FUNDAÇÃO EDUCACIONAL DO MUNICÍPIO DE ASSIS - FEMA

INSTITUTO MUNICIPAL DE ENSINO SUPERIOR DE ASSIS - IMESA

ALUNO: ANGELO RODRIGO FERREIRA JUNQUEIRA

BANCO DE DADOS: UM ESTUDO COMPARATIVO ENTRE O MODELO RELACIONAL E O MODELO ORIENTADO A OBJETOS

ASSIS
2011

FUNDAÇÃO EDUCACIONAL DO MUNICÍPIO DE ASSIS - FEMA

INSTITUTO MUNICIPAL DE ENSINO SUPERIOR DE ASSIS - IMESA

BANCO DE DADOS: Um Estudo Comparativo entre o Modelo Relacional e o Modelo Orientado a Objetos

Projeto de pesquisa apresentando ao Curso de Processamento de Dados do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial a obtenção do Certificado de Conclusão.

Orientador : Osmar Aparecido Machado
Área de Concentração: Processamento de Dados

ASSIS
2011

FICHA CATALOGRÁFICA

Junqueira, Angelo
Banco de Dados: Um Estudo Comparativo/ Angelo Rodrigo Ferreira Junqueira.
Fundação Educacional Municipal de Assis- Assis, 2011
36p.

Orientador: Osmar Aparecido Machado.
Trabalho de Conclusão de Curso- Instituto Municipal de Ensino Superior de Assis- IMESA

1. Banco de dados

CDD: 001.61
Biblioteca da FEMA

BANCO DE DADOS: UM ESTUDO COMPARATIVO DO MODELO RELACIONAL E O ORIENTADO A OBJETO

ANGELO RODRIGO FERREIRA JUNQUEIRA

TRABALHO DE CONCLUSÃO DE CURSO
APRESENTADO AO INSTITUTO MUNICIPAL
DE ENSINO SUPERIOR DE ASSIS,
COMO REQUISITO DO CURSO DE GRADUAÇÃO,
ANALISADO PELA SEGUINTE COMISSÃO EXAMINADORA

ORIENTADOR:_____

AVALIADOR:_____

ASSIS
2011

DEDICÁTORIA

Dedico este trabalho a todos que me ajudaram na sua construção, vocês sabem quem são.

RESUMO

Este trabalho descreve os conceitos de banco de dados, abordando alguns temas como a orientação a objetos, modelo relacional , linguagem SQL e sistemas gerenciadores de banco de dados juntamente com tudo aquilo compõe a estrutura e composição de um banco de dados. Devido a curiosidade no armazenamento das informações e como são armazenadas no banco de dados, analisando alguns recursos e apresentando resultados como dificuldade no entendimento da orientação objeto e a simplicidade dos conceitos da modelagem relacional.

Palavras chave: 1- Banco de Dados, 2- Orientação a Objetos

ABSTRACT

This paper describes the concepts of database, addressing some issues such as object orientation, relational model, management system and SQL database along with everything eles makes up the structure and composition of a database. Because of curiosity in the storage of information and how day are the stored in the database, analyzing results and presenting some features such as dificulty in understanding the object orientation and simplicity of the concepts of relational modeling.

KEYWORDS:1- Database, 2- Object Orientation

LISTA DE ILUSTRAÇÕES

SUMÁRIO

ASSIS
2011

1 – INTRODUÇÃO

Na década de 60 os computadores conquistaram seu espaço nas empresas, os computadores, que impulsionaram uma série de novas tecnologias que mudaram as organizações. Os bancos de dados são um exemplo destas tecnologias que surgiram por conta dos computadores. Os bancos de dados são estruturas destinadas a registrar as informações produzidas pelos softwares e desde os primeiros modelos de bancos de dados até os modelos atuais, muitas evoluções ocorrerem na forma de armazenamento da informação.

Atualmente existem duas vertentes de bancos de dados muito utilizadas no mercado, a relacional e orientação a objetos. Desta forma, o objetivo da pesquisa é fazer um comparativo entre as estruturas dos bancos de dados relacionais e a estrutura dos bancos de dados orientados a objetos. O trabalho desenvolverá um exemplo de uso dos comandos utilizados na criação e manuseio de tabelas em ambas estruturas, avaliando os pontos fortes e fracos de cada uma, além de apontar o que isso representa em ganho de desempenho e em qual cenário existe o maior ganho.

De acordo Guludtzan, Pelzer(2002) um dos principais requisitos desejados em um Sistema Gerenciador de banco de dados (SGBD’s) é ter um melhorar desempenho, tanto na utilização do hardware quanto na organização dos dados e das consultas feitas aos bancos de dados (BD’s). Sabe-se que o desempenho dos comandos pode ser afetado por vários motivos e que existem possibilidades de tratamento.

O presente estudo fará uso do banco de dados Oracle, que é um dos principais bancos de dados na atualidade e servirá como base para o desenvolvimento deste estudo.

Espera-se que o trabalho traga contribuições no sentido de sanar duvidas relacionadas à certa dificuldade encontrada por gerentes de Tecnologia da Informação (TI) na escolha do produto adequado para atender suas necessidades. Este tipo de estudo comparativo é importante, pois grande parte das pesquisas comparativas existentes são divulgadas pelos próprios fabricantes dos SGBD's que podem induzir os usuários a adoção do modelo que atenda aos interesses do fabricante.

Entende-se que a pesquisa visa avaliar os recursos do Oracle, testando com acesso nativo, em uma base media, com a linguagem SQL (Structured Query Language) ou recursos próprios de cada sistema, usando como plataforma o Sistema Operacional Windows.

1.1 Objetivos

O objetivo deste estudo é uma análise dos principais fundamentos necessários para o entendimento dos banco de dados, como a criação e manipulação de tabelas, abordando assuntos como Orientação a Objetos, modelo Relacional, Linguagem OQL e Sistemas Gerenciadores de Banco de Dados.

1.2 Motivação

A importância da informação para a tomada de decisões nas organizações, uma vez que a busca pela informação se torna mais complexa e demorada. Com base nesse fato o trabalho avalia recursos sitando um Sistema Gerenciador de Banco de Dados abordando temas necessários a sua compreensão do mesmo,

sendo testados com a linguagem SQL, usando como plataforma um Sistema operacional.

1.3 Justificativa

Uma das principais justificativas para este estudo é ampliar meus conhecimentos sobre as estruturas dos bancos de dados relacionais e orientados a objetos.

Além disso, este estudo poderá ser utilizado por alunos iniciantes no uso dos bancos de dados, utilizando-o como um guia comparativo entre os modelos relacionais e orientados a objetos, citados anteriormente.

1.4. Estrutura do Trabalho.

O estudo está organizado nos seguintes capitulos:

1. Apresentar características do Banco de Dado;
2. Utilizar os critérios de avaliação definidos em outras pesquisas que serve como base para este trabalho;
3. Entender SGBD ;
4. Comparar recursos;
5. Apresentar os resultados.

2 – BANCO DE DADOS

Os bancos de dados surgiram em meados dos anos 60. Também conhecidos com base de dados, são registros em uma estrutura regular, possibilitando a reorganização dos mesmos. Os Bancos de Dados (BD) são uma coleção de dados que mantém determinadas relações entre si, essas relações são controlados por um sistema que permite guardar, modificar e recuperar dados, conhecido como Sistema Gerenciador de Banco de Dados - SGBD (DATE, 1998).

Eles são utilizados em aplicações na área da informática, cujo objetivo é registar e manter informações consideradas significativas para as organizações (ELMSRI, 2000).

A melhor maneira de entender um banco de dados é conhecer a sua breve história, desta forma este capitulo fará um breve histórico do SGBDs, apresentando sua evolução cronológica.

Os primeiros bancos de dados surgiram bem antes dos computadores, com os registros das bibliotecas, que também utilizavam técnicas de armazenamento de documentos e com índices de livros. Como se observa o uso de índices para a busca de informações em bancos de dados não se trata de algo novo, mas sim de um método já existente há muito tempo que foi ajustado para o mundo da informática (REZENDE, 2004).

No entanto, na década de 60 os computadores conquistaram seu espaço nas empresas. Nessa época foram desenvolvidos dois modelos de banco de dados: modelo de rede e o modelo hierárquico. O modelo em rede era uma forma usada para descrever a estrutura de um banco de dados. O acesso era feito através de operações de ponteiros de nível que uniam os registros, os detalhes de armazenamento eram dependentes da informação armazenada e o usuário precisava conhecer a estrutura física do banco de dados para realizar a consulta.

No modelo de rede os dados são representados por uma coleção de registros e os relacionamentos por links, como se observa na figura 1.1.

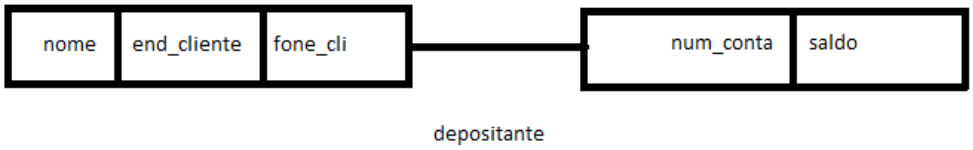


Figura 1.1 Representação de um Modelo de Dados em Rede.(HEUSER\1999)

No modelo de dados hierárquico os dados são representados por uma coleção de tabelas e os relacionamentos por links, porém sua organização é na forma de uma árvore com raiz, conforme pode ser verificado na figura 1.2.

