VANESSA GODOY KINOSHITA

BANCO DE DADOS VIA WEB: UMA ANÁLISE COMPARATIVA

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências da disciplina Projeto Orientado para obtenção do título de Bacharel em Ciência da Computação.

Orientadora

Profa. Olinda Nogueira Paes Cardoso

LAVRAS
MINAS GERAIS - BRASIL
2001

VANESSA GODOY KINOSHITA

BANCO DE DADOS VIA WEB: UMA ANÁLISE COMPARATIVA

Monografia de Graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências da disciplina Projeto Orientado para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em 28 de Junho de 2001.

_____
Prof. Heitor Augustus Xavier Costa
DCC/UFLA

_____
Profa. Olinda Nogueira Paes Cardoso
DCC/UFLA
(Orientadora)

LAVRAS
MINAS GERAIS - BRASIL

“O maior heroísmo de um jovem é manter o sorriso no rosto, mesmo que o coração estraçalhe de dor.”
(Autor desconhecido)

Dedico este trabalho a todos que me ajudam a sorrir.

RESUMO

Palavras-chave: bancos de dados, *Web*, *scripts Web*, sistemas gerenciadores de bancos de dados, análise comparativa.

A *World Wide Web* ou *Web* tem se caracterizado como um dos maiores mecanismos de disseminação de informações da atualidade. A *Web* permite que os bancos de dados possam ser acessados de qualquer lugar do mundo. Para desenvolver aplicações *Web* que permitam o acesso aos dados, algumas ferramentas são necessárias, como os *scripts Web*, os servidores *Web* e as linguagens de formatação. A escolha das ferramentas adequadas reflete no sucesso da aplicação, mas a escolha não é tão simples devido à grande quantidade de ferramentas disponíveis no mercado. O objetivo do trabalho foi apresentar as vantagens e as desvantagens de três soluções que permitem o acesso de banco de dados via *Web*, a fim de facilitar a escolha do usuário interessado em utilizar tais soluções. A primeira solução foi a integração do SGBD PostgreSQL e do *script* PHP4. A segunda solução integra o SGBD MySQL e o *script* ASP. E a última solução utiliza o SGBD Oracle8i. O trabalho apresenta uma análise comparativa entre as soluções. Aparentemente, a implementação com PostgreSQL e PHP4 mostrou ser mais robusta, apresentando bons resultados na maioria das funcionalidades analisadas.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

Atualmente, a *World Wide Web* (WWW ou *Web*), tem se caracterizado como um dos maiores mecanismos de disseminação de informações. A *Web* permite às pessoas armazenar vasta quantidade de informações para acesso público ou controlado. Uma grande porção dessas informações requer gerenciamento, muito do qual é oferecido pelos sistemas de bancos de dados.

Bancos de dados, no passado, só eram capazes de serem acessados por várias pessoas com o uso de LAN's e WAN's (*Local* e *Wide Area Networks*). Agora, com a Internet, pode-se prover acesso a bancos de dados de qualquer parte do mundo.

A utilização de um banco de dados via *Web* é imprescindível para o bom funcionamento de muitas organizações. No entanto, organizar informações de forma racional e, principalmente, recuperá-las e distribuí-las a quem de fato as necessita, traz à tona a necessidade da implementação de conceitos e ferramentas de banco de dados. Estas aplicações são desenvolvidas através do uso das mais diversas ferramentas. Utilizar as ferramentas corretas durante o desenvolvimento das aplicações pode evitar erros de processamento e determinar o sucesso do sistema. Os aspectos mais críticos que devem ser avaliados são portabilidade, desempenho e segurança.

Verifica-se que há, hoje em dia, uma grande flexibilidade na escolha das ferramentas para a estrutura das aplicações. Existem, no mercado, diferentes sistemas de gerenciamento de banco de dados, por exemplo, o Oracle, PostGreSQL e MySQL. Para acessar o banco de dados pode-se utilizar diferentes tecnologias, tais como CGI, ASP, PHP, entre outros. O cliente pode receber a informação requerida em formato HTML ou XML.

Com certeza, a escolha das ferramentas corretas para o desenvolvimento de uma aplicação tornou-se uma tarefa árdua. Determinar a melhor arquitetura

para o sistema, atendendo aos objetivos finais, pode gastar tempo e principalmente dinheiro.

O objetivo deste trabalho é apresentar as vantagens e desvantagens de três soluções que permitem o acesso de banco de dados via *Web*, a fim de facilitar a escolha do usuário interessado em utilizar tais soluções.

O texto está estruturado em oito capítulos. O primeiro, já apresentado, esclarece o objetivo do projeto. O segundo capítulo introduz os conceitos básicos sobre Internet, *Web*, banco de dados e *scripts Web*. O terceiro capítulo apresenta as ferramentas disponíveis no mercado, divididas em SGBDs, *scripts Web* e linguagens de marcação. O quarto capítulo detalha os passos das implementações desenvolvidas. O quinto capítulo apresenta os resultados da análise comparativa. O sexto capítulo conclui o trabalho. O sétimo capítulo propõe idéias para trabalhos futuros e o último capítulo apresenta as referências bibliográficas utilizadas. Além disso, há cinco anexos (de A a E) que contêm listagens e códigos usados nas implementações.

2 INTEGRANDO WEB COM BANCO DE DADOS

2.1 Internet e a *World Wide Web* (WWW)

A Internet é uma "rede de redes de comunicações", significando que muitas redes de comunicação diferentes, operadas por uma grande quantidade de organizações, estão interconectadas coletivamente para formar a Internet [Wya95], como mostra a figura 1. A Internet permite que as pessoas se comuniquem, compartilhem recursos e dados. A quantidade de informação disponível na rede é enorme, bem como o conhecimento armazenado nas pessoas que interagem nela [VD95]. Contudo, a falta de controle central pode tornar-se aqui uma desvantagem. Não existe um catálogo central onde essas informações estão armazenadas. Uma maneira de tentar gerenciar uma parte dessas informações é utilizar sistemas de banco de dados.

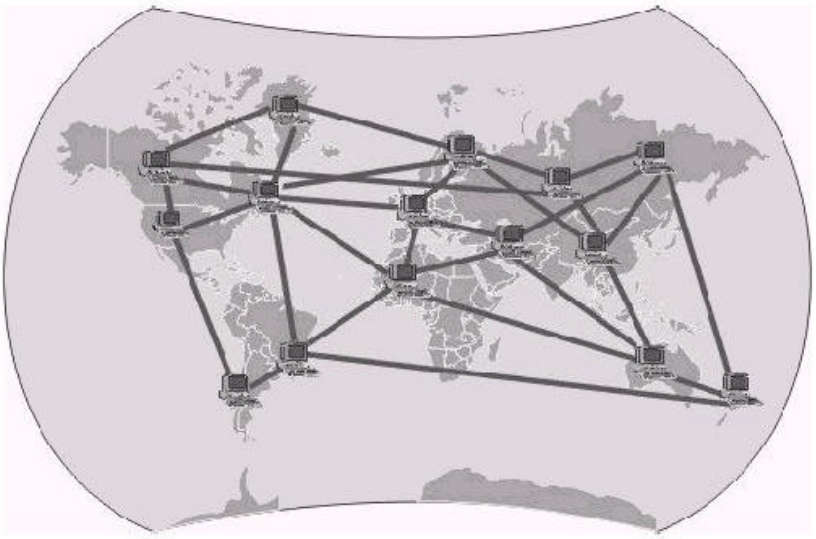

Figura 1 - Internet

Se sob o ponto de vista físico a Internet é uma conexão entre redes, para o usuário ela aparece como um grupo de serviços disponíveis para intercâmbio

de informações entre computadores ou indivíduos conectados à Internet. Segundo Tanenbaum [Tan97], tradicionalmente a Internet tem quatro serviços principais, mostrados a seguir:

- Correio eletrônico: Esse serviço permite que qualquer usuário da Internet possa enviar uma mensagem para qualquer outro usuário;
- News: Os *newsgroups* são fóruns especializados nos quais os usuários com um interesse em comum podem trocar mensagens;
- Login remoto: Utilizando o Telnet ou outros programas, os usuários de qualquer lugar da Internet podem estabelecer *login* com qualquer outra máquina na qual tenham uma conta;
- Transferência de arquivos: Utilizando o FTP (*File Transfer Protocol*), é possível copiar arquivos entre máquinas ligadas à Internet.

Até o início da década de 1990, a Internet era um verdadeiro reduto de pesquisadores ligados às universidades, ao governo e à indústria. Um novo serviço, a *World Wide Web*, mudou essa realidade e atraiu para a rede milhares de novos usuários, sem a menor pretensão acadêmica. A *World Wide Web* é uma rede virtual (não-física) "sobre" a Internet, que torna os serviços disponíveis na Internet totalmente transparentes para o usuário e ainda possibilita a manipulação multimídia da informação [VD95]. Assim, qualquer usuário pode ter acesso a uma quantidade enorme de informações na forma de imagens, textos, sons e até mesmo vídeos, navegando através de palavras-chaves, ícones e ponteiros (*links*).

A figura 2 mostra a arquitetura simplificada da *Web*, que é uma típica arquitetura cliente/servidor [Tan97]. De um lado fica o cliente, composto por *browsers Web*, que são programas capazes de exibir e solicitar documentos da rede. Opcionalmente, o cliente pode estar acompanhado por aplicativos externos usados na apresentação do documento, ou parte destes, caso o *browser* sozinho não seja capaz de interpretar algum tipo de dado (por exemplo, sons ou imagens

em movimento). Do outro lado da arquitetura *Web* fica o servidor, composto pelo servidor *Web*, cuja principal função é atender os pedidos dos clientes *Web* por documentos armazenados no sistema de arquivos da plataforma onde se encontra instalado. Dependendo do pedido do cliente *Web*, o servidor *Web* pode disparar uma aplicação externa.

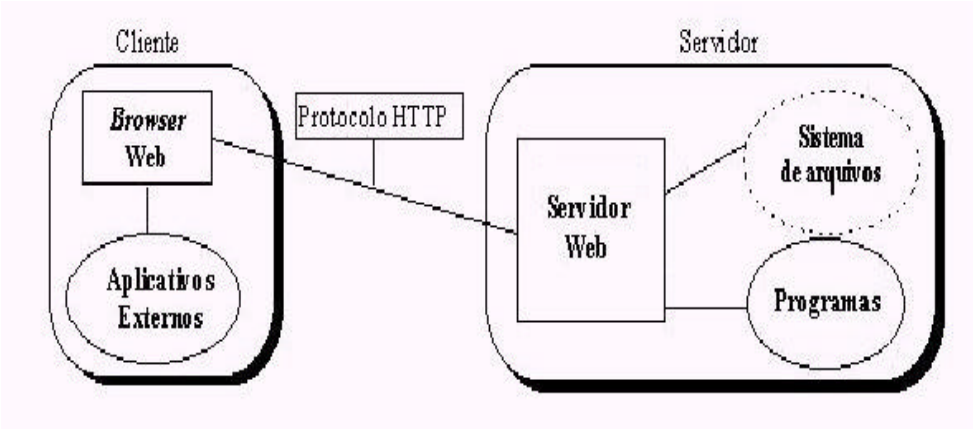

Figura 2 - Arquitetura *Web* simplificada

Uma característica muito importante da *Word Wide Web* é que ela é baseada em três padrões abertos que permitem transferir a informação, descrever a formatação da informação e localizar a informação distribuída pela rede [Sá00]:

- Transferir a informação: Para o transporte da informação entre o servidor e o cliente *Web* foi proposto um protocolo de comunicação denominado HTTP (*HyperText Transfer Protocol*). Sua principal característica é ser um protocolo aberto e especializado na transmissão de documentos *Web* sob a Internet;
- Descrever a formatação da informação: Para a apresentação e formatação de documentos na *Web* é utilizada uma das linguagens

baseadas no padrão SGML *(Standard Generalized Markup Language*), que permite que a estrutura dos documentos *Web*, bem como os vínculos *(link*s) a outros documentos e recursos da Internet, sejam incorporados diretamente no formato texto. Documentos codificados em uma dessas linguagens podem ser interpretados pelos *browsers* e formatados segundo as características de cada plataforma em que são exibidos;

- Localizar a informação: Para a identificação e localização de documentos WWW distribuídos pela Internet foi proposta a utilização de um formato para endereçamento de documentos denominado URL *(Uniform Resource Locator*). Um URL possui três partes: o protocolo (*http*), o nome DNS (*Domain Name System*) da máquina em que a página está (*www.comp.ufla.br*) e um nome local que indica a página específica (*index.html*).

A comunicação entre cliente e servidor *Web* é sempre feita na forma de pares requisição/resposta e é sempre iniciada pelo cliente. Inicialmente, o cliente envia uma mensagem de sincronização para o servidor pedindo abertura de conexão. O servidor responde aceitando ou não este pedido. Se aceitar, a conexão é estabelecida. Uma recusa pode indicar que o servidor está sobrecarregado e não consegue tratar novos pedidos de conexão que chegam.

Uma vez estabelecida a conexão, o cliente envia uma mensagem com a requisição feita pelo usuário representada por um URL. Esta mensagem é enviada através da rede para o servidor, que busca a informação no disco local ou dispara a execução de uma aplicação externa. Após o processamento do pedido, uma mensagem de resposta é enviada ao cliente que, ao recebê-la, retorna um pedido de fechamento de conexão.

Quando o cliente recebe a informação solicitada, é necessário verificar se os dados são válidos, através de informações contidas em campos do cabeçalho

da mensagem. Somente após esta verificação o cliente pode mostrar o documento para o usuário [Sá00].

A *World Wide Web* foi talvez a grande responsável pela expansão dos bancos de dados acessados remotamente. A quantidade enorme de ferramentas disponíveis para a fazer a comunicação usuário/ banco de dados permite que as informações estejam ao alcance de qualquer pessoa.

Em suma, a *Web* pode ser considerada um dos principais veículos de informação usado no mundo inteiro e a comunidade de banco de dados ganha novos contornos usando-a para se expandir.

2.2 Banco de Dados

Um dos recursos utilizados para facilitar a busca de informações é o banco de dados. Um banco de dados é um conjunto de dados relacionados [EN94]. Por exemplo, uma pessoa possui nome, data de nascimento e telefone. Cada atributo da pessoa é um dado separado, mas juntos eles possuem um significado, ou seja, caracterizam a pessoa.

Utiliza-se banco de dados pelos seguintes motivos [Sil01]:

- é compacto, pois elimina o volume de arquivos de papéis;
- é rápido, pois pode recuperar e modificar os dados muito mais rapidamente do que o ser humano;
- integra os dados e evita redundância, pois o sistema de banco de dados proporciona ao usuário o controle centralizado de seus dados operacionais. Contrasta o que acontece em empresas sem um sistema automático, em que cada aplicação possui seus próprios arquivos de tal forma que os dados operacionais são muito dispersos;
- compartilhamento, pois os dados podem ser compartilhados por diversos usuários em um banco de dados;
- permite o acesso com restrições de segurança; e

- possui padrões bem definidos, o que facilita a troca de informações.

Um sistema de banco de dados computadorizado pode ser criado e mantido por um SGBD (Sistema Gerenciador de Banco de Dados) [EN94]. A pessoa (ou grupo de pessoas) responsável pelo controle do sistema é denominada administrador de banco de dados (DBA) [Dat90]. As principais responsabilidades do DBA incluem a decisão do conteúdo do banco de dados e da estrutura de armazenamento, a estratégia de acesso e recuperação, monitoramento do desempenho, além de ter que garantir a segurança e a integridade dos dados.

Uma característica fundamental dos bancos de dados é que eles provêm um nível de abstração dos dados, escondendo detalhes de armazenamento dos dados que não são necessários para a maioria dos usuários de banco de dados. O modelo de dados é a ferramenta principal utilizada para garantir esta abstração. Um modelo de dados é um conjunto de conceitos que se usa para descrever a estrutura do banco de dados e certas restrições que o banco deve garantir [EN94].

Há três categorias de modelos de dados, o conceitual (descreve a estrutura dos dados de maneira abstrata sem se preocupar com a implementação física), o físico (descreve aspectos físicos de implementação) e o de implementação (uma categoria com características dos dois primeiros). Neste trabalho, três modelos de dados são abordados, o modelo entidade-relacionamento (conceitual), o modelo orientado a objetos (implementação) e o modelo relacional (implementação).

Em qualquer modelo de dados é importante distinguir a descrição do banco de dados do banco de dados propriamente dito. A descrição de um banco de dados é chamada de esquema de banco de dados.

O modelo de dados é o principal critério utilizado para se classificar os SGBDs. O mais tradicional é o relacional, mas alguns SGBDs recentes são

baseados em modelos orientados a objetos. Os SGBDs serão abordados no capítulo 3.

2.3 Arquitetura básica

A troca de informações na *Web*, como mostrado anteriormente, é baseada em um modelo cliente/servidor, onde um cliente HTTP (*browser* cliente *Web*) se comunica com um servidor HTTP (Servidor *Web*) requisitando arquivos. Geralmente estes arquivos encontram-se no formato HTML (*HyperText Markup Language*). Ao receber o arquivo HTML o cliente verifica cada linha de comando, solicitando ao servidor HTTP os arquivos indicados. Esse é o modelo básico e estático de apresentação de páginas na Internet, não podendo haver nenhuma interação com o usuário. A figura 3 mostra este modelo:

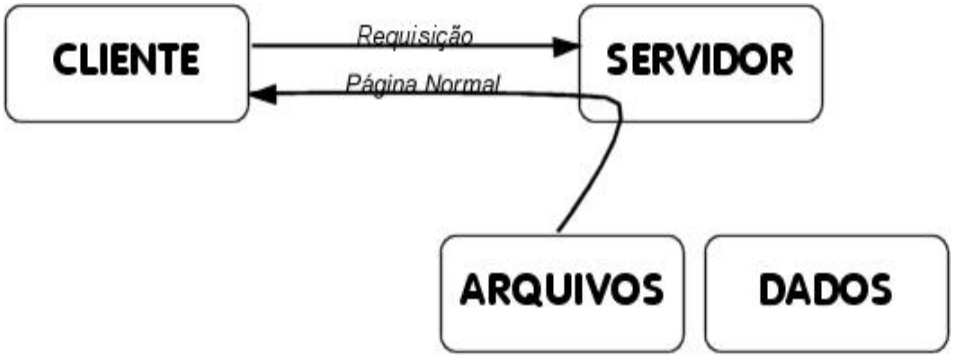

Figura 3 - Modelo estático de apresentação de páginas na Internet

No entanto, há situações em que é necessário que os documentos sejam gerados em tempo de execução *(on line*) através da interação com o usuário. Para proporcionar serviços dinâmicos como estes no ambiente *Web*, pode-se usar programas auxiliares que são executados na plataforma do servidor quando solicitados pelo usuário. Estes programas são chamados *scripts Web*.

