UNIPAC - UNIVERSIDADE PRESIDENTE ANTÔNIO CARLOS
FACULDADE DE CIÊNCIA DA COMPUTAÇÃO E
COMUNICAÇÃO SOCIAL

CURSO DE CIÊNCIA COMPUTAÇÃO

Cynthia Pereira Silva

BANCO DE DADOS DISTRIBUÍDOS

BARBACENA
DEZEMBRO – 2004

Cynthia Pereira Silva

BANCO DE DADOS DISTRIBUÍDOS

Trabalho de conclusão de curso apresentado à Universidade Presidente Antônio Carlos – UNIPAC- Barbacena, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em _____/_____/_____

BANCA EXAMINADORA

Prof. Eduardo Macedo Bhering (Orientador)

Universidade Presidente Antônio Carlos

Prof. Luis Augusto Mattos Mendes

Universidade Presidente Antônio Carlos

Prof. Gustavo Campos Menezes

Universidade Presidente Antônio Carlos

Dedico este trabalho em especial à minha mãe pelo apoio incondicional, amigos pela força, e a todos que realmente torceram por mim.

Agradeço a Deus, ao apoio minha família, amigos, ao orientador Eduardo Bhering pelo esforço, aos professores Luis Augusto e Gustavo pela participação.

RESUMO

Este trabalho apresenta um estudo sobre banco de dados distribuídos, enfatizando seu projeto. Apresentando suas características, vantagens, desvantagens, e abordando também questões sobre transações e controle de concorrência.

Palavras-chave: Banco de dados distribuídos, projeto, transações, controle de concorrência.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

BD – Banco de dados

BDD – Banco de dados distribuídos

DML – Linguagem de manipulação de dados

SGBD – Sistema gerenciador de banco de dados

SGBDD – Sistema gerenciador de banco de dados distribuídos

SQL - *Standart Query Language*

OSI/ISO - *Open Systems Interconection/ International Standards Organization*

DLL - Linguagem de Definição de Dados

ANSI - *American National Standards Institute*

ACID - Atomicidade Consistência Isolamento Durabilidade

SUMÁRIO

1 INTRODUÇÃO

O surgimento dos sistemas de banco de dados favoreceu a difusão do uso dos recursos computacionais, provendo a independência dos programas em relação aos dados armazenados. Esta independência de dados é decorrente da introdução do sistema de gerência de banco de dados (SGBD), que surgiu como uma camada lógica entre a aplicação e os dados, fornecendo um mecanismo de abstração de detalhes [9].

Os sistemas de banco de dados centralizados são aqueles executados sobre um único sistema computacional que não interagem com outros sistemas. Tais sistemas podem ter a capacidade de um banco de dados de um só usuário executado em um computador pessoal até sistemas de alto desempenho em sistemas de grande porte [8]. Porém, a estratégia de trazer os dados a um processador central pode ser mais cara do que trazer a capacidade computacional para vários processadores. Estes sistemas apresentam vulnerabilidade maior à falhas e nem sempre permitem um crescimento gradativo da capacidade computacional instalada de forma simples e adequada [3].

Um SGBD é constituído por um conjunto de dados associados a um conjunto de programas para acesso a esses dados. O principal objetivo de um SGBD é proporcionar um ambiente tanto conveniente quanto eficiente para recuperação e armazenamento das informações de dados. São projetados para gerir grandes volumes de dados. O gerenciamento implica a definição das estruturas de armazenamento e dos mecanismos para a manipulação dessas informações [8].

A evolução tecnológica causou, e ainda está causando, profundas alterações nas atividades de processamento de dados, permitindo que o usuário disponha de processamento local para muitas aplicações, dando origem aos sistemas distribuídos [4]. Neste sistema, uma coleção de computadores autônomos são ligados por uma rede e com *softwares* projetados pra produzir uma facilidade de computação integrada [7].

Visto que as empresas já são distribuídas, pelo menos logicamente e com grande probabilidade fisicamente, os dados estão normalmente distribuídos. Portanto, não existe a necessidade de cada unidade organizacional, dentro da empresa, manter dados que não são relevantes para sua própria operação. O patrimônio total de informações da empresa é desse modo disseminado naquilo que às vezes costuma-se chamar de “ilhas de informações”. Um sistema distribuído oferece pontes necessárias para interligar essas ilhas. Ou seja, permite que a estrutura do banco de dados reflita a estrutura da empresa, dados locais podem ser mantidos em

instalações locais às quais eles pertencem logicamente, enquanto ao mesmo tempo dados remotos estão disponíveis para acesso quando necessário [4].

Em decorrência desses fatos, surge o conceito de um tipo particular de sistema distribuído: o sistema gerenciador de banco de dados distribuídos (SGBDD). A tecnologia de banco de dados distribuídos surgiu como uma intercalação de duas tecnologias: tecnologia de banco de dados e tecnologia de comunicação de dados em rede [1].

No sistema centralizado os dados são armazenados em um servidor central, e diversos computadores conectados em rede fazem o uso destes dados para eventuais processamentos. O SGBDD pode ser visualizado como um conjunto de banco de dados locais geograficamente dispersos, mas interligados por uma rede de comunicação que possibilita a transferência de dados entre diversos bancos de dados locais de forma simples e transparente para os usuários [5].

Os computadores de um SGBDD não compartilham memória principal [8]. Seus processadores podem variar em tamanho e função. Podendo incluir pequenos microcomputadores, estações de trabalho, minicomputadores e sistemas de computadores de uso em geral. Esses processadores são chamados por diversos nomes, como nós e sites dependendo do contexto [6].

Em um banco de dados distribuído, são armazenados dados por vários locais. Para acessar um dado que está armazenado em um local remoto o sistema estará sujeito ao custo do transcurso da mensagem, e para reduzir este problema, um único dado pode ser dividido ou pode ser fragmentado por vários locais, ou pode ser reproduzido onde a demanda é alta [6].

O processamento é feito de acordo com a distribuição dos dados, podendo ser realizado no site local ou nos demais sites do sistema.

Quanto às implementações comerciais, a maior parte de produtos SQL (*Standart Query Language* - é uma linguagem padrão para se lidar com BD relacionais) sofrem alguma espécie de suporte para banco de dados distribuídos, com graus variados de funcionalidades. Alguns dos mais conhecidos incluem: o *Ingres/Star* (componente do BDD do *Ingres*); a opção do banco de dados distribuídos do *Oracle*; e o recurso de banco de dados distribuídos do DB2. Estes sistemas listados são relacionais, pelo menos todos eles admitem SQL [4].

O modelo relacional fornece as operações necessárias para realizar fragmentação e reconstrução, que são duas das várias razões pelas quais os sistemas de banco de dados distribuídos podem ser implementados no modelo relacional. [4] A implementação deste sistema no modelo relacional fica menos complexa, apesar disso a tendência é de implementações no modelo objeto-relacional.

A prática deste tipo de sistema pode ser definida em aplicações que requerem o compartilhamento de informação entre usuários, programas ou organizações (que tenham uma estrutura descentralizada) onde os usuários da informação, ou mesmo as fontes de informação, estão geograficamente dispersos, por exemplo, em sistemas de consultas de saldos bancários, redes de hotéis, empresas aéreas, etc [3].

Existem várias vantagens na implantação de SGBDD's, das quais podemos destacar:

- A contribuição de forma significativa para o *aumento da produtividade* em desenvolvimento das aplicações. De fato, tais sistemas simplificam a tarefa de se definir aplicações que requerem compartilhamento de informação entre usuários, programas ou organizações onde os usuários da informação, ou mesmo as fontes de informação estão geograficamente dispersas [3];
- *Sob o ponto de vista administrativo*: Os SGBDD's permitem que cada setor de uma organização geograficamente dispersa mantenha controle de seus próprios dados, mesmo oferecendo compartilhamento a nível global no uso destes dados [3];
- *Sob o ponto de vista econômico*: os SGBDD's podem diminuir os custos de comunicações, que hoje em dia tendem a ser maiores que o próprio custo do equipamento, com o tradicional declínio do custo de "hardware" [3]. Por outro lado, é menor o custo de um conjunto de máquinas de pequeno porte que possua a capacidade de processamento igual ao de um equipamento de grande porte do que o custo de um único equipamento com capacidade equivalente [9];
- *Flexibilidade para expansão*: sob o ponto de vista técnico também são atrativos, pois facilitam o crescimento modular da aplicação, simplesmente acrescentando-se novos processadores e novos módulos ao sistema [3], podendo ser feito em pequenos incrementos sem causar interrupções no funcionamento geral do SGBDD [5];
- *Aumento da Confiabilidade e disponibilidade*: a confiabilidade é geralmente definida como a probabilidade de que um sistema esteja funcionando em um certo momento, [1] realizando sua função como projetado ou como previsto [10]. Enquanto a disponibilidade é a probabilidade de que um sistema esteja continuamente disponível durante um intervalo de tempo [1], realizando sua

função sempre que ela é exigida [10]. Em um sistema centralizado, falhas em um único site fazem com que todo o sistema se torne indisponível para todos os usuários. Enquanto em um sistema banco de dados distribuídos, em caso de falhas os demais sites continuam a operar. Alguns dados podem ser inatingíveis, mas os usuários podem acessar outras partes do banco de dados [1] que contenham os dados do site onde ocorreu a falha, devido a aperfeiçoamentos que são conseguidos através de criteriosa replicação de dados e de software em mais de um site. No entanto, os efeitos no caso de falha em algum site são limitados [10]. Isso influi tanto nos aspectos de confiabilidade quanto em disponibilidade, trazendo melhorias;

- *Aumento de desempenho (performance)*: Através do emprego das técnicas de fragmentação e de replicação os dados são mantidos próximos do seu local onde a freqüência de uso é maior. Quando os dados estão localizados de tal forma, existe uma redução na disputa entre serviços da CPU e de entrada e saída, ao mesmo tempo em que reduz a demora no acesso a sistemas de rede de áreas distantes [1], principalmente se a maior parte dos acessos gerados em um site puderem ser resolvidos localmente [3]. Ou seja, quando um grande bloco de dados é distribuído em vários sites, existem bancos de dados menores em cada site. Portanto, consultas e transações locais que acessam dados em sites locais têm melhor desempenho por não haver necessidade de acesso a dados remotos [1];
- *Autonomia*: Proporciona o controle local dos dados permitindo compartilhamento. Um grupo de usuários pode ter os dados que compartilham localizados no seu próprio ambiente de trabalho, executando localmente várias das funções de administração de banco de dados, como o controle de autorização e de acesso e a escolha das estruturas de armazenamento de dados utilizadas. Com a distribuição, o próprio entendimento dos dados (ou seja, o que cada dado significa e suas inter-relações) é facilitado pela menor complexidade, favorecendo o seu controle [9].

E as desvantagens são:

- *Distribuição de controle*: Consomem recursos e podem ter uma performance duvidosa [4]. A autonomia, que é considerada como uma vantagem sob diversos aspectos, traz a desvantagem dos problemas de sincronismo das ações e coordenação das partes. Por exemplo, a atualização dos dados em uma relação fragmentada entre dois servidores implica coordenação rigorosa do processo, para assegurar que, mesmo no caso de falha em um dos processadores, os dados resultantes serão consistentes. Outro fator é o de restauração do BD: sem sincronismo das ações de recuperação, a consistência dos dados é inevitavelmente perdida [9];
- *Segurança*: Os sistemas de banco de dados distribuídos possuem vários pontos (locais) de controle e uma rede envolvida; sendo que essa rede é um meio que tem seus próprios requisitos de segurança com sérios problemas de se mantê-la. Por tais razões, os problemas de segurança em sistemas de bancos de dados distribuídos são mais complicados que dos centralizados [9]. Contudo, é mais fácil manter a segurança de uma grande quantidade de dados centralizados do que em um conjunto de ilhas de informações com localidade dispersa [10].
- *Perca de Confiabilidade:* Devido ao fato de um sistema distribuído conter uma rede de comunicação, existe um sinal de desconfiança quando de trata em confiabilidade [10].
- *Dificuldade de transição:* Em razão de algumas empresas ainda possuírem sistemas herdados (aplicações empresariais críticas construídas sob tecnologia que incorpora anos de experiências personalizadas e acumuladas, bastante diferentes daquelas que hoje temos disponíveis para uso), não podem ser trocados da noite para o dia. Mesmo que a tecnologia atual fosse estável e o projeto determinado, a transição para um sistema distribuído levaria algum tempo [10].

Observando que a tecnologia de sistemas de bancos de dados distribuídos é um dos principais desenvolvimentos recentes na área de sistemas de bancos de dados e tendo em vista que, este tema não foi abordado no decorrer do curso. Portanto, o objetivo deste trabalho é fazer um estudo sobre sistemas de banco de dados distribuídos. Sendo que, detalhes sobre tecnologia de sistemas de rede se encontram fora do escopo.

