Garantindo o Armazenamento Persistente para Objetos Programas

Os bancos de dados podem ser usados para oferecer um armazenamento persistente aos objetos programas e estruturas de dados. Essa é uma das

principais justificativas para os sistemas de banco de dados orientados a objeto. As linguagens de programação têm uma estrutura de dados complexa,

como os tipos de registro em Pascal ou as definições de classe em C++ ou Java. Os valores das variáveis dos programas são descartados, uma vez que o

programa termina sua execução, a não ser que o programador os armazene, explicitamente, em arquivos permanentes, os quais, normalmente, envolvem a

conversão de estruturas complexas em um formato adequado para o armazenamento em arquivos. Quando surge a necessidade de ler os dados mais uma

vez, o programador deve convertê-los do formato de arquivo para uma estrutura variável do programa. Os sistemas de banco de dados orientado a objeto

são compatíveis com as linguagens de programação como C++ e Java, e o software SGBD, automaticamente, executa qualquer conversão necessária.

Conseqüentemente, um objeto complexo em C++ pode ser armazenado permanentemente em um SGBD orientado a objeto. Esse objeto é conhecido como persistente, desde que exista após o término de execução dos programas e possa, depois, ser acessado por outro programa em C+ + . O armazenamento persistente de programas e as estruturas de dados são uma importante função do sistema de banco de dados. Os sistemas

tradicionais de banco de dados geralmente possuem o chamado problema de separação por impedância, quando as estruturas de dados fornecidas pelo

SGBD são incompatíveis com as estruturas de dados da linguagem de programação. Os sistemas de banco de dados orientados a objeto oferecem estruturas de dados compatíveis com uma ou mais linguagens de programação orientadas a objeto.

1.6.4 Garantindo o Armazenamento de Estruturas para o Processamento Eficiente de Consultas

Os sistemas de banco de dados devem fornecer funcionalidades para a execução de atualizações e consultas eficientemente. Pelo fato de o banco de dados

ser armazenado, tipicamente, em disco, o SGBD deve possuir estruturas de dados especializadas para aumentar a velocidade de pesquisa no disco dos

registros desejados. Os arquivos auxiliares, chamados indexes (indexados), são utilizados com esse objetivo. Os indexes são baseados em estruturas de

dados árvores (tree) ou estruturas de dados hash, adequadamente adaptados para a pesquisa em disco. Com o intuito de processar os registros necessários do

banco de dados para uma consulta particular, aqueles registros devem ser copiados do disco para a memória. Conseqüentemente, os SGBD em geral têm um

módulo de armazenamento temporário (buffering) que mantém partes do banco de dados armazenado na memória principal. Em outros casos, o SGBD pode utilizar o sistema operacional para fazer o armazenamento temporário dos dados no disco. O módulo do SGBD para o processamento de consulta e otimização é responsável pela escolha eficiente do plano de execução da consulta (query)

baseado nas estruturas de armazenamento existentes. A opção de qual index criar e manter é parte do projeto físico do banco de dados e seu ajuste (tunning), que é de responsabilidade do DBA e sua equipe.

1.6.5 Garantindo Backup e Restauração

Um SGBD deve prover facilidades para a restauração de falhas de hardware ou de software. O subsistema de backup e recuperação dos subsistemas do

SGBD é responsável pela recuperação dessas falhas. Por exemplo, se um sistema de computador falhar no meio de uma transação complexa de atualização,

o subsistema de recuperação é responsável por garantir que o banco de dados seja recolocado no mesmo estado em que estava, antes do início da execução

da transação. Alternativamente, o subsistema pode assegurar que a transação seja resumida do ponto em que foi interrompida — sendo assim, seu efeito completo seria armazenado no banco de dados.

1.6.6 Fornecendo Múltiplas Interfaces para os Usuários

Como diversos tipos de usuários com níveis de conhecimento técnico diferentes utilizam o banco de dados, o SGBD deve fornecer interfaces diferentes para

esses usuários. Essas interfaces incluem linguagens de consulta para os usuários casuais; interfaces de linguagens de programação para programadores de aplicações; formulários e seqüências de comandos para usuários

14

14

14 Capítulo 1 Banco de Dados e os Usuários de Banco de Dados

parametrizáveis; interfaces de menus, interfaces de linguagem natural para usuários autônomos. Ambas, as interfaces com menus e aquelas com

formulários, são comumente conhecidas como interfaces gráficas para os usuários — Graphical User Interfaces (GUIs). Muitos ambientes e linguagens

especializadas existem para a especificação de GUIs. As capacidades para gerar interfaces Web GUI para um banco de dados — ou capacitando um banco de dados para a Web (Web-enabling) — também são muito comuns.

