações do mundo real.

Vantagens e Desvantagens das BDDs

Desvantagens

- Custos de desenvolvimento do Software;
- Maior possiblidade de erros no Software;
- Aumento da carga de processamento;
- Aumento da complexidade para assegurar a coordenação adequada dos nós.

Noções importantes

Comandos Remotos ou Distribuídos (quais são e de que forma são feitos):

- Interrogação, actualização.

Especificar como as relações vão ser armazenadas:

- Duplicação (Maior disponibilidade, aumenta o paralelismo e a sobrecarga em actualizações);
- Fragmentação (Horizontal ou Vertical).

Processar consultas passa por considerar:

- Custo de transmissão de dados na rede;
- Existe um ganho potêncial em desempenho.

Objectivo

O grande objectivo das BDDs consiste em que, para o utilizador, um sistema distribuído pareça exactamente igual a um sistema centralizado:

- Transparência;
- Autonomia local;
- Operações contínuas;
- Não existe um nó central;
- Gestão de transacções distribuídas;
- Processamento distribuído das ”Queries”;
- Independência de Localização, de fragmentação e de duplicação;
- Independência do Hardware, do Sistema operativo e da Rede.

Noções no desenho de uma BDDs

- Como podem as BDD e as aplicações serem colocadas nos locais?
- Como podem os dados estacionários serem colocados nos locais?
- **Considerações sobre:**
  - **Fragmentação:** Divide uma relação em subrelações:
    - A fragmentação está ligada à forma de como uma tabela pode ser dividida pelos nós constituintes da BDD.
  - **Replicação:** Mantém cópias dos fragmentos;
  - **Alocação:** Onde armazenar as relações/fragmentos.

BDDs homogéneas

- Sistemas onde a BDD é composta por várias Bases de Dados idênticas e distribuídas por uma Rede com equipamentos de igual arquitectura;
- No seu conjunto o sistema apresenta ao utilizador a concepção de um ”schema”Interno Global.

BDDs heterogéneas

- Cada nó pode conter um ou mais SGBDs locais;
- Partilham BDs pré existentes com esquemas conceptuais diferentes;
- **Vantagens:**
  - Preserva-se o investimento existente em hardware, software de sistema e aplicações;
  - Autonomia local e controlo administrativo;
  - Permite o uso de SGBDs dedicados;
  - É uma passo no sentido de uma unificação homogénea de SGBDs.

Dificuldades da integração total em SGBDs heterogéneos

- Dificuldades técnicas e custo de conversão;
- **Dificuldades organizacionais/políticas:**
  - As organizações não querem perder o controlo dos seus dados;
  - As bases de dados locais querem manter um elevado grau de autonomia.

Arquitectura Cliente/Servidor

Além das arquitecturas mencionadas anteriormente, existe também a arquitectura cliente/servidor.

Esquema Centralizado - Servidor de nomes:

- **Estrutura:**
  - O servidor de nomes atribui todos os nomes;
  - Cada local mantém um registo dos itens de dados locais;
  - Os locais pedem ao servidor de nomes para localizar itens de dados não locais.

Arquitectura Cliente/Servidor (Cont.)

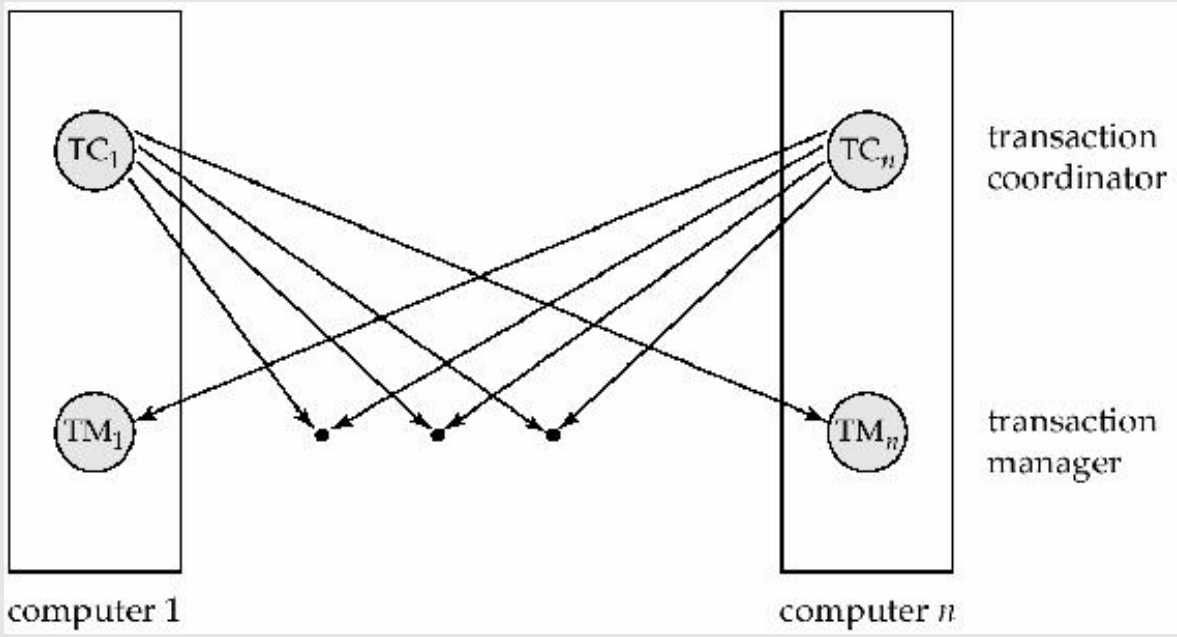
- **Vantagens:**
  - Satisfaz as regras 1 a 3.
- **Desvantagens:**
  - Não satisfaz a regra 4;
  - O servidor de nomes constitui potencialmente um bottleneck em termo de desempenho;
    - **Bottleneck:** é um fenómeno onde a realização ou a capacidade total de um sistema são severamente limitadas por um único componente.
  - O servidor de nomes é um ponto de falha único.

Arquitectura Cliente/Servidor (Cont.)

Uso de pseudónimos: (Aliases)

- **Alternativa ao esquema centralizado:** cada local prefixa com o seu identificador todos os nomes de itens de dados por si gerados:
  - Satisfaz o requisito de identificador único e evita problemas relacionados com controlo central;
  - Não permite obter transparência de rede.
  - **Solução:** Criar um conjunto de pseudónimos para itens de dados; Guardar a correspondência de pseudónimos para nomes reais em cada local.
  - O utilizador pode não ter consciência da localização física dos itens de dados, não é problemático se um item de dados é transferido de um local para outro.

Tipos de transacções e arquitectura



- **Cada local tem um gestor de transacções local responsável por:**
  - Manter um log para efeitos de recuperação;
  - Participar na coordenação da execução concorrente de transacções nesse local.

Tipos de transacções e arquitectura (Cont.)

- **Cada local tem um coordenador de transacções, que é responsável por:**
  - Iniciar a execução de transacções originadas no local;
  - Distribuir subtransacções apropriadas para execução por local;
  - Coordenar a conclusão de cada transacção originada no local.

