Universidade de Brasília Faculdade UnB Gama

Sistemas de Banco de Dados 2

Tecnologias de Banco de Dados (TI-BD)

Bancos de Dados orientados a Objetos

Leonardo de Araujo Medeiros, 17/0038891

Brasília, DF

2019

Introdução

Um Banco de Dados orientado a objetos (BDOO) é basicamente um sistema em que a unidade de armazenamento é o objeto, com o mesmo conceito das linguagens de POO. A diferença fundamental está no fato de que em BDOO, os dados não deixam de existir após o encerramento do programa, ou seja, os objetos continuam a existir mesmo se o sistema venha a ser encerrado.

Definição

Um Banco de Dados orientado a objetos é um banco de dados no qual as informações são armazenadas na forma de Objetos, ou seja, utiliza o paradigma da Orientação a Objetos (OO) para modelar suas estruturas de dados.

Neste modelo, seguindo o princípio da Orientação a Objetos (OO), os Objetos são modelados na forma de classes, que representam suas características e comportamentos abstraídos do mundo real. Uma classe é um modelo onde os objetos são originados, ela possui a definição dos atributos e das ações de um determinado tipo de objeto.

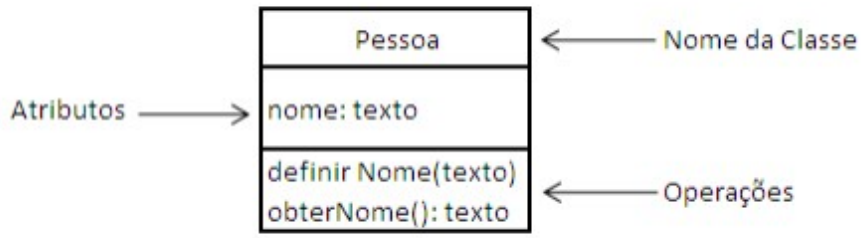

Figura 1: A figura ilustra a declaração de classes.

A interface de um Objeto é caracterizada pelo seu conjunto de mensagens, assim sendo, a interface identifica quais são as mensagens que um objeto deve responder. Como a interface é a única parte acessível (visível) do objeto, é possível realizar alterações no código que implementa as mensagens que compõem a interface (métodos) sem impactar a forma de comunicação entre os objetos.

Neste modelo, o objeto é formado como se fosse uma tripla (i, c, v), onde o **i** é o OID do objeto, o **c** é um construtor, ou seja, que tipo de valor ele vai receber ex.: *atom*, *tuple*, *set*, *list*, *bag*, *array* e **v** é o valor corrente. Então o objeto passa a suportar aquilo que foi definido para ele. Se ele vai receber um valor atômico ele só aceitará valores atômicos.

Os construtores de tipos *sets*, *bags*, *lists* e *arrays* são caracterizados como tipos de coleções e a diferença entre eles é a seguinte: *sets* e *bags*, o primeiro só aceita valores distintos enquanto o segundo aceita valores duplicados. *Lists* e *arrays*, o primeiro só aceita números arbitrários, enquanto o segundo, o tamanho deve ser preestabelecido.

A idéia do encapsulamento em um BD OO, já que não dá para ser aplicado a rigor, é pelo menos tratar o comportamento do objeto com funções pré-definidas. Por exemplo, *insert*, *delete*, *update* etc. Ou seja, a estrutura interna do objeto é escondida, e os usuários externos só conhecem a interface do tipo de objeto como os argumentos (parâmetros), de cada operação. Então a implementação é oculta para usuários externos que está incluído a definição da estrutura interna, de dados do objeto e a implementação das operações que acessam essas estruturas. Enfim o BD OO propõe o seguinte, dividir estrutura do objeto em partes visíveis e ocultas então para operações que exigem atualização da base de dados torna se oculta e para operações que exige consultas, torna-se visível (ELMASRI, 2005.).

Em 2004 os bancos de dados orientados a objeto tiveram um crescimento devido ao surgimento de banco de dados OO livres. A Object Data Management Group (ODMG) com a Object Query Language (OQL) padronizou uma linguagem de consulta para objetos. Uma característica que vale a pena ser ressaltada, é que o acesso a dados pode ser bem mais rápido, porque não é necessário junções. Já que o acesso é feito diretamente ao objeto seguindo os ponteiros.

Object Definition Language (ODL) – Linguagem de Definição de Dados

A ODL foi feita para dar suporte aos construtores semânticos do modelo de objetos ODMG e é independente de qualquer linguagem de programação em particular. O objetivo da ODL é criar especificações de objetos, isto é, classes e interfaces. A ODL não é considerada uma linguagem de programação completa. Ela permite que o usuário especifique um banco de dados independente da linguagem de programação.

Object Query Language (OQL) – Linguagem de Consulta de Objetos

É a linguagem de consulta declarativa definida pela ODMG. Prevê suporte ao tratamento de objetos complexos, invocação de métodos, herança e polimorfismo. É projetada para trabalhar acoplada com as linguagens de programação com as quais o modelo ODMG define uma ligação, como C++, SMALLTALK e JAVA. Isso faz com que qualquer consulta OQL embutida em uma dessas linguagens de programação pode retornar objetos compatíveis com os sistemas de tipos dessa linguagem.