No capítulo 2, será feita uma conceituação dos SGBDD's, estabelecendo um conjunto de definições básicas que caracterizem e permitam a análise das principais características funcionais e os tipos de tais sistemas.

No capítulo 3 será feita uma abordagem sobre o processo de projeto de banco de dados distribuídos, descrevendo uma visão geral. Onde, a fase do processo de projeto de distribuição possuirá ênfase, pois se difere com grandeza das demais fases quando comparado ao projeto dos sistemas centralizados. Serão mostradas as técnicas de distribuição de dados, conhecidas como fragmentação e replicação.

No capítulo 4 será abordado sobre transações, se tratando de gerência e execução.

No capítulo 5 será abordado sobre o controle de concorrência das transações, especificando os métodos para tal controle.

2 BANCO DE DADOS DISTRIBUÍDOS

"Um sistema de banco de dados distribuído é uma coleção de dados que é distribuída por diferentes computadores, possivelmente em diferentes locais. Os computadores estão conectados por uma rede de comunicação. O sistema deve suportar aplicações locais em cada computador, bem como aplicações globais nas quais mais de um computador estejam envolvidos" [Ceri e Pelagatti,1985 Apud 9]. A figura 1 fornece um exemplo deste sistema em uma visão geral.

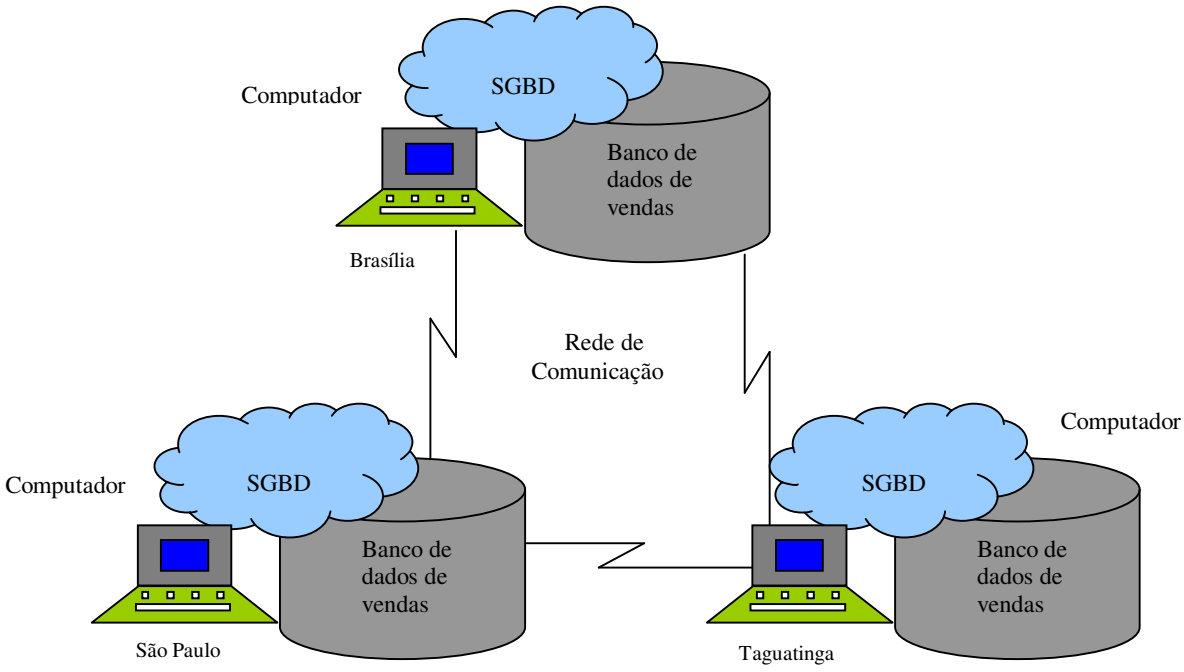

Figura 1 – Visão Geral de um Sistema de Banco de Dados Distribuídos

2.1 – Conceitos Básicos

O suporte para BDD implica que uma única aplicação deve ser capaz de operar de modo transparente sobre dados dispersos em uma variedade de BD's diferentes, gerenciados por vários SGBD's diferentes, em execução em uma variedade de máquinas diferentes, admitidos por variedade de sistemas operacionais diferentes e conectados entre si por uma variedade de redes de comunicação diferentes. Onde modo transparente significa que a aplicação opera de um

ponto de vista lógico como se os dados fossem todos gerenciados por um único SGBD, funcionando em uma única máquina [4].

Em um SGBDD é distribuído apenas: processamento lógico; funções; dados; controle. Não existe compartilhamento de memória principal e nem relógio [8].

O subconjunto do BDD armazenado em um site é chamado banco de dados local [5]. Um SGBD local é um SGBD centralizado gerenciado de forma autônoma do BD global, exceto que poderá receber comandos tanto de usuários locais quanto da cópia local do SGBD global. O SGBD local faz uso do sistema operacional local que provê as seguintes facilidades básicas: métodos de acesso, gerência de processos e gerência de memória. Sendo que, a coletividade dos BD's locais constitui uma implementação do BDD [3].

O SGBD global roda como uma aplicação sob o sistema operacional da rede de comunicação de dados e, portanto, pertence à camada de aplicação na nomenclatura do modelo de referência OSI/ISO (*Open Systems Interconection/ International Standards Organization* - é uma arquitetura de rede voltada para facilitar a interconexão de redes de comunicação heterogêneas). Isto significa que todos os problemas de comunicação de dados e distribuição de recursos são transparentes ao SGBD global [3].

Cada site de rede conterá então um SBGD local e uma cópia do SGBD global, caso armazene parte do banco, ou apenas o gerente de transações, caso não armazene parte do banco [3]. A propósito, é comum supor que os sites componentes estão fisicamente dispersos - talvez dispersos também geograficamente – embora na verdade seja suficiente que estejam dispersos logicamente. Dois sites poderiam coexistir na mesma máquina física, em especial durante o período de testes iniciais do sistema [4].

Entretanto, o projeto do BDD se difere do BD com o acréscimo do processo de projeto de distribuição que possui duas técnicas básicas de particionamento de relações, podendo ser empregadas de forma isolada ou combinadas. Estas técnicas são:

- *Fragmentação*: É um processo para o particionamento do BD, onde a relação é particionada em vários fragmentos, sendo que cada um é armazenado em um site diferente. Esses fragmentos contem informação suficiente para a reconstrução da relação original.
- *Replicação*: O sistema mantém réplicas idênticas (cópias) da relação. Se uma relação é replicada, uma cópia da relação é armazenada em dois ou mais sites diferentes. Temos a *replicação total*, na qual cada cópia é armazenada em todos os sites do sistema.

Existem duas formas básicas de se fazer cópias de dados no ambiente de BD [9]:

- *Extrato:* Qualquer duplicação de dados ou processos, onde nunca será atualizada, ou seja, a cópia é planejada para operações somente de leitura;
- *Réplica:* Qualquer duplicação de dados ou processos, onde se admite atualização.

2.2 - Tipos de SGBD's Distribuídos

Os banco de dados distribuídos pode ser classificado quanto ao tipo, em de duas formas básicas: heterogêneos (*bottom-up*) e homogêneos (*top-down*). Sendo que, no dia-a-dia das organizações, atualmente, é comum encontrarmos projetos que combinam ambas as abordagens geralmente como conseqüência do esforço de migração para outros sistemas de bancos de dados residentes em ambientes de grande porte para outras plataformas [9].

2.2.1 – SGBDD's Heterogêneos

São usualmente utilizados em ambientes onde proliferam os bancos de dados departamentais e individuais, criando "ilhas de informação" que, com o avanço da tecnologia de redes podem ser integradas, originando banco de dados globais virtuais [9].

SGBDD's heterogêneos surgem usualmente quando há necessidade de integrar sistemas já existentes e independentes, devido aos seguintes fatores [5]:

- Aproveitamento do hardware;
- Custos de aproveitamento do ambiente existente;
- Hábitos dos usuários e grau de colaboração.

Um SGBDD heterogêneo compreende a utilização de diversos SGBD's, que atuam cooperativamente, mas que continuam autônomos [9].

Estes sistemas surgem, também, se houver sistemas SBGD's diferentes rodando em diversos sites, tornando difícil o gerenciamento de transações globais [8].

2.2.2 – SGBDD's Homogêneos

Os SGBDD's homogêneos aparecem com maior freqüência quando a aplicação a que se destinam não existia antes [3]. Pode ser aplicado em busca da redução dos custos e da melhoria do desempenho [9].

Um SGBDD será chamado homogêneo se todos os seus SGBD's locais são [3]:

- Oferecem interfaces idênticas ou, pelo menos, da mesma família; ou seja, usam a mesma Linguagem de Definição de Dados (DDL) e Linguagem de Manipulação de Dados (DML); ou pelo menos, DDL da mesma família;
- Usam o mesmo modelo de dados (redes, hierárquico, relacional etc.) para descrever o banco de dados;
- Executam as mesmas funções através dos mesmos algoritmos [5], ou seja, fornecem os mesmos serviços aos usuários em diferentes nós [3].

O SGBDD é o responsável pelo fornecimento de dados e tradução de exigências aos processos num ambiente distribuído homogêneo [5].

2.3 - Características dos Sistemas de Banco de Dados Distribuídos

Um requisito fundamental para BDD's é que as aplicações sejam preservadas da necessidade de conhecer a distribuição dos dados pelos diferentes servidores, os detalhes dos caminhos de acesso aos servidores, os controles de duplicidade de dados e as diferentes formas de representação de dados utilizadas nos servidores [9]. Para preservar as aplicações de lidar com esses detalhes, o sistema deve conter tais características que serão abordadas a seguir.

2.3.1 - Autonomia local

Todos os dados distribuídos pela rede devem ser de propriedade e gerência local. Um SGBDD possibilita o acesso aos dados de forma integrada, embora cada site continue a ser administrado independentemente do sistema local distribuído. [9] Um usuário local deve acessar dados locais como se constituísse um BD centralizado independente. [3]

2.3.2 - Independência de replicação

Os usuários devem ser capazes de se comportar, pelo menos do ponto de vista lógico, como se os dados não fossem replicados de modo algum [4], sem necessitar lidar com as redundâncias introduzidas pelo processo físico [9]. É desejável porque simplifica os programas e atividades em terminal do usuário; permite que réplicas sejam criadas ou destruídas a qualquer momento, em resposta a mudanças de exigências, sem invalidar quaisquer programas e usuários [4]. Isto significa que a consistência entre cópias, no caso de atualizações, deve ser mantida pelos SGBDD's [9]. Isso implica na responsabilidade do otimizador do sistema determinar quais réplicas terão acesso físico para satisfazer a uma solicitação de um determinado usuário. [4]

2.3.3 - Independência de fragmentação

Significa que as aplicações e os usuários têm sempre uma visão única dos dados independente da forma como as relações estejam fragmentadas entre os diferentes sites. Os SGBDD's devem garantir ao usuário a visão de que todas as relações existem em um só local [9]. Estas técnicas de distribuição serão abordadas posteriormente no capítulo 3.

2.3.4 - Não dependência de um site central:

Não existe um local que possui autoridade sobre os demais. Isso implica que em cada site deve haver uma cópia do dicionário de dados e dos mecanismos de segurança. [9]

A dependência de um site central seria indesejável pelo menos por estas duas razões [4]:

- Esse site central poderia ser um gargalo;
- O sistema seria vulnerável – se o site central caísse, todo sistema cairia.

2.3.5. - Operação Contínua

Cada local possui autonomia, ou seja, é independente dos demais. No entanto, para que formem um sistema, devem cooperar entre si. Assim, os procedimentos normais de operação de um sistema local não podem comprometer a disponibilidade do todo. [9]

Para que o sistema possua uma operação contínua, tais características são relevantes:

- Confiabilidade: traz melhorias porque sistemas distribuídos não seguem a proposta do tudo ou nada. Eles podem continuar a funcionar, em nível reduzido, mesmo diante da falha de algum componente individual, como um site isolado. [4];
- Disponibilidade: também traz melhorias, em parte pela mesma razão que foi tratada no item anterior, e em parte devido à possibilidade de replicação de dados.

2.3.6 - Independência de Localização

As aplicações e os usuários não precisam conhecer a localização dos BD's nas redes de computadores ou lidar com protocolos utilizados na comunicação entre os diferentes sistemas. Portanto, uma aplicação é realizada em sites diferentes sem que o usuário ou o programador da aplicação tenha que especificar os locais para acesso [9], simplificando programas e atividades em terminal dos usuários [4]. E as diferenças entre as formas de representação de dados utilizados pelos sistemas componentes não são percebidas pelas aplicações e usuários, devendo ser resolvidas pelos SGBDD's [9].