Transacções distribuídas

- Uma transacção distribuída termina com commit em todos os servidores que participam na transacção, ou então termina com rollback em todos os servidores;
- O mecanismo/protocolo mais conhecido para garantir a execução de transacções distribuídas é o two-phase-commit;
- Uma transacção pode aceder a dados em vários locais;

Transacções distribuídas (Cont.)

- **Atomicidade nas transacções distribuídas:**
  - **Problema:** Garantir que numa transacção distribuída os nós envolvidos na transacção ou fazem todos commit ou fazem todos rollback.
  - **Surge Quando:** Uma transacção contém comandos DML (Data Manipulation Language) que referenciam objectos remotos contidos em tabelas diferentes com o objectivo de modificar os dados neles contidos.

Protocolos de COMMIT

Uso e tipos

Uso:

- **Os protocolos de commit são usados para assegurar a atomicidade entre locais:**
  - Uma transacção que executa em múltiplos locais tem que fazer o commit em todos os locais ou falhar em todos os locais;
  - Não é aceitável uma mesma transacção fazer o commit num local e falhar noutro.

Tipos:

- **Protocolo two-phase commit:** é amplamente usado;
- **Protocolo three-phase commit:** é mais complicado e pesado, mas evita alguns problemas do 2PC.

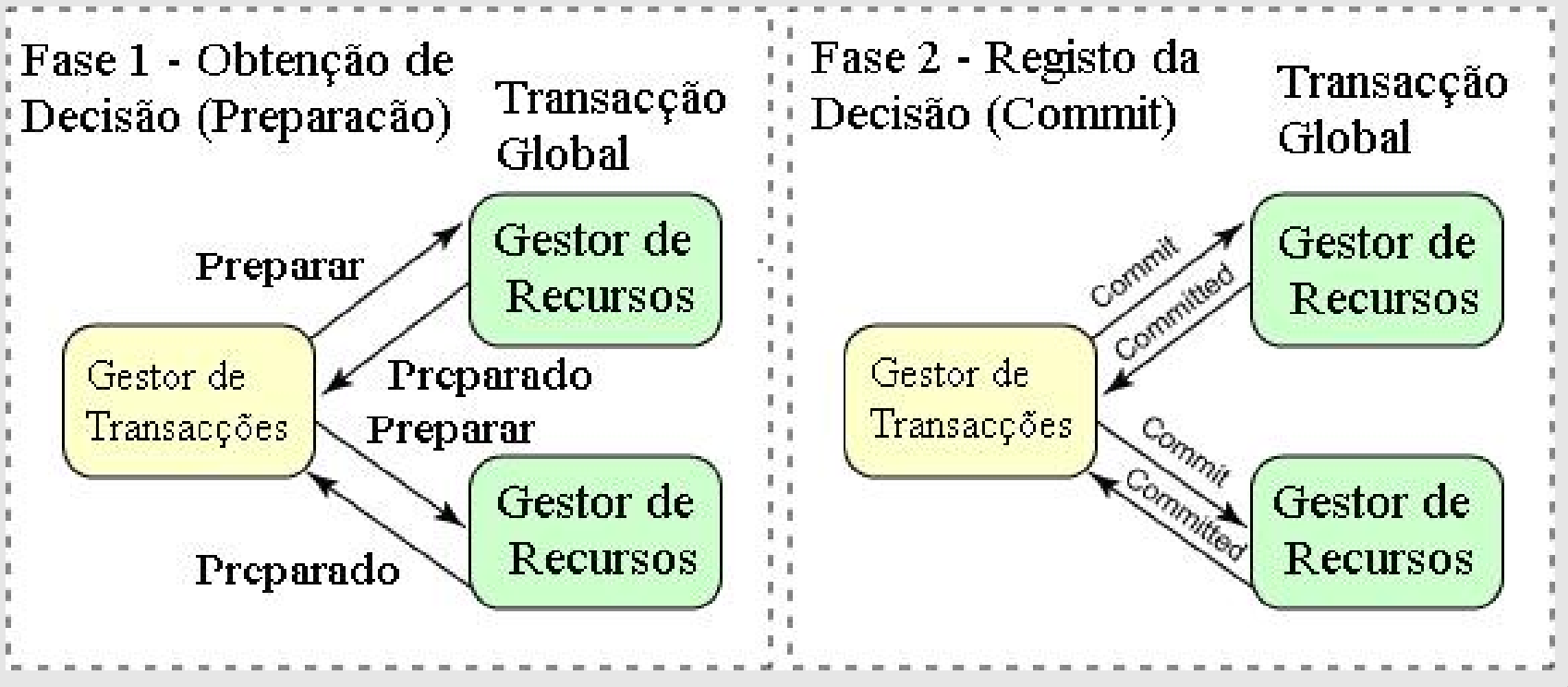
Protocolo two-phase commit (2PC)

Noções

- **Assume o modelo falha-pára: (fail-stop)** os nós que falham simplesmente param de trabalhar e não provocam mais erros;
- A execução do protocolo é iniciada pelo coordenador após o último passo da transacção;
- O protocolo envolve todos os locais onde a transacção foi executada.

Protocolo two-phase commit (2PC) (Cont.)

Fases



Protocolo two-phase commit (2PC) (Cont.)

Fase de Preparação

Em resposta ao nó coordenador, cada nó participante pode responder de três maneiras:

- **Preparado:** Todos os dados do nó, referidos pelos comandos DML, da transacção distribuída foram modificados e o nó encontra-se preparado para terminar a transacção;
- **Só de leitura:** Os comandos executados na transacção não alteram os dados do nó, pelo que não é necessária qualquer preparação;
- **Abortar:** Ocorreu um erro e o nó não está preparado para terminar a transacção com êxito.

Protocolo two-phase commit (2PC) (Cont.)

Fase de Preparação (Cont.)

Passos na fase de preparação:

- **Depois de ter recebido ordem para se preparar, um nó participante executa os seguintes passos:**
  - Indica aos seus descendentes, se os houver, para se prepararem;
  - Reserva todos os recursos necessários para fazer o commit;
  - Efectua uma série de acções, de modo a garantir que os dados bloqueados (os que foram alterados) vão ultrapassar eventuais falhas;
  - Responde (ao nó que lhe ordenou para se preparar) de acordo com o resultado da sua fase de preparação.

Protocolo two-phase commit (2PC) (Cont.)

Fase de Preparação (Cont.)

Insucesso na fase de preparação:

- **Quando um nó não se consegue preparar para terminar a transacção com êxito:**
  - Efectua rollback da parte local da transacção;
  - Liberta todos os recursos reservados para essa transacção;
  - Responde (ao nó que lhe solicitou para se preparar) a indicar que abortou a transacção.

Estas acções são propagadas aos outros nós envolvidos na transacção.

Protocolo two-phase commit (2PC) (Cont.)