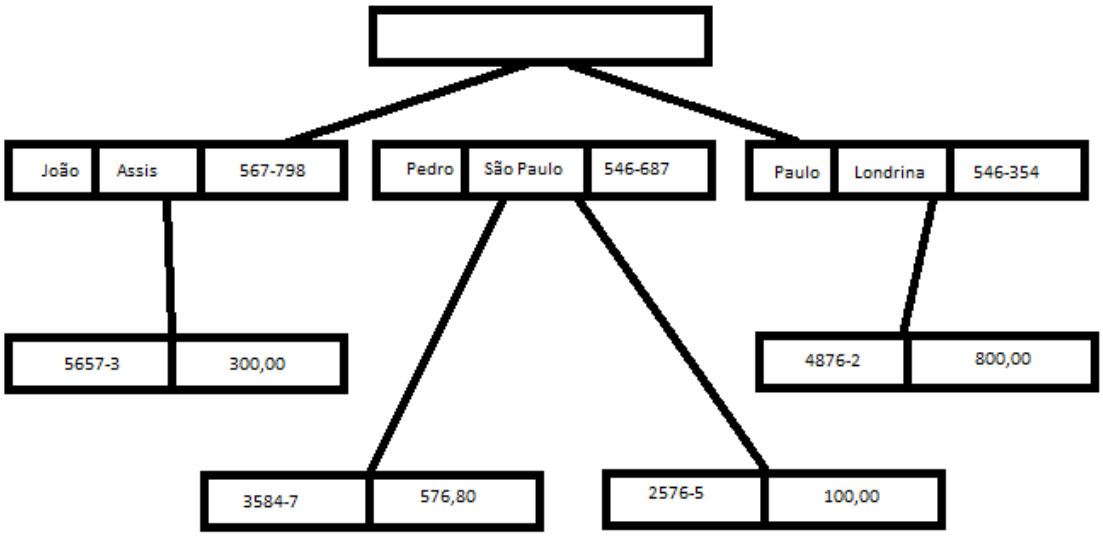


Figura 1.2 Representação de um Modelo de Dados Hierárquico.(HEUSER\1999)

No inicio dos anos 70, Edgar Frank Codd propôs o modelo relacional, que foi uma grande ideia para a época, um marco no modo de pensar em banco de dados, tornando-se um padrão para o modelo de dados (REZENDE, 2004).

Alguns protótipos de sistemas relacionais, foram:

- Ingres, desenvolvido pela UCB, que serviu como base para Sybase e o Microsoft SQL SERVER;
- System R, desenvolvido pela IBM, servindo como base para IBM DB2 e o Oracle. Nesse período que o termo SGBDR(Sistema Gerenciador de Banco de Dados Relacionais) foi definido.

Na mesma década Peter Chen propôs o modelo de entidade-relacionamento, ou simplesmente modelo ER (Figura 1.3), uma estrutura que possibilita que o projetista concentre-se na utilização dos dados, sem se preocupar a estrutura logica das tabelas (REZENDE, 2004), como se observa na figura 1.3.

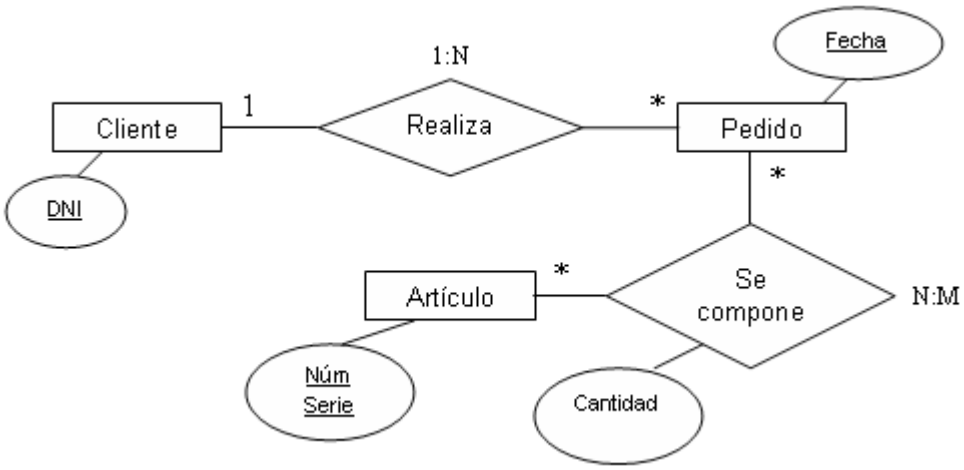


Figura 1.3 Exemplo de Diagrama ER.(HEUSER\1999)

Nos anos 80, a linguagem estruturada de consulta SQL (Structured Query Language - Linguagem Estruturada de Consulta), se tornou padrão mundial sendo o principal produto da IBM e o sistema gerenciador de banco de dados relacional (SGBDR), ganhou uma linguagem padronizada concebida para a criação e manutenção de banco de dados. A SQL, possui dois padrões estipulados, o primeiro é o SQL-92, lançado em 1992 e o outro, é o SQL - 99, lançado em 1999 (GUNDERLOYO, 2001). Mesmo não seguindo por completo a sua especificação a maioria dos SGBDR's são compatíveis com os padrões SQL-92 e SQL-99. No decorrer do trabalho serão utilizadas alguns comandos de linguagem. A linguagem SQL é dividida em duas sub-linguagens:

- DDL(Linguagem Definição de Dados), é uma linguagem que define as aplicações, arquivos e campos que irão compor o banco de dados.

- DML(Linguagem Marcação de Dados), é uma linguagem que define os comandos de manipulação e operação de dados.

Na década de 90, teve inicio as ferramentas desktop para aplicações, como PowerBuilder (Sybase), Oracle Developer e o Visual Basic (Microsoft). O padrão de modelo cliente-servidor para futuras decisões de negocio, o driver ODBC foi desenvolvido e foram lançados os primeiros protótipos dos Sistemas Gerenciadores de Banco de Dados Orientado a Objetos (SGDBOO) ou ODBMS, houve a explosão do uso da internet junto com o uso de soluções de código aberto.

2.1 - Sistemas Gerenciadores de Banco de Dados (SGBD)

SGBD são sistemas de manutenção de registros por computadores (Date, 1990), é como um sistema de coleção de objetos que mantem e manipulam os dados (Soukup, 1998) ou sistemas que gerenciam um coleção de uma ou mais tabelas de informações (SUEHRING, 2002), conforme mostra a figura 1.4.

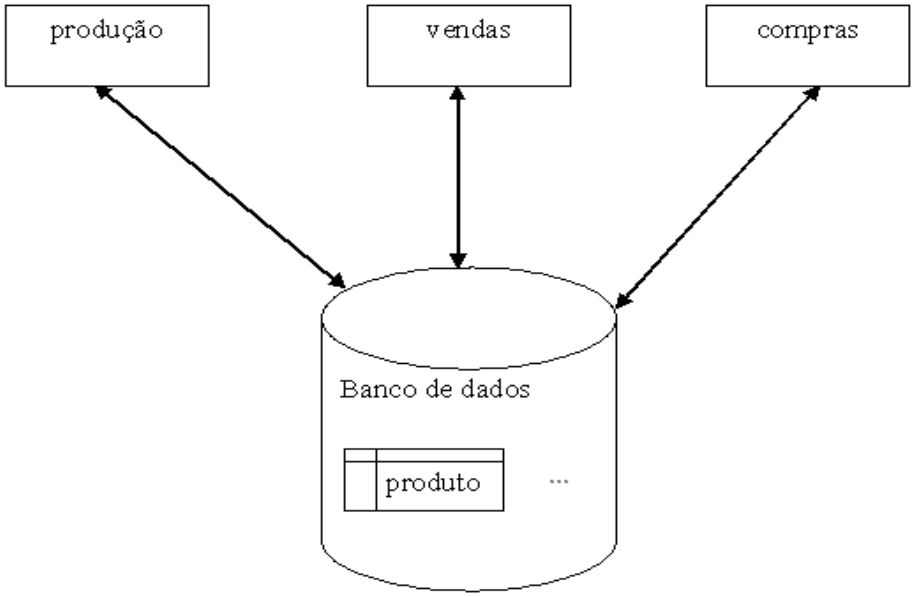


Figura1.4 Projeto de banco de dados (SUEHRING, 1999).

Os SGBDRs se tornaram um padrão de armazenamento de dados, tem como diferencial o controle de integridade dos dados, suas tabelas estão relacionadas, permitindo controle rápido e seguro (DATE, 1990).

Eles são responsáveis por controlar a concorrência, manter a integridade referencial, escolher o menor caminho para a busca de uma informação, gerenciar permissões de usuários e atualizar os índices.