2.3.7 - Processamento de Consultas Distribuídas

O usuário deve poder efetuar consultas sobre o conjunto de dados, cabendo ao sistema realizar o processamento necessário para viabilizar o acesso aos subconjuntos de dados de cada local. Isto significa, por exemplo, que seria possível a realização de uma consulta única que resultasse em acesso aos dados dispersos por vários sites [9].

2.3.8 - Gerência de Transações Distribuídas

O SGBDD deve ser capaz de possibilitar alterações nos estados de BD locais por uma única transação de forma consistente. Isto significa que, independente das possibilidades de falhas que possam ocorrer no processo em um ambiente distribuído, o sistema deve possibilitar a garantia [9] de dois aspectos principais do gerenciamento de transações [4]:

- Controle de recuperação: garantia de que uma determinada transação é atômica (tudo ou nada) no ambiente distribuído;
- Controle de concorrência: na maioria dos sistemas distribuídos este controle baseia-se em geral no bloqueio, exatamente como em sistemas não distribuídos.

Estes aspectos serão abordados em detalhes nos capítulos 4 e 5.

2.3.9 - Independência de Hardware

Uma aplicação que necessite ter acesso a dados dispersos por máquinas diferentes (Ex: IBM, DEC, HP, PC's e estações de trabalho de várias espécies etc.) deve poder realizar o seu processamento sem restrições decorrentes das diferenças entre as plataformas [9].

2.3.10 - Independência de Sistema Operacional

De forma semelhante ao que foi observado com relação ao hardware, é necessário que uma aplicação tenha acesso aos dados dispersos por plataformas com sistemas operacionais distintos, sem restrições decorrentes de suas diferenças [9].

2.3.11 - Independência de Rede

Um SGBDD pode implicar a utilização de diversos protocolos, requerendo eventualmente a conversão de protocolos. Todos esses detalhes não devem ser visíveis aos usuários devendo ser resolvidos pelo próprio SGBDD. [9]

2.3.12 - Independência de Banco de Dados

Deve ser possível armazenar e recuperar dados de diferentes BD locais, independente da arquitetura do BD e do próprio SGBD utilizado em cada um daqueles locais [9].

3 O PROJETO DE BANCO DE DADOS DISTRIBUÍDOS

Neste capítulo serão discutidas questões relativas do projeto de um banco de dados distribuído. Isso será feito através da analogia com o processo de projeto de um sistema gerenciador de banco de dados centralizado. Desde a análise da arquitetura do sistema de banco de dados até a fase de projeto lógico.

3.1 - Arquitetura de Esquemas para Sistemas de Banco de Dados

Um comitê de padronização do ANSI (*American National Standards Institute*) propôs, em 1978, uma arquitetura para sistemas de banco de dados que ficou conhecida como arquitetura ANSI/SPARC ou arquitetura de três camadas, utilizada em ambientes centralizados, assim como mostra a figura 2. O objetivo desta arquitetura é separar o banco de dados físico das aplicações do usuário, por meio de três diferentes níveis do esquema [9]:

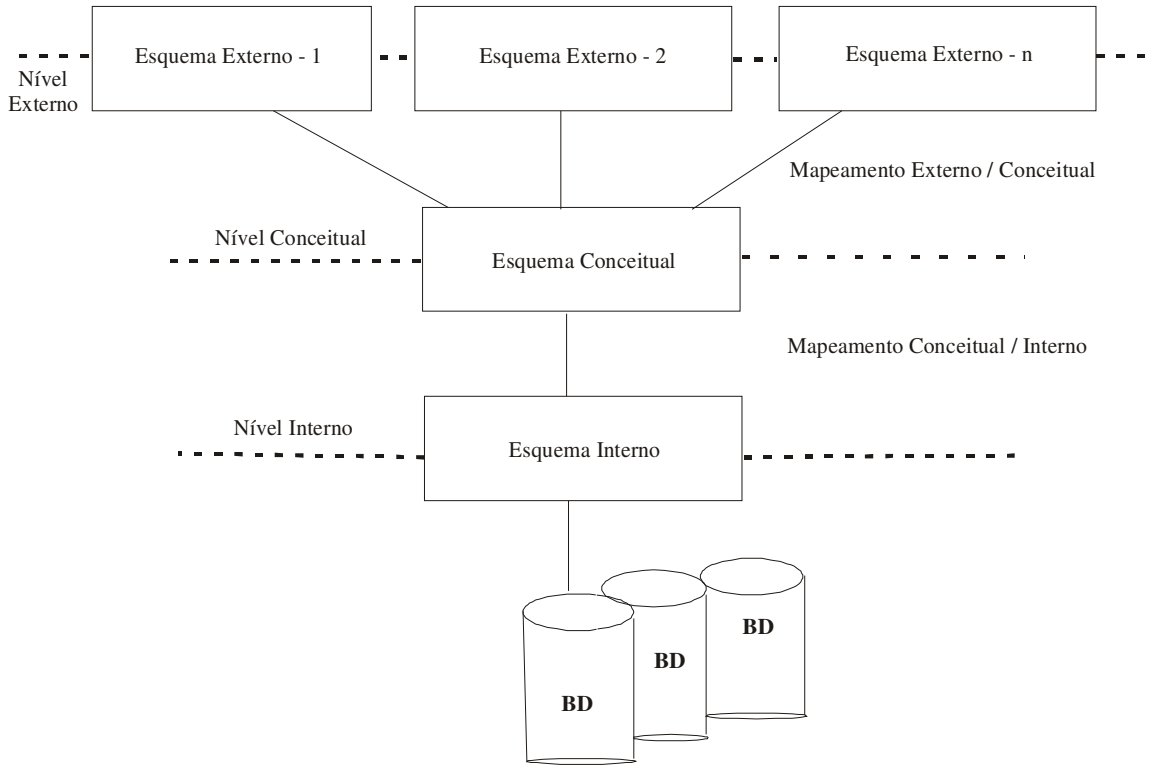

Figura 2 - Arquitetura de esquemas em três níveis utilizada em ambientes centralizados[9]

- *Esquema interno:* descreve a estrutura física de armazenamento do banco de dados, a sua organização de arquivos e os seus métodos de acesso.

- *Esquema conceitual:* descreve o banco de dados sob o ponto de vista do usuário, escondendo detalhes de armazenamento e concentrando-se na descrição de entidades, atributos, relacionamentos, operações do usuário e restrições sobre dados.
- *Esquemas externos:* também chamados visões de usuário, que descrevem as partes do banco de dados que são do interesse de um grupo de usuários, escondendo as demais [9]. Um esquema externo não é apenas um subconjunto do esquema conceitual, tem sua maneira própria de agrupar as informações e só pode ser manipulado através de operações determinadas [5].

Os esquemas são meramente descrições de dados: o único nível que armazena dados é o interno [9].

3.1.1 - Arquitetura de Esquemas para Bancos de Dados Distribuídos

A arquitetura de esquemas em três níveis (Arquitetura ANSI/SPARC) é inadequada para ambiente distribuído.No entanto, ainda não existe para este ambiente um padrão formal definindo uma estrutura de referência. Utilizando como referência a arquitetura em quatro níveis introduzida por [Ozsu e Valduriez, 1991], mostrada na figura 3. Podemos observar que, comparativamente ao modelo ANSI/SPARC, um nível de esquemas foi adicionado, correspondente aos esquemas conceituais locais. Isto mostra a necessidade de se fazer também o mapeamento do esquema conceitual global para o esquema conceitual local. Este nível foi acrescido com a finalidade de garantir a independência de dados a aplicações globais [9].

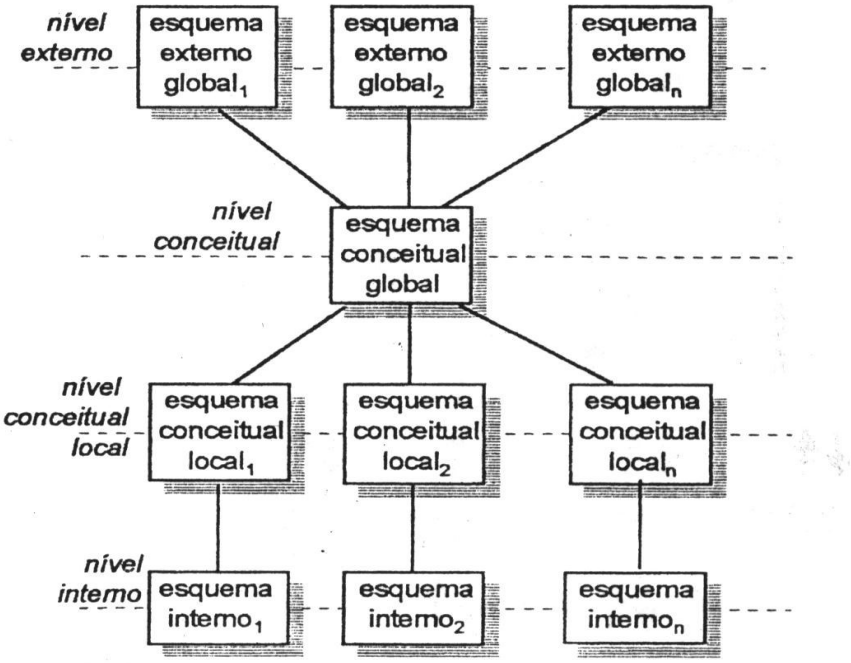

Figura 3 – Arquitetura do Esquema de um Banco de Dados Distribuídos [9].

Um SGBDD deve ser capaz de receber de uma aplicação ou de um usuário global uma requisição de operação definida sobre uma visão externa do esquema global, transformá-la em uma operação sobre o esquema global, identificar os locais onde estão os dados correspondentes, derivar as operações locais necessárias e emitir as requisições de execução para diferentes SGBD's. O SGBDD deve, ainda, coordenar o processo de execução, compondo os resultados parciais obtidos pelos diferentes SGBD's e apresentado-os de forma integrada à aplicação ou usuários [9].

3.2 - Projeto

No caso geral, o projeto é uma especificação informal de alto nível do esquema de um banco de dados, com o uso de notações e com a implementação de projetos na proporção de definição de dados [2].

Inicialmente, deve-se verificar se a solução distribuída é de fato uma opção viável. Isto significa, detectar se o banco é fortemente integrado, ou se pode ser divido em partes mais ou menos independentes; se este for o caso, deve-se então determinar qual a vantagem de descentralizar o banco [3].

Um estudo do perfil da população de transações existentes no sistema centralizado em uso (se este for o caso) deverá ser feito, tentando determinar se é possível dividir o sistema – banco de dados e transações - em subsistemas mais ou menos independentes. Se este for o caso, o custo da transmissão de dados deverá ser reduzido, descentralizando-se o banco e suas funções. Acessos que cortem fronteiras geográficas ainda serão suportados, desde que não sejam muito freqüentes. [3]

Em primeiro lugar, o projeto do esquema conceitual global e dos esquemas externos globais é inteiramente semelhante ao centralizado, já que o BDD deverá se comportar como centralizado perante os usuários globais. Além disto, o projeto dos esquemas internos locais é também idêntico ao de BD centralizados, exceto que a carga imposta por acessos remotos aos dados locais também deve ser levada em consideração [3]. A novidade decorre da introdução da atividade do projeto da distribuição dos dados, produzindo-se tantos esquemas conceituais locais quantos forem os servidores de BD a serem utilizados na implementação do BDD [9]. Portanto, o processo de projeto de distribuição será enfatizado.

A Figura 4 ilustra o processo de projeto de banco de dados distribuídos em uma visão geral.

Figura 4 - Visão geral do processo de projeto de BDD's homogêneos [9].

Para que o projeto da distribuição possa ser efetuado, devem ser identificados os requisitos das aplicações e dos usuários, com a identificação das suas visões de dados e identificação de suas necessidades de acesso a dados. Após o projeto das visões, deve ser feita a sua integração, produzindo um esquema conceitual global. Este esquema e as informações de acesso a dados, em conjunto com as informações sobre as restrições tecnológicas, são utilizados no projeto da distribuição, produzindo os diversos esquemas conceituais locais que, em uma etapa subseqüente, são utilizados para o projeto físico dos BD's locais [9].

A tabela 1 mostra um resumo das estratégias de distribuição, para se direcionar o projeto físico, observando qual técnica será viável de acordo com cada aplicação, nos seguintes aspectos: freqüência com que uma transação necessita de dados que não estão armazenados (% de exceções) localmente, e o tamanho do arquivo [3].

% de Exceções	Tamanho do Arquivo	Método de Distribuição
---	Pequeno	Replicação
Pequena	Grande	Fragmentação
Alta	Grande	Centralizado

Tabela 1 - Resumo das estratégias de distribuição [3].

3.3 - Especificação de Requisitos

Uma especificação pode ser um documento escrito, um modelo gráfico, uma coleção de cenários e uso, um protótipo ou qualquer combinação desses elementos. Sendo que, algumas vezes é necessário permanecer flexível quando uma especificação precisa ser desenvolvida. Para sistemas grandes, um documento escrito, combinando descrições em linguagem natural e modelos gráficos podem ser a melhor abordagem [12].