Fase de conclusão (commit phase)

Nesta fase há garantia de que todos os nós envolvidos estão em condições de terminar a transacção com êxito:

- Todos os nós recebem a ordem para efectuar commit;
- Em cada nó o SGBD faz o commit da parte local da transacção e liberta os bloqueios correspondentes;
- Regista o término da transacção nas estructuras que garantem tolerância a falhas.

Protocolo two-phase commit (2PC) (Cont.)

Falhas durante o two-phase-commit

Uma grande variedade de falhas pode ocorrer durante a execução do mecanismo two-phase commit:

- Falhas nos nós;
- Falhas na rede;
- Falhas de local;
- Falhas de software;
- Falha de coordenador.

O objectivo primordial das técnicas de tolerância a falhas nas BDDs é:

- Garantir que não há perda/corrompimento de dados.

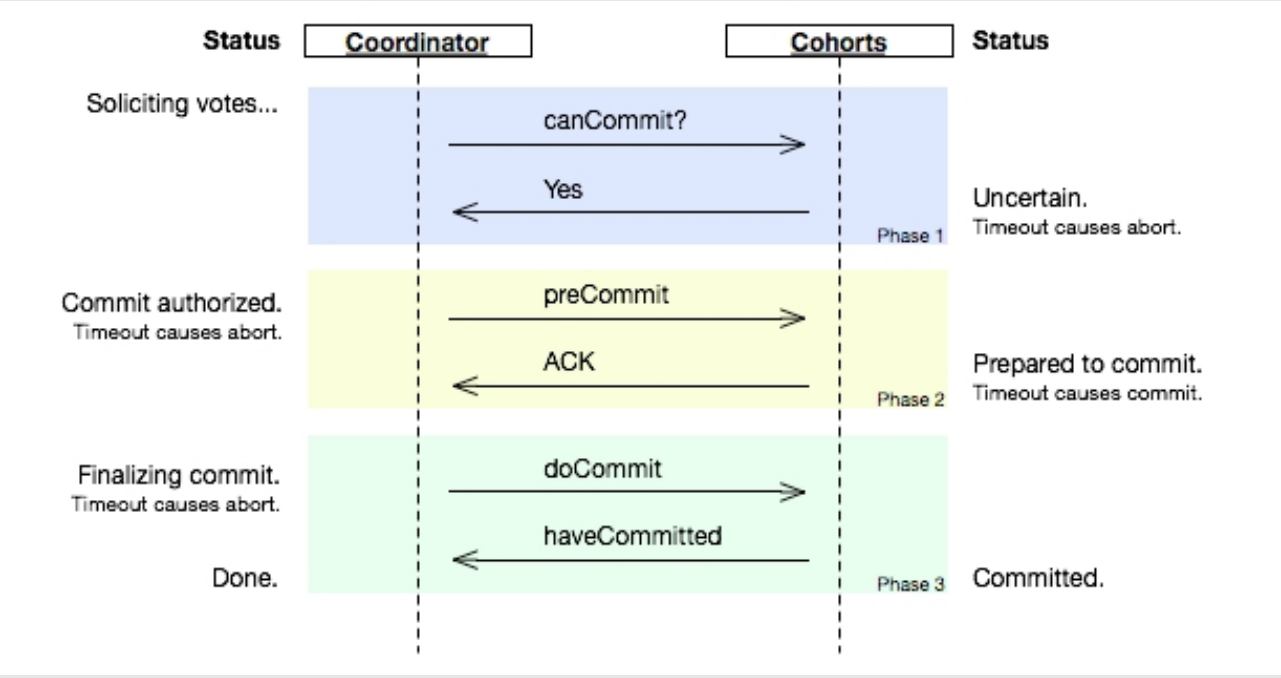
Protocolo two-phase commit (2PC) (Cont.)

Desvantagens

- ”Demasiado restritivo, basta que um dos nós participantes falhe ou não esteja disponível, para que todos os outros façam RollBack”; [1]
- ”Se houver uma falha após a 1^a fase, antes do envio das instruções de commit ou abort, todas as substransacções envolvidas ficam bloqueadas (num estado de espera activo mantendo os locks que possuíam)”; [1]

Nota: Como alternativa ao 2PC pode usar-se o 3PC, que resolve as desvantagens citadas.

Protocolo three-phase commit (3PC)



- Não há particionamento da rede;
- Em qualquer ponto, pelo menos um local deve estar disponível.

Protocolo three-phase commit (3PC) (Cont.)

Desvantagens:

- Maiores overheads (mais informação a ser transmitida);
- Deadlocks, sendo a detecção e resolução destes ainda mais complexa que nos sistemas centralizados.

Transparência

Transparência numa BDD

Tipos de transparência:

- **Na localização física de objectos;**
- **Nas interrogações e actualizações distribuídas;**
- **Nas transacções distribuídas;**
- **Na replicação de objectos;**
- Transparência de dados.

Transparência tipos

Transparência na localização física de objecto

Sinónimos e transparência na localização física de objecto:

- Um sinónimo é um nome alternativo para uma tabela, vista, sequência, procedimento, snapshot ou outro sinónimo.
- Os sinónimos também podem atribuir transparência na localização dos objectos. Se o objecto referido pelo sinónimo for mudado para outro local, só é necessário alterar a sua definição;
- Os sinónimos devem ser públicos e têm que ser declarados em todas as bases de dados locais;

Transparência tipos (Cont.)

- **Nas interrogações e actualizações distribuídas:** Os comandos DML (Select, Insert, Update, Delete,...) devem poder ser utilizados para acessos remotos e distribuídos como se fosse para acessos locais.
- **Nas transacções distribuídas:** As transacções distribuídas são controladas pelos mesmos comandos standard (Commit, Savepoint, Rollback) das transacções locais, sem necessidade de qualquer outra acção.

  Rollback voltar atrás, cancelar/anular transacção.

- **Na replicação de objectos:** O SGBD deve ter mecanismos para, de uma maneira transparente, replicar certos objectos críticos.

Transparência de transacção

Transparência de transacção

- Assegura que todas as transacções distribuídas mantêm a integridade e a consistência da BDD;
- A transacção distribuída acessa os dados armazenados em vários locais;
- Cada transacção é dividida num número de subtransacções, uma para cada local que tem que ser alcançado;
- O SGBDD deve assegurar que transacção global e cada subtransacção não podem ser divisíveis.
- **Tipos de transparência de transacção:**
  - Transparência de concorrência;
  - Transparência de falha.

Transparência de Distribuição

Transparência de Distribuição

Definição:

- A transparência da distribuição permite que o utilizador interprete a base de dados como uma entidade lógica;
- **Se o SGBDD exibir a transparência de distribuição, o utilizador não necessita de saber:**
  - Os dados que são fragmentados (transparência de fragmentação);
  - Localização dos dados (transparência de localização);

Transparência de Distribuição (Cont.)

Transparência de Distribuição (Cont.)