As empresas oracle, Microsoft e IBM são grandes empresas que desenvolvem os SGBD pagos e outros de software livre como (Mysql,

PostgreSQL e FireBirds). Estes produtos mantém as características apresentadas anteriormente, mas diferem em outros aspectos, como porte, compatibilidade, técnicas de otimização de desempenho, recursos de configuráveis, desempenho.

As próximas seções haverá mais sobre o assunto.

Segundo Navathe (2005) os SGBD's e os Bancos de Dados tornaram-se componentes indispensáveis às necessidades do modernismo da sociedade. Na atualidade entende-se que os bancos de dados são um conjunto de dados em uma estrutura que organiza as informações. Conforme mostra a figura 1.4.

Os bancos de dados tem acesso pelo seu manipulador, que tem como objetivo principal tirar do cliente a responsabilidade de gerenciar o acesso, manipulação e organização dos dados, usando uma interface para que seus usuários possam incluir excluir, alterar ou consultar os dados. (DATE, 1998).

O termo banco de dados é aplicado aos dados enquanto o SGBD é aplicado ao software com capacidade de manipular bancos de dados.

Surgindo em meados de 70, baseado no modelo relacional escrito por Edgar. Frank Cood, o modelo relacional não fornece um suporte adequado para certas exigências, pois trabalha com tabelas (relações), devido as melhorias ocorridas no desenvolvimento do hardware, o aumento de memória nos computadores, acabou propiciando aplicações mais complexas. O modelo relacional não supre todas as necessidades como a utilização de objetos.

2.3 Modelo de Dados

O conjunto de conceitos que são usados para descrever a estrutura de um banco de dados é o modelo de dados, são classificados de acordo com a maneira

que estão dispostos aos seus usuários, ou seja, seu modelo de dados.(KORTH/ SILBERSCHATZ, 1994).

Modelo de Dados é um sub-conjunto de modelo de implementação que descreve a representação logica e física dos dados presentes no sistema. Também abrange qualquer comportamento definido no banco de dados, como procedimentos armazenados, triggers, restrições etc.

O modelo de dados é criado na faze de elaboração com base em classes persistentes, significativa do ponto de vista da arquitetura. Este modelo é refinado e expandido durante a sua fase de construção.

Os modelos mais citados são o Modelo Relacional e o Modelo Orientado a Objeto.

2.4 Modelo Relacional

O modelo relacional é baseado na teoria criada por *Edgar Cood* em 1970. Essa teoria tem por finalidade representar os dados como uma coleção de relações, sendo definida por um conjunto de operações logicas (álgebra e calculo relacional) que são a base da linguagem SQL.

Bancos de Dados relacionais possuem tabelas que por sua vez, são formadas por colunas que correspondem a um fragmento diferente de dados e por linhas que correspondem a registros individuais. A teoria relacional tem a possibilidade de definição de regras de restrição de integridade. Estas regras definem os conjuntos de estado e mudança de estado consistente do banco de dados, determinando os valores que podem e os que não podem ser armazenados (HTMLSTAFF, 2006).

No modelo relacional a principal construção para representação dos dados é a relação, uma tabela com linhas não ordenadas e colunas. Uma relação consiste de um esquema e de uma instância. O esquema especifica o nome da relação e o

nome e o domínio de cada coluna, também denominada atributo ou campo da relação. O domínio do atributo é referenciado no esquema por seu nome e serve para restringir os valores que este atributo pode assumir. O esquema de uma relação é invariável ao longo do tempo, sendo modificado apenas por comandos específicos.

A instância de uma relação é o conjunto de linhas, também denominadas tuplas ou registros, distintas entre si, que compõem a relação em um dado momento. Ela é variável, já que o número de tuplas e o conteúdo de seus atributos podem variar ao longo do tempo. A instância de uma relação deve seguir sempre o seu respectivo esquema, respeitando o número de atributos definidos, bem como os seus domínios.

Esta restrição, denominada restrição de domínio, é muito importante. O número de tuplas que uma dada instância possui denomina-se cardinalidade da relação e o número de atributos é o seu grau.

Um banco de dados relacional é um conjunto de uma ou mais relações com nomes distintos. O esquema do banco de dados relacional é a coleção dos esquemas década relação que compõe o banco de dados.

Para *Korth* e *Silberschatz* (1994), este modelo relacional representa os dados e os relacionamento entre eles por meio de um conjunto de tabelas, cada tabela tem um numero de colunas com nomes únicos.

As implantações comerciais do modelo relacional foram disponibilizadas no inicio da década de 80, com o SGBD Oracle e o sistema SQL/DS do sistema operacional MVS, da IBM. Desde então, o modelo tem sido implementado em um grande numero de sistemas comerciais. Os SGBD's mais conhecidos atualmente são o BD2 e Infomix Dynamic Server (da IBM), o Oracle, e SQL Server (Microsoft) (NAVATHE, 2005).

As arquiteturas baseadas na orientação a objetos estão sendo usadas para desenvolver as aplicações atuais. A orientação a objetos propicia o desenvolvimento

de aplicações com dados complexos e as torna mais modulares, que trás como beneficio a reutilização de código entre outras coisas (ATKINSON et al.,1989).

Apesar do surgimento dos Sistemas Gerenciador de Banco de Dados Orientado a Objeto (SGBDOO), o Sistemas Gerenciadores de banco de Dados Relacional (SGBDR) ainda é utilizado em algumas aplicações por gerenciar grandes quantidades de dados.

Com a dificuldade em abandonar os sistemas relacionais existentes e da necessidade de dados mais complexos nas aplicações, surgem os Sistemas Gerenciadores de Banco de Dados Objeto Relacional (SGBDOR), que são sistemas que estão surgindo como alternativa para acomodar a evolução das aplicações bem como para atender as suas novas necessidades.

Fornecedores de sistema gerenciador de banco de dados estão a algum tempo acrescentando características objeto relacional nos seus sistemas com base no que está elaborado até então no padrão SQL3.

SQL 3 é uma nova especificação do padrão SQL,que está sendo elaborado para acrescentar extensões pertinentes á orientação a objetos.

Alguns fornecedores já aderiram a este novo tipo sistema gerenciador de banco de dados, sendo possível citar alguns fornecedores como: Oracle, Informix e UniSQL. Os dois sistema que se enquadram neste novo modelo são o Oracle 9i (ORACLE, 2001) e o PostgreSQL (2003).

O sistema de gerenciamento de banco de dados relacional tradicional suporta um modelo de dados que consiste em uma coleção de relações, atributos de um tipo especifico. Nos sistemas comerciais em uso, os tipos possíveis incluem numero de ponto flutuante, inteiro, cadeia de caracteres, monetário e data.

Banco de dados orientado a objetos suas informações são armazenadas na forma de objetos. Possui dois fatores que o diferencia do modelo relacional, em primeiro lugar o banco de dados relacional se torna difícil de manipular e possui

dados complexos, em segundo, os dados são manipulados pela aplicação escrita usando linguagem de programação orientada a objetos, como C++, C#, Java, Python ou Delphi. No final dos anos 80, inicio dos anos 90 que se deu o crescimento de sistemas gerenciadores de banco de dados orientado a objetos, buscando uma base de dados para objetos gráfico-estruturados.

O Sistema de gerenciamento de banco de dados relacional de objetos (SGBDRO) é um sistema gerenciador semelhante ao banco de dados relacional, porém possui um modelo de banco de dados orientado a objetos: objetos, classes e herança são suportados no banco de dados e na linguagem de consulta. Suporta também extensões do modelo de dados com a personificação de tipos de dados e métodos.

Foi introduzido na década de 90 o modelo de dados Objeto Relacional que possui alta confiabilidade e larga disseminação com o aumento dos conceitos da Orientação a Objeto, segundo Stonebraker e Moore (1996) esse modelo teve origem devido a falta de desempenho e definições nos bancos de dados Orientado a Objeto, e por não ser capaz de atender as novas aplicações de dados complexos nos bancos de dados relacionais. O modelo objeto-Relacional segue padrões da linguagem SQL (Structured Query Language).

Suas principais características são permitir, especificar e utilizar tipos abstratos de dados da mesma forma que os tipos de dados pré-definidos, estender a tabela convencional para permitir tipos abstratos e valores alfanuméricos como domínio de coluna, baseia suas consultas em caminhos de referencia mais compactas do que as consultas feitas com junção, utiliza construtores : set list, multi setlist ou array para organizar coleções de objetos (DA COSTA, 2002).

O modelo Objeto-Relacional incorpora novas funcionalidades e capacidade de modelagem para tratar dados complexos (objetos) sobre estruturas físicas relacionais (tabelas).

2.5 Modelo Orientado a Objeto

Segundo Navathe (2005) o modelo Orientado a Objetos foi desenvolvido para atender as necessidades das aplicações mais complexas, oferecendo uma maior flexibilidade para lidar com requisitos sem ser limitada pelos tipos de dados e linguagem de consultas disponíveis em sistemas de bancos de dados tradicionais.

