

UNIVERSIDADE DE BRASÍLIA
Faculdade do Gama

Sistemas de Banco de Dados 2

Tecnologias de Banco de Dados (TI-BD)

**Bancos de Dados Orientado a Objeto e
Objeto-Relacional**

Nome: Rossicler Rodrigues Pires Júnior
Matrícula: 16/0154197

Brasília, DF
2019


Definição das tecnologias

Banco de dados orientado a objetos:

Um banco de dados orientado a objetos é um banco de dados em que cada informação é armazenada na forma de objetos, ou seja, utiliza a estrutura de dados denominada orientação a objetos, a qual permeia as linguagens mais modernas. Começou a ser comercialmente viável em 1980. O gerenciador do banco de dados para um orientado a objeto é referenciado por vários como ODBMS ou OODBMS.

Banco de dados objeto-relacional:

O banco de dados objeto-relacional, do inglês *object-relational database* (ORD) ou *object-relational database management system* (ORDBMS), é um sistema de gerenciamento de banco de dados (SGBD) semelhante a um banco de dados relacional, porém com um modelo de banco de dados orientado a objetos, classes e heranças são suportadas diretamente nos esquemas do banco de dados e na linguagem de consulta. Além disso, ele suporta a extensão do modelo de dados com a personalização de tipos de dados e métodos.



Modelo Relacional	Modelo OO
Tabelas (entidades)	Objetos
Linhas (registros)	Tuplas
Query's(consultas,etc)	Native Query's
Sql Ansci	Métodos, construtores

CARACTERÍSTICAS DOS SGBDOO'S

*Figura 1: Modelo Relacional x Modelo OO

De acordo com a figura 1, podemos observar as diferenças nos conceitos entre o modelo relacional de banco de dados e o modelo orientado a objetos, onde no relacional, os dados são armazenados em tabelas que são compostas por linhas, e no modelo orientado a objetos é utilizado objetos e tuplas.

Objetivos:

Um dos objetivos de um SGBDO (Sistema de Gerenciamento de Bancos de Dados de Objeto) é manter uma correspondência direta entre objetos do mundo real e do banco de dados, de modo que objetos não percam sua integridade e identidade e possam facilmente ser identificados e operados. Assim, o SGBDO oferece uma identidade única e imutável a cada objeto armazenado no banco de dados chamado OID (Identificador de objeto). O valor de um OID não é visível ao usuário externo, somente ao sistema para poder gerenciar a referência dos objetos. Também possui o objetivo de fazer um tratamento mais adequado para objetos complexos, como por exemplo, textos, gráficos. O que acaba possuindo uma maior naturalidade conceitual por estarem em consonância com fortes tendências em linguagens de programação e engenharia de software. Além disso, com o advento do paradigma orientado a objetos e o tradicional modelo relacional, foi criada uma “ponte” entre essas duas formas de representar um problema real em um sistema de informação, o que acarretou na necessidade de se “mapear” estes dois modelos, surgindo então os frameworks objeto-relacionais, o que sendo uma mistura entre estes dois paradigmas.

Vantagens:

Uma vantagem que é muito importante é que para determinadas tarefas, a performance do banco de dados orientado a objetos é superior. O motivo para isso é que esses tipos de banco usam interfaces navegacionais para a realização de várias operações, ao invés de interfaces relacionais, fazendo então que a performance

seja melhor, por conta de que interfaces navegacionais geralmente é implementado de forma muito eficiente por ponteiros.

A segunda vantagem ao qual já foi citada no tópico de 'Objetivos', é que usando um banco que possui o conceito de orientação a objetos, é mais fácil fazer com o banco se relacione com a linguagem de programação, visto que grande parte das linguagens de programação atual utilizam e seguem os princípios da orientação a objeto. Logo, a maioria dos programadores conseguem absorver melhor as funcionalidades do banco, diferentemente de bancos relacionais, onde o conceito por trás se difere significativamente dos conceitos utilizados por linguagens de programação no geral, assim fazendo com que um programador sem experiência com banco de dados tenha uma curva de aprendizado mais baixa, se comparado ao banco de dados orientado a objetos.



BDOO x BDR - Principais Diferenças

<i>Banco de Dados Relacional (BDR)</i>	<i>Banco de Orientado à Objetos (BDOO)</i>
Atributos são limitados à valores simples	Atributos multivalorados são comuns
Não há estrutura pré-definida para herança	Estruturas de herança estão imbutidas no modelo
Definição de operações pode ser postergada	Operações devem ser especificadas antecipadamente, pois fazem parte das especificações das classes.

*Figura 2: Principais diferenças entre BDOO x BDR

De acordo com a figura 2, podemos ver algumas das principais

diferenças entre os dois modelos de banco de dados. Onde podemos observar algumas vantagens e desvantagens. Uma das vantagens observadas é que no modelo BDOO a utilização de atributos multivalorados é comum e de simples compreensão, não sendo necessário criar uma nova “tabela” para armazenar esse tipo de atributo. Podendo também aproveitar o conceito de herança, deixando de maneira mais simples a relação entre os objetos/tabelas.