Tipos de transparência de Distribuição:

- Transparência de fragmentação;
- Transparência de localização;
- Transparência de replicação;
- Transparência de mapeamento local;
- **Transparência de nomeação;**
- **Transparência de desempenho.**

Transparência de nomeação

Transparência de nomeação

- Cada item numa BDD deve ter um nome único;
- O SGBDD deve assegurar-se de que dois locais não criam um objecto da base de dados com o mesmo nome;
- **Uma solução é criar um servidor de nomes central, isto resulta em:**
  - Perda de alguma autonomia local;
  - O local central pode transformar-se num bottleneck;
  - **Baixa disponibilidade:** se o local central falhar, os locais restantes não podem criar nenhuns objectos novos.

Transparência de nomeação (Cont.)

Transparência de nomeação (Cont.)

- **Solução alternativa:** prefixar o objecto, com o identificador do local que o criou:
  - Necessita de identificar cada fragmento e as suas cópias;
  - Isto resulta na perda da transparência de distribuição;
- **Outra solução, que resolve estes problemas, usa pseudônimos para cada objecto da base de dados:**
  - O SGBDD tem a tarefa de traçar (mapear) um pseudônimo para obter o objecto da base de dados.

Transparência de Desempenho

Transparência de Desempenho

- O SGBDD deve executar como se fosse um SGBD centralizado;
- O SGBDD não deve sofrer nenhuma degradação do desempenho devido à arquitectura distribuída;
- O SGBDD deve determinar a melhor estratégia de custo de execução de uma consulta;
- O processamento de consultas distribuído (Dsitributed Query processing - DQP) mapeia a consulta dos dados numa ordem sequêncial de operações em bases de dados locais;
- Deve considerar a fragmentação, a replicação e o esquema de alocação;

Transparência de Desempenho (Cont.)

Transparência de Desempenho (Cont.)

- **Um DQP tem que decidir:**
  - A que fragmento aceder;
  - Que cópia dum fragmento vai usar;
  - Que local vai usar.
- DQP implementa a estratégia de execução optimizada relativamente a alguma função de custo;
- **Tipicamente, os custos associados a uma consulta distribuída incluem:**
  - Custo do I/O;
  - Custo do processador central;
  - Custo de comunicação.

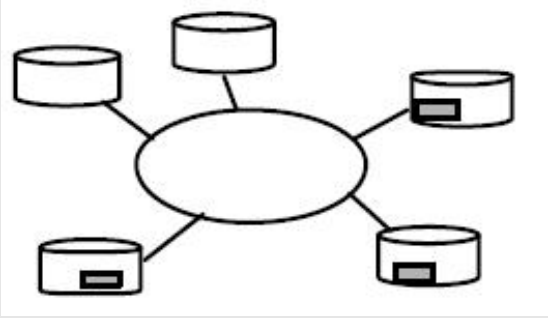
Armazenamento de dados distribuídos

- **Recorrer à Replicação, Fragmentação, ou a ambas combinadas:**
  - Uma relação é particionada em vários fragmentos, mantendo o sistema várias cópias destes.

Replicação de dados

Conceitos

- Uma relação ou fragmento de uma relação está replicado se está guardado redundantemente em vários locais;
- **Replicação total de uma relação:** acontece quando uma relação está guardada em todos os locais;
- **Bases de dados com redundância total:** são aquelas onde cada local tem uma cópia de toda a base de dados;



- **Tipos:**
  - Base de dados completamente replicada;
  - Sem Replicação;
  - Replicação Parcial.

Replicação de dados (Cont.)

Vantagens e Desvantagens da replicação

Vantagens:

- Melhoramento da disponibilidade e da fiabilidade dos dados;
- Paralelismo (tratamento de parte dos dados em nó);
- Aumenta o desempenho de queries globais;
- Redução dos dados a transferir.

Desvantagens:

- Custo de manter uma consistência mútua;
- Aumento do custo de actualizações (mais lentas);
- Aumento da complexidade do controlo da concorrência.

Fragmentação

A definição e a alocação dos fragmentos requer estratégias relativas a:

- Localização de referência;
- Implementar fiabilidade, disponibilidade e desempenho;
- Balancemanto entre capacidade de armazenamento e custo;
- Custo de comunicação mínimo.

Nota: Se a relação é pequena e não é actualizada com frequência, é melhor não fragmentar!

Fragmentação (Cont.)

- Envolve a análise das aplicações mais importantes e é baseada na quantidade e qualidade da informação;
- **A quantidade de informação inclui:**
  - Frequência de execução da aplicação;
  - Local (site) onde é executada.
- Critérios de desempenho para aplicações e transacções;
- **A qualidade da informação:** envolve as transacções que são executadas pelas aplicações, tipo de acesso (leitura e gravação) e propriedades das operações de leitura.

Fragmentação (Cont.)

Porquê fragmentar?

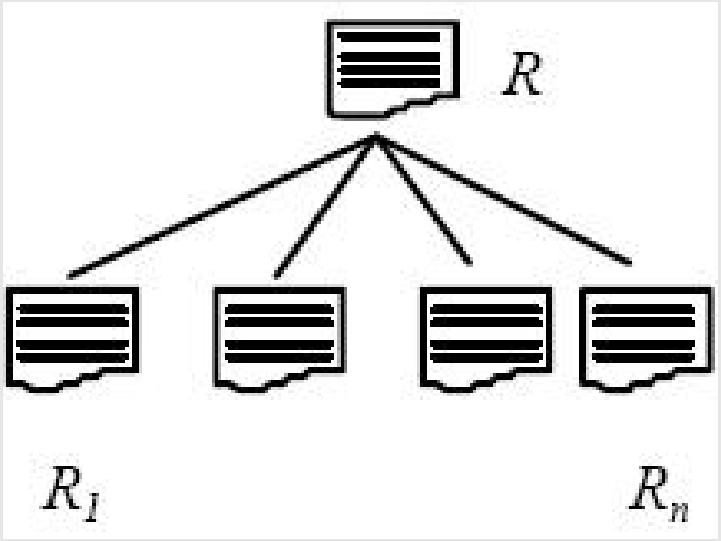
Vantagens:

- **Utilização:** Aplicações que trabalham com subconjuntos e não com a relação completa;
- **Eficiência:** Os dados são armazenados onde são utilizados mais frequentemente;
- **Paralelismo:** As transacções podem ser divididas em diversas consultas (subconsultas), que operam sobre os fragmentos;
- **Segurança:** Dados não necessários localmente não são armazenados e não estão disponíveis para utilizadores não autorizados.

Desvantagens: Desempenho, Integridade.

Fragmentação (Cont.)