Justifica-se o aumento da orientação a objetos o uso de linguagens de programação orientadas a objetos para o desenvolvimento de aplicações de software como C++, SMALLTALK ou Java (KORTH/ SILBERSCHATZ, 1994).

O Modelo Orientado a Objetos é baseado no código e em dados encapsulados em uma unidade, chamada objeto. Sua interface entre um objeto e o resto do sistema é definida como um conjunto de mensagens. Os objetos tendo os mesmos valores e os mesmos métodos são agrupados em classes. Cada classe pode ser vista como uma definição de tipo para objetos.

A orientação a objeto é uma metodologia, baseado em objetos únicos para que se possa fazer uma modelagem de sistemas. É utilizada em engenharia de software, banco de dados e programação. Seus princípios básicos são.

2. 5.1 Abstração

Abstração é a habilidade de ignorar os aspectos de um assunto irrelevante para o próprio desejado, para que haja uma maior atenção na parte principal do assunto (OXFORD, 86). Ela é muito importante na Analise Orientada a Objeto, definindo seus atributos que são representações de informações sobre um objeto.

2.5.2 Objetos

Tudo aquilo ao qual se possa discutir, referir ou experimentar pode ser considerado um objeto (MARTIN; ODELL, 95). Dentro da Orientação a Objeto um objeto é a abstração de alguma coisa que reflete a capacidade de um sistema para manter as informações sobre ela e interagir com ela, encapsulando valores dos atributos e seus serviços (COAD; YOURDON, 93).

Um objeto pode ser real ou abstrato como; um carro, uma venda, um desenho, um relatório. Os objetos podem ser compostos de outros objetos, sendo que cada um deles tem na sua identificação uma espécie de impressão digital conhecido como OID (Object Identifier).

2.5.3 Classe

Conhecida como uma coleção de objetos que tem certa semelhança entre si e que possuem o mesmo tipo de atributo, serviços e relacionamento. Essa classe é representada graficamente na notação UML (Unifier Modeling Language), que é uma linguagem de modelagem unificada para especificar a Orientação a Objeto. (Figura 2.1).

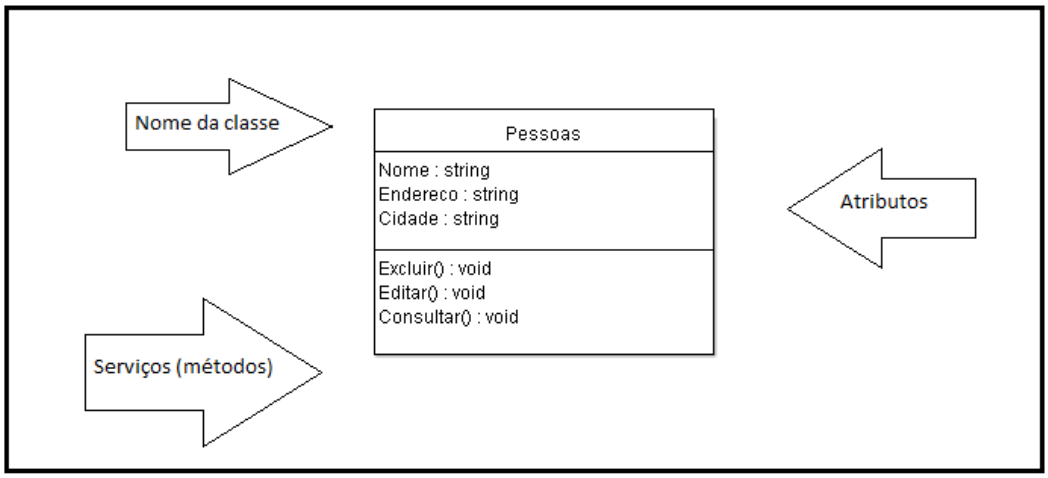


Figura 2.1 Notação UML para classe.

Um atributo é um valor guardado pelos objetos de uma classe (RUMBAGH,94, et. al). Cada objeto pertence a uma classe e essa classe é chamada de instância da classe.

2.5.4 Herança

Herança é um mecanismo para expressar a similaridade entre classes, simplificando a definição de classes iguais as outras que já foram definidas(COAD;YOURDON, 93).

Na herança, ela possui subclasses como mostra. (figura 2.2).

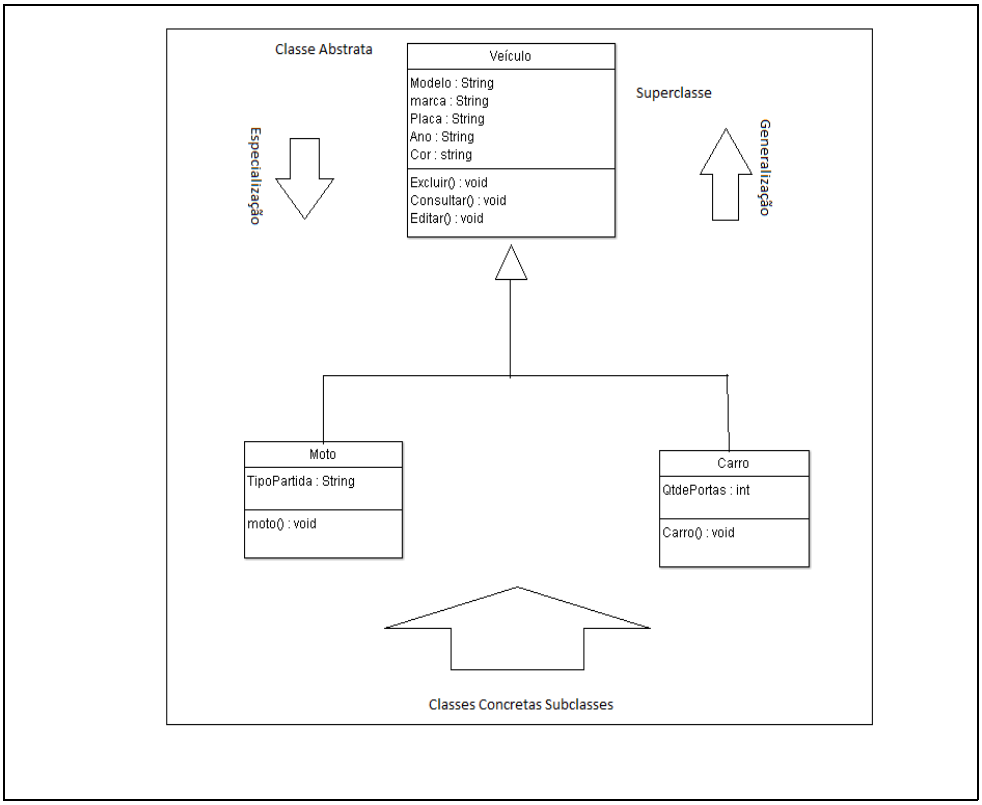


Figura 2.2 Notação UML para Herança

2.5.5 Polimorfismo

No polimorfismo é estabelecido que um objeto pode assumir variadas formas e os outros objetos podem interagir com ele sem se preocupar com sua forma num determinado momento (SCOTT, 97).

Uma solicitação no polimorfismo pode ser feita sem saber qual método devera ser invocado (MARTIN; ODELL, 95). O método pode ser invocado com a mesma assinatura e ocorrer comportamentos distintos de acordo com as informações fornecidas. (Figura 2.3).