Um *script Web* é um programa que pode ser executado pelo servidor *Web* em resposta a um pedido de um cliente *Web* [Sá00]. Pode ser um código compilado a partir de um programa escrito em C, C++ ou Java, um código interpretado como Perl ou PHP ou, por exemplo, um *shell script* do Unix. Um *script Web* pode chamar outros programas ou contactar outros servidores. Seu principal objetivo é gerar uma informação dinâmica (em tempo de execução) para o cliente *Web* que o solicitou. Por exemplo, um *script* pode usar os campos de entrada de um formulário HTML para gerar e enviar uma consulta apropriada a um servidor de Banco de Dados remoto. Cabe ao *script* retornar os dados em um formato que o cliente *Web* consiga interpretar, como por exemplo, no formato HTML. A figura 4 mostra o modelo dinâmico de apresentação de páginas na Internet:

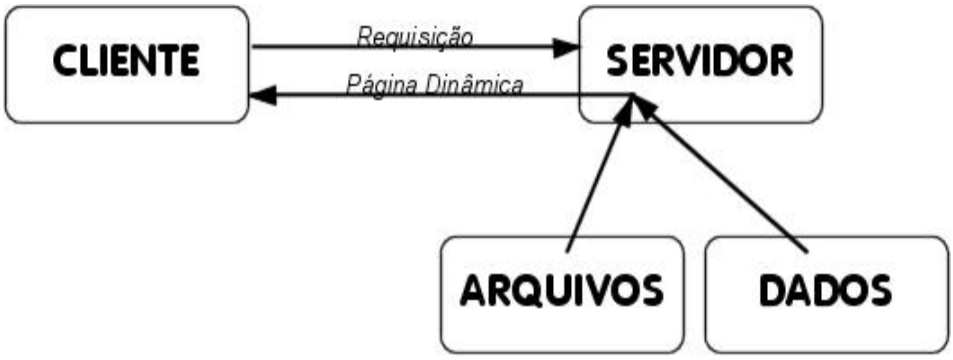

Figura 4 - Modelo dinâmico de apresentação de páginas na Internet.

É importante que as aplicações *Web* preservem determinadas características presentes em inúmeras aplicações sobre Banco de Dados. Neste sentido, serão identificadas a seguir uma série de funcionalidades que as aplicações *Web* devem apresentar [LL98]:

- Gerenciamento do estado da aplicação: O estado da aplicação no contexto de banco de dados é um conjunto de informações mantidas

pelo SGBD, relativas à conexão atual do usuário, como, por exemplo, quem é o usuário e quais objetos do banco de dados estão sendo gerenciados pelo SGBD. O gerenciamento do estado da aplicação possibilita a autenticação de usuários e controle de sessão do usuário;

- Segurança: Diz respeito às restrições de acesso aos objetos (dados e procedimentos) gerenciados pelo SGBD. A *Web* é reconhecidamente um ambiente não seguro e a disponibilização de aplicações *Web* deve levar em consideração os requisitos básicos de restrições de acesso aos objetos do banco de dados;
- Desempenho: Em particular, o tempo de resposta para uma determinada transação *Web*. Devem ser analisados os fatores que deterioram ou melhoram o desempenho das aplicações *Web*;
- Implementação: Nível de dificuldade técnica para implementar uma aplicação *Web* segundo as características da arquitetura apresentada;
- Portabilidade: Diz respeito à independência da plataforma na qual foi desenvolvida a aplicação *Web*. Também é considerada a facilidade de integração da aplicação com outras ferramentas, tais como novos clientes e/ou servidores *Web* e novos SGBDs;
- Custo: Diz respeito ao preço total da aplicação *Web*.

3 VISÃO GERAL DAS FERRAMENTAS DISPONÍVEIS NO MERCADO

Em primeiro lugar, vale ressaltar que o aparecimento de novas tecnologias para a Internet (especialmente para *Web*) permite utilizar ferramentas mais poderosas, que não existiam há alguns anos. Este capítulo tem o objetivo de apresentar algumas destas ferramentas desenvolvidas, divididas em três áreas: os SGBDs, os *scripts Web* e as linguagens de marcação (que permitem gerar documentos em um formato que o cliente *Web* consiga interpretar).

3.1 Sistemas Gerenciadores de Bancos de Dados (SGBDs)

A década de 60 marcou a área de processamento de dados pela criação e utilização em ambiente comercial dos primeiros programas especializados no gerenciamento de bancos de dados [Net00]. Entretanto, estes produtos careciam de versatilidade e facilidades para recuperação das informações e, fundamentalmente, não possuíam uma base teórica que permitisse a implementação de modelos matemáticos da realidade.

Nos anos 80 surgiu o modelo relacional como uma nova técnica para modelagem de banco de dados que viria a determinar a evolução do mercado de SGBD, com uma sólida base teórica definida pelo Dr. E. F. Codd. A utilização de Sistemas Gerenciadores de Banco de Dados Relacionais (SGBDRs) tem sido de longe a opção mais utilizada pelos gerentes e administradores de informática para o desenvolvimento de sistemas de informação comerciais desde então.

Apesar de serem amplamente utilizados, os tradicionais SGBDRs apresentam limitações na implementação de aplicações que manipulem tipos complexos de informação como dados compostos e multivalorados, imagens, sons, textos, vídeos (multimídia), dados geoespaciais ou de tempo, ou qualquer dado que um usuário deseje definir. Apesar da tecnologia de orientação a objeto ser largamente aplicada no mercado de software desde a década de 80, foi na década de 90 que alguns produtos para gerenciamento de bancos de dados

orientados a objetos (SGBDOOs) surgiram no mercado, permitindo solucionar duas das limitações dos bancos de dados relacionais: suporte para dados e aplicações multimídia e a modelagem de dados mais próxima do mundo real.

Estes produtos incorporam poderosos mecanismos para manipulação de objetos e dados não estruturados ou complexos. Apesar disto, ainda não chegaram a um grau de maturidade aceitável para o desenvolvimento de sistemas que gerenciem grandes volumes de dados ou processem um elevado número de transações. É bastante provável entretanto que os bancos de dados orientados a objeto não substituam os bancos relacionais em todas as áreas de negócio.

Dentro do processo de evolução dos SGBDRs está a incorporação de recursos para armazenamento e manipulação de objetos, baseados no modelo relacional. Da extensão da capacidade de manipulação de dados complexos surgiu o termo Gerenciador de Banco de Dados Objeto Relacional (SGBDOR) [GZS01].

Estes produtos se aproveitam da versatilidade, robustez e tecnologia consagradas dos SGBDR´s incorporando a flexibilidade e potencialidade da orientação a objetos num único produto, permitindo a recuperação de objetos complexos através da extensão do endereçamento do conteúdo relacional. Uma das principais características é permitir que o usuário defina tipos adicionais de dados, especificando a estrutura e a forma de operá-lo, e use estes tipos no modelo relacional [GZS01].

Hoje praticamente todas as grandes empresas de software de gerenciamento de banco de dados do mercado oferecem produtos seguindo esta linha de evolução integrado em seus produtos, ou através de pacotes adicionais. Isto tem ocorrido de fato por que muitas empresas que usam SGBDRs não precisam dos recursos do modelo orientado a objeto em todas as suas aplicações,

não existindo, portanto, um interesse unânime numa revolução tecnológica mas sim em uma evolução dos atuais produtos.

Na tabela 1, são apresentados os SGBDs mais vendidos, divididos em SGBDR, SGBDOR e SGBDOO:

Tabela 1 - SGBDs (Fabricantes x Produtos)

	Produtos		
Fabricantes	**SGBDR**	**SGBDOR**	**SGBDOO**
Oracle	Oracle 7.x	Oracle 8.x	
Sybase	System 10/11		
Informix	Dynamic Server	Universal Server	
IBM	DB/2	Universal Database	
UniSQL		UniSQL/X	
Unisys		OSMOS	
Comp. Associates	OpenIngres		Jasmin
Gemstone			Gemstone
O2			O2
Object Design			Object Design
Objectivity			Objectivity/DB
Versant			Versant ODBMS

Fonte: International Data Corporation, 1997

3.2 *Scrips Web*

O *script Web* é a ferramenta responsável em fazer a comunicação entre o banco de dados e o servidor *Web*. Pode ser um programa compilado como o CGI ou interpretado como o PHP e o ASP.

3.2.1 Programas CGI

Para que um servidor *Web* possa gerar uma informação dinâmica foi especificada a interface CGI (*Common Gateway Interface*). CGI é uma interface padrão para execução de programas externos ao servidor *Web* [LL98]. A interface descreve regras de como os programas externos ao servidor *Web* serão inicializados e como dados são passados entre o servidor *Web* e o aplicativo.

A arquitetura mais imediata para se desenvolver aplicações *Web* é através de programas CGI, uma vez que para implementar aplicativos através da interface CGI é necessária uma linguagem de programação que possa ler da entrada padrão, obter as variáveis de ambientes e escrever na saída padrão. Como a grande maioria dos SGBDs existentes incorpora alguma linguagem de programação hospedeira com estas características, praticamente todos eles podem interagir com a *Web* por meio de programas CGI. A figura 5 ilustra a arquitetura.

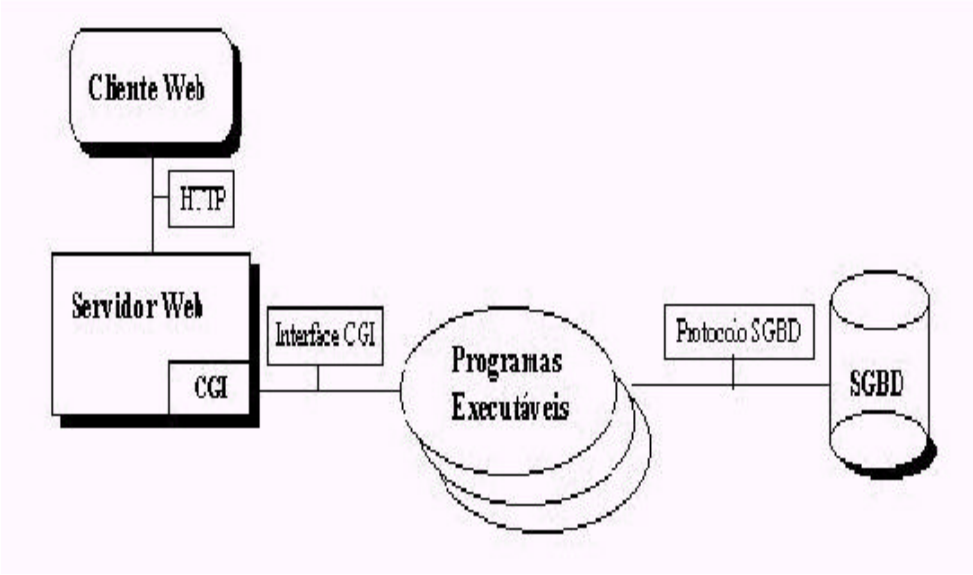

Figura 5 - Arquitetura de Programas CGI

O funcionamento da arquitetura é mostrado a seguir [Sá00]:

1. Cliente *Web* solicita ao servidor *Web* a execução de uma aplicação externa, que neste caso é um programa executável na plataforma do servidor *Web*;
2. Servidor *Web* dispara um processo para execução do programa através da interface padrão CGI, enviando os dados recebidos do cliente *Web* de acordo com a implementação do servidor *Web* para a interface CGI;
3. O programa recebe os dados passados pelo servidor *Web* via interface CGI, decodifica-os, formula o comando SQL e abre uma conexão com o banco de dados para sua execução;
4. O SGBD retorna os dados e a conexão é finalizada. O programa executável trata os dados recebidos e os repassa ao servidor *Web* via interface CGI, num formato que o cliente *Web* entenda, como por exemplo, no formato HTML. O processo iniciado pelo servidor *Web* para execução do programa é finalizado neste momento;
5. O Servidor *Web* retorna os dados repassados pelo programa ao cliente *Web*.

A maior desvantagem dos programas CGI está relacionada ao desempenho. Isto se deve ao fato de que, para cada pedido de um cliente *Web*, um novo processo é iniciado em separado pelo servidor *Web* [LL98]. Sendo assim, um servidor que recebesse várias requisições simultâneas facilmente se sobrecarregaria e travaria, pois cada cliente que invoca este executável cria uma nova instância diferente do mesmo.

Outra desvantagem apresentada está relacionada à implementação. Os programas CGI são pouco legíveis, sendo relativamente difíceis de serem depurados. A segurança também é limitada, pois como é o servidor *Web* que solicita a execução dos programas CGI, o acesso é realizado, na maioria das

vezes, sob a conta do servidor *Web* e não a conta do usuário cliente em particular.

No entanto, a arquitetura apresenta boas características de portabilidade, uma vez que é baseada na interface padrão CGI que atualmente é suportada por quase todos os servidores *Web*.

3.2.2 ASP

Uma alternativa ao CGI, é o ASP (*Active Server Page*) da Microsoft, que funciona de forma diferente. A partir do cliente (*browser*) faz-se uma requisição para uma página ASP hospedada em determinado servidor Windows NT usando o protocolo HTTP que recebe e processa a requisição. O arquivo solicitado é lido no servidor que determina o que deve ser interpretado por ele e depois devolvido ao cliente.

A arquitetura, apresentada na figura 6, funciona da seguinte maneira: o cliente requisita uma página asp (ex.: teste.asp), então o servidor processa todo o código desta página, criando uma página HTML e mandando ao cliente apenas esta página. O código, então, não é exibido ao cliente, garantindo a segurança de informações sigilosas.

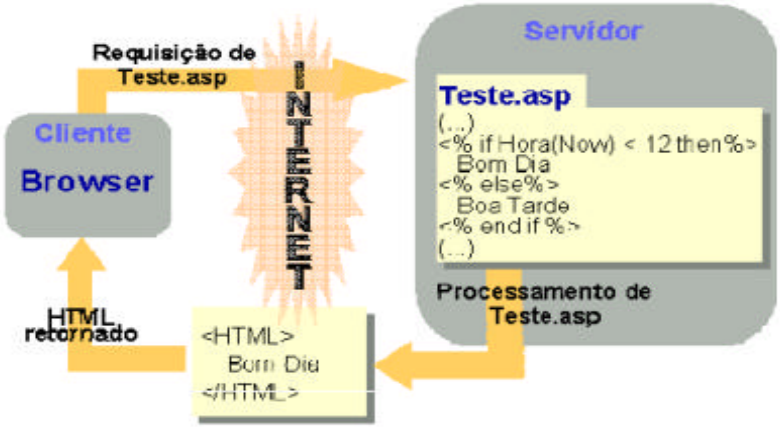

Figura 6 - Arquitetura do *script* ASP

A plataforma mais utilizada para hospedar uma página ASP é o Windows NT Server 4.0 (ou superior) com o Internet Information Service 3.0 (ou superior), sendo esse último o programa servidor *Web* da Microsoft. Para efeito de teste, também é possível executar uma página ASP com o *MS Personal Web Server* (PWS) para Windows NT Workstation ou Windows 9.x, embora não seja muito recomendado. Para plataformas Unix/Linux, já existem módulos que fornecem suporte ao ASP. A empresa CHILI!SOFT está disponibilizando para compra online desde 08/03/2000, a versão final de seu software para plataforma Linux, o CHILI!ASP [Mar00].

As principais vantagens do ASP sobre o CGI são [Bar00a]: segurança na execução, facilidade na programação, utilização de uma linguagem interpretada e não compilada, facilidade de acesso a base de dados e proteção do código fonte da página, pois o servidor retorna somente o resultado HTML. O ASP fornece todos os recursos de aplicações CGI de uma forma mais fácil e mais robusta. Com ASP, é bem mais fácil criar conexões entre o *browser* e os dados em formatos normalmente incompatíveis com HTML, como bancos de dados. O ASP é mais robusto por não criar um processo no servidor para cada pedido do usuário, como acontece com o CGI. Usando ASP ao invés de CGI, um servidor pode atender a um grande número de pedidos de usuários de forma mais rápida e usando menos memória. Além disso, criar páginas ASP é em geral muito mais fácil do que criar aplicações CGI.

Mas por outro lado, ASP só roda em sistemas e servidores da Microsoft, enquanto CGI é aceito em todos os servidores (exceto alguns gratuitos).

3.2.3 PHP

Assim como o ASP, o PHP (*Personal Home Page*) [Bar00b] é um *script Web* cujo código PHP é executado no servidor, sendo enviado para o cliente apenas

HTML puro. Desta maneira é possível interagir com bancos de dados e aplicações existentes no servidor, com a vantagem de não expor o código fonte para o cliente. A sintaxe do PHP é baseada nas linguagens C, Java e Perl, facilitando o desenvolvimento para pessoas familiarizadas com tais linguagens.

Uma dos pontos fortes do PHP é o suporte a um grande número de bancos de dados, como Informix, Microsoft SQL Server, MySQL, Oracle, Sybase, PostgreSQL e vários outros. Além disso, pode ser utilizado tanto em Windows como em Unix (portabilidade) e possui código aberto.

3.2.4 Outros *Scripts Web*

Há vários outros *scripts Web*, mas os apresentados neste trabalho são os mais populares. Entre as alternativas que podem ser utilizadas para a geração de páginas dinâmicas estão o JSP (*JavaServer Pages*), Javascript, SSI (*Server-Side Includes*), API (*Application Programming Interface*), ColdFusion e Java servlets. Uma lista completa pode ser encontrada em [Ash99].

3.3 Linguagens de Marcação

3.3.1 HTML

As páginas da *Web* são normalmente desenvolvidas em uma linguagem chamada HTML (*HyperText Markup Language*) [Sav96]. A HTML é uma linguagem de marcação, ou seja, uma linguagem para descrever como os documentos devem ser formatados. Trata-se de uma linguagem versátil, que permite que seus usuários criem páginas da *Web* que incluam textos, elementos gráficos e ponteiros para outras páginas.

Os criadores da HTML se basearam na SGML (*Structured Generalized Markup Language*), que é uma linguagem desenvolvida pela ISO (*International Standards Organization*) amplamente utilizada no mundo inteiro para marcação

de nível superior. Então, a HTML pode ser considerada um subconjunto do padrão SGML adaptada à *Web*.

Uma página HTML consiste em um cabeçalho e um corpo entre as *tags* (comandos de formatação) <HTML> e </HTML>. O cabeçalho começa e termina com as *tags* <HEAD> e </HEAD> respectivamente, enquanto com o corpo as *tags* usadas são <BODY> e </BODY>. Os comandos dentro das *tags* são chamados de diretivas. A maioria das *tags* da HTML tem esse formato, ou seja, <ALGO> para marcar o início de alguma formatação e </ALGO> para marcar seu fim. Há várias outras diretivas que podem ser utilizadas tanto dentro do cabeçalho quanto dentro do corpo do documento [Sav96].