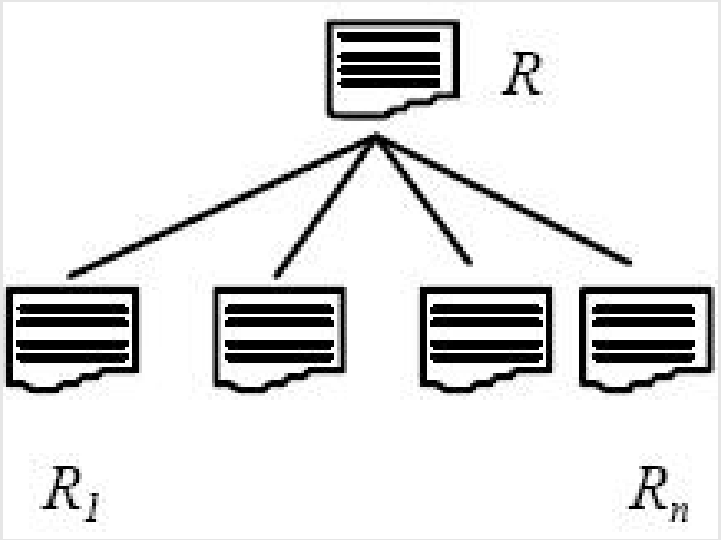
Regras para a correcção da fragmentação



- **Integralidade:** Se uma relação R for decomposta em fragmentos R_1, R_2, ... R_n, cada item de dados que pode ser encontrado em R deve aparecer em pelo menos um fragmento.

Fragmentação (Cont.)

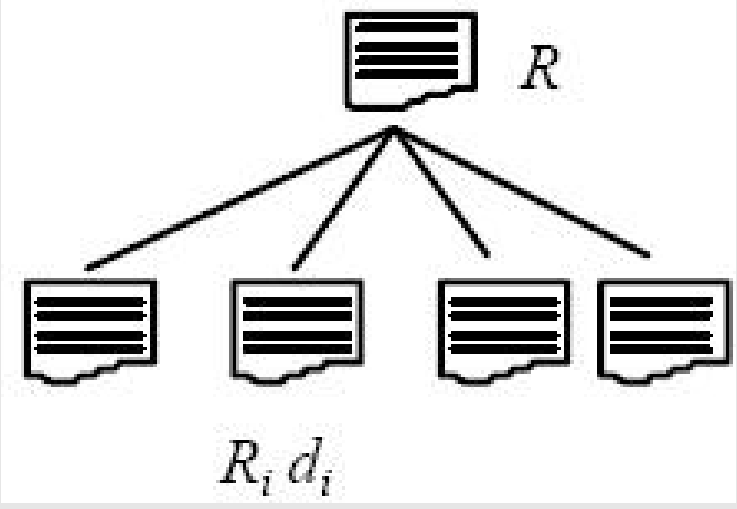
Regras para a correcção da fragmentação (Cont.)



- **Reconstrução:**
  - Tem ser possível definir uma operação de relação que vai reconstruir *R* a partir dos fragmentos;
  - Reconstrução para fragmentação horizontal designa-se União e para fragementação vertical Junção.

Fragmentação (Cont.)

Regras para a correcção da fragmentação (Cont.)



- **Disjunção:** Se o item de dados $_{di}$ aparecer no fragmento R_i, então ele não deve aparecer em nenhum outro fragmento. **Excepção:** Fragmentação vertical, onde os atributos da chave primária têm de ser repetidos para permitir a reconstrução.

Fragmentação (Cont.)

Tipos de Fragmentação

- **Fragmentação horizontal:**
  - Consiste num subconjunto de tuplos de uma relação;
  - Define-se usando a operação de selecção da álgebra relacional.

Fragmentação (Cont.)

Tipos de Fragmentação (Cont.)

- **Fragmentação vertical:**
  - Consiste num subconjunto de atributos de uma relação;
  - Define-se usando a operação de projecção da álgebra relacional;
  - Agrupa atributos que sejam usados por uma aplicação.
- **Fragmentação mista.**

Fragmentação (Cont.)

Vantagens

- **Vertical:**
  - Permite que os tuplos sejam divididos de forma a que cada parte do tuplo esteja quardada onde é mais frequentemente acedida;
  - O atributo de identificação do tuplo permite a junção eficiente de fragmentos verticais;
  - Permite o processamento paralelo de uma relação.

Fragmentação (Cont.)

Vantagens (Cont.)

- **Horizontal:**
  - Permite o processamento paralelo de fragmentos de uma relação;
  - Permite que uma relação seja dividida de forma a que os tuplos estejam onde são mais frequentemente acedidos.

Alocação de Dados

Estratégias de armazenamento de dados

- **Centralizado:** Consiste numa única BD armazenada num local com utilizadores distribuídos sobre uma rede de comunicação;
- **Particionado - Distribuído:** A base de dados é dividida em fragmentos (disjuntos) e cada parte é pré-alocoda num local;
- **Replicação completa:** Consiste em manter uma cópia completa da base de dados em cada local;
- **Replicação selectiva:** É uma combinação da centralização, fragmentação e replicação.

Alocação de Dados (Cont.)

Comparativo de Estratégias

	Centralizado	Fragmentado	Replicação completa
Localidade da referência	Muito baixo	Alto	Muito alto
Fiabilidade e disponibilidade	Muito baixo	Baixo para um item Alto para o sistema	Muito alto
Desempenho	Não satisfatório	Satisfatório	Melhor para leitura
Custos de armazenamento	Muito baixo	Muito baixo	Muito alto
Custos de comunicação	Muito alto	Baixo	Alto (actualizações) Baixo (leitura)

Processamento de Consultas (Queries)

Processamento de Consultas

Remoção de informação duplicada quando os locais têm informação repetida:

- Decidir que locais executam a consulta;
- Optimização global de consultas.

Processamento de Consultas distribuídas:

- Para sistemas centralizados, o critério primário para medir o custo de uma determinada estratégia é o número de acessos ao disco. **Num sistema distribuído, outros aspecto devem ser tidos em conta:**
  - O custo da transmissão de dados pela rede;
  - O ganho potêncial em desempenho, devido a haver vários locais a processar partes de uma consulta em paralelo.