3.3.1 - Restrições Tecnológicas

- *Informações sobre aplicações*: fragmentos de tabelas utilizadas, freqüência de execução, número de acesso por fragmento/execução, quantidade de tuplas recuperadas por seleção, tempo máximo de resposta esperado;
- *Informações sobre a rede de comunicação*: capacidade do meio (banda) e grau de utilização atual;
- *Informações sobre as plataformas servidoras:* disponibilidade de espaço em disco.

Dependendo do caso considerado existem outras informações que aumentam a complexidade do processo de análise, tais como a existência de picos de utilização da rede, requisitos de disponibilidade de segurança, que não foi abordado, mas que podem ser determinantes do projeto de distribuição final [9].

3.3.2 - Requisitos de Funções

Uma aplicação pode ser vista como um conjunto de funções que possibilitam ao usuário acesso aos dados. Estas funções são descritas a seguir [9]:

- *Funções de processamento da lógica da interface do usuário:* são aquelas relacionadas com a apresentação e com as atividades de entrada e saída de dados do usuário final. São projetadas para interagir com dispositivos físicos. As funções de apresentação executam tarefas como a formatação de telas, a leitura e a exibição de dados nas telas, a gerência de janelas, a gerência de diálogos e a manipulação de teclado e mouse. Outras funções possibilitam a análise dos dados fornecidos pelo usuário, executando funções usuais de validação de valores, verificação de tipos de dado e fornecimento de ajuda sensível ao contexto.
- *Funções de processamento da lógica do negócio*: a própria razão da existência da aplicação. Estas funções implementam regras do negócio e práticas administrativas de uma organização.
- *Funções de gerência de dados*: compreendem as funções de manipulação de dados e as funções que realizam serviços de acesso a esses dados. As funções de manipulação dos dados compõem a parte da aplicação responsável pelo armazenamento e pela recuperação de dados existentes em BD's ou em sistemas de arquivos. As funções que realizam acesso a dados provêem o acesso físico dos dados, como o acesso a índices, a localização de registros e a sua recuperação.

3.3.3 – Requisitos de dados

A etapa de análise de requisitos de dados é feita através da captura as necessidades da organização em termos de armazenamento de dados de forma independente de implementação O resultado desse processo de análise deve ser representado através de um modelo conceitual (diagrama entidade-relacionamento, diagrama de classes, etc) [11].

Dada uma coleção de requisitos de dados, podendo ser apresentado sob a forma de especificação em linguagem natural ou gráfica, o papel do projetista do BD é criar um esquema que os satisfaça. [9].

As estratégias básicas para projetar esquemas conceituais são [9]:

- *Estratégia do projeto descendente (top-down)*: o projetista inicia o esquema contendo conceitos de alto nível de abstração (algumas entidades mais relevantes), e então aplica sucessivos refinamentos, como a especificação de atributos, relacionamentos, especializações, agregações e novas entidades.
- *Estratégia ascendente (bottom-up)*: o projetista inicia o esquema contendo conceitos básicos (atributos ou itens de dados dispersos nos requisitos), e então aplica combinações e extensões destes conceitos, como agrupamento de atributos em entidades e relacionamentos, agregações e generalizações.

3.4 – Projeto e Integração de visões

Vários esquemas podem resultar do processo de projeto conceitual, cada um representando a visão de um usuário ou do grupo de usuários, ou de uma aplicação particular do sistema. Por isso, existe uma metodologia para a integração de visões, em um esquema conceitual global, que é dividida nos seguintes passos [9]:

- Identificação de conceitos correspondentes e conflitos de nomes, de tipos de dados e restrições entre os esquemas.
- Modificação das visões para conformar os esquemas entre si.
- Combinação das visões juntando os esquemas individuais de acordo com os conceitos correspondentes.
- Reestruturação do esquema global, se necessário para remover redundâncias e simplificar a modelagem.

3.5 – Projeto de distribuição

O projeto de distribuição de dados é iniciado com a aplicação das técnicas de distribuição, a partir das definições das necessidades de acesso a dados das aplicações. A alocação dos fragmentos das tabelas e suas possíveis réplicas entre os diferentes locais são feitos na segunda [9]

3.5.1 - Fragmentação

Um esquema de fragmentação de um BD é a definição de um conjunto de fragmentos que incluem todos os atributos e tuplas do BD e satisfaça à condição de que todo o BD pode ser reconstruído aplicando operações na álgebra relacional [1]. A fragmentação é requerida quando [5]:

- O arquivo é muito grande para ser colocado em um único site ou sua taxa de acesso, por ser muito alta, não é suportada por um único site do sistema.
- A freqüência de acesso às diferentes partes dos dados pode variar em função dos diferentes sites, as informações devem estar localizadas no site em que o acesso das mesmas é maior, para minimizar o custo das comunicações.
- A possibilidade de melhorar o nível de concorrência entre transações – fazendo com que cada uma tenha acesso as relações que contenham apenas os dados de seu interesse - e as diferentes exigências de acesso imediato aos dados [9].
- *Vantagens*: aumento da confiabilidade e eficiência (desempenho) do sistema, pois os dados podem ser armazenados no local em que são mais freqüentemente utilizados, de modo que a maior parte das operações seja apenas local, e o tráfego de rede seja reduzido [1].

As duas formas básicas de fragmentação são: fragmentação horizontal e fragmentação vertical que podem ser utilizadas de forma combinada entre si ou com outras técnicas que serão apresentadas a seguir. A aplicação conjunta das técnicas de fragmentação vertical e de fragmentação horizontal é chamada de fragmentação híbrida ou mista [9].

Em razões da facilidade de fragmentação e reconstrução, em maior parte, os sistemas distribuídos são relacionais; pois o modelo relacional oferece exatamente as operações necessárias para essas tarefas [4]

Para exemplificar de fragmentação será usado a Tabela 2 que representa uma relação de uma aplicação do sistema bancário. Esta relação (denominada "conta") mantém as informações das contas de um conjunto de cliente de uma agência bancária, apresentada a seguir:

Conta			
Agência	**Número Conta**	**Cliente**	**Saldo**
Congonhas	B-100	Maria	1200
Cons. Lafaiete	B-110	Ana	2500
Congonhas	B-130	Paula	1800
Congonhas	B-150	Pedro	750
Cons. Lafaiete	B-160	José	2300

Tabela 2 – Relação "Conta"

3.5.1.1 - Fragmentação horizontal

A relação é dividida "horizontalmente" agrupando linhas para criar um subconjunto de tuplas, nas quais cada subconjunto possui um significado lógico. As tuplas são especificadas através de uma condição em um ou mais atributos; freqüentemente, apenas um atributo está envolvido [1]. Um fragmento horizontal de uma tabela pode ser especificado através de uma operação de restrição (seleção) da álgebra relacional [1]

Cada tupla da relação deve pertencer a pelo menos um fragmento, de modo que a tabela original possa ser reconstruída, se necessário [8].

Ri= R1 U R2 U R3 U ... U Rn

A tabela CONTA pode ser divida em n fragmentos diferentes, cada um dos quais contendo tuplas de agências em particular:

Aplicando a fragmentação horizontal: Ri = σ Pi (R)

Conta_1 = σ Agência = "Congonhas" (Conta)

Conta_2 = σ Agência = "Cons. Lafaiete" (Conta)

Tem-se como resultado da fragmentação horizontal o novo esquema de conta apresentado a seguir:

Conta_1			
Agência	**Número Conta**	**Cliente**	**Saldo**
Congonhas	B-100	Maria	1200
Congonhas	B-130	Paula	1800
Congonhas	B-150	Pedro	750

Tabela 3 – Relação Conta 1.

Conta_2			
Agência	**Número Conta**	**Cliente**	**Saldo**
Cons. Lafaiete	B-110	Ana	2500
Cons. Lafaiete	B-160	José	2300

Tabela 4 – Relação Conta 2.

3.5.1.2 - Fragmentação Vertical

A relação é dividida "verticalmente" em colunas, decompondo o esquema. Um fragmento vertical de uma relação pode ser especificado através de uma operação de projeção da álgebra relacional [1].

Uma relação ao ser decomposta em fragmentos verticais poderá ser recomposta em sua forma original através de uma operação de junção das relações, utilizando-se o atributo em comum (chave) [9]. Porém, se os fragmentos estiverem armazenados de maneira separada não será possível a recomposição da relação original, visto que não existe nenhum atributo comum entre os fragmentos. Então, faz-se necessário incluir, em cada uma das novas relações, a chave primária da relação original ou algum atributo de chave candidata [1]. Muitas vezes é conveniente acrescentar um atributo especial, chamado identificador de tupla ou tupla_id, no esquema da relação original. O valor da tupla_id é um valor único, usado para distinguir uma tupla de todas as outras. O atributo tupla_id serve, assim, como chave candidata incrementada ao esquema [8].

A tabela 5 mostra o resultado da operação de adição de atributo tupla_id à tabela 2.

Conta + Tupla_id				
Agência	**Número Conta**	**Cliente**	**Saldo**	**Tupla_id**
Congonhas	B-100	Maria	1200	1
Cons. Lafaiete	B-110	Ana	2500	2
Congonhas	B-130	Paula	1800	3
Congonhas	B-150	Pedro	750	4
Cons. Lafaiete	B-160	José	2300	5

Tabela 5 – Relação Conta acrescida do atributo tupla_id.

As tabelas 6 e 7 mostram o resultado do processo de fragmentação vertical utilizando-se o esquema abaixo:

Conta_3 = ⊓ Agência, Número_Conta, Tupla_id (Conta)

Conta_4 = ⊓ Cliente, Saldo, Tupla_id (Conta)

Conta_3		
Agência	**Número Conta**	**Tupla_id**
Congonhas	B-100	1
Cons. Lafaiete	B-110	2
Congonhas	B-130	3
Congonhas	B-150	4
Cons. Lafaiete	B-160	5

Tabela 6 – Relação Conta 3.

Conta_4		
Cliente	**Saldo**	**Tupla_id**
Maria	1200	1
Ana	2500	2
Paula	1800	3
Pedro	750	4
José	2300	5

Tabela 7 – Relação Conta 4.

3.5.1.3 – Fragmentação Híbrida

A fragmentação mista é obtida através da combinação da fragmentação horizontal e vertical.

Considerando o fragmento Conta_3 gerado na seção anterior através da fragmentação vertical. Pode-se e utilizando-se a fragmentação horizontal, nesse fragmento, obtém-se as duas relações representadas nas tabelas 8 e 9

Conta_3 a = σ Agência = "Congonhas" (Conta_3)

Conta_3 b = σ Agência = " Cons. Lafaiete" (Conta_3)

Conta_3[a]		
Agência	**Número Conta**	**Tupla_id**
Congonhas	B-100	1
Congonhas	B-150	4
Congonhas	B-130	3

Tabela 8 – Relação Conta 3 a.

Conta_3b		
Agência	**Número Conta**	**Tupla_id**
Cons. Lafaiete	B-110	2
Cons. Lafaiete	B-160	5

Tabela 9 – Relação Conta 3 b.

3.5.2 - Replicação

A replicação de dados é uma técnica simples e de baixo custo, mas que deve ter a utilização cuidadosamente analisada, pois contrai o pressuposto básico de não fazer cópia dos dados. Dados altamente voláteis não devem ser replicados, enquanto dados que nunca (ou quase nunca) mudam podem ser replicados minimizando o perigo de trazer inconsistência para ao banco de dados. Dados que nunca mudam são casos raros. Portanto, cada caso deve ser analisado, avaliando-se fatores como aumento da confiabilidade e da disponibilidade do sistema de banco de dados, a freqüência de atualização dos dados, a origem e a prioridade das transações de acesso aos dados [9].

Duas vantagens de aplicação da replicação podem ser citadas:

- *Disponibilidade e confiabilidade*: como já foi dito, se um dos sites que contenha uma determinada relação houver falha, então esta relação pode ser encontrada em outro site. Assim, apesar da ocorrência da falha de um site, o sistema pode continuar a processar consultas que envolvam a relação, [8] pelo menos para acesso, enquanto houver no mínimo uma cópia disponível [4].
- *Aumento do paralelismo*: se a maioria dos acessos a uma determinada relação implica somente na leitura da mesma, então, diversos sites podem processar consultas à mesma relação em paralelo. A replicação minimiza o movimento dos dados dos dados entre os sites, pois ao realizar cópias, as chances de que os dados necessários sejam encontrados no site em que a transação está sendo executada são maiores, aumentando desempenho nestas operações [8]. Ou seja, aplicações podem operar sobre cópias locais, em vez de terem de se comunicar com sites remotos [4].