A HTML 1.0 só permitia a comunicação em uma direção. Os usuários podiam acessar páginas dos provedores de informações, mas era difícil enviar informações na direção inversa. Como a necessidade de um tráfego em duas vias tornou-se cada vez maior, devido às empresas que desejavam ter a possibilidade de receber informações dos clientes, houve a inclusão de formulários a partir da HTML 2.0. Os formulários têm caixas ou botões que permitem aos usuários fornecer informações ou fazer escolhas e retornar esses dados para o proprietário da página. Os formulários mais comuns são campos em branco para digitação de dados, caixas que podem ser selecionadas, mapas ativos e botões SUBMIT. Um exemplo simples de um documento HTML é apresentado na figura 7.

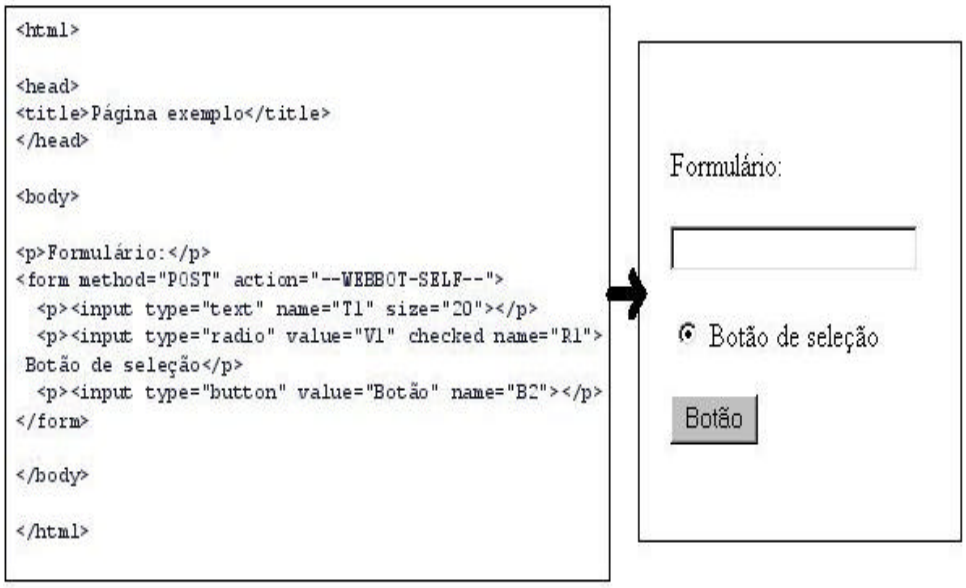

Figura 7 - Exemplo de um documento HTML

Os formulários HTML podem ser enviados ao servidor por meio dos métodos GET e POST [Sil01]. A diferença dos métodos de envio de informações do formulário ao servidor vai influenciar diretamente na forma como os *scripts Web* irão receber essas informações.

Utilizando o método GET, o servidor armazena os dados codificados em uma variável chamada QUERY_STRING. O método GET também tem a característica de colocar o conteúdo do formulário após o URL delimitado por um ponto de interrogação (“?”). Por exemplo, se o nome da caixa de texto for *universidade* e o usuário digitar *UFLA*, a variável será enviada da seguinte forma: “http://nome_do_servidor/nome_arquivo?universidade=UFLA”. Apesar de ser possível passar parâmetros utilizando o método GET, e com isso gerar páginas dinamicamente, este método tem pelo menos dois problemas que em determinadas circunstâncias podem ser considerados sérios. O primeiro é que o

GET permite uma quantidade de dados passados limitada a 1024 caracteres, o que pode gerar perda de informações em certos casos. O segundo é devido ao fato de que as informações fazem parte do URL, todos os dados podem ser vistos pelo usuário. Isso pode ser extremamente perigoso quando informações sigilosas estão envolvidas (senha, por exemplo).

O envio de formulários com o método POST armazena as informações na entrada padrão do sistema operacional. O script pode reconhecer essa entrada como STDIN. Com esse método nada é colocado na variável QUERY_STRING e as outras variáveis ficam disponíveis normalmente. A maior vantagem deste método é a possibilidade de maior quantidade de informação poder ser enviada com mais segurança.

3.3.2 XML

A XML (*eXtensible Markup Language*) também é um subconjunto da SGML que permite que uma nova marcação seja criada para especificar idéias e compartilhá-las na rede [Suh00]. Algumas vantagens da XML são apresentadas a seguir:

- Inteligência**:** um dos maiores poderes da XML, já mencionado anteriormente, é que ela permite que o usuário crie seus próprios nomes para as marcas. Pode-se criar marcas para qualquer nível de complexidade. A marcação pode ser alterada de uma marcação mais geral como "<CÃO> Lassie </CÃO>" para uma mais detalhista, como "<CÃO> <COLLIE> Lassie </COLLIE> </CÃO>";
- Adaptabilidade**:** a XML é a língua-mãe de outras linguagens. Assim, linguagens como a XHTML (eX*tensible HyperText Markup Language*) e SMIL (*Synchronized Multimedia Integration Language*) tornaram-se possíveis;

- Manutenção**:** a XML é fácil de manter, pois ela contém somente idéias e marcações. Folhas de estilos e *links* vêm em separado e não escondidas no documento. Tanto as folhas de estilos como os *links* podem ser alterados separadamente quando preciso, com facilidade;
- Portabilidade**:** a XML é de fácil portabilidade. A razão da sua existência é força e portabilidade. A SGML tem força, por ser mais genérica. A HTML tem portabilidade. A XML tem ambas. Tudo que um *browser* precisa para apresentar um documento XML é ter o documento formatado de acordo com as especificações XML e a folha de estilos para controlar a aparência.

A figura 8 apresenta um documento XML:

```
<?xml version='1.0'?>
<?xml-stylesheet type="text/xsl" href="supermercado.xsl" ?>
<supermercado>
 <nomedosupermercado>Supermercado Peg-Leve</nomedosupermercado>
 <telefone>344-5333</telefone>
 <endereco>Rua Getulio Vargas,45</endereco>
<lista>
 <produtos>
  <nomedoproduto>Feijao Boa Colheita - pcte 1Kg</nomedoproduto>
  <descricao>Carioquinha</descricao>
  <preco>R$ 1,50</preco>
 </produtos>
 <produtos>
  <nomedoproduto>Arroz Americano - pcte 5Kg</nomedoproduto>
  <descricao>tipo 1 - agulhinha</descricao>
  <preco>R$ 3,45</preco>
 </produtos>
</lista>
</supermercado>
```

Figura 8 - Exemplo de um documento XML

A apresentação de um documento XML é dependente de uma folha de estilos. A linguagem de folha de estilos padrão para os documentos XML é a eX*tensible Style Language* (XSL).

A figura 9 mostra a apresentação final de um documento XML com a folha de estilos XSL do anexo A:

Figura 9 - Apresentação final de um documento XML

4 DESCRIÇÃO DAS IMPLEMENTAÇÕES

As implementações basearam-se no modelo de um site de reserva de hotéis. A figura 10 apresenta o mapa do site:

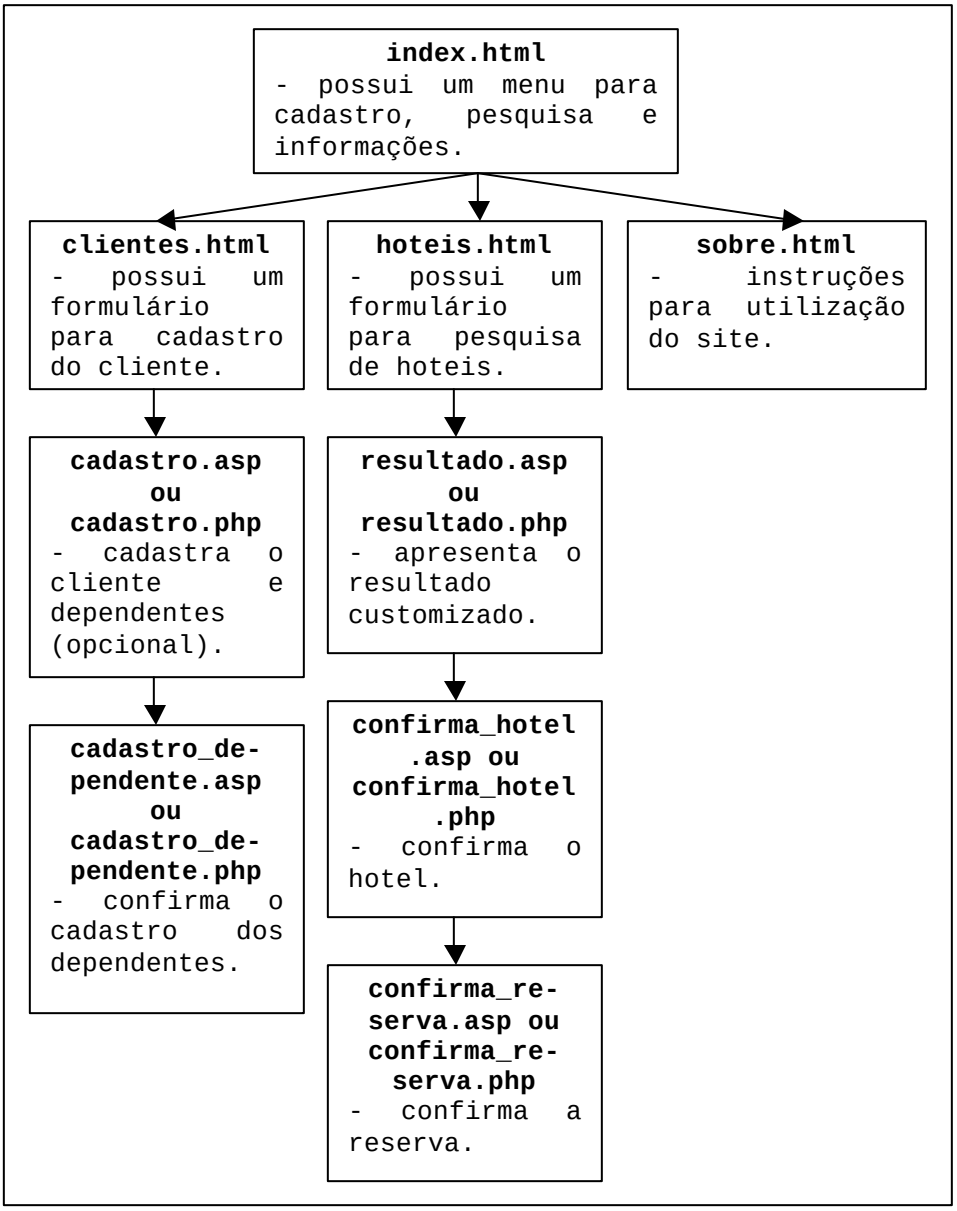

Figura 10 – Mapa do site de reserva de hotéis

O site funciona da seguinte maneira:

- O cliente se cadastra no site através de um formulário desenvolvido em formato HTML. Pelo formulário é possível cadastrar nome, endereço, telefones, data de nascimento, CPF e senha. Há a opção para cadastro de dependentes. Com estes dados, o *script Web* requisita uma conexão com o banco de dados e efetiva o cadastro.
- O cliente, depois de se cadastrar, pode fazer pesquisas customizadas, a fim de escolher em qual hotel a reserva será feita. A customização é realizada através da escolha do estado ou da cidade, da categoria do quarto (standard, luxo ou suíte), do tipo de acomodação (solteiro, duplo, triplo ou quádruplo), do preço do quarto e das datas de entrada e de saída. Após as escolhas, o *script Web* requisita uma conexão com o banco de dados e realiza a consulta. O resultado é apresentado ao cliente em formato HTML.
- O cliente escolhe um dos hotéis que aparecem no resultado e digita a senha para confirmar a reserva. O *script Web*, então, efetiva o cadastro da reserva.

A primeira etapa da implementação foi a modelagem dos dados. Utilizou-se o modelo entidade-relacionamento. A segunda etapa englobou o mapeamento do modelo entidade-relacionamento para o modelo relacional, seguido da implementação em PostgreSQL integrado com o *script* PHP. A terceira etapa utilizou o mesmo modelo relacional para a implementação em MySQL integrado com o *script* ASP. A quarta etapa, e última, utilizou o modelo orientado a objetos para a implementação em Oracle8i. Os próximos subcapítulos apresentam detalhes das implementações. O código fonte dos formulários em formato HTML podem ser encontrados no anexo B (index.html, clientes.html e hoteis.html).

4.1 Modelo Entidade – Relacionamento (ER)

Um dos modelos mais populares é o modelo de dados entidade-relacionamento. O modelo descreve os dados como entidades, relacionamentos e atributos [EN94].

O objeto básico representado pelo modelo entidade-relacionamento é uma entidade, que identifica uma "coisa" do mundo real com existência independente. O modelo do site de reserva *online* possui quatro entidades: o cliente, os dependentes, o hotel e os quartos do hotel. Uma entidade é denominada entidade fraca se sua existência depender da existência de outra entidade. Por exemplo, só existem quartos se existir um hotel, então o quarto é considerado uma entidade fraca. O mesmo acontece com os dependentes com relação ao cliente.

Cada entidade possui propriedades ou atributos utilizados para descrevê-la. Por exemplo, o cliente possui nome, endereço, telefone, CPF, entre outros. Os atributos podem ser simples, como o nome, ou compostos, como o endereço. O endereço é considerado atributo composto porque é formado por rua, bairro, cidade, estado, CEP e país. Os atributos também podem ser classificados como monovalorados, como o CPF, ou multivalorados, como o telefone (o atributo pode possuir *n* valores).

Um tipo de entidade define um conjunto de entidades que possuem os mesmos atributos [EN94]. Por exemplo, há vários clientes com os mesmos atributos, porém com valores diferentes para cada atributo.

As chaves são os identificadores de um tipo de entidade. É o atributo de um tipo de entidade que possui um valor único para cada entidade [EN94]. Por exemplo, o CPF do cliente ou o CNPJ do hotel são chaves.

Uma associação entre duas ou mais entidades distintas com um determinado significado é denominada relacionamento [EN94]. Um relacionamento pode ser de um para um, de um para *n* ou de *n* para *m*. Por exemplo, a entidade cliente possui dependentes (relacionamento de um para *n*), o hotel possui quartos (relacionamento de um para *n*) e os clientes podem reservar quartos (relacionamento *n* para *m*).

O modelo entidade-relacionamento pode ser representado graficamente por um diagrama com a seguinte notação:

- Cada tipo de entidade é mostrada em uma caixa retangular, etiquetada com o nome do tipo de entidade em questão;
- As caixas retangulares que representam entidades fracas possuem linha dupla;
- Se todas as intâncias de uma entidade fraca estão obrigatoriamente relacionadas a uma entidade, as entidades devem ser conectadas por uma linha dupla (totalidade);
- Os atributos são mostrados como elipses e conectados ao tipo de entidade por uma linha reta;
- Um atributo chave é representado pelo nome do atributo sublinhado;
- Cada tipo de relacionamento é mostrado em losango, etiquetado com a denominação do tipo de relacionamento em questão;
- Cada caixa de relacionamento é conectada por linhas às caixas de entidades, participando do relacionamento em questão;
- Cada linha é etiquetada com “um” ou “muitos”, a fim de definir qual o tipo de relacionamento entre as entidades.

A figura 11 apresenta o modelo entidade-relacionamento do site de reserva de hotéis:

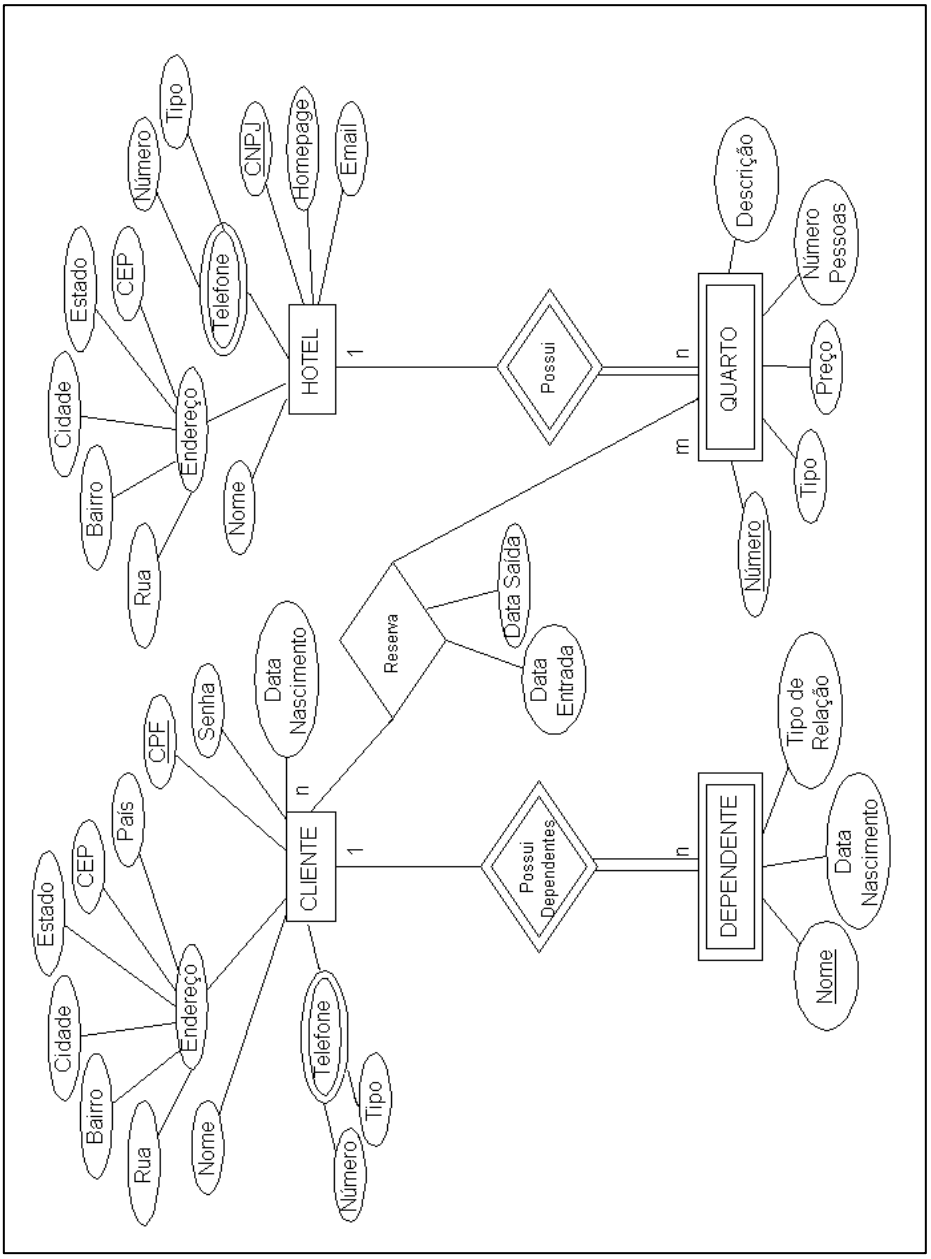

Figura 11 – Modelo entidade-relacionamento do site de reserva de hotéis

4.2 Modelo Relacional

O modelo de dados relacional representa um banco de dados como uma coleção de tabelas, onde cada tabela pode ser armazenada como um arquivo separado [EN94]. Cada linha na tabela representa uma coleção de dados relacionados, cujos valores podem ser interpretados como fatos que descrevem uma entidade ou relacionamento do mundo real. O nome da tabela e os nomes das colunas são utilizados para ajudar na interpretação do significado dos dados de cada linha. Por exemplo, a tabela CLIENTE possui este nome porque cada linha possui dados relativos a um cliente.