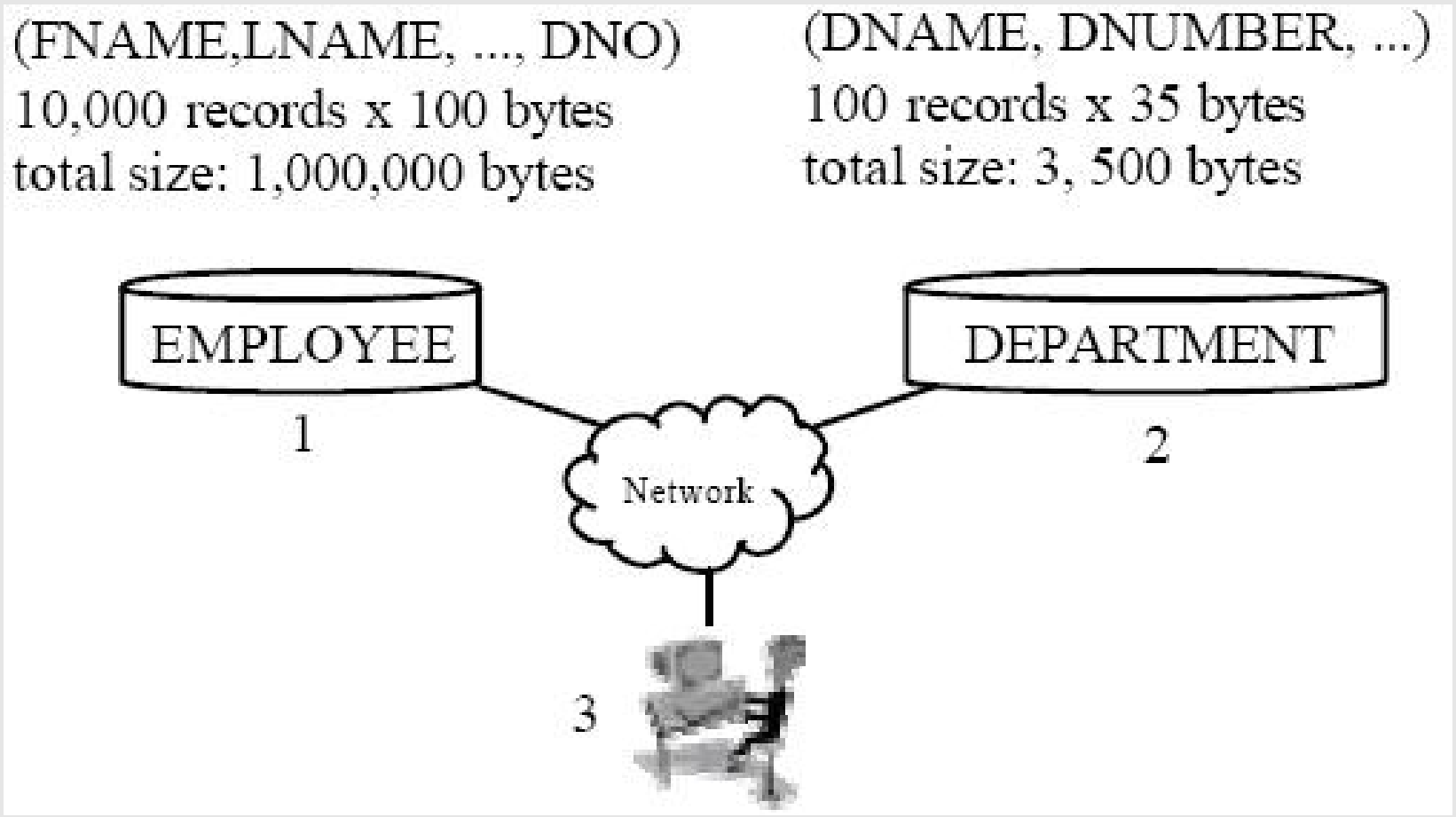
Transformação de consultas

Transformação de consultas

Consultas algébricas sobre fragmentos:

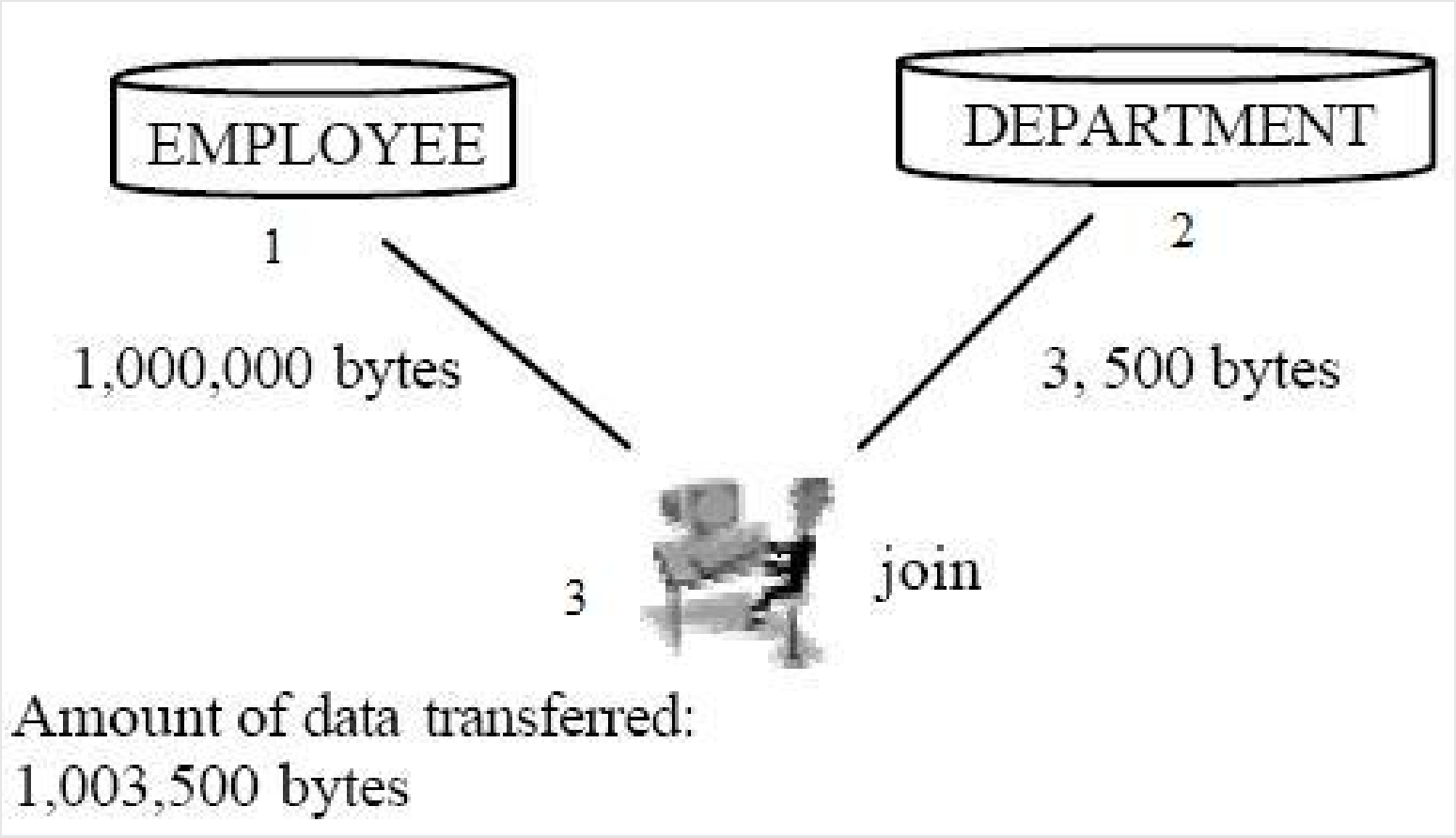
- Tem de ser possível reconstruir uma relação *r* dos seus fragmentos;
- Substituir a relação *r* pela expressão que constrói a relação *r* dos seus fragmentos.