1.6.7 Representando Relacionamentos Complexos entre os Dados

Um banco de dados pode incluir uma grande variedade de dados que estão inter-relacionados de muitas maneiras. Veja o exemplo mostrado na Figura 1.2.

O registro de Brown no arquivo ALUNO está relacionado com quatro registros no arquivo HISTORICO\_ESCOLAR. De forma similar, cada registro

Disciplina está relacionado com um registro de cursos e ainda com os registros HISTORICO\_ESCOLAR, um para cada aluno que completou uma

disciplina. O SGBD deve ter a capacidade de representar a variedade de relacionamentos complexos entre os dados, bem como recuperar e atualizar os dados relacionados fácil e eficientemente.

1.6.8 Forçando as Restrições de Integridade

A maioria das aplicações de um banco de dados tem certas restrições de integridade que devem complementar os dados. O SGBD deve prover

funcionalidades para a definição e a garantia dessas restrições. O tipo mais simples de restrição de integridade envolve a especificação de um tipo de dado

para cada item de dados. Por exemplo, na Figura 1.2, podemos especificar que o valor do item de dados Turma, em cada registro ALUNO, tem de ser um

inteiro entre 1 e 5, e o valor de Nome deve ser uma string (cadeia de caracteres) menor que 50 caracteres alfabéticos. Os tipos mais complexos de restrições

podem ocorrer, com freqüência, envolvendo a definição de que o registro em um arquivo deve estar relacionado aos registros de outros arquivos. Por

exemplo, na Figura 1.2, podemos especificar que 'todo registro de disciplina deve estar relacionado com um registro de Curso'. Outro tipo de restrição

especifica a singularidade no valor do item de dado, como 'todo registro de curso deve ter um único valor para NumerodoCurso'. Essas restrições são

derivadas do significado ou da semântica dos dados e do minimundo que representam. Os projetistas do banco de dados são responsáveis por identificar as

restrições de integridade durante o projeto do banco. Algumas restrições podem ser especificadas para o SGBD e executadas automaticamente. Outras podem ser testadas pelos programas de atualização ou no momento da entrada dos dados. Um item de dados pode ser inserido incorretamente e ainda assim satisfazer as restrições de integridade definidas. Por exemplo, se um aluno recebe

nota A, mas é inserida, no banco de dados, a nota C, o SGBD não pode descobrir esse erro, automaticamente, porque C é um valor válido para os tipos de

dados de NOTA. Esse erro pode ser descoberto manualmente (quando o aluno receber a nota e reclamar), sendo corrigido, depois, por meio da atualização do banco de dados. Porém, a nota Z pode ser rejeitada automaticamente pelo SGBD, pois ela é um valor inválido para os tipos de dados de NOTA.

1.6.9 Permitindo Inferências e Ações Usando as Regras

Alguns sistemas de banco de dados oferecem capacidades para definir as regras de dedução por inferência gerando novas informações de fatos

armazenados no banco de dados. Esses sistemas são chamados sistemas de banco de dados dedutivos. Por exemplo, podem existir regras complexas no

minimundo da aplicação para determinar quando um aluno está em recuperação. Isso pode ser especificado declaradamente como uma regra, e quando for

compilada e mantida pelo SGBD, poderá determinar todos os alunos em recuperação. No SGBD tradicional, um código explícito de programa procedural

teria de ser escrito para dar suporte a essas aplicações. No entanto, se as regras do minimundo mudarem, geralmente é mais conveniente mudar as regras de

dedução declaradas que reescrever os programas procedurais. Os sistemas de banco de dados ativos oferecem funcionalidades mais potentes, pois permitem regras ativas que podem disparar automaticamente ações quando certos eventos e condições ocorrerem.

1.6.10 Implicações Adicionais do Uso da Abordagem de um Banco de Dados Esta seção discute algumas implicações adicionais da utilização da abordagem do uso de um banco de dados que beneficiam a maioria das organizações:

• Potencial para Garantir Padrões. A abordagem de um banco de dados permite ao DBA definir e forçar o uso de padrões entre os usuários de um

banco de dados em uma grande organização. Isso facilita a comunicação e cooperação entre os vários departamentos, projetos e os usuários na organização. Os padrões podem ser definidos para os nomes e