Na terminologia de modelo relacional, uma tabela é chamada de relação, uma linha da tabela é chamada de tupla e o cabeçalho de uma coluna de atributo. O conjunto de valores que podem aparecer em cada coluna é chamado de domínio.

Para fazer o mapeamento do modelo entidade-relacionamento para o modelo relacional, os seguintes passos foram utilizados:

1. Para cada tipo de entidade regular, uma relação contendo todos os atributos simples da entidade foi criada. No site de reserva de hotéis, duas relações foram criadas, o hotel e o cliente.

HOTEL

hot_nome	hot_rua	hot_bairro	hot_cidade	hot_estado	hot_cep
hot_cnpj	hot_homepage	hot_email			

CLIENTE

cl_nome	cl_rua	cl_bairro	cl_cidade	cl_estado	cl_cep
cl_pais	cl_cpf	cl_dtnasc	cl_senha		

2. Para cada tipo de entidade fraca, uma relação contendo todos os atributos simples da entidade foi criada. Além disso, os atributos que formam a chave da entidade regular (da qual a entidade fraca depende) foram incluídas e estes atributos passaram a fazer parte da chave da

relação criada. No site de reserva de hotéis, duas relações foram criadas, o dependente e o quarto.

DEPENDENTE

dep_cl_cpf	dep_nome	dep_dtnasc	dep_relacao

QUARTO

qto_hot_cnpj	qto_numero	qto_tipo	qto_preco	qto_nopessoas	qto_descricao

3. Para cada atributo multivalorado, uma relação foi criada. Além disso, os atributos que formam a chave da entidade regular (da qual o atributo pertence) foram incluídas e estes atributos passaram a fazer parte da chave da relação criada. No site de reserva de hotéis, duas relações foram criadas, o cliente_telefone e o hotel_telefone.

CLIENTE_TELEFONE

cl_tel_cpf	cl_tel_numero	cl_tel_tipo

HOTEL_TELEFONE

hot_tel_cnpj	hot_tel_numero	hot_tel_tipo

4. Para o relacionamento n para m (relacionamento reserva), criou-se uma relação que possui como chave a combinação das chaves das entidades participantes do relacionamento.

RESERVA

res_cl_cpf	res_hot_cnpj	res_noquarto	res_entrada	res_saida

4.3 Linguagem SQL

Há várias linguagens que podem ser utilizadas para definir, gerenciar, manipular e recuperar dados de um banco de dados. A mais conhecida é a SQL (*Structured Query Language*). SQL usa os termos tabela (*table*), linha (*row*) e coluna (*column*) para relação, tupla e atributo, respectivamente. Os comandos mais importantes são apresentados a seguir:

1. Comando CREATE TABLE – comando utilizado para especificar uma nova relação, seus atributos e restrições.

```
CREATE TABLE <nome da tabela> (
<definição das colunas>
<definição da chave primária>
<definição de chaves alternativas>
<definição de chaves estrangeiras>)
```

2. Comando DROP TABLE – comando utilizado para apagar uma relação e todas as suas chaves e restrições.

```
DROP TABLE <nome da tabela>
```

3. Comando ALTER TABLE – comando utilizado para efetuar alterações na estrutura de uma tabela. Por exemplo, definir chaves ou incluir atributos.

```
ALTER TABLE <nome da tabela> <modificações>
```

4. Comando SELECT – comando usado para executar uma consulta ao banco de dados.

```
SELECT <lista de atributos>
FROM <lista de tabelas>
WHERE <condição>
```

5. Comando INSERT – comando usado para inserir tuplas em uma tabela, individualmente.

```
INSERT INTO <nome da tabela> (atributo1, atributo2,
 ..., atributoN)
VALUES (valor1, valor2, ..., valorN)
```

6. Comando UPDATE – comando usado para alterar uma ou várias tuplas em uma relação.

```
UPDATE <nome da tabela>
SET atributo1 = valor1, atributo2 = valor2, ...,
 atributoN = valorN
WHERE <condições>
```

7. Comando DELETE – comando usado para apagar uma ou várias tuplas em uma relação. Obedece às restrições de remoção impostas na criação da tabela.

```
DELETE FROM <nome da tabela>
WHERE <condições>
```

A linguagem SQL é fundamental para fazer os cadastros e as consultas nos bancos de dados utilizados nas implementações.

4.4 Implementação utilizando PostgreSQL e PHP4

A parte inicial desta etapa foi configurar o servidor *Web* (Apache) do Linux Red Hat 7.0 com o PHP4 para servir de plataforma de testes. Além disso, o PostgreSQL foi instalado e um usuário foi criado para que o banco de dados pudesse ser populado. O PostgreSQL é um SGBDOR que suporta herança, classes, tipos e funções. Porém, estas características não foram utilizadas na implementação. Para a criação das tabelas, as seguintes instruções em SQL foram definidas:

```
1. CREATE TABLE hotel (
   hot_nome VARCHAR(50) NOT NULL,
   hot_rua VARCHAR(50),
   hot_bairro VARCHAR(25),
   hot_cidade VARCHAR(20),
   hot_estado CHAR(2),
```

```
    hot_cep VARCHAR(9),
    hot_cnpj INT8 NOT NULL,
    hot_homepage VARCHAR(50),
    hot_email VARCHAR(50),
    PRIMARY KEY (hot_cnpj) );

2.  CREATE TABLE hotel_telefone (
    hot_tel_cnpj INT8 NOT NULL,
    hot_tel_numero CHAR(15) NOT NULL,
    hot_tel_tipo VARCHAR(15),
    PRIMARY KEY (hot_tel_cnpj, hot_tel_numero),
    FOREIGN KEY (hot_tel_cnpj) REFERENCES hotel(hot_cnpj) );

3.  CREATE TABLE quarto (
    qto_hot_cnpj INT8 NOT NULL,
    qto_numero INTEGER NOT NULL,
    qto_tipo VARCHAR(25),
    qto_preco DECIMAL(9,2),
    qto_nopessoas INTEGER,
    qto_descricao VARCHAR(200),
    PRIMARY KEY (qto_hot_cnpj, qto_numero),
    FOREIGN KEY (qto_hot_cnpj) REFERENCES hotel(hot_cnpj) );

4.  CREATE TABLE cliente (
    cl_nome VARCHAR(60) NOT NULL,
    cl_rua VARCHAR(50),
    cl_bairro VARCHAR(25),
    cl_cidade VARCHAR(20),
    cl_estado CHAR(2),
    cl_cep CHAR(9),
    cl_pais VARCHAR(20),
    cl_cpf INT8 NOT NULL,
    cl_dtnasc DATE,
    cl_senha VARCHAR (20) NOT NULL,
    PRIMARY KEY (cl_cpf) );
```

```
5.  CREATE TABLE cliente_telefone (
    cl_tel_cpf INT8 NOT NULL,
    cl_tel_numero CHAR(15) NOT NULL,
    cl_tel_tipo VARCHAR(15),
    PRIMARY KEY (cl_tel_cpf, cl_tel_numero),
    FOREIGN KEY (cl_tel_cpf) REFERENCES cliente(cl_cpf) );

6.  CREATE TABLE dependente (
    dep_cl_cpf INT8 NOT NULL,
    dep_nome VARCHAR(60) NOT NULL,
    dep_dtnasc DATE,
    dep_relacao VARCHAR(40),
    PRIMARY KEY (dep_cl_cpf, dep_nome),
    FOREIGN KEY (dep_cl_cpf) REFERENCES cliente(cl_cpf) );

7.  CREATE TABLE reserva (
    res_cl_cpf INT8 NOT NULL,
    res_hot_cnpj INT8 NOT NULL,
    res_noquarto INTEGER NOT NULL,
    res_entrada DATE NOT NULL,
    res_saida DATE NOT NULL,
    PRIMARY KEY (res_cl_cpf, res_hot_cnpj, res_noquarto),
    FOREIGN KEY (res_cl_cpf) REFERENCES cliente(cl_cpf),
    FOREIGN KEY (res_hot_cnpj) REFERENCES hotel(hot_cnpj),
    FOREIGN KEY (res_noquarto) REFERENCES   quarto(qto_numero));
```

As páginas com *script* PHP foram desenvolvidas após a criação das tabelas.

Através do método POST, os dados dos formulários HTML são enviados ao servidor e utilizados pelo *script* PHP. As variáveis podem ser utilizadas diretamente pelo *script*, ou seja, se há um campo denominado senha no formulário, o PHP pode utilizar a variável senha no código, apenas inserindo o símbolo $ na frente do nome da variável ($senha). Assim, após o preenchimento

do formulário contido no arquivo clientes.html, os dados são enviados para o servidor e utilizados pelo arquivo cadastro.php. Com os dados, o cadastro.php faz as seguintes ações:

1. Verifica se o nome, CPF ou senha são nulos. Se um dos campos for nulo, solicita ao cliente para voltar à página anterior e preencher o campo.
2. Se os campos acima estiverem preenchidos corretamente, o cadastro é efetivado. Para isso, a seguinte instrução SQL é utilizada: `INSERT INTO cliente VALUES ('$nome','$rua', '$bairro', '$cidade', '$estado', '$cep', '$pais', $cpf, '$dtnascimento', '$senha').` A comunicação com o banco de dados é realizada através dos comandos:

   ```
   $database = pg_Connect ("", "", "", "", "reservaonline");
   $resultado = pg_exec($database, $insercao);
   ```

   O primeiro comando faz a conexão com o SGBD, selecionando o banco de dados a ser acessado. O segundo comando executa a inserção.
3. Para cada telefone preenchido pelo cliente, um cadastro é efetivado com a seguinte instrução SQL: `INSERT INTO cliente_telefone VALUES ($cpf,'$telefone1', '$tipo_tel1').`
4. Se o cliente optar pela inclusão de dependentes em seu cadastro, um formulário é apresentado. Caso contrário, uma mensagem de agradecimento aparece.

O arquivo cadastro_dependente.php efetiva o cadastro dos dependentes. De forma similar ao cadastro.php, o *script* conecta-se ao banco de dados e insere cada dependente através da instrução `INSERT INTO dependente VALUES ($cpf_cli, '$dependente1','$dtnasc_dep1', '$rel_dep1').`

Quando o cliente realiza uma pesquisa customizada, os dados sobre o estado ou cidade, categoria do quarto, tipo de acomodação, preço e datas de

entrada e saída são enviados ao servidor. Com estes dados, no arquivo resultado.php, uma instrução SQL é montada dinamicamente para que a consulta ao banco de dados possa ser efetivada. A instrução SQL completa possui todas as condições para que a consulta seja customizada: `SELECT hot_cnpj, hot_nome, hot_rua, hot_bairro, hot_cidade, hot_estado, hot_CEP, hot_homepage, hot_email, qto_numero, qto_tipo, qto_nopessoas, qto_preco, qto_descricao FROM hotel, quarto WHERE hot_cnpj = qto_hot_cnpj AND hot_cidade = 'cidade_escolhida' AND qto_tipo = 'tipo_escolhido' AND qto_nopessoas = número_de_pessoas_escolhido AND qto_preco < preco_escolhido AND (qto_hot_cnpj, qto_numero) NOT IN (SELECT res_hot_cnpj, res_noquarto FROM reserva WHERE (('data_de_entrada' < res_entrada AND 'data_de_saída' > res_saida) OR (''data_de_entrada' >= res_entrada AND ' data_de_saída' <= res_saida) OR (''data_de_entrada' < res_entrada AND (' data_de_saída' > res_entrada AND ' data_de_saída' < res_saida)) OR (((''data_de_entrada' > res_entrada AND ''data_de_entrada' < res_saida) AND 'data_de_saída' > res_saida))).`

Depois que o cliente escolhe o hotel, ele deve confirmar a reserva preenchendo o CPF e a senha, anteriormente cadastrados. Isto é feito no arquivo confirma_hotel.php e os dados são enviados para o arquivo confirma_ reserva.php, onde a senha é validada e a reserva é efetivada através da instrução `INSERT INTO reserva VALUES($cpf, $cnpjhotel, $noquarto, '$data_ entrada', '$data_ saida').`

Os códigos completos podem ser encontrados no anexo C.

4.5 Implementação utilizando MySQL e ASP

A parte inicial desta etapa foi configurar o servidor *Web* (*Personal Web Server* da Microsoft) no Windows 98 para servir de plataforma de testes. O servidor já possui ASP. O MySQL para Windows também foi instalado, juntamente com o MySQL ODBC. Uma das principais características do MySQL é a sua conveniência com o ambiente multiusuário e multitarefa, o que o torna ideal

para a Internet. O MySQL é um SGBDR e para a criação das tabelas, as mesmas instruções SQL utilizadas para o PostgreSQL foram usadas.

Após a criação das tabelas, as páginas com *script* ASP foram desenvolvidas e seguem as mesmas estruturas dos arquivos em PHP. Algumas das diferenças mais relevantes são apresentadas a seguir:

1. As variáveis dos formulários não podem ser utilizadas diretamente como no *script* PHP, devem ser armazenadas em outra variável. Por exemplo, para recuperar o valor da variável senha, deve-se utilizar o comando `Request (“senha”)`.
2. Outras variáveis utilizadas durante o processamento, devem ser declaradas.
3. A conexão com o banco de dados é feita via ODBC (*Open Data Base Connectivity*). O ODBC é uma ferramenta criada pela Microsoft para servir de intercâmbio entre a base de dados e o *script.* O ODBC pode ser configurado no servidor ou criado em tempo real como acontece no ASP através do método `CreateObject`. Este método cria a conexão com a fonte de dados ODBC por meio de um componente denominado ADO (*ActiveX Data Objects*). Com o vínculo criado, deve-se abrir a fonte de dados ODBC, o que é feito com o método `open` aplicado ao objeto ADO criado. No site de reserva de hotéis, os seguintes comandos são utilizados:

```
Set Banco = Server.CreateObject("ADODB.connection")
Banco.open "hotel", "root", ""
```

Todas as operações de inclusão, alteração, exclusão e consulta são feitas por meio do método `execute` aplicado ao banco de dados, como no exemplo:

```
Set Tabela = Banco.Execute("<instrução SQL>")
```

4. O acesso aos resultados das consultas acontece de forma linear. Enquanto o PHP acessa de forma randômica, como em um vetor, o ASP movimenta o ponteiro linearmente com o comando `Movenext`.
5. Como o MySQL não aceita um comando SELECT dentro da condição de outra consulta, a verificação dos quartos com reserva foi implementada no *script*. Isto pode afetar o desempenho da arquitetura.

Os códigos completos podem ser encontrados no anexo D.

4.6 Modelo Orientado a Objetos (OO)

Uma diferença fundamental entre o modelo relacional e o orientado a objetos (OO) é a ausência de fundamentação teórica do modelo OO. Ao contrário do modelo relacional, não existe uma padronização para o modelo OO, e isso dificulta muito o estabelecimento de estratégias de execução de consultas, e a realização de comparações de desempenho entre os sistemas [GPL01].

O modelo orientado a objetos define um banco de dados em termos de objetos, suas propriedades e suas operações [EN94].

Um objeto é uma entidade real ou abstrata, que possui uma identidade e que exibe um comportamento bem definido. Um objeto tem um estado que inclui todas as suas propriedades (estruturas de dados e métodos que agem sobre estas estruturas) e seus valores (estado abstrato do objeto). A definição do objeto deve conter a estrutura de dados (atributos) mais os métodos (operações) relacionados com esta estrutura.

Em SGBDOO, um objeto é a unidade de informação que contém dado, é equivalente a registros nos bancos de dados tradicionais e comportamento de dados (métodos). Cada objeto possui : identidade, valor e métodos.

- Identidade do Objeto: em geral, a identidade do objeto é gerada pelo sistema. Essa identidade nada mais é do que um identificador implícito

único de cada objeto que é independente do valor desse objeto e de seu local físico de armazenamento.

- Valor do Objeto: os dados contidos em um objeto são chamados de valores dos objetos e a estrutura do valor é chamada tipo.
- Métodos: é através deste conceito que clientes solicitam serviços a um objeto por meio de uma mensagem. Um método, em relação a um objeto, corresponde ao comportamento dos objetos, implementando uma operação associada a uma ou mais classes, de forma similar aos códigos dos procedimentos usados em linguagens de programação tradicionais, que manipula o objeto ou parte deste.

Os objetos complexos são formados por construtores (conjuntos, listas, tuplas, registros, coleções, arrays) aplicados a objetos simples (inteiros, booleanos, strings). Em SGBDOO, também pode-se utilizar estes tipos de dados estruturados, assim sendo, a consulta ao banco de dados precisa ser mais complexa, pois ao invés de acesso a tabelas e registros, é necessário o acesso a listas, tuplas, arrays, entre outros.

Um conjunto de objetos que compartilha a mesma estrutura e o mesmo comportamento pode ser agrupado para formar uma classe. Um objeto é uma instância de uma classe, ou seja, suas características são definidas na classe a qual foi instanciado.

Assim como na orientação a objetos, nos SGBDOO também há a utilização de conceitos de herança e encapsulamento [Bar94]. Além disso, tornou-se necessária a aplicação do conceito de persistência [Kho94]:

- Herança: Espécie de relacionamento entre classes, onde, uma classe compartilha (herda) estrutura de dados (atributos) e métodos (operações) de outras classes. Por exemplo: Se a classe B herda da classe A, todas as características de A são utilizáveis em B; e mais, a classe B pode

acrescentar outras características que só dizem respeito a ela. Estas novas características não são utilizáveis na classe A;

- Encapsulamento: O conceito de encapsulamento utilizado em Orientação a Objetos também é aplicado aos SGBDOO's .Um objeto é definido como os atributos mais as operações sobre eles. Com esta característica, os atributos só podem ser acessados por estes métodos. Assim, o objetivo do encapsulamento é esconder detalhes da implementação dos métodos e da estrutura dos objetos. Outro ponto positivo do encapsulamento é a garantia de que nenhum dado seja acessado, ou alterado, por algum método desconhecido. Os usuários dos objetos têm conhecimento apenas de quais operações este objeto pode realizar, e não de como ele realiza a operação;
- Persistência: Um objeto é considerado persistente em uma base de dados quando ele continua existindo após o final da transação que o criou. Para garantir que um objeto necessário às futuras transações no banco de dados continuará existindo, é necessário torná-lo persistente. Para isso, deve-se associá-lo a um nome, sob o qual o sistema de banco de dados e suas aplicações o referenciarão. A partir dessa identificação, um objeto estará realmente armazenado. Com isso, todos os seus componentes também serão tornados persistentes.