Processamento de queries em BDD



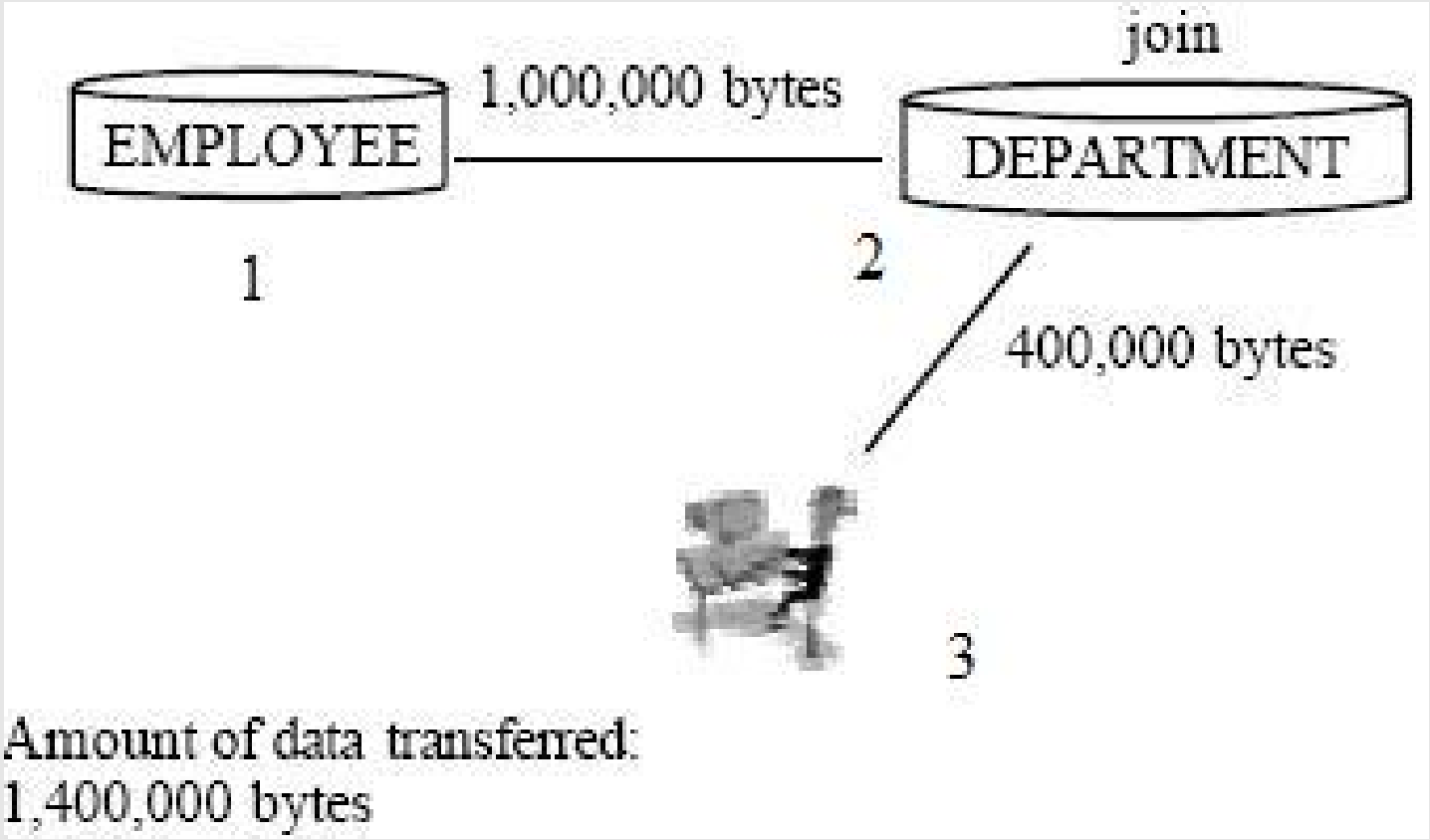
Processamento de queries em BDD (Cont.)

Estratégia 1 - Executar Join no local 3



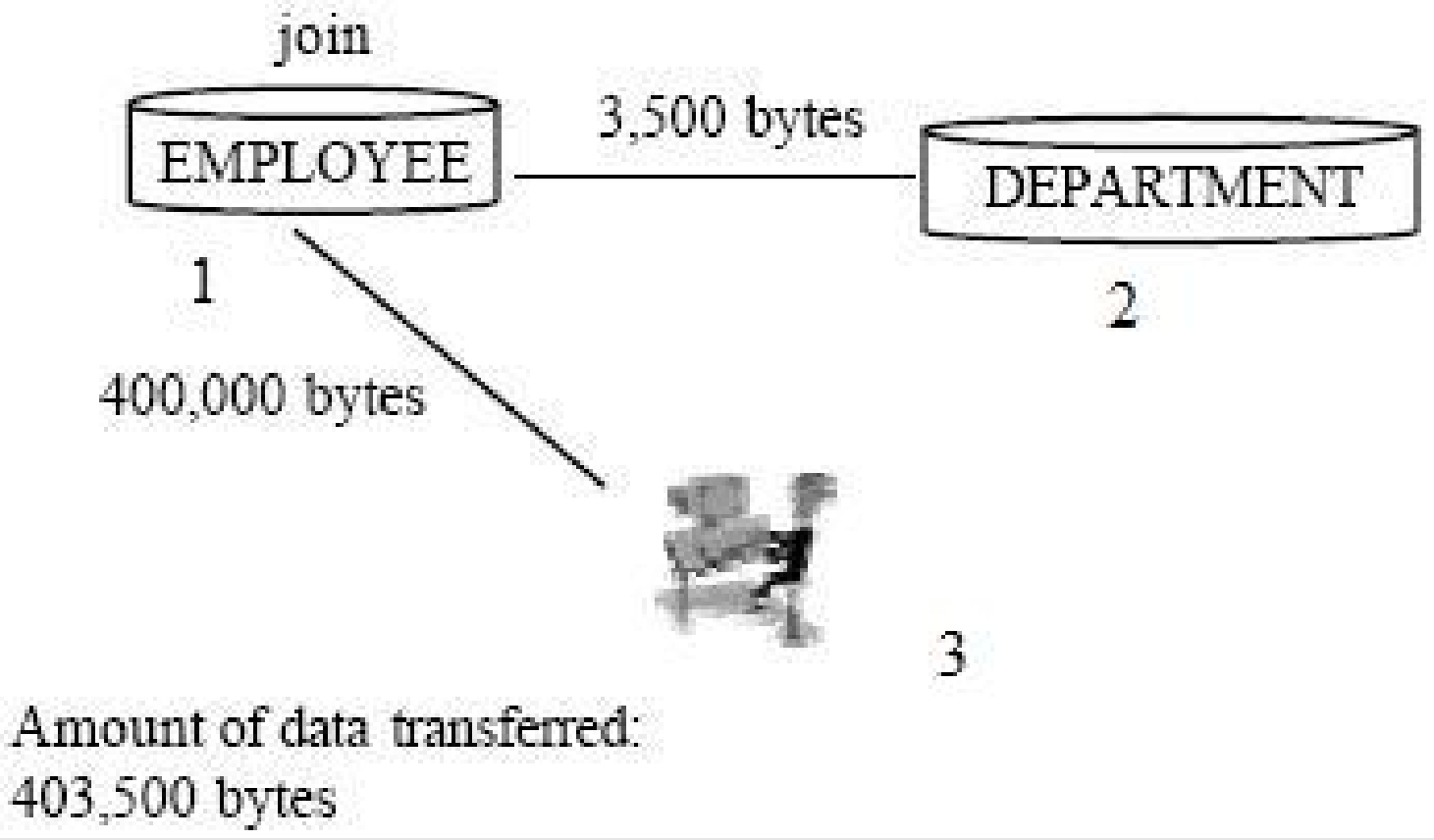
Processamento de queries em BDD (Cont.)