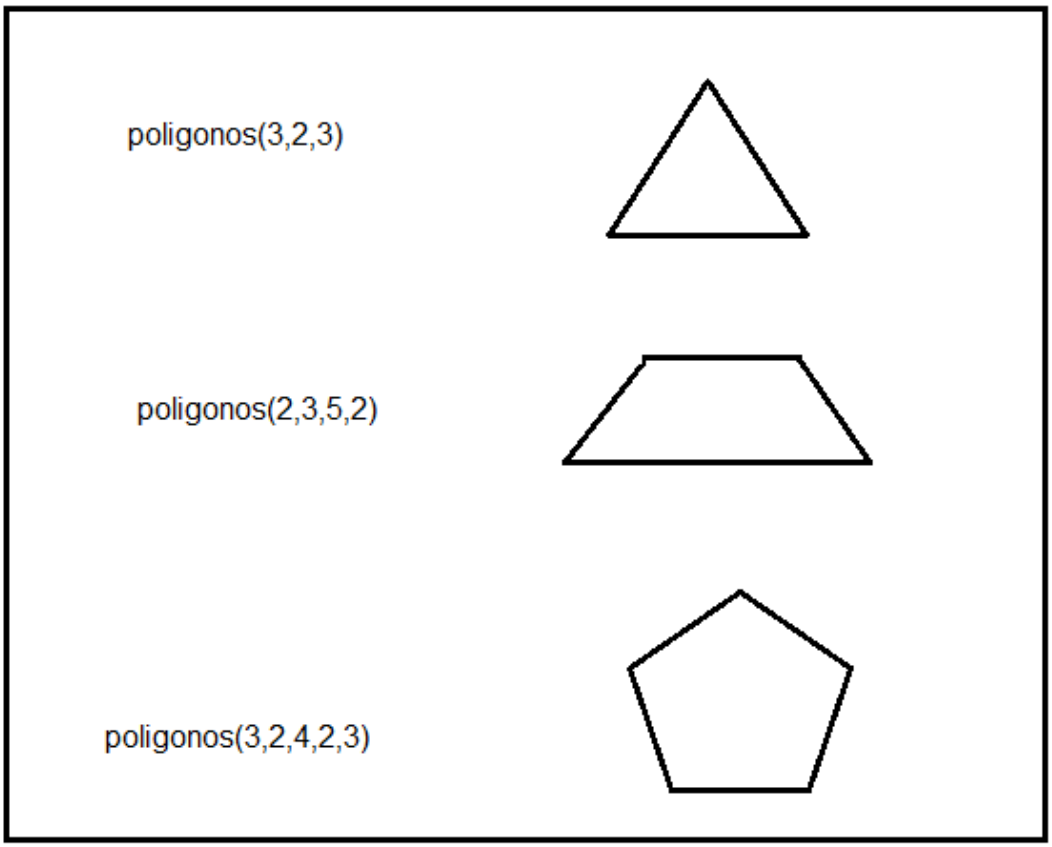


Figura 2.3 Exemplo Polimorfismo

2.5.6 Encapsulamento

Encapsulamento é o ato de empacotar ao mesmo tempo atributos e métodos, o objeto esconde seus atributos de outros objetos e permite que os atributos sejam acessados por intermédio de seus próprios métodos, chamado de ocultação de informações (MARTIN; ODELL. 95).

O Encapsulamento tem como objetivo proteger o atributo de alterações, ocultar detalhes da sua implementação aos usuários.

Um exemplo é uma calculadora que deixa disponíveis seus serviços como; somar, multiplicar e outros, retornando os resultados, mas sem mostrar os detalhes pelos quais se obteve o resultado.

2.5.7 Agregação

Há objetos que são construídos a partir de outro com isso um objeto é uma agregação de outro objeto. A agregação representa um relacionamento do tipo “faz parte de” (SCOTT, 97), possuindo dois tipos de agregação.

- Agregação por referencia: o objeto agregador pode existir sem os seus objetos constituintes e vice-versa. (Figura 2.4).

Figura 2.4 Notação UML agregação entre classes

- Agregação por valor ou composição: os objetos agregadores e agregados são dependentes um do outro. (Figura 2.5).

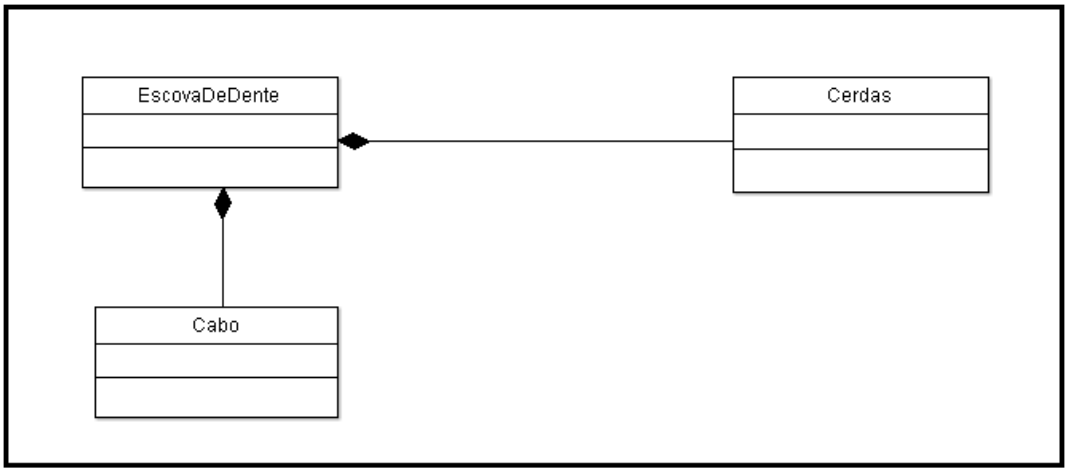


Figura 2.5. Notação UML composição entre classes

2.5.8 Associação

Para realizar determinada tarefa um objeto que necessita de outro objeto e ambos apresentarem existência independente, ou seja, um não é parte integrante do outro, então é definida a existência de uma associação entre suas respectivas classes. (TENORIO, 04). (figura 2.6).

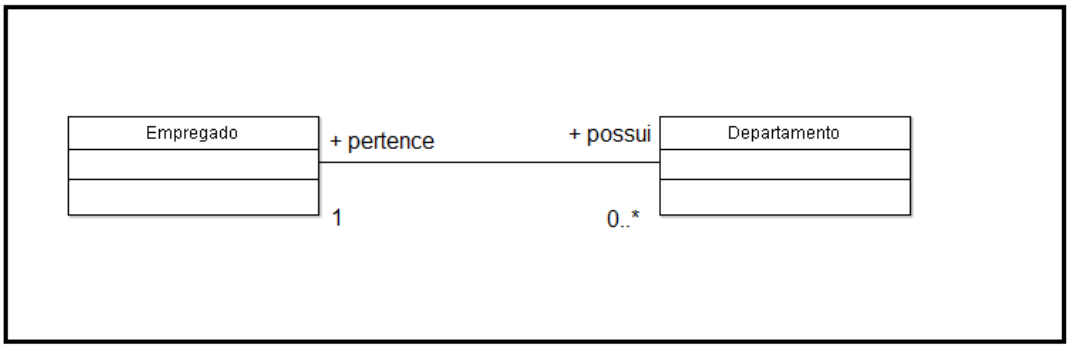


Figura 2.6. Associação notação UML

Conhecido como cardinalidade é que especifica o número de instâncias de uma classe em relação a outra em um relacionamento é mostrado através do numero mínimo e máximo.(Figura 2.7).

Notação	Significado
0..1	Zero ou uma instância
1	Somente uma instância
0..*	Zero ou mais instâncias
*	Default, número mínimo e máximo de instâncias
1..*	Uma ou mais instâncias

Figura 2.7 Cardinalidade

2.5.9 Mensagens

Mensagens são requisitos enviados de um objeto “emissor” para outro “receptor”, onde o “receptor” executa uma mensagem através de seus métodos, retornando ou não alguma resposta.

É através da mensagem que os objetos se comunicam, invocando as operações desejadas.

3 - LINGUAGEM OQL

Devido à estrutura complexa dos objetos no modelo orientado a objetos e a dificuldade no processamento de consultas sendo diferente do modelo relacional, sendo que suas relações são formadas por atributos de tipo simples, o modelo de objetos permite que um “objeto complexo” tenha atributos de referência para outros objetos. O atributo de coleção existente no modelo orientado a objetos faz com que dificulte as tarefas, com isso as linguagens de manipulação OQL (Object Query Language) oferece mecanismos capazes de especificar consultas que possuem as chamadas expressões de caminho que são um recurso simples e elegante para navegar pela estrutura dos objetos, pois fornecem um mecanismo uniforme para a formulação de consultas que envolvem diferentes características do modelo orientado a objetos, incluindo relacionamentos binários, derivados e herança (Bertino 1990).

OQL (Object Query Language) combina os aspectos declarativos da linguagem SQL com o paradigma da orientação a objetos, mas não oferece abstrações como visões SQL.

Linguagem de consulta OQL (Object Query Language), que é uma linguagem declarativa para consultar o conteúdo da base dados em um Sistema Gerenciador de Banco de dados Orientado Objeto (SGBDOO), sendo baseada no SQL. A OQL não é uma linguagem computacionalmente completa e pode ser embutida em linguagem de programação orientada a objetos, como Java e C++.

Assim como a SQL a OQL é uma linguagem declarativa, ou seja, não é necessário especificar como os dados serão retornados, mas somente que dados devem ser retornados.

A OQL é uma linguagem que não possui operadores para inserção, atualização e deleção de objetos, isto se deve ao fato de o paradigma da orientação

objeto não permitir tais manipulações de forma direta, ou seja, objetos podem ser manipulados por meio de métodos por eles definidos.

O resultado de uma consulta OQL é um conjunto de objetos que serão tratados pela linguagem orientada a objeto utilizada.

4 – COMPARATIVO DOS MODELOS RELACIONAL E O ORIENTADO A OBJETO

4.1 Modelos Relacional

Após o levantamento de dados, percebe se todo o processo de crescimento e evolução dos bancos de dados, quanto as suas linguagens e métodos sendo necessário no decorrer da evolução. Sendo o banco de dados Oracle um SGBD (Sistema Gerenciador de Banco de Dados) e suportando todo tipo de banco.