Uma das maneiras de representar graficamente o modelo orientado a objetos é por meio do modelo relacional extendido, proposto por Delobel [DLR95]. O modelo possui a seguinte notação:

- Os atributos são representados por quadrados rotulados com o nome do atributo;
- Os objetos são representados por circunferências com um X no meio. Além disso, possuem *n* atributos conectados por setas;

- Um conjunto de atributos ou de objetos é representado por uma circunferência com um círculo no meio.

A figura 12 apresenta o modelo relacional extendido do site de reserva de hotéis:

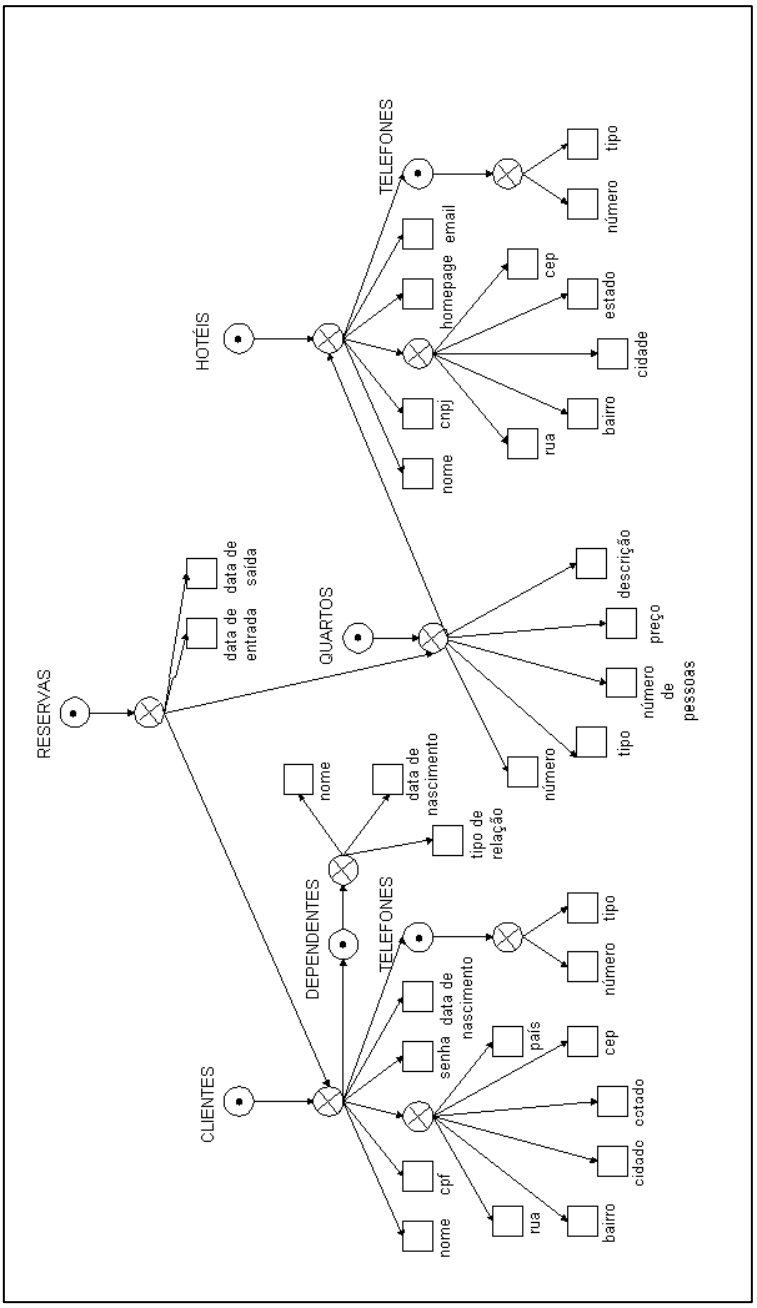

Figura 12 – Modelo relacional extendido do site de reserva de hotéis

4.7 Implementação utilizando Oracle8i

O Oracle8i é um dos SGBDORs mais populares no mercado. Como mencionado anteriormente, a principal característica de um SGBDOR é permitir que o usuário crie seus próprios tipos de dados.

Com o comando CREATE TYPE, pode-se criar novos tipos de dados e fazer referência a estes dentro de outros objetos. Um exemplo da utilização do comando pode ser observado na criação do objeto endereço:

```
CREATE TYPE "ENDERECO_CLI" AS OBJECT (
"CL_RUA" VARCHAR2(50),
"CL_BAIRRO" VARCHAR2(25),
"CL_CIDADE" VARCHAR2(20),
"CL_ESTADO" CHAR(2),
"CL_CEP" CHAR(9),
"CL_PAIS" VARCHAR2(20))
```

Uma tabela criada a partir de um OBJECT TYPE é denominada uma tabela de objetos (OBJECT TABLE) e cada linha armazenada nesta tabela é denominada uma linha objeto (OBJECT ROW).

Para definir um relacionamento entre objetos, utiliza-se referência. O objeto referenciado deve ser uma linha objeto (OBJECT ROW). Uma coluna do tipo REF armazena a referência ou um ponteiro para uma linha objeto.

Um novo tipo reconhecido no Oracle8i é VARRAY. Um vetor é uma lista ordenada de elementos com tamanho máximo definido e no Oracle8i são chamados de VARRAY porque o vetor é de tamanho variável. Um exemplo de VARRAY é o telefone:

```
CREATE TYPE "TELS" AS VARRAY (5) OF TELEFONE
```

Estes foram os comandos utilizados na criação dos objetos do site de reserva de hotéis. A listagem completa dos comandos se encontra no anexo E.

Para construir sites dinâmicos, a Oracle desenvolveu uma solução denominada WebDB. O WebDB armazena todos os dados necessários para as páginas da *Web* no banco de dados. Assim, quando o conteúdo de uma página da

Web mudar, a página, quando vista através do navegador, refletirá o novo conteúdo [ACA00].

Não foi possível terminar a implementação em Oracle8i devido a problemas na configuração do WebDB.

5 RESULTADOS

Este capítulo apresenta os resultados da análise comparativa dos sistemas desenvolvidos em PHP4, ASP e Oracle8i. Para cada característica analisada, os sistemas receberam uma das quatro classificações: "baixo", "médio", "alto" e "não avaliado". A primeira significa que a funcionalidade é fraca, difícil ou não é robusta para a arquitetura em estudo. A segunda significa que a funcionalidade é boa e apresenta melhores características do que as avaliadas com "baixo". A terceira significa que a funcionalidade é muito boa e, comparativamente, apresenta as melhores características. A última significa que não foi possível avaliar a funcionalidade. A tabela 2 sintetiza a análise realizada.

Tabela 2 – Quadro comparativo entre as aplicações *Web* analisadas

Aspecto Analisado: **Gerenciamento do estado da aplicação**

Arquitetura	Classificação
PostgreSQL e PHP4	baixo
MySQL e ASP	baixo
Oracle8i	não avaliado

Comentários: Em ambas aplicações a funcionalidade é muito deficiente. Como a conexão com o banco de dados é fechada quando o arquivo php ou asp é executado pelo servidor, não é possível que o SGBD mantenha o estado da aplicação.

Aspecto Analisado: **Segurança**

Arquitetura	Classificação
PostgreSQL e PHP4	baixo
MySQL e ASP	médio
Oracle8i	não avaliado

Comentários: A implementação de segurança nestas arquiteturas é muito limitada. A segurança fica restrita aos mecanismos tradicionais dos SGBDs. Assim, foi avaliada a autenticação do usuário. Pode-se conectar ao PostgreSQL apenas com os dados do host, porta e nome do banco de dados. Já o ASP necessita do nome do banco de dados, login do usuário e senha.

Aspecto Analisado: **Desempenho**

Arquitetura	Classificação
PostgreSQL e PHP4	alto
MySQL e ASP	baixo
Oracle8i	não avaliado

Comentários: Para avaliar o desempenho, uma consulta base foi utilizada (consulta com todos os hotéis possíveis). A implementação em PHP4 foi desenvolvida em um computador Intel Celeron 300 MHz com 64 M de RAM e o tempo de resposta foi de aproximadamente 23 segundos. A implementação em ASP foi desenvolvida em um computador Intel Pentium III 500 MHz com 128 M de RAM e o tempo de resposta foi de aproximadamente 43 segundos. Como os testes foram realizados em computadores diferentes, não é possível determinar corretamente qual a melhor arquitetura, mas aparentemente o PHP4 apresenta melhores resultados. Um dos motivos do baixo desempenho do ASP pode ser a inclusão de algumas estruturas de repetição durante a implementação (para verificar quais quartos estavam reservados). O ASP também gera um processamento a mais durante a criação dos objetos para comunicação com o banco de dados.

Aspecto Analisado: **Implementação**

Arquitetura	Classificação
PostgreSQL e PHP4	alto
MySQL e ASP	médio
Oracle8i	médio

Comentários: Para os usuários familiarizados com as linguagens C e Java, utilizar o PHP4 é simples. Além disso, o PHP4 conecta-se diretamente com o banco de dados, cria variáveis automaticamente quando formulários HTML são utilizados e não exige declaração de tipos dos dados. Já o ASP é baseado no Visual Basic da Microsoft, exigindo declaração das variáveis e dos tipos destas variáveis. Para acessar o banco de dados precisa-se criar um objeto denominado ADO e a conexão é feita via ODBC. O Oracle8i facilita a criação de objetos e de sites, uma vez que possui interface gráfica. Porém, demanda maiores conhecimentos durante a instalação do software.

Aspecto Analisado: **Portabilidade**

Arquitetura	Classificação
PostgreSQL e PHP4	alto
MySQL e ASP	médio
Oracle8i	alto

Comentários: Tanto o PostgreSQL quanto o PHP4 podem ser utilizados em várias plataformas incluindo Windows, Linux, Solaris e AIX. O ASP trabalha melhor em plataforma Windows, apesar de existir tentativas para utilizá-lo em Linux. O Oracle pode ser instalado tanto em Windows NT, como Unix e Solaris.

Aspecto Analisado: **Custo**

Arquitetura	Classificação
PostgreSQL e PHP4	alto
MySQL e ASP	baixo
Oracle8i	baixo

Comentários: Uma aplicação desenvolvida em PostgreSQL e PHP4 possui custo zero, pois os softwares podem ser adquiridos gratuitamente através de *download*. Para utilizar o ASP é necessário possuir Microsoft Windows 2000 (R$559,00) ou Windows NT (R$1.781,00) instalado. Há uma versão monousuário gratuita para download do Oracle8i, porém a versão para mais usuários deve ser adquirida ao preço de R$1.576,00 (versão para cinco usuários).

Fonte: Revista Info Exame, março 2001

6 CONCLUSÃO

A integração de *scripts Web* e SGBDs visando o desenvolvimento de aplicações que funcionem na Internet é uma área de pesquisa que ainda está em pleno desenvolvimento. A diversidade de arquiteturas disponíveis para integrar as tecnologias tem dificultado a construção de sistemas *Web* que satisfaçam as exigências dos usuários.

Uma contribuição deste trabalho foi a apresentação do funcionamento de três arquiteturas que possibilitam o acesso à bancos de dados via *Web*. Estas arquiteturas foram analisadas segundo alguns requisitos funcionais, o que possibilitou a definição das vantagens e desvantagens da utilização de tais arquiteturas. O quadro comparativo permitiu avaliá-las de modo a facilitar a definição da melhor arquitetura, ou da mais adaptada, para atender aos interesses do usuário.

De um modo geral, a implementação com PostgreSQL e PHP4 aparenta ser mais robusta, apresentando bons resultados na maioria das funcionalidades analisadas.

Considerando que a área de pesquisa é relativamente nova, muitas questões ainda devem ser melhor estudadas para que este ambiente se torne confiável e possa ser utilizado mais intensamente. Isto é verificado principalmente em questões como segurança e desempenho das aplicações.

7 TRABALHOS FUTUROS

As atuais implentações estão em suas primeiras versões, podendo ser melhoradas em vários pontos, como a reestruturação das instruções SQL, que podem melhorar o desempenho.

A implementação em Oracle8i deve ser finalizada, para que todas as características possam ser analisadas.

Além disso, outros *scripts* (como o JSP e ColdFusion), formatos de documentos (como o XML e XHTML) e SGBDs (como o Informix e Sybase) podem ser incorporados à análise comparativa, possibilitando ao usuário maior flexibilidade na escolha das ferramentas.

8 REFERÊNCIA BIBLIOGRÁFICA

[ACA00] ABBEY, M.; COREY, M.J.; ABRAMSON, I. **Oracle8i – Guia Introdutório**. Rio de Janeiro: Ed. Campus, 2000. 701p.

[Ash99] ASHENFELTER, J.P. **Web Database Tools for Web Development**. 1999. URL: http://www.webdatabase.org

[Bar94] BARBIERI, C. **Modelagem de Dados**. Rio de Janeiro: Ed. Infobook, 1994. 270p.

[Bar00a] BARRETO, A. **ASP – Active Server Pages**. 2000. URL: http://www.lemon.com.br/canais/tutoriais/bd.cfm?id=5&Sub=2

[Bar00b] BARRETO, M.V. de S. **Aplicações Web com PHP**. Aracaju, 2000. URL: http://www.tutoriais.com.br/php

[Dat90] DATE, C.J. **Introdução a Sistemas de Bancos de Dados**. Rio de Janeiro: Ed. Campus, 1990. 674p.

[DLR95] DELOBEL, C.; LÉCLUSE, C.; RICHARD, P. **Databases: From Relational to Object-Oriented Systems**. London: Ed. Thomson Computer Press, 1995. 382p.

[EN94] ELMASRI, R.; NAVATHE, S.B. **Fundamentals of Database Systems**. Redwood City: Ed. A. Wesley, 1994, 873p.

[GPL01] GONÇALVES, G.I.; PESENTE, G.R.; LIMA, R. **Banco de Dados Orientado a Objetos.** Pontifícia Universidade Católica do Rio Grande do Sul/Faculdade de Informática. Acessado em Abril de 2001.

[GZS01] GUERRA, D.C.; ZAMBAN, L.B.; SANTOS, M.S. dos - **Bancos de Dados Objeto-Relacionais**. Pontifícia Universidade Católica do Rio Grande do Sul/Faculdade de Informática. Acessado em Abril de 2001.

[Kho94] KHOSHAFIAN, S. **Banco de Dados Orientado a Objeto.** Rio de Janeiro: Ed. Infobook, 1994. 354p.

[LL98] LIFSCHITZ, S.; LIMA, I.N. de. **Arquiteturas de Integração Web SGBD: um Estudo do Ponto de Vista de Sistemas de Banco de Dados**. Belo Horizonte: Anais do XVIII Congresso da Sociedade Brasileira de Computação, 1998. v. 1. p. 297-318.

[Mar00] MARZANO, R. **ASP for Lynux by Chili?Soft.** 2000. URL: http://aspbrasil.zip.net/ver_artigos.asp?id=10

[Net00] NETO, E.W. **Administração de Sistemas Gerenciadores de Banco de Dados Objeto Relacional em Ambiente Distribuído**, 2000. URL: http://www.mestradoinfo.ucb.br/Aluno/eneto/sbd/sgbdor.htm

[Sá00] SÁ, P.S.S. **Gerador Automático de Arquivos HTML de Ajuda para Aplicação em Educação a Distância (GAAHA)**. São Carlos: USP, 2000. 103p. (Dissertação - Mestrado em Ciências da Computação e Matemática Computacional). URL: http://java.icmc.sc.usp.br/research/master/Paulo_Sa/tese.pdf

[Sav96] SAVOLA, T. **Usando HTML: O Guia de Referência Mais Completo**. Rio de Janeiro: Ed. Campus, 1996. 682p.

[Sil01] SILVA, L.C. da. **Banco de Dados para Web: Do planejamento à Implementação**. São Paulo: Ed. Érica, 2001. 240p.

[Suh00] SUHETT, M. R. **Introdução a XML**. 2000. URL: http://www.mrshp.hpg.com.br/prog/aposprog.htm

[Tan97] TANENBAUM, A.S. **Redes de Computadores**. Rio de Janeiro: Ed. Campus, 1997. 876p.

[VD95] VALENTE, A. de S. M.; DAMSKI, J.C. **Internet: Guia do Usuário Brasileiro**. São Paulo: Ed. Makron Books, 1995. 167p.

[Wya95] WYATT, A.L. **Sucesso com Internet**. São Paulo: Ed. Érica, 1995. 401p.