E como desvantagens, tem-se:

- *Aumento do overhead para atualização*: o sistema deverá assegurar que todas as réplicas da relação sejam consistentes. A atualização da relação deverá ser propagada a todos os sites que contenham réplicas da mesma, tendo como resultado as transações geram o aumento do custo operacional (overhead) [8]. A replicação total, quando cada BD local contém uma cópia completa do banco, torna as técnicas de recuperação e concorrência mais caras [1].

Existem duas formas básicas de se fazer cópias de dados no ambiente de BD [9]: e*xtrato e réplica.*

3.5.2.1 – Extratos

Extratos podem ser bastante úteis para aplicações que demandem apenas a recuperação de dados (consultas). Podem ser classificados em diversos tipos de acordo com as necessidades das aplicações para as quais são projetados. [9]

Extrato simples: A maneira usualmente empregada para se produzir extratos deste tipo consiste na execução de algum utilitário ou programa de aplicação batch, que tem acesso aos dados de um BD. As tuplas das tabelas selecionadas são copiadas do BD origem para o BD destino, gerando uma tabela que possa apenas ser consultada. Freqüentemente esta operação é chamada de exportação de dados (conforme ilustrado na figura 5), podendo haver diferença entre os formatos e estruturas de armazenamento dos dados utilizadas no BD de origem e aquelas utilizadas no BD destino [9]. Este tipo de extrato é utilizado para compor BD utilizados por organizações em sistemas de suporte a decisão [9] e também se encaixa às cópias feitas para dados não-voláteis, para dados históricos, etc, [10].

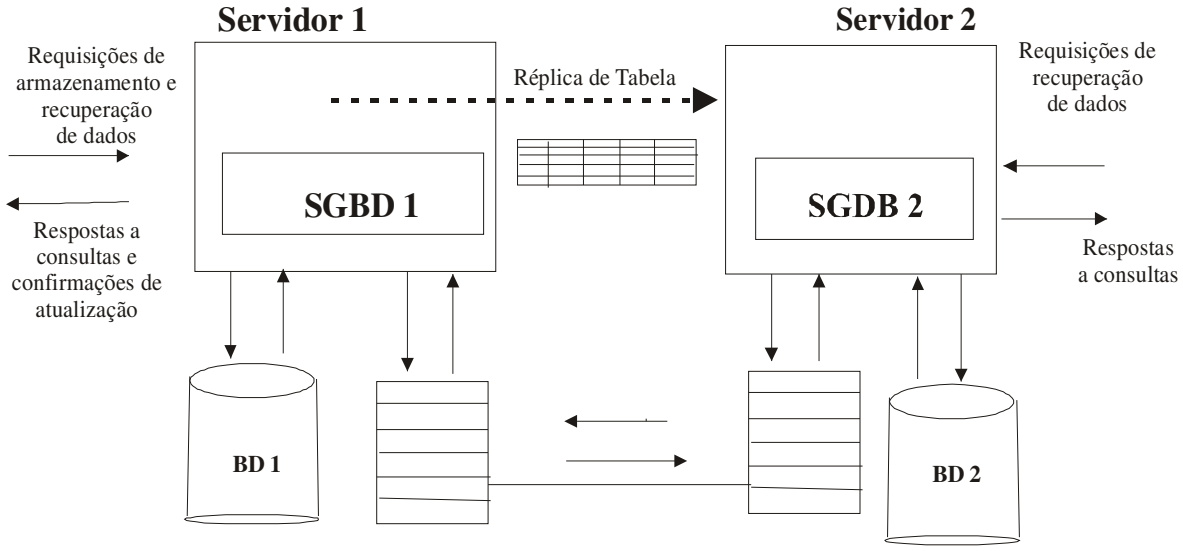

Figura 5 – Esquema de exportação de dados no extrato simples [9].

- **Extrato Associado a uma Data Hora:** Consiste em uma variação do extrato simples. Contendo a utilização de uma referência a um ponto no tempo que permite que a aplicação ou usuário avalie a utilidade do dado e decida pela necessidade de sua substituição por

outro extrato, mais atualizado. São produzidos através do uso de utilitários ou programas de aplicação específicos [9].

- **Extrato controlado:** É uma variação do tipo anterior. Onde além das tuplas o BD destino recebe uma informação que vincula o extrato a um instante no tempo ou a um somatório de valores, com a substituição da referência a um ponto no tempo por um valor agregado. Possibilitando à aplicação estabelecer uma correlação entre a porção copiada e o valor total. Os mecanismos de produção também são baseados no princípio de exportação de dados (conforme ilustrado na figura 6). [9]

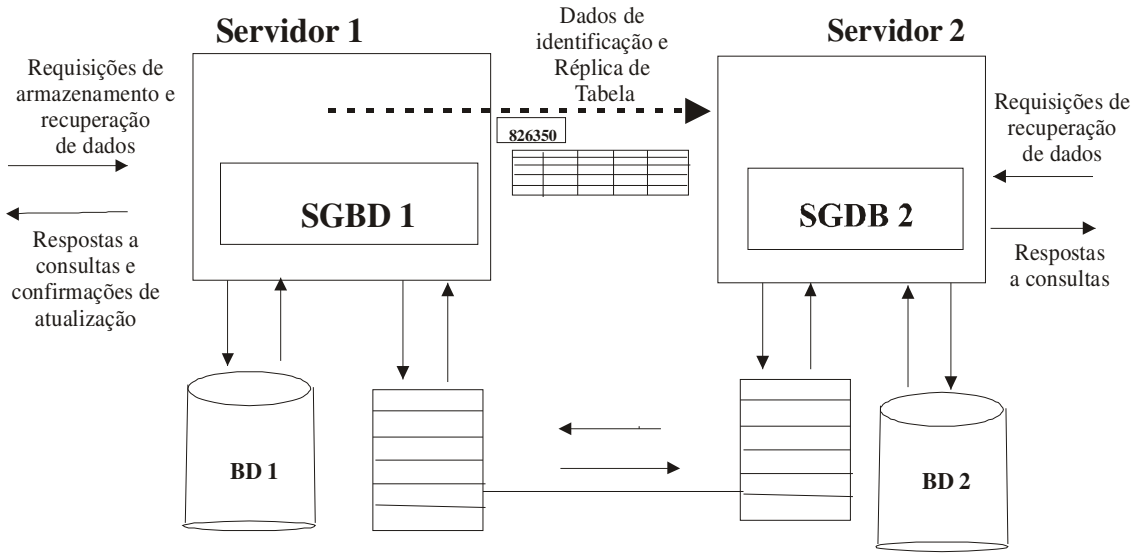

Figura 6 – Esquema de exportação de dados no extrato controlado [9].

- **Extrato Periodicamente Atualizado:** O funcionamento é semelhante ao extrato simples, a diferença é controle da passagem do tempo necessário para a realização da cópia substitutiva [9]. Este tipo de extrato pode ser periodicamente substituído (descartado) de acordo com as especificações dos usuários. O mecanismo básico utilizado consiste na criação de um procedimento *batch*, executado periodicamente para efetuar a substituição da cópia existente (*refresh*). Este procedimento pode ser um utilitário do próprio SGBD ou programa de aplicação específico (conforme ilustrado da figura 7) [9]. A execução dos processos que efetuam o *refresh* não deve ser deixada sob controle do usuário final, mas gerenciada pelo grupo de administração do BD [9].

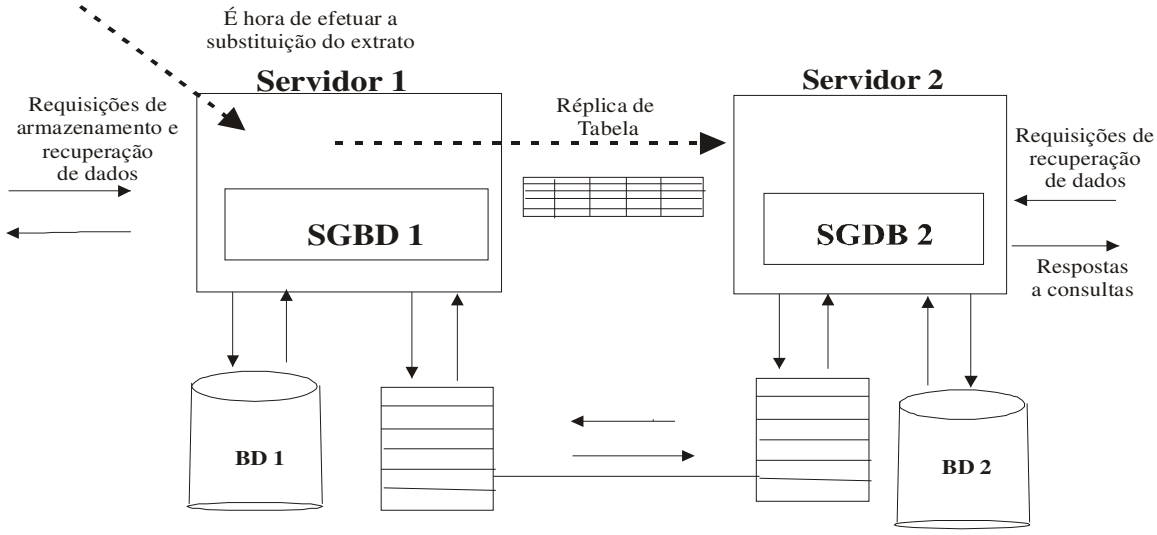

Figura 7 – Esquema de manipulação de dados no extrato periodicamente atualizado [9].

Existem dois métodos para criar um extrato atualizado:

- Realizar uma cópia completa para o BD de destino em cada intervalo. Este método é mais simples e pode ser preferido para pequenos extratos [10].
- Realizar uma cópia completa e então executar cópias incrementais em cada intervalo. Este método minimiza o fluxo de dados [10].

3.5.2.2 – Réplicas

- **Atualização Síncrona (contínua):** Implica na atualização simultânea de todas as cópias pelas transações. Tem como vantagem a garantia de que todos os usuários e todas as transações possuem uma visão consistente dos dados, não importando sobre qual cópia estão operando. E como desvantagens [9]:
    - A intolerâncias à falhas: considerando o caso de um BDD em que existem réplicas de uma relação em sites diferentes, por exemplo. Ao executar uma transação, se houver uma falha (de comunicação ou pela indisponibilidade do

processador etc) em qualquer um destes sites a transação de atualização será cancelada. Neste caso, o sistema distribuído passa a se comportar como centralizado, mas com maior grau de vulnerabilidade a falhas. Ou seja, aumenta a indisponibilidade do sistema global para a realização de transações de atualização dos BD's. [9].

- Perda de desempenho de transações: no caso de uma transação que atualize o mesmo dado de relações replicadas em sites diferentes, a mesma terá seu desempenho afetado pelos custos de comunicação. Além disso, em cada um destes sites há a concorrência de outras transações, o que pode significar espera pelo acesso a dados ou mesmo o envolvimento em situações de bloqueio mútuo (*deadlock*), podendo determinar o cancelamento da transação [9].

- **Atualização Assíncrona (Periódica):** Esta outra forma de manter a consistência entre as cópias de um BD, que funciona da seguinte forma:
  - Inicialmente, são geradas cópias do BD de origem, que são exportadas para o BD destino;
  - As atualizações são feitas apenas nas atualizações do BD de origem, as alterações são registradas em um log, periodicamente, são aplicadas sobre as cópias, restaurando a consistência entre os BD's [9].

Tem como vantagem o fato de possibilitar maior flexibilidade para execução de transações, comparando com o a atualização síncrona; caso um SGBD remoto não esteja ativo no momento da execução da transação, ele poderá ser conectado posteriormente e ter o seu BD atualizado a partir do log produzido pelo SGBD de origem da transação global [9].

Quando muitos sites são capazes de atualizar os dados e os mesmos possuem alta volatilidade, é sugerimos retornar ao banco de dados centralizado. O esforço para manter, atualizar e resincronizar é muito grande para dados muito voláteis [10].

3.6 - O Problema de Alocação de Fragmentos

Um esquema de alocação é um mapeamento que especifica para cada fragmento o (s) site (s) no (s) está armazenado [1].

Este problema ocorre quando obtemos o conjunto dos fragmentos a serem alocados aos diferentes BD's. Neste caso, os conjuntos são disjuntos, isto é, existem dados que são utilizados por programas de aplicação executados em mais de um local. Devido a estas situações, deve-se optar entre alocação em um único BD, utilizando a centralização dos fragmentos, e a alocação de cópias dos dados nos diferentes locais onde eles são requeridos, utilizando técnicas de replicação [9].

A escolha dos sites e o grau de replicação dependem das metas de desempenho e disponibilidade do sistema e dos tipos de freqüência de transações submetidas em cada site [1].

Para ser decido quanto à localização dos fragmentos existe a necessidade de considerar também as restrições da tecnologia utilizada para implementar o sistema de banco de dados distribuídos.

3.7 – Esquemas Conceituais Locais

De acordo com a distribuição dos segmentos entre sites, obtém-se os esquemas conceituais locais, que descrevem os esquemas existentes em cada um desses sites.