Uma ocorrência da relação de Pessoa no banco de dados contém os seguintes dados:

Pessoa			
Nome	Endereço	Cidade	Estado
Pedro Paulo	Av. Rui Barbosa	Assis	SP
João da Silva	R. Luz	Belo Horizonte	BH
Daniel Corvo	R. Platina	Rio de Janeiro	RJ
João Paulo	R. Piratininga	Recife	PE
Iracema	Av. Interior	Londrina	PR

Figura 2.8. Tabela Pessoa

Os quatro atributos para cada tupla (é uma linha individual da tabela) são Nome, Endereço, Cidade e Estado. Uma ocorrência da relação animal pode ser:

Animal		
Nome Animal	Tipo Animal	Raça
Bidú	Cachorro	Ihasa apso
Rambo	Gato	Siamês
Lulú	Cobra	Cascavél

Scooby	Cachorro	Fox
Loló	Gato	Siamês

Figura 2.9. Tabela Animal

A primeira tupla estabelece o relacionamento entre Pedro Paulo e Bidú, isto é, indica que Pedro Paulo possui Bidú.

Possui	
Nome	Nome Animal
Pedro Paulo	Bidú
João da Silva	Rambo
Daniel Corvo	Lulú
João Paulo	Scooby
Iracema	Loló

Figura 2.10. Tabela Possui

A figura mostra um diagrama E-R para o banco de dados. Este diagrama diz que pessoas e cachorros são entidades. As pessoas têm os atributos Nome, Endereço, Cidade e Estado. Os animais têm atributos Nome Animal, Tipo Animal e Raça. O diagrama também informa que as pessoas possuem animais. Imaginando que as entidades são conjuntos, os conjuntos Pessoas e os conjuntos Animais, “possui” é uma relação binária em Pessoas X Animais, essa relação é obtida por pares ordenados (pessoa, animal). O “1” e “N” nas linhas de conexão indica que essa relação é binária e é do tipo um-para-vários,isto é, uma pessoa pode possuir diversos animais.

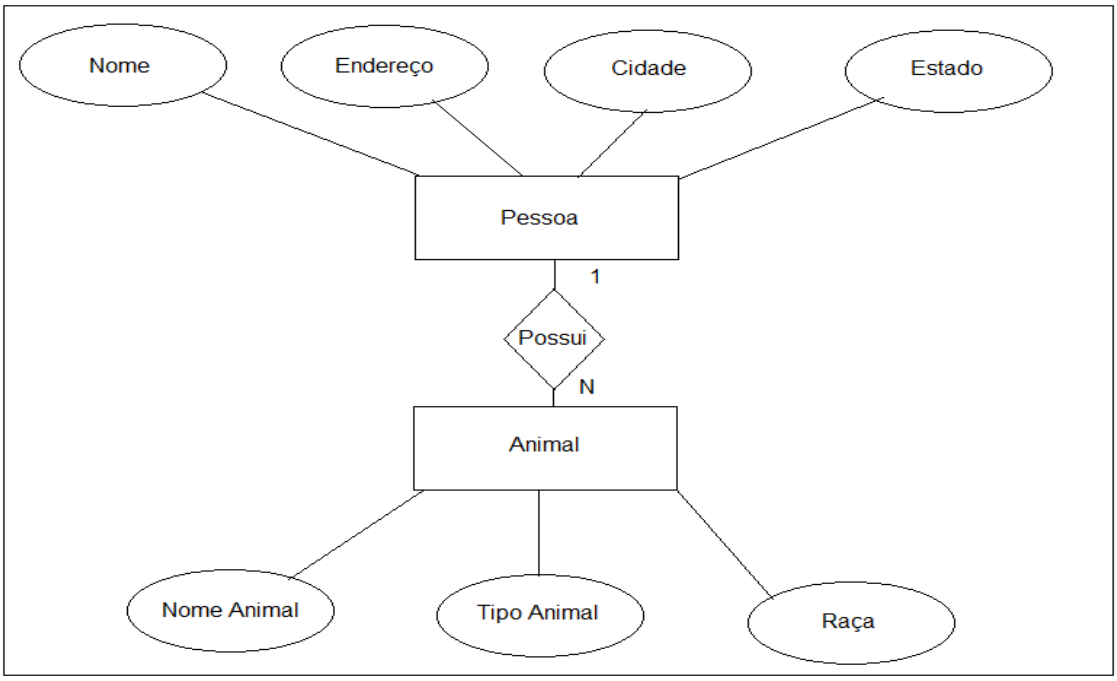


Figura 2.11. Diagrama E-R

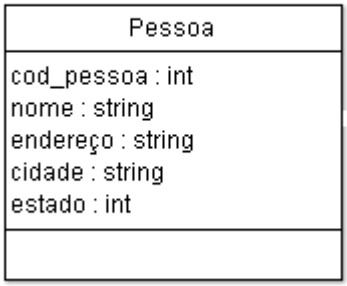
4.2 Orientado a Objetos

De modo geral cada classe persistente dá origem a uma tabela, cada atributo da classe corresponde à uma coluna, e os valores dos atributos para cada objeto na classe corresponde a uma linha.

Neste mapeamento, deve-se ter cuidado de criar a coluna de identificação, a chave primaria de cada linha ou registro armazenado, recomenda-se armazenar no banco de dados o identificador do objeto como chave primaria, ou qualquer outro atributo do objeto que o identifique de forma única.

A seguir é mostrado um exemplo de conversão de classe persistente em uma tabela. No exemplo a classe “Pessoa “ , acrescenta-se a coluna cod_pessoa, pois a classe não possui um atributo ou um conjunto de atributos que possa ser

considerado “chave”. Dessa forma é compatível com as linguagens baseadas em objetos onde os objetos têm identidades independentes das suas propriedades.



Pessoa				
Cod Pessoa	Nome	Endereço	Cidade	Estado
001	Pedro Paulo	Av. Rui Barbosa	Assis	SP
002	João da Silva	R. Luz	Belo Horizonte	BH
003	Daniel Corvo	R. Platina	Rio de Janeiro	RJ
004	João Paulo	R. Piratininga	Recife	PE
005	Iracema	Av. Interior	Londrina	PR

Figura 2.12. Mapeamento de Classes em tabela.

Existem duas maneiras de se mapear associações “um-para-muitos” entre objetos persistentes:

1º caso: Neste caso transpõe-se a chave primaria da tabela “Pessoa”, para a tabela correspondente a classe do lado.

Pessoa				
Cod_Pessoa	Nome	Endereço	Cidade	Estado
001	Pedro Paulo	Av. Rui Barbosa	Assis	SP
002	João da Silva	R. Luz	Belo Horizonte	BH
003	Daniel Corvo	R. Platina	Rio de Janeiro	RJ
004	João Paulo	R. Piratininga	Recife	PE
005	Iracema	Av. Interior	Londrina	PR

Animal				
Cod_Animal	Nome_Animal	Tipo _Animal	Raça	Cod_Pessoa
01	Bidú	Cachorro	Ilhasa apso	001
02	Rambo	Gato	Siamês	002
03	Lulú	Cobra	Cascavel	003
04	Scooby	Cachorro	Fox	004
05	Loló	Gato	Siamês	005

Figura 2.13. Mapeamento de associações “um para muitos” do caso 1.

2º caso: É criada uma tabela correspondente á associação, transpondo para elas as chaves das duas classes.

Tabela Pessoa

Pessoa				
Cod Pessoa	Nome	Endereço	Cidade	Estado
001	Pedro Paulo	Av. Rui Barbosa	Assis	SP
002	João da Silva	R. Luz	Belo Horizonte	BH
003	Daniel Corvo	R. Platina	Rio de Janeiro	RJ
004	João Paulo	R. Piratininga	Recife	PE
005	Iracema	Av. Interior	Londrina	PR

Tabela Animal:

Animal			
Cod_Animal	Nome_Animal	Tipo_Animal	Raça
01	Bidú	Cachorro	Ilhasa apso
02	Rambo	Gato	Siamês
03	Lulú	Cobra	Cascavel
04	Scooby	Cachorro	Fox
05	Loló	Gato	Siamês

Tabela Pessoa-Animal:

Pessoa_Animal	
Cod_Pessoa	Cod_Animal
001	01
002	02
003	03
004	04
005	05

Figura 2.14. Mapeamento de associação “muitos para um” do caso 2.

5 - DESENVOLVIMENTO DO TRABALHO

Nos anos 80 e 90 a comunidade acadêmica de banco de dados trabalhou com duas vertentes de sistemas gerenciador de banco de dados, a abordagem relacional e a orientada a objetos.

Atualmente estas duas vertentes tem em comum a incorporação dos seus conceitos nos principais bancos de dados do mercado, ou seja, os conceitos da orientação a objetos e a relacional. Em outras palavras, significa que os SGBD do mercado tem incorporado os modelos relacionais, mas que podem ser estendidos para que possam receber recursos da orientação objeto fazendo o que pode promover melhorias no banco de dados e ampliar as possibilidades de uso.

Estas mudanças ou capacidade dos bancos de dados em atender aos dois modelos ou paradigmas, visam atender às novas exigências impostas pela evolução tecnológica, tendo como principais indutores, as linguagens orientadas a objetos como Java, C#, C++.

Todas essas áreas de aplicações compartilham necessidades comuns, de estruturas de representações mais sofisticadas que as tabelas do modelo relacional. Operações especificas sobre estruturas de representação complexa também são necessárias para uma gerencia eficiente desses dados, ou seja, a solução para novos desafios de banco de dados aponta para a tecnologia de orientação a objetos.