ANEXO A

1 Folha de estilos do documento XML da figura 8

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/TR/WD-xsl"
<xmlns:HTML="http://www.w3.org/Profiles/XHTML-transitional">
<xsl:template><xsl:apply-templates/></xsl:template>
<xsl:template          match="text()"><xsl:value-of/><xsl:for-each/>
</xsl:template>
<xsl:template match="/">
<!--.descricao{ text-align: justify; font-size: 14pt; background-
color: red; color=white; } -->
<HTML><HEAD><TITLE>Supermercado PEG-LEVE - O seu centro de
compras</TITLE>
<STYLE>
BODY{ margin:0px; background:ffffff; width: 30em; font-
family:arial, Helvetica, sans-serif; font-size:small; }
.nomedosupermercado{ text-align: left; font-family:verdana; font-
size: margin-top: .25em; font-weight: bold; color: darkblue;
background-color: lightblue;}
.telefone{ font-style: italic; font-size: smaller; text-align:
left}
.endereco{ text-align: left; font-size:8pt; font-family:arial; }
.nomedoproduto {font-size: 10pt; text-align: right; }
.hist2{ font-weight: bold; font-size: 8pt; background-color: red;
color=white; text-align: left; }
.preco{ font-size: 10pt; font-weight: bold; color=darkblue; text-
align: left}
.hist1{ font-size: 10pt; background-color: lightblue;
color=darkred; text-align: center; text-decoration: blink;}
</STYLE></HEAD>
<BODY><TABLE WIDTH = "770" CELLSPACING = "5" CELLPADDING = "0"
BORDER = "0">
<TR><TD WIDTH = "770">
<P CLASS = "nomedosupermercado"><xsl:value-of select=
"supermercado/nomedosupermercado"/></P>
<P CLASS = "endereco"><xsl:value-of select="supermercado/
endereco"/></P>
<P CLASS = "hist2">Telefone</P>
<P CLASS = "telefone"><xsl:value-of select = "supermercado/
telefone"/></P>
</TD>
```

```
</TR>
<TR>
<TD WIDTH = "770">
<P CLASS = "hist1">Produtos</P>
</TD>
</TR>
</TABLE>
<TABLE>
<xsl:for-each select="supermercado/lista/produtos">
<TR>
<TD WIDTH = "350">
<P  CLASS  =  "nomedoproduto"><xsl:value-of  select="nomedoproduto"/>
</P>
</TD>
<TD WIDTH = "350">
<P CLASS = "preco"><xsl:value-of select="preco"/></P>
</TD>
</TR>
</xsl:for-each>
</TABLE><P/></BODY></HTML>
</xsl:template>
</xsl:stylesheet>
```

ANEXO B

1 Arquivo index.html

```
<html>
<head>
<title>Reserva On Line</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps" size="7">RESERVA ON LINE</font></p>
<table border="0" width="100%">
<tr>
<td width="27%"><a href="hoteis.htm"><img border="0" src="hoteis.jpg" width="124" height="128"></a><br>
<a href="clientes.htm">
<img border="0" src="cliente.jpg" width="123" height="58"></a><br>
<img border="0" src="contato.jpg" width="123" height="192"></td>
<td width="73%">
<p align="center"><img border="0" src="paisagem.jpg" width="337" height="376" align="left"></p>
<p> </td>
</tr>
<tr>
<td width="27%"></td>
<td width="73%">
</td>
</tr>
</table>
<p align="center" style="text-indent: 0; margin-left: 0; margin-right: 0"> </p>
<p align="center"> </p>
</body>
</html>
```

2 Arquivo clientes.html

```
<html>
<head>
<title>Cadastro de Clientes</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps" size="7">Cadastro de clientes</font></p>
<p align="center"><font face="Verdana">Os campos assinalados com <font color="#FF0000">*</font> não podem ser nulos</font></p>
<form method="POST" action="cadastro.php4">
<table border="0" width="88%" height="132" cellspacing="3">
<tr>
<td width="55%" height="24"><font face="Verdana"><font size="4"> Nome <font color="#FF0000"> *</font>: </font></font></td>
<td width="56%" height="24"><input type="text" name="nome" size="35"></td>
</tr> <tr>
<td width="55%" height="25"><font face="Verdana"><font size="4">Rua</font></font> <font face="Verdana" size="4">: </font> </td>
<td width="56%" height="25"><input type="text" name="rua" size="35"></td>
</tr> <tr>
<td width="55%" height="21"><font face="Verdana" size="4">Bairro:</font></td>
<td width="56%" height="21"><input type="text" name="bairro" size="25"></td>
</tr> <tr>
<td width="55%" height="21"><font face="Verdana" size="4">Cidade:</font></td>
<td width="56%" height="21"><input type="text" name="cidade" size="25"></td>
</tr> <tr>
<td width="55%" height="21"><font face="Verdana" size="4">Estado: </font></td>
<td width="56%" height="21"><font face="Arial Black" size="3"><select name="estado" size="1">
            <option>Acre</option>
            <option>Amazonas</option>
            <option>Amapá</option>
            <option>Alagoas</option>
```

```
            <option>Bahia</option>
            <option>Ceará</option>
            <option>Distrito Federal</option>
            <option>Espírito Santo</option>
            <option>Goiás</option>
            <option>Maranhão</option>
            <option>Mato Grosso</option>
            <option>Mato Grosso do Sul</option>
            <option selected>Minas Gerais</option>
            <option>Pará</option>
            <option>Paraíba</option>
            <option>Paraná</option>
            <option>Pernambuco</option>
            <option>Piauí</option>
            <option>Rio de Janeiro</option>
            <option>Rio Grande do Norte</option>
            <option>Rio Grande do Sul</option>
            <option>Rondônia</option>
            <option>Roraima</option>
            <option>Santa Catarina</option>
            <option>São Paulo</option>
            <option>Sergipe</option>
            <option>Tocantins</option>
          </select></font></td>
</tr> <tr>
<td width="55%" height="21"><font face="Verdana" size="4">CEP:
</font></td>
<td width="56%" height="21"><input type="text" name="cep"
size="9"></td>
</tr> <tr>
<td width="55%" height="21"><font face="Verdana" size="4">
País:</font></td>
<td width="56%" height="21"><input type="text" name="pais"
size="25"></td>
</tr> <tr>
<td width="55%" height="21"><font face="Verdana" size="4">
Telefones:</font></td>
<td width="56%" height="21"><input type="text"
name="telefone1" size="15"><font face="Arial Black" size="3">
<select name="tipo_tel1" size="1">
            <option selected>residencial</option>
            <option>comercial</option>
```

```
            <option>fax</option>
            <option>celular</option>
</select></font></td>
</tr> <tr>
<td width="55%" height="21"></td>
<td       width="56%"       height="21"><input       type="text"
name="telefone2"      size="15"><font      face="Arial      Black"
size="3"><select name="tipo_tel2" size="1">
            <option selected>residencial</option>
            <option>comercial</option>
            <option>fax</option>
            <option>celular</option>
</select></font></td>
</tr> <tr>
<td width="55%" height="21"></td>
<td       width="56%"       height="21"><input       type="text"
name="telefone3"      size="15"><font      face="Arial      Black"
size="3"><select name="tipo_tel3" size="1">
            <option selected>residencial</option>
            <option>comercial</option>
            <option>fax</option>
            <option>celular</option>
</select></font></td>
</tr> <tr>
<td  width="55%"  height="21"><font  face="Verdana"  size="4">Data
de nascimento:</font></td>
<td    width="56%"    height="21"><font    face="Arial    Black"
size="3"><select name="dia_entrada" size="1">
            <option selected>1</option>
            <option>2</option>
            <option>3</option>
              AQUI A SEQUÊNCIA CONTINUA...
            <option>31</option>
</select><select name="mes_entrada" size="1">
            <option selected>1</option>
            <option>2</option>
              AQUI A SEQUÊNCIA CONTINUA...
            <option>12</option>
</select><select name="ano_entrada" size="1">
            <option>1930</option>
              AQUI A SEQUÊNCIA CONTINUA...
            <option>1990</option>
```

```
         </select></font></td>
</tr> <tr>
<td width="55%" height="21"> <font face="Verdana" size="4">CPF
<font color="#FF0000">*</font>:</font></td>
<td   width="56%"   height="21"><input   type="text"   name="cpf"
size="12"></td>
</tr> <tr>
<td    width="55%"    height="21">        <font    face="Verdana"
size="4">Senha <font color="#FF0000">*</font>:</font></td>
<td      width="56%"      height="21"><input      type="password"
name="senha" size="12"></td>
</tr> <tr>
<td       width="55%"       height="21"><font       face="Verdana"
size="4">Confirmar   senha   <font   color="#FF0000">*   </font>:
</font> </td>
<td   width="56%"   height="21"><input   type="password"   name=
"senha2" size="12"> </td>
</tr>
<tr>
<td     width="55%"     height="21">     <font     face="Verdana"
size="4">Dependentes:</font></td>
<td><input    type="radio"    value="V1"    name="inclusao"><font
face="Verdana" size="4">      incluir </font>
<input  type="radio"  value="V2"  checked  name="inclusao"><font
face="Verdana" size="4"> nao incluir</font></td>
</tr>
</table>
<p     align="center"><b><font     face="Arial     Black"><input
type="submit" value="Cadastrar" name="B1"></font></b></p>
</form>
</body>
</html>
```

3 Arquivo hoteis.html

```
<html>
<head>
<title>Reserva On Line</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p  align="center"><font  color="#000080"  face="Comix  Regular  Caps"
size="7">PESQUISA DE HOTÉIS</font></p>
<form method="POST" action="resultado.php4">
```

```
<table border="0" width="88%" height="132" cellspacing="3">
<tr>
<td  width="59%"  height="24"><font  face="Verdana"><font  size="4">
Localidade</font><input     type="radio"     value="V1"     checked
name="R1"><font face="Verdana" size="4">Estado-</font></font></td>
<td     width="52%"     height="24"><font     face="Arial     Black"
size="3"><select name="estado" size="1">
        <option selected>Indiferente</option>
        <option>Acre</option>
        <option>Amazonas</option>
        <option>Amapá</option>
        <option>Alagoas</option>
        <option>Bahia</option>
        <option>Ceará</option>
        <option>Distrito Federal</option>
        <option>Espírito Santo</option>
        <option>Goiás</option>
        <option>Maranhão</option>
        <option>Mato Grosso</option>
        <option>Mato Grosso do Sul</option>
        <option>Minas Gerais</option>
        <option>Pará</option>
        <option>Paraíba</option>
        <option>Paraná</option>
        <option>Pernambuco</option>
        <option>Piauí</option>
        <option>Rio de Janeiro</option>
        <option>Rio Grande do Norte</option>
        <option>Rio Grande do Sul</option>
        <option>Rondônia</option>
        <option>Roraima</option>
        <option>Santa Catarina</option>
        <option>São Paulo</option>
        <option>Sergipe</option>
        <option>Tocantins</option>
     </select></font></td>
  </tr>
  <tr>
<td     width="59%"     height="25"><font     face="Verdana"><font
size="4"></font><input   type="radio"   value="V2"   name="R1"><font
face="Verdana" size="4"> Cidade -</font></font></td>
```

```
<td  width="52%"  height="25"><font  face="Arial  Black"  size="3">
<select name="cidade" size="1">
        <option selected>Indiferente</option>
        <option>AC - Rio Branco</option>
        <option>AM - Manaus</option>
        <option>AP - Macapá</option>
        <option>AL - Maceió</option>
        <option>BA - Ilhéus</option>
        <option>BA - Porto Seguro</option>
        <option>BA - Salvador</option>
        <option>CE - Fortaleza</option>
        <option>DF - Brasília</option>
        <option>ES - Vitória</option>
        <option>GO - Goiânia</option>
        <option>MA - São Luís</option>
        <option>MT - Cuiabá</option>
        <option>MS - Campo Grande</option>
        <option>MG - Belo Horizonte</option>
        <option>MG - Diamantina</option>
        <option>MG - Lavras</option>
        <option>MG - Ouro Preto</option>
        <option>PA - Belém</option>
        <option>PB - João Pessoa</option>
        <option>PR - Curitiba</option>
        <option>PR - Foz do Iguaçu</option>
        <option>PE - Recife</option>
        <option>PI - Teresina</option>
        <option>RJ - Rio de Janeiro</option>
        <option>RN - Natal</option>
        <option>RS - Porto Alegre</option>
        <option>RO - Porto Velho</option>
        <option>RR - Boa Vista</option>
        <option>SC - Blumenau</option>
        <option>SC - Florianópolis</option>
        <option>SC - Joinville</option>
        <option>SC - Londrina</option>
        <option>SP - Araraquara</option>
        <option>SP - Campinas</option>
        <option>SP - Jundiaí</option>
        <option>SP - Presidente Prudente</option>
        <option>SP - São Carlos</option>
```

```
        <option>SP - São José dos Campos</option>
        <option>SP - São José do Rio Preto</option>
        <option>SP - São Paulo</option>
        <option>SP - Sorocaba</option>
        <option>SE - Aracaju</option>
        <option>TO - Palmas</option>
      </select></font></td>
</tr> <tr>
<td   width="59%"   height="21"><font   face="Verdana"   size="4">
Categoria do quarto: </font></td>
<td     width="52%"     height="21"><font     face="Arial     Black"
size="3"><select name="tipo" size="1">
        <option selected>Indiferente</option>
        <option>standard</option>
        <option>luxo</option>
        <option>suíte</option>
      </select></font></td>
</tr> <tr>
<td width="59%" height="21"><font face="Verdana" size="4">Tipo de
acomodação: </font></td>
<td  width="52%"  height="21"><font  face="Arial  Black"  size="3">
<select name="quantidade" size="1">
        <option selected>Indiferente</option>
        <option>Solteiro</option>
        <option>Duplo</option>
        <option>Triplo</option>
        <option>Quádruplo</option>
      </select></font></td>
</tr> <tr>
<td width="59%" height="21"><font face="Verdana" size="4">Preço do
quarto: </font></td>
<td  width="52%"  height="21"><font  face="Arial  Black"  size="3">
<select name="preco" size="1">
        <option>Indiferente</option>
        <option>até R$ 30,00</option>
        <option>de R$ 30,00 até R$ 50,00</option>
        <option>de R$ 50,00 até R$ 70,00</option>
        <option>de R$ 70,00 até R$ 100,00</option>
        <option>acima de R$ 100,00</option>
      </select></font></td>
  </tr> <tr>
```

```
<td width="59%" height="21"><font face="Verdana" size="4"> Data de
entrada </font><font face="Verdana" size="4">: </font></td>
<td width="52%" height="21"><font face="Arial Black" size="3">
<select name="dia_entrada" size="1">
        <option selected>1</option>
        <option>2</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
</select><select name="mes_entrada" size="1">
        <option selected>1</option>
        <option>2</option>
      AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
</select><select name="ano_entrada" size="1">
        <option selected>2001</option>
        <option>2002</option>
</select></font></td>
</tr> <tr>
<td width="59%" height="21"> <font face="Verdana" size="4"> Data
de saída:</font></td>
<td width="52%" height="21"><font face="Arial Black" size="3">
<select name="dia_saida" size="1">
        <option selected>1</option>
        <option>2</option>
      AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
</select><select name="mes_saida" size="1">
        <option selected>1</option>
        <option>2</option>
      AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
</select><select name="ano_saida" size="1">
        <option selected>2001</option>
        <option>2002</option>
</select></font></td>
</tr>
</table>
<p align="center"><b><font face="Arial Black"><input type="submit"
value="Pesquisar" name="B1"></a></font></b></p>
</form>
</body>
</html>
```

ANEXO C

1 Arquivo cadastro.php4

```
<?
//O campo nome nao foi preenchido
if ($nome == ''){
?>
       <html>
       <head><title>Cadastro de Clientes - Erro no
cadastro</title></head>
       <body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
       <p align="center"><font color="#000080" face="Comix Regular
Caps" size="7">O nome nao foi preenchido. Retorne e preencha o
campo.</font></p>
       </body>
       </html>
<?
}
//cpf nao foi preenchido
else if ($cpf == ''){
?>
       <html>
       <head><title>Cadastro de Clientes - Erro no
cadastro</title></head>
       <body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
       <p align="center"><font color="#000080" face="Comix Regular
Caps" size="7">O CPF nao foi preenchido. Retorne e preencha o
campo.</font></p>
       </body>
       </html>
<?
}
//senha nao foi preenchida ou foi confirmada incorretamente
else if (($senha == '') or ($senha != $senha2)){
?>
       <html>
       <head><title>Cadastro de Clientes - Erro no
cadastro</title></head>
       <body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
```

```
	<p align="center"><font color="#000080" face="Comix Regular
Caps" size="7">A senha nao foi preenchida ou foi confirmada
incorretamente. Retorne preencha o campo.</font></p>
	</body>
	</html>
<?
}
else {
//inclusão do cliente
	$dtnascimento = "$ano_entrada";
	$dtnascimento .= "-$mes_entrada";
	$dtnascimento .= "-$dia_entrada";
	if ($estado == "Acre")
		$estado = 'AC';
	else if ($estado == "Amazonas")
		$estado = 'AM';
	else if ($estado == "Amapá")
		$estado = 'AP';
	else if ($estado == "Alagoas")
		$estado = 'AL';
	else if ($estado == "Bahia")
		$estado = 'BA';
	else if ($estado == "Ceará")
		$estado = 'CE';
	else if ($estado == "Distrito Federal")
		$estado = 'DF';
	else if ($estado == "Espírito Santo")
		$estado = 'ES';
	else if ($estado == "Goiás")
		$estado = 'GO';
	else if ($estado == "Maranhão")
		$estado = 'MA';
	else if ($estado == "Mato Grosso")
		$estado = 'MT';
	else if ($estado == "Mato Grosso do Sul")
		$estado = 'MS';
	else if ($estado == "Minas Gerais")
		$estado = 'MG';
	else if ($estado == "Pará")
		$estado = 'PA';
	else if ($estado == "Paraíba")
```

```
            $estado = 'PB';
      else if ($estado == "Paraná")
            $estado = 'PR';
      else if ($estado == "Pernambuco")
            $estado = 'PE';
      else if ($estado == "Piauí")
            $estado = 'PI';
      else if ($estado == "Rio de Janeiro")
            $estado = 'RJ';
      else if ($estado == "Rio Grande do Norte")
            $estado = 'RN';
      else if ($estado == "Rio Grande do Sul")
            $estado = 'RS';
      else if ($estado == "Rondônia")
            $estado = 'RO';
      else if ($estado == "Roraima")
            $estado = 'RR';
      else if ($estado == "Santa Catarina")
            $estado = 'SC';
      else if ($estado == "São Paulo")
            $estado = 'SP';
      else if ($estado == "Sergipe")
            $estado = 'SE';
      else if ($estado == "Tocantins")
            $estado = 'TO';
      $insercao = "INSERT INTO cliente VALUES ('$nome','$rua',
'$bairro', '$cidade', '$estado', '$cep', '$pais', $cpf,
'$dtnascimento', '$senha'); ";
      $database = pg_Connect ("", "", "", "", "reservaonline");
      $resultado = pg_exec($database, $insercao);
      if($telefone1 != ''){
            $insercaotel1 = "INSERT INTO cliente_telefone VALUES
($cpf,'$telefone1', '$tipo_tel1'); ";
            $database = pg_Connect ("", "", "", "",
"reservaonline");
            $resultado = pg_exec($database, $insercaotel1);
      }
      if($telefone2 != ''){
            $insercaotel2 = "INSERT INTO cliente_telefone VALUES
($cpf,'$telefone2', '$tipo_tel2'); ";
            $database = pg_Connect ("", "", "", "",
"reservaonline");
```

```
            $resultado = pg_exec($database, $insercaotel2);
      }
      if($telefone3 != ''){
            $insercaotel3 = "INSERT INTO cliente_telefone VALUES
($cpf,'$telefone3', '$tipo_tel3'); ";
            $database = pg_Connect ("", "", "", "",
"reservaonline");
            $resultado = pg_exec($database, $insercaotel3);
      }
//Se o cliente deseja incluir dependentes, vai para o formulário
de dependentes
      if ($inclusao == V1){
?>
<html>
<head><title>Cadastro de Clientes - Erro no
cadastro</title></head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Cadastro
de Dependentes</font></p>
<form method="POST" action="cadastro_dependente.php4">
<?
//Variável oculta para enviar o cpf do cliente
      echo"<input type='hidden' name='cpf_cli' value='$cpf'>";
?>
 <table border="0" width="100%">
  <tr>
    <td width="47%" height="30">
      <p align="center"><font face="Verdana" size="4">Nome do
dependente</font></td>
    <td width="27%" height="30">
      <p align="center"><font face="Verdana" size="4">Data de
nascimento</font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Verdana" size="4">Tipo de
relação</font></td>
  </tr>
  <tr>
    <td width="47%" height="30">
      <p align="left"><input type="text" name="dependente1"
size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
```

```
          name="dia_dep1" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep1" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep1" size="1">
        <option>1930</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep1" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente2" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep2" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep2" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
```