Principais vantagens

Existem pelo menos dois fatores que levam a adoção deste modelo, a primeira é que banco de dados relacional se torna difícil trabalhar com dados complexos. A segunda é que aplicações são construídas em linguagens orientadas a objetos (java, C++, C#) e o código precisa ser traduzido para uma linguagem que o modelo de banco de dados relacional entenda, uma tarefa complicada e tediosa (ELMASRI, 2005. Adaptado).

Além disto, a habilidade de modificar a definição de um objeto sem afetar o resto do sistema é considerada uma das maiores vantagens do paradigma de orientação a objetos.

Principais desvantagens

Umas das características dos sistemas OO é ocultar informação e tipos abstratos de dados, sendo que é muito complicado aplicar esse modelo na prática. Por exemplo, nos sistemas relacionais, para uma consulta em uma determinada tabela é necessário saber todos os atributos da tabela, para formar a consulta. Em um sistema OO, que preza pelo encapsulamento nem toda tabela pode enxergar a outra, o que dificulta muito as consultas.

Outro problema seria a inconsistência desse modelo em trabalhar com outras ferramentas como OLAP, backup e padrões de recuperação.

Exemplos de BDOO

Postgres

O sistema Postgres foi construído em linguagem C e teve como referência o Ingress (banco de dados relacional).

O postgres é baseado em um modelo relacional estendido, oferecendo objetos, OID, objetos compostos, herança múltipla, versões, dados históricos e uma linguagem de consulta denominada Postquel.

A declaração de dados no Postgres tem como base o modelo relacional. O sitema oferece um tipo abstrato de dados, para que se possa definir um novo tipo de base de dados.

```
create pessoa( nome = char[25],
               Endereco = char[30],
               RG = char[15]
key(RG) )
```

Figura 2: A figura ilustra a declaração de dados do Postgres

Para a manipulação de dados, o sistema Postgres possui uma linguagem de consulta derivada do Quel, linguagem de consulta do banco de dados relacional antecessor (Ingress).

```
reatrieve (A.nome, M.nota)
where A.nome = "Eugenio Akihiro Nassu"
```

Figura 3: A figura ilustra a manipulação de dados do Postgres.

O_2

O sistema O_2 fornece um ambiente gráfico para criação de telas (O_2Look), um browser para navegar entre os objetos do banco de dados. E para facilitar a migração vinda de SGBD relacionais, há uma ferramenta extra, que realiza a conversão e migração (O_2DBAcess).

O sistema O_2 fornece integração com as linguagens de programação C, C++ e Java.

A declaração de dados ocorre através de uma linguagem (O_2C) que tem origem a partir da linguagem C. Esta linguagem possui herança múltipla e não possui distinção na declaração dos objetos persistentes e não-persistentes.

```
Class Pessoa
   Type tuple(nome: string
              RG: string
              Endereço: string)

   Method Muda_Endereco(novoEndereco: string)
end
```

Figura 4: A figura ilustra a declaração de dados em O2

O sistema O_2 garante automaticamente a integridade referencial em seus relacionamentos, ou seja, se um dos objetos for excluído do banco de dados, o sistema automaticamente torna as referências para este objeto nulas.

A manipulação de dados no sistema O_2 pode ser efetuada de duas formas, uma é pela própria linguagem de programação nativa do sitema e outra é pela já citada OQL.

```
/* Cria objeto aluno persistente */
add name Eugenio: Aluno;
/* Cria conjunto de alunos persistente */
add name AlunosDoBCC: set(Aluno);
/* Cria um aluno, a princípio, não persistente */
Francisco: Aluno;

run body
{
Eugenio.Nome = "Eugênio Akihiro Nassu";
/* modificando/iniciando o objeto Aluno */
...
AlunosDoBCC += set(Eugenio);
/* colocando o Aluno em um conjunto */
AlunosDoBCC += set(Francisco);
/* o aluno Francisco também se torna persistente
pois é referenciado por um objeto persistente */
...
/* modifica o endereço do Aluno Eugenio */
Eugenio.Muda_Endereco("Alameda Barros 380");
...
}

select tuple(Aluno: m.Al.Nome, Nota: m.nota)
from m in Matriculas
where m.Disc.nome = "MAC-110" and m.SemAno = "1/95"
```

Figura 5: A figura ilustra a manipulação de dados do O2

Referências

ELMASRI, Ramez. **Sistemas de banco de dados.** Ramez Elmasri e Shamkant B. Navathe; Revisor técnico Luis Ricardo de Figueiredo. [S.I.]. Addison Wesley, 2005.

FOWLER, Martin. **UML essencial: um breve guia para a linguagem padrão de modelagem de objetos.** Martin Fowler; tradução João Tortello. 3ª ed. [S.I.]. Bookman, 2005.

MANSUELI. Victor. **DevMedia – Bancos de Dados Orientados a Objetos - SQL Magazine 78.** [S.I.]. 2010. Disponível em: https://www.devmedia.com.br/bancos-de-dados-orientados-a-objetos-sql-magazine-78/17717. Acesso em: 07 set. 2019.

CARVALHO, Guilherme. **Sistema de Bancos de Dados Orientados a Objetos.** [S.I.]. 2011. Disponível em: http://www.fatecsp.br/dti/tcc/tcc0002.pdf. Acesso em: 08 set. 2019.