Estratégia 2 - Executar Join no local 2



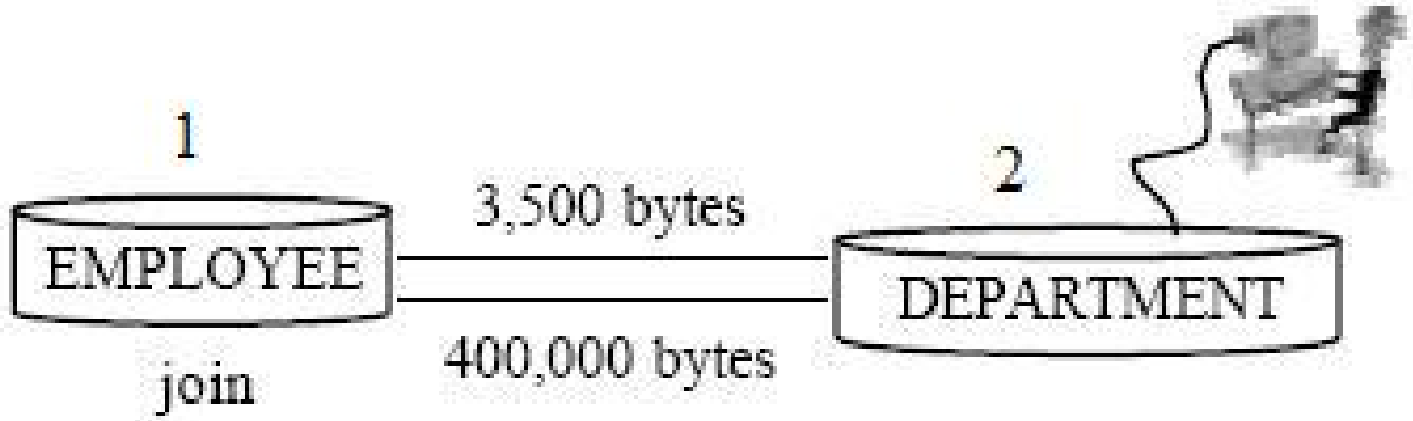
Processamento de queries em BDD (Cont.)

Estratégia 3 - Executar Join no local 1



Processamento de queries em BDD (Cont.)

Utilizador no local 2



Amount of data transferred:
403,500 bytes

Definição

- Modificação dos esquemas de controlo de concorrência para uso em ambientes distribuídos;
- Assumimos que cada local participa na execução do protocolo de commit para assegurar a atomicidade global das transacções;
- Assumimos que todas as réplicas de qualquer item estão actualizadas.

Abordagens para o controlo de concorrência

- Abordagem Single Lock Manager; [2]
- Lock Manager Distribuído; [2]
- Primary copy; [2]
- Quorum Census Protocol; [2]
- Abordagem Centralizada. [2]

Falhas do sistema

Falhas próprias de sistemas distribuídos:

- Falhas de um local;
- Perda de mensagens:
  - Tratada pelos protocolos de rede tais como TCP-IP.
- **Falha de um caminho de comunicação:**
  - Tratada por protocolos de rede, pelo envio de mensagens por caminhos alternativos.

Noção de partição da rede

- Uma rede diz-se particionada quando está dividida em dois ou mais subsistemas que não possuem qualquer ligação entre eles.
- Partição de rede e falhas de locais são normalmente indistinguíveis.

Problemas de Segurança

- **A segurança das base de dados distribuídas deve satisfazer os seguintes requerimentos:**
  - **Integridade física:** que é a protecção da perda de dados;
  - **Integridade lógica:** é a protecção da estrutura lógica da base de dados;
  - **Integridade de elementos:** é assegurar dados exactos;
  - Disponibilidade fácil;
  - Controlo de acessos a alguns níveis dependendo da sensibilidade dos dados do utilizador;
  - O utilizador autenticar-se para garantir que é quem diz ser.
- O objectivo destes requerimentos é garantir que os dados armazenados nos SGBDD estão protegidos contra modificação não autorizada, e updates não exactos.

Falhas do sistema e Problemas de Segurança (Cont.)

Problemas de Segurança (Cont.)

- **Ameaças:**
  - Alteração de dados;
  - Estar á escuta (eavesdropping);
  - Roubo de dados;
  - Falsificar a identidade de um utilizador;
  - Administrar muitas passwords.
- **A segurança nas bases de dados distribuídas pode ser fornecida através de:**
  - Controlo de acesso;
  - Autenticação do utilizador;
  - Transparência local;
  - Transparência de vista.

Conclusões

Conclusões

- As BDDs são um tema vasto e com muitas peculiaridades;
- Somente Oracle faculta um serviço de BDDs;
- Os novos modelos são extensões significativas aos modelos anteriores;
- Evitar SBDD quando o problema (tendo em atenção as necessidades futuras) não o exige;
- As BDDs apresentam vantagens claras e significativas face às BD, simultâneamente uma quantidade enorme de novos problemas e questões a ter em conta na implementação;

Bibliografia

Conceitos (Ultima consulta a 16-04-2008)

- **Distributed DBMS**
- **3ª Geração: Novos modelos de BD**
- **Bases de Dados**
- **Distributed Database Security**
- **Modelos de Bases de Dados**
- **Modelagem de Dados em Multicamadas para a Implementação de Bases de Dados Distribuídas**
- **Transaction Management in Distributed Database Systems: the Case of Oracle's Two-Phase Commit**
- [1] [*Pereira*97] José Luís Pereira, "Técnologias de Bases de dados"; FCA, 1997, ISBN:9727220738

Bibliografia (Cont.)

Estudos práticos (Ultima consulta a 07-03-2008)

- **Components of a Distributed Database**
- **Security Implications of the Choice of Distributed Database Management System Model: Relational vs. Object-Oriented**
- **ISSUES IN DISTRIBUTED DATABASE SECURITY**
- **Distributed Databases**
- **Bases de Dados Distríbuidas**
- **Bases de Dados**
- **Uma reflexão sobre BDD's Orientadas a Objetos**
- [2] **Bases de Dados Distribuídas**

FIM

Questões fágnificas?
Opiniões fabulásticas?
Comentários espantásticos?