3.8 – Projeto Lógico

A fase do projeto lógico objetiva transformar o modelo conceitual, obtido na primeira fase, em um modelo lógico. O modelo lógico define como o BD será implementado em um SBGD específico [11]. Depois de conhecido o SGBD, o processo de projeto prossegue até a geração dos comandos de definição de dados. Em BD's centralizados, esta geração de comandos é única, enquanto em BDD's pode ser necessária uma geração de comandos para cada local onde serão armazenados os dados [9].

4 TRANSAÇÕES

As consultas e outras ações estão agrupadas em transações, onde uma transação é uma unidade lógica de trabalho. O procedimento de reiniciação recuperará qualquer transação concluída com sucesso que não tenha conseguido fazer suas atualizações serem gravadas fisicamente antes da queda [4].

Em um SGBDD, há dois tipos de transações que devemos considerar:

- Transações locais: são aquelas que mantêm acesso a um único site, justamente no qual ela se inicia, e as atualizações são feitas somente a base de dados local [4] [8];
- Transações globais: por outro lado, são aquelas que mantêm acesso a diferentes sites, ou a outro site diferente daquele em que a transação iniciou e as atualizações são feitas em diversas bases de dados locais [4] [8].

No ambiente de BDD os usuários devem ter acesso aos dados de forma transparente. Isso significa que, na elaboração de consultas, o usuário possui a visão de um único BD. As consultas podem ser formuladas diretamente pelos usuários finais, através de ferramentas de acesso do BD, podem estar contidas em aplicações desenvolvidas por programadores. Independente da origem da consulta, sua composição é feita, parcial ou integralmente, por comandos de manipulação de dados. Cabendo ao SGBDD a transformação da consulta expressa em linguagem de alto nível em uma estratégia de execução eficiente, expressa em termos de comandos de baixo nível aplicados sobre os diversos BD's locais. Esses comandos são submetidos ao SGBD para execução um-a-um. Isso significa que, o SGBD só receberá um novo comando para processamento após haver retornado o resultado da execução do comando anterior [9].

Quando se trata do um modelo relacional, uma consulta consiste em duas partes [5]:

- Especifica os domínios da relação a ser recuperada;
- Especifica o predicado, o qual é uma qualificação representando as propriedades que se deseja do conjunto a acessar. Sendo que, a complexidade aumenta com o número de operadores lógicos e de operadores de comparação incluídos na qualificação.

As consultas em um BD relacional contêm operações que podem ser classificadas nas seguintes categorias [5]:

- Operações de recuperação:
    - Recuperações simples: O predicado representando a propriedade do conjunto a ser recuperado é definido sobre a mesma relação.
    - Recuperações múltiplas: O predicado é definido sobre múltiplas relações.
- Operações de atualização:
    - Atualização simples: Só atualizam valores de uma tupla especifica de uma relação.
    - Atualização múltipla: Atualizam os valores de múltiplas tuplas.
    - Adição de tuplas.
    - Eliminação de tuplas.

As transações nos ambientes distribuídos possuem as mesmas características das transações executadas sobre os BD's centralizados. Portanto, as transações executadas sobre os BDD's também devem possuir as propriedades ACID (Atomicidade Consistência Isolamento Durabilidade), e os SGBDD's devem implementar as funções de controle correspondentes [9].

Estas propriedades, que se destinam a assegurar a integridade dos dados, são as seguintes:

- Atomicidade: as transações são atômicas, ou seja, tudo ou nada [4]. O sistema de BD mantém um registro (em disco) dos antigos valores de quaisquer dados sobre os quais a transação executa uma gravação e, se a transação não se completar, os valores antigos são restabelecidos para fazer com que pareça que ela nunca foi executada [8].
- Consistência: uma transação transforma um estado consistente em outro estado consistente, preservando o estado consistente do banco de dados, sem necessariamente preservar a consistência de todos os pontos intermediários [4].

- Isolamento: as transações são isoladas uma das outras. Isto é, embora em geral haja muitas transações sendo executadas concorrentemente, as atualizações de qualquer transação dada são ocultas de todas as outras até que essa transação seja finalizada [4].
- Durabilidade: uma vez iniciada uma transação, suas atualizações sobrevivem no BD mesmo que haja falhas no sistema [4]. Podemos garantir a durabilidade considerando um dos itens a seguir [8]:
    - As atualizações realizadas pela transação foram gravadas, antes da transação completar-se.
    - Informações gravadas são suficientes para que o BD possa reconstruir essas atualizações quando o sistema for reiniciado após uma falha.

4.1 - Estrutura do Sistema

Cada site possui seu próprio gerenciador local de transações, cuja função é garantir as propriedades ACID das transações executadas localmente. Os diversos gerenciadores de transações cooperam para executar as transações globais. Onde, cada site contém dois subsistemas [8]:

- Gerenciador de transações: administra a execução daquelas transações que mantêm acesso aos dados locais armazenados no site local. Sendo que, cada uma dessas transações pode ser uma transação local ou parte de uma transação global. Onde, cada gerenciador é responsável por [8]:
    - Manutenção de um log para propósitos de recuperação;
    - Participação em um esquema de controle de concorrência adequado para coordenação da execução de transações concorrentes em um mesmo site.
- Coordenador de transações: coordena a execução de várias transações iniciadas naquele site, podendo ser tanto locais quanto globais. Para cada uma dessas transações, o coordenador é responsável por [8]:
    - Iniciar a execução da transação;
    - Quebrar uma transação em um número determinado de subtransações e distribuí-las pelos sites apropriados para execução;
    - Coordenar o término das transações, que podem resultar em efetivações ou em interrupção em todos os sites.

A estrutura do gerenciador de transações é similar em vários aspectos à estrutura usada nos sistemas centralizados. O subsistema de coordenação de transações não é necessário em um ambiente centralizado, já que uma transação tem acesso somente a um único site [8].

Neste sistema distribuído, se o coordenador e os participantes forem executados em máquinas diferentes, então a falha por parte do coordenador poderá manter algum participante esperando durante um longo tempo pela decisão por parte do coordenador – e, quando estiver esperando, quaisquer atualizações feitas pela transação através desse participante deverão ser mantidas ocultas de outras transações (isto é, atualizações provavelmente terão de ser mantidas bloqueadas) [4].

4.2 - Atomicidade de Transações

Em um SGBD que implementa padrão SQL, uma transação é um bloco de comandos da DML (Linguagem de Manipulação de Dados) encerrado por um comando da DLC que indica se os resultados devem ser tornados permanentes no banco de dados (COMMIT) ou desfeitos (ROLLBACK) [9].

- COMMIT: é a operação que indica o término de uma transação bem-sucedida. Ela informa ao gerenciador de transações que uma unidade lógica foi concluída com sucesso, o banco de dados está novamente em uma situação consistente e todas as atualizações feitas por esta unidade de trabalho podem agora ser validadas ou tornadas permanentes.
- ROLLBACK: é a operação que assinala o término de uma transação mal-sucedida. Ela informa ao gerenciador de transações que algo saiu errado, que o banco de dados pode estar em um estado inconsistente, e que todas as atualizações feitas pela unidade lógica de trabalho até agora devem ser retomadas ou desfeitas.

Quando os resultados de uma transação são efetivados, as modificações que eventualmente tenham sido realizadas no BD são tornadas visíveis para outras transações. Neste caso dizemos que o BD mudou de estado, isto é, passou para um estado diferente do inicial. Caso a transação tenha os seus resultados desfeitos, dizemos que o BD permanece no estado inicial [9].

O SGBD de origem da transação, aquele com quem o usuário ou o programa de aplicação iniciou o processo de interação, atua como um coordenador da execução distribuída. Cada um dos SGBD's acionados como coordenador durante o processo de execução da transação é visto como um participante do processo de execução distribuída [9].

Para a garantia da atomicidade, todos os sites envolvidos devem refletir ou remover todos os efeitos das ações elementares executadas em favor da particular transação [5]. É preciso que transação seja efetiva em todos os sites ou então abortada em todos eles. Para assegurar que isso ocorra, quando do término da transação, o coordenador da transação precisa executar o protocolo de efetivação (*commit protocol*) para que o processo de desenvolva de forma ordenada [8], mesmo na presença de falhas repetidas e imprevisíveis que podem afetar a operação normal de vários dentre os nós participantes [5]. Porém, se um único SGBD adotar um caminho diferente daquele que foi seguido pelos demais participantes, a atomicidade da transação será violada e, conseqüentemente, a consistência do BD global [9].

O protocolo de sincronismo normalmente utilizado para sincronismo de processos em ambientes distribuídos onde dois ou mais sistemas estão envolvidos [9], entre os mais simples e mais utilizados protocolos de efetivação está o protocolo de duas fases (*two-phase commit protocol* – 2PC). Existem implementações de protocolos de sincronismo que acrescentam mais passos, buscando aumentar, ainda mais, a garantia da atomicidade das transações na presença de falhas múltiplas durante o processo de efetivação [9]. Outra alternativa é o protocolo de três fases (*three-phase commit protocol* – 3PC), que elimina algumas desvantagens dos 2PC, apesar de ser mais complexo e aumente o *overhead* [8].

4.2.1 - Efetivação de Duas Fases (2PC)

Este protocolo é chamado de efetivação em duas fases por executar a ordem de efetivação (COMMIT) emitida pela aplicação em duas fases [9]. Estas fases serão abordadas a seguir:

- **Primeira Fase:**

    - *O coordenador:* o SGBD que iniciou a transação, avisa a todos os SGBD's participantes sobre a intenção de efetuar a efetivação dos resultados da transação, e a guarda que todos os SGBD's notificados se manifestem;
    - *Cada SGBD participante*: quando recebe o aviso do coordenados, tenta salvar os resultados parciais em memória permanente. Caso consiga efetuar o salvamento, responde favoravelmente à efetivação e, caso não consiga, responde negativamente. Sendo que, neste ponto mesmo após haver se

manifestado pela efetivação, o SGBD não tornará os resultados visíveis para outras transações, pois a decisão final ainda virá do coordenador, neste caso o SGBD local está preparado [9].

- **Segunda Fase:**

  - *O coordenador*, após receber os votos de todos os SGBD's participantes, faz a apuração do resultado. Se houver pelo menos um voto contrário à efetivação, o coordenador avisa a todos os SGBD's participantes que a transação deve ser cancelada (*abort*), isto é, que os resultado devem ser desfeitos por todos os SGBD's. Caso contrário, quando todos os votos são favoráveis, o coordenador notifica os SGBD's participantes que os resultados devem ser efetivados nos BD's [9].
  - *Cada Participante*, quando recebe a decisão final do coordenador, adota a ação indicada. Após completar a execução, ele notifica o coordenador, possibilitando que haja o controle centralizado do encerramento do processamento e, todos os locais [9].

4.2.2 – Efetivação em Três Fases (3PC)

O protocolo foi projetado para evitar a possibilidade de obstrução nos casos restritos de falhas possíveis. Este protocolo exige que [8]:

- Não ocorra nenhum particionamento da rede.
- Um número máximo de participantes pode falhar enquanto o protocolo 3PC estiver sendo executado por transação.
- Em um tempo qualquer, ao menos um site mais do que no número máximo precisam estar ativos.

Este protocolo consegue impedir a obstrução adicionando uma fase extra, onde é tomada uma decisão preliminar sobre o destino da transação. Essa informação distribuída aos sites participantes, como conseqüência dessa decisão preliminar, permite que uma decisão seja tomada no caso de falha do coordenador [8].

4.2.3 - Comparação Entre Protocolos

O protocolo 2PC é bastante utilizado, apesar de sua potencial obstrução. A probabilidade de ocorrência de obstrução é relativamente inferior ao custo extra do protocolo 3PC, invalidando seu uso. A vulnerabilidade do 3PC em combinar falhas é outro problema prático. Essa desvantagem pode ser sobreposta pelo protocolo em nível de rede, mas essa solução aumenta o overhead.

Podemos otimizar ambos os protocolos para reduzir o número de mensagens enviadas e reduzir o número de vezes em que os registros precisam ser escritos em memória estável. O protocolo 3PC pode ser implementado, permitindo que se aumente o número máximo de participantes que possam falhar antes que um novo coordenador tome a decisão de efetivação [8].

4.3 - Execução de Consultas e Gerência de Transações

O processamento de consultas num BDD corresponde à tradução de requerimentos formulados numa linguagem de alto nível sobre um site da rede, numa seqüência de instruções elementares sobre os dados armazenados em vários BD locais. Cada consulta formulada por um usuário tem de ser separada em subconsultas, executáveis pelos diferentes SGBD's locais. Além disso, a partir dos resultados das subconsultas o SGBD resolve um conjunto de operações lógicas expressos na consulta. O resultado dessas operações constitui o resultado final que será retornado ao usuário. Sendo que, a evolução da consulta corresponde à seqüência de traduções até o esquema interno [5].