```
      </select><select
          name="ano_dep2" size="1">
        <option>1930</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep2" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente3" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep3" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep3" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep3" size="1">
        <option>1930</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep3" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
```

```
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente4" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep4" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep4" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep4" size="1">
        <option>1930</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep4" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente5" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep5" size="1">
        <option selected>01</option>
        <option>02</option>
```

```
         AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep5" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep5" size="1">
        <option>1930</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep5" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente6" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep6" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep6" size="1">
        <option selected>01</option>
        <option>02</option>
       AQUI A SEQUÊNCIA CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep6" size="1">
        <option>1930</option>
```

```
      AQUI A SEQUÊNCIA CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep6" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
 </table>
 <p align="center"><b><font face="Arial Black"><input
type="submit" value="Cadastrar" name="B1"></font></b></p>
</form>
<p align="center"> </p>
</body>
</html>
<? }
       else if ($inclusao == V2){ //nao vai cadastrar dependentes
?>
       <html>
       <head><title>Cadastro de Clientes - Erro no
cadastro</title></head>
       <body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
       <p align="center"><font color="#000080" face="Comix Regular
Caps" size="7">Obrigado por se cadastrar
</font></p>
       </body>
       </html>
<?
} //fim da inclusao
} //fim do else dos erros
?>
```

2 Arquivo cadastro_dependente.php4

```
<html>
<head>
<title>Cadastro de Dependentes</title>
</head>
```

```
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<?php
//inclusão do primeiro dependente
	$database = pg_Connect ("", "", "", "", "reservaonline");
	if ($dependente1 != ''){
		$dtnasc_dep1 = "$ano_dep1";
		$dtnasc_dep1 .= "-$mes_dep1";
		$dtnasc_dep1 .= "-$dia_dep1";
		$insercao_dep1 = "INSERT INTO dependente VALUES (
$cpf_cli, '$dependente1','$dtnasc_dep1', '$rel_dep1'); ";
		$resultado = pg_exec($database, $insercao_dep1);
	}
	if ($dependente2 != ''){
		$dtnasc_dep2 = "$ano_dep2";
		$dtnasc_dep2 .= "-$mes_dep2";
		$dtnasc_dep2 .= "-$dia_dep2";
		$insercao_dep2 = "INSERT INTO dependente VALUES (
$cpf_cli, '$dependente2','$dtnasc_dep2', '$rel_dep2'); ";
		$resultado = pg_exec($database, $insercao_dep2);
	}
	if ($dependente3 != ''){
		$dtnasc_dep3 = "$ano_dep3";
		$dtnasc_dep3 .= "-$mes_dep3";
		$dtnasc_dep3 .= "-$dia_dep3";
		$insercao_dep3 = "INSERT INTO dependente VALUES (
$cpf_cli, '$dependente3','$dtnasc_dep3', '$rel_dep3'); ";
		$resultado = pg_exec($database, $insercao_dep3);
	}
	if ($dependente4 != ''){
		$dtnasc_dep4 = "$ano_dep4";
		$dtnasc_dep4 .= "-$mes_dep4";
		$dtnasc_dep4 .= "-$dia_dep4";
		$insercao_dep4 = "INSERT INTO dependente VALUES (
$cpf_cli, '$dependente4','$dtnasc_dep4', '$rel_dep4'); ";
		$resultado = pg_exec($database, $insercao_dep4);
	}
	if ($dependente5 != ''){
		$dtnasc_dep5 = "$ano_dep5";
		$dtnasc_dep5 .= "-$mes_dep5";
		$dtnasc_dep5 .= "-$dia_dep5";
```

```
		$insercao_dep5 = "INSERT INTO dependente VALUES (
$cpf_cli, '$dependente5','$dtnasc_dep5', '$rel_dep5'); ";
		$resultado = pg_exec($database, $insercao_dep5);
	}
	if ($dependente6 != ''){
		$dtnasc_dep6 = "$ano_dep6";
		$dtnasc_dep6 .= "-$mes_dep6";
		$dtnasc_dep6 .= "-$dia_dep6";
		$insercao_dep6 = "INSERT INTO dependente VALUES (
$cpf_cli, '$dependente6','$dtnasc_dep6', '$rel_dep6'); ";
		$resultado = pg_exec($database, $insercao_dep6);
	}
?>
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Obrigado por se cadastrar.</font></p>
</body>
</html>
```

3 Arquivo resultado.php4

```
<html>
<head>
<title>Resultado da Pesquisa de Hotéis</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Resultado da Pesquisa</font></p>
<p align="center"> </p>
<form method="POST" action="confirma_hotel.php4">
<?php
	$dataatual = getdate();
	$valor = $dataatual['year'];
	$data = "$valor-";
	$valor = $dataatual['mon'];
	$data .= "$valor-";
	$valor = $dataatual['mday'];
	$data .= "$valor";
	$data_entrada = "$ano_entrada-$mes_entrada-$dia_entrada";
	$data_saida = "$ano_saida-$mes_saida-$dia_saida";
```

```
//Verifica se as datas estao corretas, ou seja, se entrada e menor
do que saida e se sao maiores que a data atual
       if(($data_entrada >= $data) and ($data_saida >= $data) and
($data_entrada <= $data_saida)){
              $database = pg_Connect ("", "", "", "",
"reservaonline");
              $consulta = "SELECT hot_cnpj, hot_nome, hot_rua,
hot_bairro, hot_cidade, hot_estado, hot_CEP, hot_homepage,
hot_email, qto_numero, qto_tipo, qto_nopessoas, qto_preco,
qto_descricao FROM hotel, quarto WHERE hot_cnpj = qto_hot_cnpj ";
              //$consulta = "SELECT hot_nome FROM hotel ";
              if ($R1 == V1) {    //marcou o estado
                     if ($estado != "Indiferente"){
                            if ($estado == "Acre")
                                   $estado = 'AC';
                            else if ($estado == "Amazonas")
                                   $estado = 'AM';
                            else if ($estado == "Amapá")
                                   $estado = 'AP';
                            else if ($estado == "Alagoas")
                                   $estado = 'AL';
                            else if ($estado == "Bahia")
                                   $estado = 'BA';
                            else if ($estado == "Ceará")
                                   $estado = 'CE';
                            else if ($estado == "Distrito Federal")
                                   $estado = 'DF';
                            else if ($estado == "Espírito Santo")
                                   $estado = 'ES';
                            else if ($estado == "Goiás")
                                   $estado = 'GO';
                            else if ($estado == "Maranhão")
                                   $estado = 'MA';
                            else if ($estado == "Mato Grosso")
                                   $estado = 'MT';
                            else if ($estado == "Mato Grosso do
Sul")
                                   $estado = 'MS';
                            else if ($estado == "Minas Gerais")
                                   $estado = 'MG';
                            else if ($estado == "Pará")
                                   $estado = 'PA';
```

```
                        else if ($estado == "Paraíba")
                               $estado = 'PB';
                        else if ($estado == "Paraná")
                               $estado = 'PR';
                        else if ($estado == "Pernambuco")
                               $estado = 'PE';
                        else if ($estado == "Piauí")
                               $estado = 'PI';
                        else if ($estado == "Rio de Janeiro")
                               $estado = 'RJ';
                        else if ($estado == "Rio Grande do
Norte")
                               $estado = 'RN';
                        else if ($estado == "Rio Grande do
Sul")
                               $estado = 'RS';
                        else if ($estado == "Rondônia")
                               $estado = 'RO';
                        else if ($estado == "Roraima")
                               $estado = 'RR';
                        else if ($estado == "Santa Catarina")
                               $estado = 'SC';
                        else if ($estado == "São Paulo")
                               $estado = 'SP';
                        else if ($estado == "Sergipe")
                               $estado = 'SE';
                        else if ($estado == "Tocantins")
                               $estado = 'TO';
                        $consulta .= "AND hot_estado =
'$estado' ";
                }
//fim do marcou indiferente
            }
            else {   //marcou cidade
                   if ($cidade != "Indiferente"){
                          $cidade = substr ($cidade, 5);
                          $consulta .= "AND hot_cidade =
'$cidade' ";
                   }
            }
//fim do else da marcacao da cidade
```

```
// tipo do quarto
             if ($tipo != "Indiferente"){
                    $consulta .= " AND qto_tipo = '$tipo' ";
             }

//quantidade de pessoas
             if ($quantidade != "Indiferente"){
                    if ($quantidade == "Solteiro")
                           $qtd_pessoas = 1;
                    else if ($quantidade == "Duplo")
                           $qtd_pessoas = 2;
                    else if ($quantidade == "Triplo")
                           $qtd_pessoas = 3;
                    else if ($quantidade == "Quádruplo")
                           $qtd_pessoas = 4;
                    $consulta .= "AND qto_nopessoas = $qtd_pessoas
";

             }

//Preço
             if ($preco != "Indiferente"){
                    if ($preco == "até R$ 30,00")
                           $consulta .= "AND qto_preco < 30 ";
                    else if ($preco == "de R$ 30,00 até R$ 50,00")
                           $consulta .= "AND qto_preco >= 30 AND
qto_preco < 50 ";
                    else if ($preco == "de R$ 50,00 até R$ 70,00")
                           $consulta .= "AND qto_preco >= 50 AND
qto_preco < 70 ";
                    else if ($preco == "de R$ 70,00 até R$
100,00")
                           $consulta .= "AND qto_preco >= 70 AND
qto_preco < 100 ";
                    else if ($preco == "acima de R$ 100,00")
                           $consulta .= "AND qto_preco >= 100 ";
             }

//data
```

```
            $consulta .= "AND (qto_hot_cnpj, qto_numero) NOT IN
";
            $consulta .= "(SELECT res_hot_cnpj, res_noquarto
FROM reserva WHERE ( ";
            $consulta .= "('$data_entrada' < res_entrada AND
'$data_saida' > res_saida) OR ";
            $consulta .= "('$data_entrada' >= res_entrada AND
'$data_saida' <= res_saida) OR ";
            $consulta .= "('$data_entrada' < res_entrada AND
('$data_saida' > res_entrada AND '$data_saida' < res_saida)) OR ";

            $consulta .= "(('$data_entrada' > res_entrada AND
'$data_entrada' < res_saida) AND '$data_saida' > res_saida))) ";

            $consulta .= " ; ";
            $resultado = pg_exec($database, $consulta);
            $numero_linhas = pg_NumRows($resultado);
            if ($numero_linhas == 0){
?>
<p align="center"><font face="Comix Regular Caps" size="5">Nenhum
resultado foi encontrado.</font></p>
<?
            }//fim  do numero_linhas == 0
            $resposta = 0; //armazena a quantidade de selecoes
disponiveis para o usuario
            for ($i = 0; $i < $numero_linhas; $i++){
                  $linhas = pg_fetch_array($resultado,$i);
                  $hotelcorrente = $linhas['hot_cnpj'];
?>
      <table border="1" width="72%">
      <tr>
            <td width="100%">
                  <p>Nome do Hotel:  <? echo
$linhas['hot_nome']; ?>  </p>
                  <p>Endereço:  <? echo $linhas['hot_rua'] ," /
",  $linhas['hot_bairro']; ?>  </p>
                  <p>Cidade:  <? echo $linhas['hot_cidade']," -
",  $linhas['hot_estado']; ?>  </p>
                  <p>CEP:  <? echo $linhas['hot_cep']; ?>  </p>
                  <p>Homepage:  <? echo $linhas['hot_homepage'];
?>  </p>
                  <p>Email:  <? echo $linhas['hot_email']; ?>
</p>
```

```
<? //mostrar o telefone
             $consulta_tel = "SELECT hot_tel_numero, hot_tel_tipo
FROM hotel_telefone WHERE hot_tel_cnpj = $hotelcorrente ; ";
             $resultado_tel = pg_exec($database, $consulta_tel);
             $numero_linhas_tel = pg_NumRows($resultado_tel);
             for ($k = 0; $k < $numero_linhas_tel; $k++){
                    $linhas_tel =
pg_fetch_array($resultado_tel,$k);
?>
                    <p>Telefone <? echo
$linhas_tel['hot_tel_tipo']; ?>:  <? echo
$linhas_tel['hot_tel_numero']; ?> </p>
<?
             } //fim dos telefones.
?>
                    <table border="1" width="72%">
                    <tr>
                           <td>Selecao</td>
                           <td>Tipo do quarto</td>
                           <td>Numero de pessoas</td>
                           <td>Preco</td>
                           <td>Descricao</td>
                    </tr>
                    <tr>
                           <td><input type="radio" value="<? echo
$resposta ?>" name="selecao"></td>
                           <?
                                  $nome = "hot_cnpj";
                                  $nome .= "$resposta";
                                  echo "<input type='hidden'
name='$nome' value='$hotelcorrente'>";
                                  $nome = "numero_quarto";
                                  $nome .= "$resposta";
                                  $numeroquarto =
$linhas['qto_numero'];
                                  echo "<input type='hidden'
name='$nome' value='$numeroquarto'>";
                                  $resposta++;
                           ?>
                           <td> <? echo $linhas['qto_tipo']; $tipo
= $linhas['qto_tipo']; ?> </td>
                           <td> <? echo $linhas['qto_nopessoas'];
$nopessoas = $linhas['qto_nopessoas'];  ?> </td>
```

```
                        <td><? echo $linhas['qto_preco'];
$preco = $linhas['qto_preco']; ?></td>
                        <td><? echo $linhas['qto_descricao'];
?></td>
                    </tr>
<?
                    //Aqui vou colocar todos os tipos na tabela do
hotel
                        do{
                              if (($i + 1) < $numero_linhas) {
//para verificar que o proximo nao e vazio
                                    $linhas =
pg_fetch_array($resultado,$i+1);
                                    $hotel =
$linhas['hot_cnpj'];
                                    if ($hotel ==
$hotelcorrente){ //continua no mesmo hotel
                                          $flag = 1; //para
sair do loop do while
                                          $i++;
                                    if (($tipo !=
$linhas['qto_tipo']) or ($preco != $linhas['qto_preco'])or
($nopessoas != $linhas['qto_nopessoas'])){
?>
      <tr>
      <td><input type="radio" value="<? echo $resposta ?>"
name="selecao"></td>
<?

      $nome = "hot_cnpj";
      $nome .= "$resposta";
      echo "<input type='hidden' name='$nome'
value='$hotelcorrente'>";
      $nome = "numero_quarto";
      $nome .= "$resposta";
      $numeroquarto = $linhas['qto_numero'];
      echo "<input type='hidden' name='$nome'
value='$numeroquarto'>";
      $resposta++;
?>
<td> <? echo $linhas['qto_tipo']; $tipo = $linhas['qto_tipo']; ?>
</td>
<td> <? echo $linhas['qto_nopessoas']; $nopessoas =
$linhas['qto_nopessoas']; ?> </td>
```

```
<td><? echo $linhas['qto_preco']; $preco = $linhas['qto_preco'];
?></td>
<td><? echo $linhas['qto_descricao']; ?></td>
</tr>
<?
              }//fim do if dos quartos
       }//fim do if do hotel
       else $flag = 0;
       }//fim do if
       else $flag = 0;
                           }while ($flag == 1);
?>
              </table>
              </td>
       </tr>
       </table>
<?
       } //fim do loop
       echo "<input type='hidden' name='qtdrespostas'
value='$resposta'>"; //armazena a quantidade de quartos
dispon[iveis para o cliente escolher
       echo "<input type='hidden' name='data_entrada'
value='$data_entrada'>";
       echo "<input type='hidden' name='data_saida'
value='$data_saida'>";
?>
<p align="center"><input type="submit" value="Reservar"
name="B1"></p>
<?
       }//fim da validacao da data
       else {
?>
<p align="center"><font face="Comix Regular Caps"
size="5">Verifique as datas de entrada e de saida</font></p>
<?
       } //fim do else de validacao da data
?>
</form>
</body>
</html>
```

4 Arquivo confirma_hotel.php4

```
<html>
<head>
<title>Reserva do quarto</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Reserva do quarto</font></p>
<p align="center"> </p>
<form method="POST" action="confirma_reserva.php4">
<?php
       for ($i = 0; $i < $qtdrespostas; $i++){
              if($selecao == $i){
                     $cnpjhotel = "hot_cnpj$i";
                     $cnpjhotel = $$cnpjhotel;
                     $noquarto = "numero_quarto$i";
                     $noquarto = $$noquarto;
                     $consulta = "SELECT hot_nome, hot_cidade,
hot_estado, qto_tipo, qto_preco, qto_nopessoas, qto_descricao FROM
hotel, quarto WHERE hot_cnpj = $cnpjhotel AND hot_cnpj =
qto_hot_cnpj AND qto_numero = $noquarto; ";
                     $database = pg_Connect ("", "", "", "",
"reservaonline");
                     $resultado = pg_exec($database, $consulta);
                     $linha = pg_fetch_array($resultado,0);
?>
   <p     align="center"><font     face="Comix     Regular     Caps"
size="5">Voce selecionou um quarto <? echo $linha['qto_tipo'];?>
do hotel <? echo $linha['hot_nome'];?> da cidade de <? echo
$linha['hot_cidade'];?> (<? echo $linha['hot_estado'];?>) para <?
echo $linha['qto_nopessoas'];?> pessoas. O quarto possui <? echo
$linha['qto_descricao'];?> e o preco sera de R$ <? echo
$linha['qto_preco'];?> reais. Para confirmar a reserva do quarto,
voce devera preencher os campos abaixo.</font></p>
<p align="center"> </p>
<?
                     $i = $qtdrespostas; //para parar o loop
                     echo "<input type='hidden' name='cnpjhotel'
value='$cnpjhotel'>";
                     echo "<input type='hidden' name='noquarto'
value='$noquarto'>";
                     echo "<input type='hidden' name='data_entrada'
value='$data_entrada'>";
```

```
                    echo "<input type='hidden' name='data_saida'
value='$data_saida'>";
             }
      }
?>
<table border="0" width="88%" height="132" cellspacing="3">
      <tr>
             <td width="55%" height="24"><font
face="Verdana"><font size="7">CPF:</font>
             </td>
             <td width="55%" height="24"><input type="text"
name="cpf" size="12">
             </td>
      </tr>
      <tr>
             <td width="55%" height="24"><font
face="Verdana"><font size="7">Senha:</font>
             </td>
             <td width="55%" height="24"> <input type="password"
name="senha" size="12">
             </td>
      </tr>
</table>
<p align="center"><input type="submit" value="Confirmar reserva"
name="B1"></p>
</form>
</body>
</html>
```

5 Arquivo confirma_reserva.php4

```
<html>
<head>
<title>Reserva do quarto</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Reserva do quarto</font></p>
<p align="center"> </p>