Desvantagens:

Uma desvantagens que afeta bastante o uso dessa tecnologia é que, para consultas gerais, a performance das técnicas baseadas em ponteiros tenderão a ser mais lentas e mais difíceis de se formular do que as relacionais. Desta maneira, a abordagem navegacional parece simplificar para usos específicos conhecidos às custas do uso geral, ignorando usos futuros.

A segunda desvantagem é a perda de interoperabilidade com um grande número de ferramentas/características que são tidas como certas no mundo SQL, incluindo a indústria de padrões de conectividade, ferramentas de relatório, ferramentas de OLAP e backup, e padrões de recuperação. Onde todos esses fatores trás diversas vantagens para o banco de dados relacional.

A terceira desvantagem é que o banco de dados orientado a objetos perdem o fundamento formal matemático, ao contrário do modelo relacional, e isto às vezes conduz a fraqueza na sustentação da consulta. Entretanto esta objeção é descartada pelo fato que alguns ODBMS's suportam totalmente o SQL em adição ao acesso navegacional (Objectivity/SQL++). Mas, o uso eficaz pode requerer acordos para manter ambos os paradigmas sincronizados, que é a dificuldade que faz com que isso seja uma desvantagem comparado ao modelo relacional.

Uma outra desvantagem que pode ser notada na figura 2, é que as operações nos bancos de dados orientados a objetos devem ser definidas antecipadamente, visto que fazem parte da especificação

da classe, diferente do modelo relacional, onde essas operações podem ser definidas posteriormente a criação do modelo.

Banco de Dados Orientado a Objetos no mercado

CACHÉ: Trabalha com as seguintes linguagens: Java, .Net, C++, XML e outras. É um banco de dados comercial.

VERSANT: Trabalha com as seguintes linguagens: Java e C++. É bastante utilizado nos sistemas de telecomunicações, redes de transporte, áreas médias e financeiras. É um banco de dados comercial.

DB4Objects: Trabalha com as seguintes linguagens: Java e .Net. Sua linguagem de Consulta é o Object Query Language (OQL) e é um banco de dados distribuído em duas licenças, a GPL (licença pública geral) e uma licença comercial.

O2: Trabalha com as seguintes linguagens: C, C++ e o ambiente O2. Sua linguagem de Consulta O2Query, OQL. Seu gerenciador de Banco de Dados é o O2Engine, e é um banco de dados comercial.

GEMSTONE: Trabalha com as seguintes linguagens: Java, C++, C#, XML e outras. Sua linguagem de Consulta é o DML. É um banco de dados comercial.

JASMINE: Possui alta conectividade com Web, suporte à linguagem Java. Pode-se ainda desenvolver aplicações em Visual Basic usando Active/X em HTML usando as ferramentas de conectividade para Web disponíveis no Jasmine, em C e C++ usando APIs e em Java usando interfaces de middleware embutidas no Jasmine. É um banco de dados comercial.

MATISSE: Trabalha as seguintes linguagens: Java, C#, C++, VB, Delphi, Perl, PHP, Eiffel, SmallTalk. É um banco de dados comercial.

Objectivity/DB: Trabalha com as seguintes linguagens: C#, C++, Java, Python, Smalltalk, SQL++ (SQL com objeto - extensões orientadas) e XML (para a importação e exportação somente). É um banco de dados comercial.

Ozone: Trabalha com as seguintes linguagens: Java e XML. É um banco de dados opensource.

Caso de sucesso

O Versant Object Database é uma das aplicações de banco de dados que possui um bom caso de sucesso, visto que possui grandes clientes de diversas áreas. É difícil saber em quais empresas específicas foram aplicadas, pois eles não revelaram essas informações, porém foi citado que esse banco de dados é utilizado em plataformas globais de vendas com grande troca de estoque, configuração de rede de grandes empresas de telecomunicação, sistema de reserva para grandes agências de vôo/hotel, análise de gerência de riscos para empresas de bancos e transportes, grandes jogos online multiplayer, detecção de fraudes, redes sociais e outros.

Referência Bibliográfica:

HARRINGTON, J. L. Object-oriented Database Design Clearly Explained: 1. ed. Academic Press, 1999

Banco de dados orientado a objetos. Disponível em :
<https://pt.wikipedia.org/wiki/Banco_de_dados_orientado_a_objetos>.
Acesso em 06 set. 2019

Banco de dados objeto-relacional. Disponível em:
<https://pt.wikipedia.org/wiki/Banco_de_dados_objeto-relacional>.
Acesso em 06 set. 2019

Banco orientado a objetos PDF. Disponível em:
<<https://www.docsity.com/pt/banco-de-dados-orientado-a-objeto-1/4734488/>>. Acesso em 07 set. 2019

Versant Object Database Disponível em:
<https://en.wikipedia.org/wiki/Versant_Object_Database>. Acesso em
09 set. 2019

Banco de Dados Orientados a Objetos -SQL Magazine 78.
Disponível em: <<https://www.devmedia.com.br/bancos-de-dados-orientados-a-objetos-sql-magazine-78/17717>>. Acesso em 09 set.
2019