O processamento em um BDD se difere do caso centralizado em três aspectos básicos [5]:

- O diretório de dados é distribuído e a sua forma de armazenamento afeta fortemente a eficiência do processador de consultas;
- Como o banco é distribuído, uma relação do esquema conceitual pode estar fragmentada e replicada ao longo da rede. O processador deverá selecionar os fragmentos, localizar as cópias apropriadas e eventualmente movê-las para que a consulta possa ser processada;
- Se o sistema for heterogêneo, o processador deverá, ainda, efetuar traduções entre modelos de dados distintos.

O problema de processamento de consultas num sistema heterogêneo é afetado pela complexidade dos mapeamentos, assim como das diferentes representações e estratégias de acesso, as quais devem convergir para uma única função objetivo [5].

O processamento de consultas inclui um módulo de otimização cujo objetivo é escolher uma seqüência de subconsultas e transferência de dados, e de cópias de dados no caso de replicação, a serem usadas com a finalidade de minimizar uma certa função de custo. A saída do módulo de otimização inclui acessos a serem realizados em nós remotos e transferências de dados de um site para outro [5].

A otimização pode ser feita pelos SGBD's basicamente através de duas maneiras [9]:

- O uso de heurísticas: regras derivadas de experiência de execução das consultas. Envolve o reconhecimento da forma de comando empregada, o que pode determinar um conjunto de métodos indicados para a recuperação dos dados, e a maximização do uso doa índices disponíveis para obtenção do conjunto-resposta (resultado final).
- O uso de abordagens baseadas em otimização de custos: utiliza estatísticas sobre o BD para descobrir a estratégia de execução de menor custo, produzindo um plano de consulta. Os custos são medidos em termos das unidades de tempo gastas para execução das ações necessárias à produção da resposta desejada pelo usuário ou aplicação. Os custos mais relevantes nos ambientes centralizados são o custo de acesso aos dados e o custo de processamento. Além dos fatores considerados na geração do plano de consulta otimizados em BD's centralizados, um SGBDD também deve considerar o custo do acesso aos dados através das redes de comunicação. No caso de redes de longa distância, os otimizadores de consulta devem ser orientados para evitar o custo da movimentação de grandes volumes de dados entre diferentes locais para realização das operações O plano de consulta gerado determina o método de obtenção das tuplas (linhas) de uma tabela, com efeitos diretos sobre o custo de processamento [9].

Cada um dos comandos é também analisado pelos SGBD's locais dos sites remotos e a estratégia local ótima de recuperação de dados é executada [5].

A taxa dos dados e o retardo de acesso são ambos fatores importantes na escolha de uma estratégia. Os tempos de processamento e E/S (Entrada/Saída) provavelmente serão desprezíveis comparados com o tempo de comunicação das estratégias inviáveis. Além disso, algumas estratégias permitem processamento paralelo em sites. Assim, o tempo de resposta para o usuário poderia ser menor que em um sistema centralizado [4].

4.4 - Fatores Críticos no Processamento de Consultas

Um dos mais importantes problemas no processamento de consultas num ambiente distribuído é a eficiência. Dentre outros fatores críticos que podem afetar essa eficiência, temos primeiramente a transferência de dados entre os sites. Essa transferência deve ser minimizada, o que é um dos objetivos principais do sistema. Para atingir esse objetivo, são empregadas técnicas de otimização para separar consultas múltiplas em tantas subconsultas simples quanto possível, de modo que o movimento de tabelas de um site para outro seja mínimo [5].

O custo de transmissão é significativamente maior do que o custo de processamento local e do que a transferência de dados entre os dispositivos de armazenamento de um sites. Contudo, a distribuição do sistema torna possível o processamento em paralelo das subconsultas e das operações lógicas, o que contribui para a diminuição do tempo de resposta [5].

4.5 - Estratégias de Processamento de Consultas

O acesso a dados que estão espalhados sobre sites de uma rede implica necessariamente na transmissão de dados sobre linhas de comunicação. O SGBDD deve considerar esse fato e decidir a forma eficaz quanto ao processamento local e às transmissões dos dados com a finalidade de processar eficientemente consultas distribuídas [5].

Essa ordem de processamento de dados e transmissões é chamada de estratégia de processamento de consultas. Pode-se considerar dois tipos de estratégias [5]:

- Estratégia distribuída: se distribuem, para certa consulta, todas as respostas parciais das subconsultas, executando-se operações lógicas no diferentes sites de distribuição. A resposta final é gerada como produto da união dos diferentes processamentos locais nos diferentes sites.
- Estratégia centralizada: consistem em agrupar para uma determinada consulta todos o resultados parciais das subconsultas e, em seguida, executar todas as operações lógicas num simples site. A característica principal desta estratégia é que na fase final só se inclui um processamento local da consulta, isto é, depois de se ter em um único site a reunião de todos os dados distribuídos.

4.6 - Critério de Correção da Execução

No caso da execução das transações distribuídas, o critério usualmente empregado para verificação da sua correção também é o da serializabilidade das execuções. Ou seja, das duas transações T1 e T2, sua execução concorrentes deve produzir os mesmos resultados que seriam obtidos caso fossem executadas isoladamente, em seqüência [9].

Portanto, surge um complicador, representado pelo diferentes locais onde as partes das transações globais conflitantes, sua subtransações, são executadas concorrentemente, as execuções sejam serializáveis [9].

No caso de um BDD onde tenha sido utilizada a replicação de dados, há necessidade de se observar também um outro requisito: a ordem de serialização das subtransações de T1 e T2 deve ser a mesma em todos os locais [9].

5 CONTROLE DE CONCORRÊNCIA

Um SGBD, suportando bancos de dados com várias aplicações, deverá permitir acesso concorrente aos dados. Pois, quanto maior for o nível de concorrência permitido, tanto melhor será o tempo de resposta do sistema como um todo. Os mecanismos que controlam o acesso concorrente ao banco impõem uma sobrecarga adicional sobre o desempenho do SGBD. Os procedimentos que harmonizam o paralelismo no SGBD serão conhecidos por mecanismos de controle de concorrência [5].

A tarefa fundamental dos módulos de controle de concorrência é propiciar uma serialização das ações elementares, uma transação não tem como acessar objetos modificados por outra transação que ainda não tenha completado. Assim, num dado instante, o uso de cada objeto do BD é privativo da transação ativa naquele instante que irá modificar o seu valor. Um mesmo objeto pode ser lido por várias transações, desde que nenhuma outra transação em andamento venha a modificar seu valor [5].

Os sites da rede operam de forma bastante independente, embora o controle de concorrência deva ser efetivado de forma global, abrangendo informação que pode estar espalhada por vários sites [5].

5.1 - Mecanismos Básicos de Controle de Concorrência

O controle de execução concorrente das transações pode ser feito com o uso de diferentes mecanismos, que são classificados em:

- *Métodos pessimistas*: partem da premissa de que as transações conflitam entre si e estabelecem protocolos para acesso exclusivo aos dados, assegurando o isolamento das ações da transação e a correção de suas execuções;
- *Métodos otimistas*: baseiam na premissa que as transações não possuem operações conflitantes entre si, possibilitando livre acesso aos dados. Ao final da execução de uma transação fazem uma avaliação, caso haja alguma violação do isolamento a transação é desfeita.

5.2 – Visão Geral de Controle e Recuperação

Para o controle de concorrência e recuperação surgem inúmeros problemas que são [1]:

- Lida com várias cópias dos itens de dados: o método de controle de concorrência é responsável por manter consistência e o método de recuperação por tornar uma cópia consistente com outras cópias se o site no qual a cópia estiver armazenada falhar;
- Falhas de sites individuais: o SGBD deve continuar a operar quando um ou mais sites individuais falharem. Quando um site se recuperar, eu banco de dados locais deve ser atualizado antes de se juntar ao sistema;
- Falhas dos *links* de comunicação: o sistema deve ter capacidade de lidar com falhas ou nos *links* de comunicação que conectam os sites. Como por exemplo, pode ocorrer um particionamento da rede, ou seja, os sites são desmembrados em partições nas quais os sites dentro de cada partição podem se comunicar apenas um com outro;
- Commit Distribuidos: podem surgir problemas no commit de uma transação que esteja acessando um banco de dados em vários sites, para lidar com esse problema é usado o protocolo de commit de duas fases;
- DeadLoock Distribuído: pode ocorrer DeadLoock em vários sites;

5.3 – Protocolo da Maioria

Neste protocolo, cada site mantém um administrador de bloqueios local, cuja função é gerir as soluções de bloqueio e desbloqueio para os itens de dados que estão armazenados naquele site. Quando uma transação deseja o bloqueio do item de dado Q, que não é replicado e reside no site $S_{i,}$ é enviada uma mensagem para o administrador de bloqueios do sites S_i pedindo o bloqueio (em um modo de bloqueio em particular). Se oi modo de bloqueio solicitado para determinado item de dado Q não é compatível, a solução espera até que possa ser conseguido.Uma vez determinado que o bloqueio pode ser realizado, o administrador de bloqueios manda uma mensagem de volta ao solicitador indicando que ele obteve o bloqueio solicitado. O esquema possui a vantagem de ser de fácil implementação. Exige a troca de duas mensagens para o tratamento das solicitações de desbloqueio.

Se o item de dado Q é replicado em n diferentes sites, então as mensagens de solicitação de bloqueio devem ser enviadas para mais de metade dos n sites nos quais Q é armazenado. Cada administrador de bloqueio determina se o bloqueio pode ser atendido imediatamente. Como antes, a resposta é adiada até que a solicitação pode ser atendida. A transação não opera em Q até que tenha obtido com sucesso sobre as replicas de Q.

5.4 – Protocolo Parcial (Biase)

Similar ao protocolo da maioria, a diferença é que é dado um melhor tratamento ás solicitações para bloqueios compartilhados que o dado para as solicitações de bloqueio exclusivo.

O sistema mantém um administrador de bloqueios em cada site, cada administrador gerencia os bloqueios de todos os itens de dados armazenados naquele site. Os bloqueios compartilhados e exclusivos são tratados diferentemente.

- Bloqueios compartilhados: Quando uma transação precisa do bloqueio de um item de dado Q, ela simplesmente pede o bloqueio de Q para o administrador de bloqueios em um site que contém uma réplica de Q.
- Bloqueios exclusivos: Quando uma transação precisa do bloqueio de um item de dado Q, ela pede o bloqueio de Q para o administrador de bloqueios de todos os sites que possuam uma réplica de Q.

Como antes, a resposta à solicitação é adiada até o bloqueio possa ser realizado.

5.5- Cópia primária

No caso de replicação de dados, podemos escolher uma das réplicas como cópia primária, assim para cada item de dado Q, a cópia primária de Q deve residir precisamente em um site, ao qual designamos site primário de Q.

Assim, a cópia primária permite que o controle de concorrência para dados replicados possa ser feito de modo similar ao usado para dados em replicação. Essa similaridade leva em conta uma implementação simples. Entretanto, o site primário de Q falha, torna-se inacessível, mesmo que outros sites que contenham uma réplica de Q possam estar ativos.

5.6 - Controle de Concorrência Baseado em Cópia Distinta de um Item de Dado

A idéia de designar uma determinada cópia de cada item de dado, como uma cópia distinta. Esta idéia mantém os bloqueios para aquele item e todas as solicitações de bloqueio e desbloqueio são todas enviadas aos sites que mantém essa cópia. Para isto são usadas técnicas que serão descritas a seguir:

- Técnica do site primário (principal): todas as copias distintas são mantidas no mesmo site, com um site de backup (cópia). Neste método, um único site primário é designado para ser o site coordenador para todos os itens do banco de dados. Pelo fato de que todas as solicitações são envidadas a um único site, possivelmente sobrecarregando esse site e causando um gargalo no sistema. Falhas no site primário paralisam o sistema, pois todas as informações sobre bloqueio são mantidos nesse site. Isso pode limitar a confiabilidade e a disponibilidade do sistema .
- Site Primário com Site de Backup: esta abordagem descreve uma das desvantagens do método do site primário, designando um segundo site para ser site de backup. Todas as informações sobre são mantidas em ambos os sites, o primário e o backup. Em caso de falha do site primário, o site de backup assumo como site primário, e um novo site de backup é escolhido. Todavia isso desacelera o processo de aquisição de bloqueio, porque todas as solicitações e concessões de bloqueios devem ser gravadas em ambos os sites, antes que uma resposta seja enviada para a transação solicitante.
- Técnica da cópia primária: esse método tenta distribuir a carga da coordenação de bloqueio entre vários sites fazendo com que as copias distintas de diferentes itens de dados sejam armazenadas em diferentes sites. Falhas em um site afetam quaisquer transações que estejam acessando bloqueios em itens cujas cópias primarias residam naquele site, mas outras transações não são afetadas.