- Parte Relacional:

Nos computadores, são recriados copias bidimensionais do mundo tendo inicio com arquivos, registros, relacionamentos e agora objetos. Esses objetos definidos pelo usuário, podem, ser usados como qualquer outro tipo de dados (*Korth* e *Silberschatz* (1994). Com isso, a intenção é caracterizar “objeto” de tal maneira

para representa-lo no mundo bidimensional. Um objeto é único e possui atributos, que no computador realiza ações intrínsecas a ele.

Quando é dito que o objeto possui atributos, se refere ao fato de armazenar informações que o caracterizam, portanto o objeto é composto de dados, quanto a realizar ações, ele realiza procedimentos, portanto, uma parte deste objeto é composta de rotinas, (procedimentos, funções, etc.), que realizam ações. Um objeto é algo que junta dados(atributos) e códigos (métodos) em um único elemento.

O objetivo deste trabalho é mostrar uma aplicação simples, como ficaria a estrutura de uma banco relacional e a estrutura orientada a objetos dispostos no banco Oracle que lida tanto com a parte relacional quanto a parte orientada a objeto sendo suficiente apresentar comandos SQL para a criação de tabelas.

- Descrição da Aplicação :

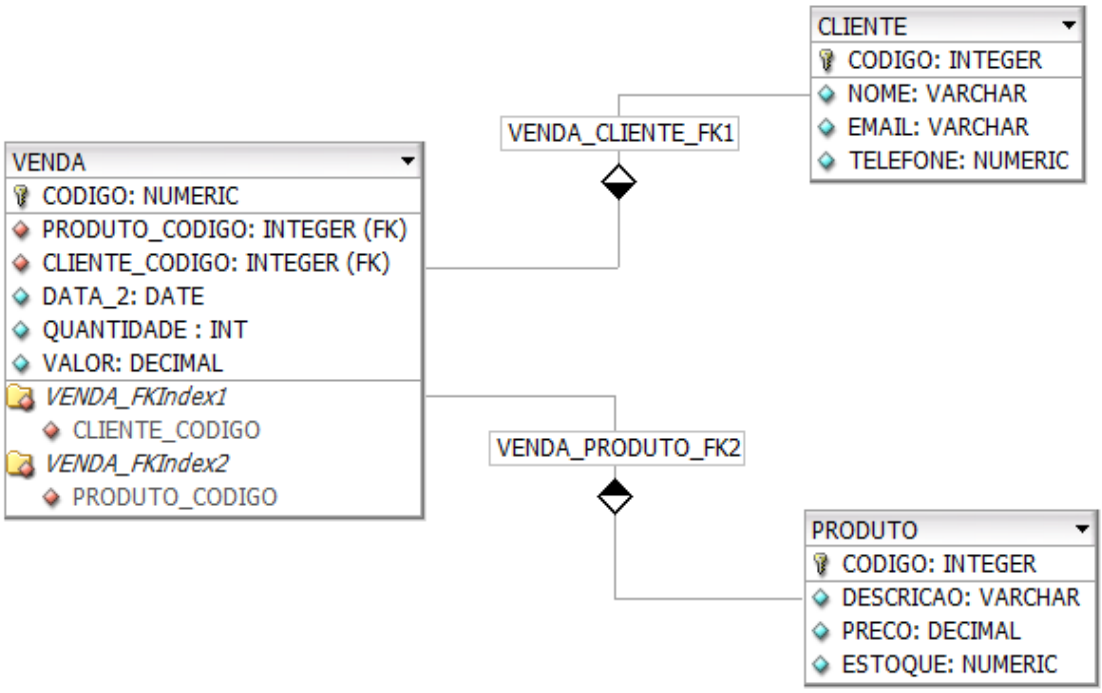
Foi implementado um Banco de Dados “Vendas” mapeado no Modelo Relacional, tendo como objetivo explicar os conceitos do Modelo Relacional através de uma aplicação real. Dentre os requisitos do Sistema Vendas (aplicação real) está, manipular os dados das tabelas do Banco de dados Vendas para obter controle sobre os produtos e vendas efetuadas para os clientes. O Banco de dados Vendas possui três tabelas: cliente, venda e produto. Sendo que a tabela cliente tem uma relação com a tabela venda e a tabela produto também possui uma relação com a tabela venda.

Estas tabelas possuem valores inseridos, compatíveis com as relações estabelecidas.

Cada tabela possui um identificador, na tabela clientes o atributo cod_cliente é a chave primária da tabela cliente, indicando que o registro é único, portanto, não é possível haver outro com o mesmo código.

As tabelas cliente, venda e produto possuem relações entre elas denominadas relacionamentos.

O esquema de Banco de dados Venda, possui um mapeamento de cardinalidade dos relacionamentos existentes entre tabelas, que informam como as tabelas se comportam e como serão associadas via relacionamento. Neste caso, a cardinalidade 1:N (um para muitos), quer dizer que um cliente pode solicitar várias Vendas e uma Vendas só pode ser requisitada apenas por um Cliente.



- Implementação Relacional

Os banco de dados relacionais são indiscutivelmente o tipo de banco mais utilizado, não sendo necessário entender a teoria relacional para utiliza-lo, aqui estão alguns conceitos:

Os bancos relacionais são compostos de relações, ou seja, tabelas. É exatamente o que o nome diz, uma tabela de dados que é composta por colunas, cada uma corresponde a um fragmento diferente de dados e linhas que corresponde

a registros individuais. Cada coluna tem nome único e dados diferentes, por ser de um formato tabular todas elas tem o mesmo atributo, as linhas também são conhecidas como tuplas. Cada linha consiste em um conjunto de valores individuais que correspondem a colunas, cada valor deve ter o tipo de dados especificados pela sua coluna. A respeito das chaves é adicionado um inteiro único para servir como chave primária, sendo o mesmo principio do numero da conta bancaria.

A seguir é apresentada a implementação da criação das relações mapeadas:

```
CREATE TABLE Cliente (
 CODIGO NUMBER(2) NOT NULL,
 cli_nome VARCHAR2(20),
 cli_mail VARCHAR2(20),
 cli_telefone VARCHAR2(10),
 CONSTRAINT CLIENTE_PRIMARY_KEY PRIMARY KEY (CODIGO));
```

```
INSERT INTO Cliente VALUES (1,'João','joao@hotmail',3333333);
INSERT INTO Cliente VALUES (2,'Carlos','carlos@yahoo',33223362);
INSERT INTO Cliente VALUES (3,'Juca','@juca@yahoo',33225362);
INSERT INTO Cliente VALUES (4,'Pedro','pedro@yahoo',33223662);
```

```
CREATE TABLE Produto (
CODIGO NUMBER(2) NOT NULL,
pro_descricao VARCHAR2(20),
pro_preco NUMBER(10,2),
pro_estoque NUMBER(6),
CONSTRAINT PK_Produto PRIMARY KEY (CODIGO));

INSERT INTO Produto VALUES (1,'computador',100,10);
```

```
INSERT INTO Produto VALUES (2,'secadora',500,3);
INSERT INTO Produto VALUES (3,'cadeira',700,10);
INSERT INTO Produto VALUES (4,'mesa',50,10);
/
CREATE TABLE VENDA (
 CODIGO NUMBER(4) NOT NULL,
  DATA DATE,
 QUANTIDADE NUMBER(4),
  VALOR NUMBER(10,2),
 CLIENTE NUMBER(2) NOT NULL,
 PRODUTO NUMBER(2) NOT NULL,
 CONSTRAINT VENDA_PRODUTO_FK FOREIGN KEY (PRODUTO) REFERENCES
PRODUTO (CODIGO),
 CONSTRAINT VENDA_CLIENTE_FK FOREIGN KEY (CLIENTE) REFERENCES CLIENTE
(CODIGO),
 CONSTRAINT VENDA_CODIGO_PK PRIMARY KEY (CODIGO));


INSERT INTO VENDA VALUES (1,'17-11-1971',200,2,1,1);
INSERT INTO VENDA VALUES (1,'17-11-1971',200,2,3,1);
```

Na implementação acima, foram criadas as tabelas e as restrições de integridade das mesmas sendo possível consulta-las.

- Orientação a Objeto

Como já foi visto que a maior parte do sistemas gerenciadores de banco de dados (SGBDs) utilizados, é baseado no modelo relacional. Outros modelos tem surgido devido a demanda de novas aplicações. Muitas aplicações requerem técnicas de acesso que melhorem o desempenho e as estruturas de dados muito

mais complexas que as tabelas relacionais,como tipos adicionais, como imagem e vídeo, fundamentos na tecnologia de orientação a objetos(Navathe (2005)).

A principal estrutura do modelo objeto-relacional são as tabelas que serão mostradas neste trabalho, estas tabelas possuem muito mais recurso do que as tabelas puramente relacionais. O objetivo é preservar os fundamentos relacionais, acrescentando novas funcionalidades.

Para utilizar os recursos de orientação a objetos a Oracle, a partir da sua versão 8 implementou diversos conceitos que definem um modelo objeto relacional tais como: tipo objeto, tabela de objetos, etc. Serão examinados os principais conceitos a seguir.