5.7 – Controle de Concorrência Distribuída Baseada em Votação

Neste método não existe cópia distinta, uma solicitação de bloqueio é enviada a todos os sites que incluem uma copia de itens de dados. Cada cópia mantém seu próprio bloqueio e pode conceder ou negar o pedido. Se uma transação que requisita um bloqueio que é concedido pela maioria de cópias, ela segura o bloqueio e informa todas as cópias que tiveram o bloqueio concedido. Se uma transação não receber a maioria de votos concedendo a mesma o bloqueio de um determinado espaço de tempo ela cancela a solicitação e informa a todos os sites sobre o cancelamento.

6 CONCLUSÃO

O avanço da tecnologia de *hardware* e de comunicação proporcionou o surgimento do sistema de banco de dados distribuídos. Neste sistema trata-se de um sistema de banco de dados, onde os dados estão dispersos logicamente e/ou fisicamente. Além dos dados, também estão distribuídos o processamento e o controle. Uma das indicações de uso deste sistema é para manipular grandes massas de dados, por isso é aplicável em grandes empresas.

Ao invés do processamento estar em um servidor central, o sistema proporciona o processamento de dados local em diversos sites, sem haver a necessidade de acessos a dados remotos. Com isso, o custo de transporte de dados pela rede é reduzido.

A tolerância a falhas é maior, pois caso haja falha em um determinado site os demais continuam a operar, sendo que, os dados podem estar armazenados em outro site podendo ser acessados. Esta é uma de suas inúmeras vantagens, que proporciona a disponibilidade e confiabilidade do sistema. Apesar disso, possui também desvantagens quando se trata de segurança, devido ao fato que, manter a segurança de dados dispersos e o envolvimento de redes de comunicação se torna complicado.

A sua implementação é complexa, apesar disso no modelo relacional torna-se menos, podendo ser implementado também, no modelo orientado a objetos, dentre outros.

As transações são feitas de modo que o usuário possa se comportar como se o sistema não fosse distribuído. Sendo que, as transações são consideradas globais quando existe acesso a dados remotos e locais quando isso não ocorre.

Em questões de projeto, se difere do caso centralizado por possuir um acréscimo: o processo de projeto de distribuição de dados, que possuem as técnicas de fragmentação e replicação. Os dados são distribuídos visando diminuir custos e aumentar o tempo de resposta, estes dados devem estar o mais próximo do local onde o número de acessos é maior.

Possui grande tendência no mercado, pois é um sistema bastante eficiente, apesar de alguns contratempos, a sua implantação é vantajosa. Contudo sua experiência prática ainda é limitada.

Como proposta para trabalhos futuros, sugere-se uma implementação de um sistema de banco de dados distribuídos. Que teria como objetivo mostrar uma visão prática deste estudo teórico. Este sistema poderá ser construído partindo do princípio de que os dados são dispersos pelos menos logicamente.

7 REFERÊNCIAS BIBLIOGRÁFICAS

[1] DATE, C. J.. Introdução a Sistemas de Banco de Dados, 2000, Editora Campus Ltda.

[2] MOLINA, Hector Garcia; ULLMAN, Jeffrey D.; WINDOM, Jennifer. Implementação de Sistemas de Banco de Dados. Edição original, 2001, Editora Campus Ltda.

[3] CASANOVA, Marco Antonio. Princípios de Sistemas de Gerência de Bancos de Dados Distribuídos. Edição revisada, 1999, Distribuição Restrita. (Disponível em: www.inf.puc-rio.br/~casanova/LivroCasanova, acessado em julho de 2004)

[4] ELMASRI, R. Navate, S. B.. Sistemas de Banco de dados; Fundamentos e Aplicações. Terceira Edição. Rio de Janeiro. LTC. 2002.

[5] COUCEIRO, Luiz Antonio Carneiro da Cunha; BARRENECHA, Hugo Fernando Spencer. Sistemas de gerência de Banco de Dados Distribuídos. Edição original, 1984, Livros Técnicos e Científicos Editora S. A.

[6] RAMKRIHAN, Raghu; GEHRKE, Johanners. Database Management Sytems. Second Edition, mcGraw-Hill international Editons

[7] RICCIONI, Paulo Roberto. Introdução a Objetos Distribuídos com Corba

[8] SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSSHAN S. Sistema de Banco de Dados. Terceira edição, Editora Makron Books, 1999.

[9] MELO, RubensN.; SILVA, Sidney Dias; TANAKA, Asterio K. Banco de Dados em Aplicações Cliente-servidor. Sexta edição, Editora Infobook, 1998.

[10] HACKATHORN, Richard D.. Conectividade de Banco de Dados Empresariais. 1993, Livraria e Editora Infobook S. A.

[11] HEUSER, Carlos Alberto. Projeto de banco de dados. Quarta edição, Editora Sagra Luzzatto, 2001.

[12] PRESMAN, Roger S.. Engenharia de Software. Quinta edição, Editora McGraw-Hill, 2002.

GLOSSÁRIO

Aplicação: é um conjunto não vazio de programas de computador que é utilizado para dar suporte a funções específicas de um negócio para um usuário (ou tipo de usuários) que ocupe cargo específico, com atribuições determinadas.

Armazenamento de dados: processo de atualização de um ou mais banco de dados co m novas transações.

Banco de dados distribuído: Banco de dados no qual os dados podem estar espalhados por diversos outros bancos menores conectados por uma rede de comunicação.

Banco de dados relacional: Banco de dados (q. v.) cuja estrutura se baseia no modelo relacional. [Nesse tipo de banco de dados, as informações são organizadas sob a forma de tabelas, sendo possível relacionar as diferentes tabelas a partir da comparação dos valores de determinadas colunas, designadas campos-chave

Banco de dados: grupo de arquivos relacionados; mais especificamente, uma coleção de dados organizados para parecerem estar em um só local de modo que possam ser acessados e utilizados em muitas aplicações diferentes.

Classes Especificas de diálogo: transferência de arquivos.

Compilador: Programa de tradução de linguagens que traduz um programa inteiro em linguagem de alto nível para linguagem de máquina.

Controle de concorrência: método de lidar com uma situação na qual duas ou mais pessoas precisam de acesso, simultaneamente, ao mesmo registro de um banco de dados.

Dados: fatos brutos que podem moldados para serem convertidos em informações.

Dicionário de dados: descrição detalhada de todos os dados usados no banco de dados.

Esquema: uma descrição da estrutura dos dados em um banco de dados, freqüentemente chamada “metadados”.

Gateway: dispositivo que opera na camada de transporte OSI ou acima desta e ligada as LANs ou redes que empregam diferentes protocolos de níveis mais altos, permitindo, assim, que redes com arquiteturas muito diferentes para se comunicar.

Hardware de computador: Equipamento físico usado para o trabalho de entrada, processamento e saída em um sistema de informação.

Imagem: Cópia exata do conteúdo de um segmento contínuo de memória (principal ou secundária) ou de arquivo.

Implementação: Processo de por em ação a solução de um problema a avaliar os resultados de modo a se realizarem melhoramentos; quinta etapa da resolução dos problemas.

Independência de dados: a capacidade de modificar a definição dos esquemas em determinado nível, sem afetar o esquema do nível superior. [8]

Informação: dados que foram modificados para uma forma significativa e útil para seres humanos.

Integração lógica: qualquer nó tem acesso potencial a todo o banco de dados.

Integridade de dados: grau de precisão de qualquer dados num arquivo.

Linguagem de definição de dados: Parte de um sistema de gerenciamento de dados que define cada elemento dado como ele aparece no banco de dados antes de ser traduzido para a forma exigida por diversos programas de aplicação.

Linguagem de manipulação de dados: ferramenta especial de um sistema de gerenciamento de banco de dados que manipula os dados de um banco de dados.

Mainframe: computador de grande porte, geralmente utilizado para problemas comerciais ou militares.

Manipulação de dados: processo de execução de cálculos e outras transformações de dados relacionadas às transações do negócio.

Modelo relacional: modelo de dado no qual todos os elementos de dados são colocados em tabelas de duas dimensões, chamadas relações que são os equivalentes lógicos dos arquivos.

Modelo: abstração ou aproximação usada para representar a realidade.

Overhead: Diz-se das despesas operacionais de um negócio, outras que não diz respeito ao trabalho e aos materiais.

Paginação: função da memória virtual que permite ao computador armazenar páginas de vários programas enquanto o restante destes programas espera no disco.

Plataforma de computação: combinação de uma configuração de hardware em particular e pacotes de software.

Ponte: dispositivo que conecta duas ou mais redes à parte do controle de acesso de uma camada do link de dados; as duas redes precisam usar o mesmo protocolo de comunicação.

Processamento centralizado: processamento de dados que ocorre em um único local ou instalação.

Processamento de dados: Manipulação dos dados em um sistema computacional, que tecnicamente equivale à execução de instruções por processador(es), e que abrange a entrada, verificação, armazenamento, recuperação, transformação e produção de novas informações a

partir dos dados iniciais.[O termo é us. esp. para aplicações comerciais a administrativas, como controle de estoque e folha de pagamentos.] [Tb. se diz apenas processamento.]

Processamento descentralizado: processamento de dados que ocorre quando os dispositivos são colocados em diversos lugares diferentes.

Processamento distribuído: processamento de dados que ocorre quando os computadores estão em lugares remotos, mas estão conectados uns aos outros por meio de dispositivo de telecomunicação.

Processamento: conversão ou transformação dos dados em resultados úteis.

Processos: programas em execução.

Projeto físico: Parte de um projeto que converte o modelo de sistema lógico abstrato em especificações de equipamentos, hardware, software e outros recursos físicos.

Projeto lógico: Parte do projeto que apresenta a descrição do nível geral de recursos, do processo operacional e da natureza das saídas que a solução deve requer; descreve o que a solução fará, e não como ela funciona fisicamente.

Protocolo: Conjunto de regras que controlam a transmissão entre dois componentes de uma rede de comunicação.

Recuperação: Fornecer (informação requisitada por programa ou usuário), após tê-la localizado e lido em dispositivo de memória.

Rede de longa distância (WAN): rede que une grandes regiões geográficas usando microondas e transmissão via satélite ou linhas telefônicas.

Rede local (LAN): rede que conecta computadores e dispositivos dentro da mesma área geográfica.

Redes: usadas para conectar computadores e dispositivos de hardware em um prédio, ao redor do país ou ao redor do mundo, para viabilizar a comunicação eletrônica.

Redundância de dados: presença de dados duplicados em múltiplos arquivos de dados.

Relação: uma organização de dados em uma tabela bidimensional, na qual as linhas (tuplas) representem entidades básicos ou fatos de algum tipo, e colunas (atributos) representam propriedades dessas entidades.

Roteador: dispositivo que opera ao nível no modelo OSI e apresenta um software de comunicação mais sofisticado do que as pontes. Enquanto as pontes passam tudo o que chega até elas, os roteadores determinam caminhos alternativos para um destino final.

Segurança de dados: controle que visa à prevenção do uso não-autorizado de dados e que garante que os dados não sejam alterados ou destruídos ou destruídos acidentalmente.

Servidor: computador com um grande disco rígido. Sua função é permitir que outros dispositivos compartilhem arquivos e programas.

Sistema de informações: Sistema que manipula informações por meio do uso de banco de dados.

Sistema distribuído: Aquele em que diversos computadores interconectados, cada qual com capacidade de realizar independentemente certas funções ou tarefas, podem trabalhar coordenadamente em processos que envolvam informações ou recursos remotos, de tal modo que a distribuição das informações e tarefas entre os diversos componentes não se torne aparente ao usuário, o qual se limita a operar a máquina e os recursos locais.

Sistema operacional: software de sistemas que gerencia e controla atividades do computador. Conjunto de programas que controlam o hardware do computador e agem como uma interface com os programas aplicativos.

Sistema: conjunto de elementos ou componentes que interagem para atingir um objetivo.

Software aplicativo: programas projetados para manipular o processamento de uma determinada aplicação de computador.

Software de computador: instruções pré-programadas que coordenam o trabalho dos componentes de hardware do computador para desempenhar os processos exigidos pelos sistemas de informação.

Subesquema: arquivo que contém a descrição de um subconjunto do banco de dados e identifica quais usuários podem executar modificações nos itens de dados deste subconjunto.

Telecomunicações: transmissão eletrônica de sinais para comunicações incluindo meios como telefone, rádio e televisão.

Transação: Em um sistema de informações, operação lógica que não fere a coerência dos dados armazenados.

Visão dos dados: proporciona ao usuário uma visão abstrata dos dados, isto é, o sistema acaba por ocultar determinados detalhes sobre a forma de armazenamento e manutenção desses dados. [8]

Visão física: apresentação dos dados como são realmente organizados e estruturados em meios de armazenamento físico.

Visão lógica: apresentação de dados como serão vistos pelos usuários finais ou especialistas de uma empresa.