- Modelo Objeto Relacional: Oracle

O Oracle suporta o conceito de objetos complexos, sendo possível criar tipos de dados adicionais e depois fazer a referencia a esses tipos de dados dentro de outros objetos. Essa capacidade de estender o banco de dados com tipos adicionais é um recurso muito importante, pois ajuda a simplificar, por exemplo, a complexidade do tratamento a tipos de dados complexos.

Os tipos criados são gravados no esquema armazenado no banco de dados. Então, outras declarações que acessam o banco de dados podem fazer uso das definições desses tipos. Tipicamente as definições de tipo em linguagens de programação, incluindo linguagens de programação persistente, não são armazenadas em um banco de dados e só podem ser vistas por programas que incluem um arquivo texto contendo as definições.

Segundo a Oracle (2002), o modelo tipo objeto é similar ao mecanismo de classes encontrados em linguagens C++ e JAVA, essa tecnologia é uma camada de abstração embutida na tecnologia Relacional ,podendo utilizar os conceitos da

Orientação a Objeto juntamente com dados relacionais existentes, ou criar um puramente objeto, essa tecnologia será mostrada nesse capitulo.

Usando como base de dados do banco de dados Vendas, será usado os conceitos da orientação a objeto nas próximas paginas.

- Object Type

Dentro da orientação a objeto uma reunião de objetos com características e ações semelhantes são chamadas de Classe, no banco de dados Oracle essa reunião é conhecida como Object Type.

```
CREATE TYPE ENTE AS OBJECT
(
 CODIGO NUMBER(3),
 NOME VARCHAR2(40),
 EMAIL VARCHAR2(20),
 TELEFONE VARCHAR2(20)
);
```

Como uma Classe ou Object Type corresponde a uma estrutura e não a um objeto, não é possível armazenar dados sobre ela, para isso é necessário um repositório, então para resolver isso é criado o Object Table.

- Object Table

O object table foi criado para armazenar as instâncias de uma classe, ou seja, guardar os objetos criados. Declarado desta forma.

```
CREATE TABLE TB_CLIENTE OF ENTE
(
```

```
  CODIGO PRIMARY KEY
);
```

Para o atributo multi valorado (cliente type) foi criado uma nova tabela, a tabela cliente do tipo cliente type é onde sera armazenado os valores.

Tabelas podem ser armazenadas dentro de tabelas, através de tabelas aninhadas, dentro da orientação a objeto um objeto complexo pode ser representado por uma única tupla, a inclusão dos dados é dada pelo comando a seguir.

```
INSERT INTO TB_CLIENTE
VALUES (ENTE(1,'MARCIO','MARUOLBR@YAHOO','33227217'));
INSERT INTO TB_CLIENTE
VALUES (ENTE(2,'ANGELO','ANGELO_CABELO@YAHOO','33226951'));
```

Agora são criados as tabelas para comporem o sistema em apresentação:

```
CREATE TYPE PRODUTO_TIPO AS OBJECT
(
 CODIGO NUMBER(3),
 DESCRICAO VARCHAR2(40),
 PRECO NUMBER(10,2),
 ESTOQUE NUMBER(6)
);

CREATE TABLE PRODUTO_TABELA OF PRODUTO_TIPO
(
 CODIGO PRIMARY KEY
);

INSERT INTO PRODUTO_TABELA VALUES(PRODUTO_TIPO(2,'BATERIA',2,3));
INSERT INTO PRODUTO_TABELA VALUES(PRODUTO_TIPO(7,'CORREIA',7,8));
```

```
CREATE TYPE VENDA_TIPO AS OBJECT
(
  CODIGO NUMBER(3),
  DATA DATE,
  QUANTIDADE NUMBER(4),
  VALOR NUMBER(10,2) ,
  REF_CLIENTE REF ENTE,
  REF_PRODUTO REF PRODUTO_TIPO
);

CREATE TABLE VENDA_TABELA OF VENDA_TIPO
(
  CODIGO PRIMARY KEY,
  REF_CLIENTE SCOPE IS TB_CLIENTE,
  REF_PRODUTO SCOPE IS PRODUTO_TABELA
);
```

O tipo venda foi criado e as REF's fazem referência, através do tipo REF, a uma instancia de objeto, tabelas criadas sem a REF implica em redundância. Se um cliente quiser fazer mais de uma compra implicaria na repetição dos seus dados mais de uma vez, as tabelas de objetos criadas através do tipo referencia permite que um atributo seja referencia a objeto do tipo especificado.

Aqui foi apresentado um sistema Venda, implementado no modelo relacional e no modelo objeto relacional.

6 - CONCLUSÃO

As aplicações comerciais evoluem, começam a usar estruturas mais sofisticadas de representações exigindo dos Sistemas Gerenciadores de Banco de Dados (SGBDs) maior poder semântico de representação e manipulação de dados, com o surgimento da orientação a objetos nas linguagens de programações, as aplicações conseguiram em parte resolver seus problemas encontrando no mecanismo de herança, encapsulamento entre outros, a resposta aos seus problemas de tipos de dados complexos e também a reutilização de código.

Com o uso da tecnologia objeto relacional é possível usar um tipo dentro de outro tipo. Sendo demonstrado um exemplo,o projeto Vendas, na forma relacional e objeto relacional.

São características do do modelo objeto-relacional permitir especificar e utilizar tipos abstratos de dados, na tabela convencional é estendida para permitir a referencia de objetos, TADs (tipos abstratos de dados) e valores alfanuméricos como domínio de colunas, utiliza a referencia para representar conexões inter-objetos tornando as consultas baseadas em caminhos de referencia mais compactas do que as consultas feitas com junção. Seus benefícios são possuir uma nova funcionalidade, aumentar indefinidamente o conjunto de tipos e funções fornecidos pelo Sistema gerenciador de banco de dados, desenvolvimento de aplicações simplificado e reuso de código. Consistência, permite a definição de padrões, código reusável por todas as apicoações.

Bem diferente do modelo Relacional, na orientação a objetos, o relacionamento entre dois objetos não é simétrico e possui direção.
Conclui-se que com base na teoria relacional a compreensão da teoria orientada a objetos fica complexa, pois possui outras características, no armazenamento a tabela relacional leva vantagem em relação a orientação a objeto que cria estruturas.

7 - REFERÊNCIAS BIBLIOGRÁFICAS

BAHIS VIEIRA, Mateus. Implementação da orientação a Objetos no banco de Dados Oracle 9i. Fundação Educacional do Município de Assis – FEMA – Assis, 2004.

COAD, Peter e YOURDON, Edward. Projeto Baseado em Objetos. 1ed. Rio de Janeiro: Campus, 1993.

[Elm 00] Elmasri e Navathe "Fundamentals of Database Systems", Benjamin-Cummings, 3a. Edição, 2000.

ELMASRI, Ramez. NAVATHE, Shamkant. Sistema de Banco de Dados, Person, SP, 2005.

GULUTZAN, Peter; PELZER, Trudy. SQL Performance Tuning. New York: Addison-Wesley , 2002.

GUNDERLOY, Mike, JORDEN, Joseph L. Dominando o SQL Server 2000. São Paulo: Markon Books, 2001.

GULUTZAN, Peter; PELZER, Trudy. Ajustes de Desempenho em "Consultas Simples" na SQL . SQL Magazine, Rio de Janeiro: ed. 11, ano 1, p. 24-31.

GULUTZAN, Peter; PELZER, Trudy. SQL Performance Tuning. New York: Addison-Wesley , 2002.

HEUSER,Carlos A., Projeto de Banco de Dados, Editora Sagra & Luzzato, Porto Alegre, 1999.

HTMLSTAFF, Teoria Relacional. Disponível em: <http://www.htmlstaff.org/ver.php?id=393 > Acesso em: 17 maio. 2011.

LONEY, Kevin. THERIAULT, Marlene. Oracle 9i: O Manual do DBA. Rio de Janeiro: Campus, 2002.

MySQL Manual. Disponível em: <http://dev.mysql.com/doc/mysql/pt/index.html>.

WAZLAWICK, Raul Sidnei. Análise eprojeto de sistemas de informação orientados a objetos: Elsevier.

KORTH, Henry. SILBERSCHATZ, Abraham. Sistema de Banco de Dados, Marron, SP, 1994.

REZENDE, Ricardo. Conceitos Fundamentais de Banco de Dados – Parte 2.

Disponível em:

<http://www.sqlmagazine.com.br/Colunistas/RicardoRezende/03_ConceitosBD_P2.asp>.2004.

RUMBAUGH, James et.al Modelagem e projetos baseados em Objetos. Rio de Janeiro: Campus 1994.

SCOOT, W. Amber. Analise e Projetos Orientados a Objetos.
Infobook, 1997.

STONEBRAKER, M. e Moore, D. Object-relationa DBMSs: The Next Great Wave, Morgan Kaufmann 1996.

TENÓRIO, B. Marcelo. Fundamentos de Programação Orientada a Objetos.

WAZLAWICK, Raul Sidnei. Análise e projeto de sistemas de informação orientados a objetos: Elsevier.