<?php
      $database = pg_Connect ("", "", "", "", "reservaonline");
```

```
	$consulta = "SELECT cl_senha FROM cliente WHERE cl_cpf =
$cpf;";
	$resultado = pg_exec($database, $consulta);
	$linha = pg_fetch_array($resultado,0);
	if ($senha == $linha['cl_senha']){
		$insercao = "INSERT INTO reserva VALUES ($cpf,
$cnpjhotel, $noquarto, '$data_entrada', '$data_saida');";
		$resultado = pg_exec($database, $insercao);
	}

?>

<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Reserva realizada com sucesso.
</font></p>

</body>
</html>
```

ANEXO D

1 Arquivo cadatro.asp

```
<%
nome = Request("nome")
senha = Request("senha")
senha2 = Request("senha2")
cpf = Request("cpf")
if len(nome) = 0 then
%>
<html>
<head>
<title>Cadastro de Clientes - Erro no cadastro</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<center>
<font face="Verdana" size="4">
Nome não foi preenchido. Volte à página anterior e preencha o
campo.</font>
</center>
</body>
</html>
<%
Elseif len(cpf) = 0 then
%>
<html>
<head>
<title>Cadastro de Clientes - Erro no cadastro</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<center>
<font face="Verdana" size="4">
CPF não foi preenchido. Volte à página anterior e preencha o
campo.</font>
</center>
</body>
</html>
<%
```

```
Elseif senha <> senha2 or len(senha) = 0 then
%>
<html>
<head>
<title>Cadastro de Clientes - Erro no cadastro</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<center>
<font face="Verdana" size="4">
Senha não foi preenchida ou não confirmada corretamente.Volte à
página anterior e preencha o campo.</font>
</center>
</body>
</html>
<%
Else
    rua = Request("rua")
    bairro = Request("bairro")
    cidade = Request ("cidade")
    cep = Request ("cep")
    pais = Request ("pais")
    dia_nasc = Request ("dia_entrada")
    mes_nasc = Request ("mes_entrada")
    ano_nasc = Request ("ano_entrada")
    dtnascimento = ano_nasc & "-" & mes_nasc & "-" & dia_nasc
    estado = Request("estado")
    If estado = "Acre" Then
        estado = "AC"
    ElseIf estado = "Amazonas" Then
        estado = "AM"
    ElseIf estado = "Amapá" Then
        estado = "AP"
    ElseIf estado = "Alagoas" Then
        estado = "AL"
    ElseIf estado = "Bahia" Then
        estado = "BA"
    ElseIf estado = "Ceará" Then
        estado = "CE"
    ElseIf estado = "Distrito Federal" Then
        estado = "DF"
    ElseIf estado = "Espírito Santo" Then
```

```
    estado = "ES"
ElseIf estado = "Goiás" Then
    estado = "GO"
ElseIf estado = "Maranhão" Then
    estado = "MA"
ElseIf estado = "Mato Grosso" Then
    estado = "MT"
ElseIf estado = "Mato Grosso do Sul" Then
    estado = "MS"
ElseIf estado = "Minas Gerais" Then
    estado = "MG"
ElseIf estado = "Pará" Then
    estado = "PA"
ElseIf estado = "Paraíba" Then
    estado = "PB"
ElseIf estado = "Paraná" Then
    estado = "PR"
ElseIf estado = "Pernambuco" Then
    estado = "PE"
ElseIf estado = "Piauí" Then
    estado = "PI"
ElseIf estado = "Rio de Janeiro" Then
    estado = "RJ"
ElseIf estado = "Rio Grande do Norte" Then
    estado = "RN"
ElseIf estado = "Rio Grande do Sul" Then
    estado = "RS"
ElseIf estado = "Rondônia" Then
    estado = "RO"
ElseIf estado = "Roraima" Then
    estado = "RR"
ElseIf estado = "Santa Catarina" Then
    estado = "SC"
ElseIf estado = "São Paulo" Then
    estado = "SP"
ElseIf estado = "Sergipe" Then
    estado = "SE"
ElseIf estado = "Tocantins" Then
    estado = "TO"
End If
```

```
    Dim Banco, Tabela
    Set Banco = Server.CreateObject("ADODB.connection")
    Banco.open "hotel", "root", ""
    Set Tabela = Banco.Execute("INSERT INTO cliente VALUES ('" &
nome & "','" & rua & "','" & bairro  & "','" & cidade & "','" &
estado & "','" & cep & "','" & pais & "'," & cpf & ",'" &
dtnascimento & "','" & senha & "');")
    telefone1 = Request("telefone1")
    telefone2 = Request("telefone2")
    telefone3 = Request("telefone3")
    if len(telefone1) <> 0 then
            tipo_tel1 = Request("tipo_tel1")
            Set Tabela = Banco.Execute("INSERT INTO
cliente_telefone VALUES (" & cpf & ",'" & telefone1 & "','" &
tipo_tel1  & "');")
    End if
    if len(telefone2) <> 0 then
            tipo_tel2 = Request("tipo_tel2")
            Set Tabela = Banco.Execute("INSERT INTO
cliente_telefone VALUES (" & cpf & ",'" & telefone2 & "','" &
tipo_tel2  & "');")
    End if
    if len(telefone3) <> 0 then
            tipo_tel3 = Request("tipo_tel3")
            Set Tabela = Banco.Execute("INSERT INTO
cliente_telefone VALUES (" & cpf & ",'" & telefone3 & "','" &
tipo_tel3  & "');")
    End if
    inclusao = Request("inclusao")
    if inclusao = "V1" then
%>
<html>
<head>
<title>Cadastro de Dependentes</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Cadastro
de Dependentes</font></p>
<form method="POST" action="cadastro_dependente.asp">
<%
    Response.write("<input type='hidden' name='cpf_cli' value='" &
cpf &"'>")
```

```
%>
 <table border="0" width="100%">
  <tr>
    <td width="47%" height="30">
      <p align="center"><font face="Verdana" size="4">Nome do
dependente</font></td>
    <td width="27%" height="30">
      <p align="center"><font face="Verdana" size="4">Data de
nascimento</font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Verdana" size="4">Tipo de
relação</font></td>
  </tr>
  <tr>
    <td width="47%" height="30">
      <p align="left"><input type="text" name="dependente1"
size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep1" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep1" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep1" size="1">
        <option>1930</option>
       AQUI CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep1" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
```

```
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente2" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep2" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep2" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep2" size="1">
        <option>1930</option>
         AQUI CONTINUA...
         <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep2" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente3" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep3" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
```

```
        <option>31</option>
      </select><select
          name="mes_dep3" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep3" size="1">
        <option>1930</option>
        AQUI CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep3" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente4" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep4" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep4" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep4" size="1">
        <option>1930</option>
        AQUI CONTINUA...
```

```
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep4" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
    <td width="47%" height="30"><input type="text"
name="dependente5" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep5" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep5" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep5" size="1">
        <option>1930</option>
        AQUI CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep5" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
  <tr>
```

```
    <td width="47%" height="30"><input type="text"
name="dependente6" size="43"></td>
    <td width="27%" height="30"><font face="Arial Black"
size="3"><select
          name="dia_dep6" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>31</option>
      </select><select
          name="mes_dep6" size="1">
        <option selected>01</option>
        <option>02</option>
        AQUI CONTINUA...
        <option>12</option>
      </select><select
          name="ano_dep6" size="1">
        <option>1930</option>
        AQUI CONTINUA...
        <option>2001</option>
      </select></font></td>
    <td width="26%" height="30">
      <p align="center"><font face="Arial Black" size="3"><select
          name="rel_dep6" size="1">
        <option>cônjuge</option>
        <option>filho(a)</option>
        <option>outro parentesco</option>
      </select></font></td>
  </tr>
 </table>
 <p align="center"><b><font face="Arial Black"><input
type="submit" value="Cadastrar" name="B1"></font></b></p>
</form>
<p align="center"> </p>
</center>
</body>
</html>
<%
    else
 %>
<html>
```

```
<head>
<title>Cadastro de Clientes</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<center>
<font face="Verdana" size="4">
Obrigado por se cadastrar.</font>
</center>
</body>
</html>
<%
End if
End if
%>
```

2 Arquivo cadatro_dependente.asp

```
<html>
<head>
<title>Cadastro de Dependentes</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<%
    cpf_cli = Request ("cpf_cli")
    dependente1 = Request ("dependente1")
    dependente2 = Request ("dependente2")
    dependente3 = Request ("dependente3")
    dependente4 = Request ("dependente4")
    dependente5 = Request ("dependente5")
    dependente6 = Request ("dependente6")
    Dim Banco, Tabela
    Set Banco = Server.CreateObject("ADODB.connection")
    Banco.open "hotel", "root", ""
       if len(dependente1) <> 0 then
        dia_dep1 = Request("dia_dep1")
        mes_dep1 = Request("mes_dep1")
        ano_dep1 = Request("ano_dep1")
        dtnasc_dep1 = ano_dep1 & "-" & mes_dep1 & "-" & dia_dep1
        rel_dep1 = Request("rel_dep1")
```

```
        Set Tabela = Banco.Execute("INSERT INTO dependente VALUES
(" & cpf_cli & ",'" & dependente1 & "','" & dtnasc_dep1  & "','" &
rel_dep1 & "');")
       End if
    if len(dependente2) <> 0 then
        dia_dep2 = Request("dia_dep2")
        mes_dep2 = Request("mes_dep2")
        ano_dep2 = Request("ano_dep2")
        dtnasc_dep2 = ano_dep2 & "-" & mes_dep2 & "-" & dia_dep2
        rel_dep2 = Request("rel_dep2")
        Set Tabela = Banco.Execute("INSERT INTO dependente VALUES
(" & cpf_cli & ",'" & dependente2 & "','" & dtnasc_dep2  & "','" &
rel_dep2 & "');")
    End if
    if len(dependente3) <> 0 then
        dia_dep3 = Request("dia_dep3")
        mes_dep3 = Request("mes_dep3")
        ano_dep3 = Request("ano_dep3")
        dtnasc_dep3 = ano_dep3 & "-" & mes_dep3 & "-" & dia_dep3
        rel_dep3 = Request("rel_dep3")
        Set Tabela = Banco.Execute("INSERT INTO dependente VALUES
(" & cpf_cli & ",'" & dependente3 & "','" & dtnasc_dep3  & "','" &
rel_dep3 & "');")
    End if
    if len(dependente4) <> 0 then
        dia_dep4 = Request("dia_dep4")
        mes_dep4 = Request("mes_dep4")
        ano_dep4 = Request("ano_dep4")
        dtnasc_dep4 = ano_dep4 & "-" & mes_dep4 & "-" & dia_dep4
        rel_dep4 = Request("rel_dep4")
        Set Tabela = Banco.Execute("INSERT INTO dependente VALUES
(" & cpf_cli & ",'" & dependente4 & "','" & dtnasc_dep4  & "','" &
rel_dep4 & "');")
    End if
    if len(dependente5) <> 0 then
        dia_dep5 = Request("dia_dep5")
        mes_dep5 = Request("mes_dep5")
        ano_dep5 = Request("ano_dep5")
        dtnasc_dep5 = ano_dep5 & "-" & mes_dep5 & "-" & dia_dep5
        rel_dep5 = Request("rel_dep5")
        Set Tabela = Banco.Execute("INSERT INTO dependente VALUES
(" & cpf_cli & ",'" & dependente5 & "','" & dtnasc_dep5  & "','" &
rel_dep5 & "');")
```

```
    End if
    if len(dependente6) <> 0 then
        dia_dep6 = Request("dia_dep6")
        mes_dep6 = Request("mes_dep6")
        ano_dep6 = Request("ano_dep6")
        dtnasc_dep6 = ano_dep6 & "-" & mes_dep6 & "-" & dia_dep6
        rel_dep6 = Request("rel_dep6")
        Set Tabela = Banco.Execute("INSERT INTO dependente VALUES
(" & cpf_cli & ",'" & dependente6 & "','" & dtnasc_dep6  & "','" &
rel_dep6 & "');")
    End if
%>
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="7">Obrigado por se cadastrar - dependente
</font></p>
</body>
</html>
```

3 Arquivo resultado.asp

```
<html>
<head>
<title>Resultado da Pesquisa de Hotéis</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="6">Resultado da Pesquisa</font></p>
<p align="center"> </p>
<%
    dia = Day(now)
    mes = Month(now)
    ano = Year(now)
    data = ano & "-" & mes & "-" & dia
    dia_entrada = Request("dia_entrada")
    mes_entrada = Request("mes_entrada")
    ano_entrada = Request("ano_entrada")
    data_entrada = ano_entrada & "-" & mes_entrada & "-" &
dia_entrada
    dia_saida = Request("dia_saida")
    mes_saida = Request("mes_saida")
    ano_saida = Request("ano_saida")
    data_saida = ano_saida & "-" & mes_saida & "-" & dia_saida
```

```
    if (Cdate(data_entrada) <= Cdate(data_saida)) and
(Cdate(data_entrada) >= Cdate(data)) and (Cdate(data_saida) >=
Cdate(data)) then
            Dim Banco, Tabela, Reserva
            Set Banco = Server.CreateObject("ADODB.connection")
            Banco.open "hotel", "root", ""
            consulta = "SELECT hot_cnpj, hot_nome, hot_rua,
hot_bairro, hot_cidade, hot_estado, hot_CEP, hot_homepage,
hot_email, qto_numero, qto_tipo, qto_nopessoas, qto_preco,
qto_descricao FROM hotel, quarto WHERE hot_cnpj = qto_hot_cnpj "
            R1 = Request ("R1")
            if R1 = "V1" then    'marcou o estado
                estado = Request("estado")
                if estado <> "Indiferente" then
                    if estado = "Acre" then
                        estado = "AC"
                    elseif estado = "Amazonas" then
                        estado = "AM"
                    elseif estado = "Amapá" then
                        estado = "AP"
                    elseif estado = "Alagoas" then
                        estado = "AL"
                    elseif estado = "Bahia" then
                        estado = "BA"
                    elseif estado = "Ceará" then
                        estado = "CE"
                    elseif estado = "Distrito Federal" then
                        estado = "DF"
                    elseif estado = "Espírito Santo" then
                        estado = "ES"
                    elseif estado = "Goiás" then
                        estado = "GO"
                    elseif estado = "Maranhão" then
                        estado = "MA"
                    elseif estado = "Mato Grosso" then
                        estado = "MT"
                    elseif estado = "Mato Grosso do Sul" then
                        estado = "MS"
                    elseif estado = "Minas Gerais" then
                        estado = "MG"
                    elseif estado = "Pará" then
                        estado = "PA"
```

```
                    elseif estado = "Paraíba" then
                        estado = "PB"
                    elseif estado = "Paraná" then
                        estado = "PR"
                    elseif estado = "Pernambuco" then
                        estado = "PE"
                    elseif estado = "Piauí" then
                        estado = "PI"
                    elseif estado = "Rio de Janeiro" then
                        estado = "RJ"
                    elseif estado = "Rio Grande do Norte" then
                        estado = "RN"
                    elseif estado = "Rio Grande do Sul" then
                        estado = "RS"
                    elseif estado = "Rondônia" then
                        estado = "RO"
                    elseif estado = "Roraima" then
                        estado = "RR"
                    elseif estado = "Santa Catarina" then
                        estado = "SC"
                    elseif estado = "São Paulo" then
                        estado = "SP"
                    elseif estado = "Sergipe" then
                        estado = "SE"
                    elseif estado = "Tocantins" then
                        estado = "TO"
                    end if ' dos estados
                   consulta =  consulta & " AND hot_estado = '" &
estado & "'"
                end if ' do indiferente
            else    'marcou cidade
                cidade = Request("cidade")
                if cidade <> "Indiferente" then
                    cidade = mid(cidade,6, len(cidade) - 5)
                    consulta = consulta & " AND hot_cidade = '" &
cidade & "'"
                end if
            end if 'do estado ou cidade
' tipo do quarto
        tipo = Request("tipo")
        if tipo <> "Indiferente" then
```

```
            consulta = consulta & " AND qto_tipo = '" & tipo  & "'
"
        end if
'quantidade de pessoas
        quantidade = Request("quantidade")
        if quantidade <> "Indiferente" then
            if quantidade = "Solteiro" then
                qtd_pessoas = 1
            elseif quantidade = "Duplo" then
                qtd_pessoas = 2
            elseif quantidade = "Triplo" then
                qtd_pessoas = 3
            elseif quantidade = "Quádruplo" then
                qtd_pessoas = 4
            end if
            consulta = consulta & "AND qto_nopessoas = " &
qtd_pessoas
        end if
'Preço
        preco = Request("preco")
        if preco <> "Indiferente" then
            if preco = "até R$ 30,00" then
                consulta =  consulta & " AND qto_preco < 30 "
            elseif preco = "de R$ 30,00 até R$ 50,00" then
              consulta = consulta & " AND qto_preco >= 30 AND
qto_preco < 50 "
            elseif preco = "de R$ 50,00 até R$ 70,00" then
              consulta = consulta & " AND qto_preco >= 50 AND
qto_preco < 70 "
            elseif preco = "de R$ 70,00 até R$ 100,00" then
             consulta = consulta & " AND qto_preco >= 70 AND
qto_preco < 100 "
            elseif preco = "acima de R$ 100,00" then
                consulta =  consulta & " AND qto_preco >= 100 "
            end if
        end if
'data
        'consulta2 = consulta & "AND (qto_hot_cnpj, qto_numero)
NOT IN "
        consulta2 = "SELECT res_hot_cnpj, res_noquarto FROM
reserva WHERE ( "
```

```
        consulta2 = consulta2 & "('" & data_entrada & "' <
res_entrada AND '" & data_saida  & "' > res_saida) OR "
        consulta2 = consulta2 & "('" & data_entrada & "' >=
res_entrada AND '" & data_saida & "' <= res_saida) OR "
        consulta2 = consulta2 & "('" & data_entrada & "' <
res_entrada AND ('" & data_saida & "' > res_entrada AND '" &
data_saida & "' < res_saida)) OR "
        consulta2 = consulta2 & "(('" & data_entrada & "' >
res_entrada AND '" & data_entrada & "' < res_saida) AND '" &
data_saida & "' > res_saida)) "
        Set Reserva = Banco.Execute(consulta2)
        Dim hoteis_nao()
        Dim quartos_nao()
        numero_vetor = -1
        while not Reserva.EOF
            numero_vetor = numero_vetor + 1
            Redim Preserve hoteis_nao(numero_vetor)
            Redim Preserve quartos_nao(numero_vetor)
            hoteis_nao(numero_vetor) = Reserva("res_hot_cnpj")
            quartos_nao(numero_vetor) = Reserva("res_noquarto")
            Reserva.movenext
        Wend
        Set Tabela = Banco.Execute(consulta)
        imprimir_botao = 1
        if not (Tabela.BOF AND Tabela.EOF) then
            Dim resposta, imprimir
            resposta = 0

%>
    <form method="POST" action="confirma_hotel.asp">
    <table border="1" width="72%" align = center>
<%
            while not Tabela.EOF
                if not Tabela.EOF then
                    for y = 0 to numero_vetor
                        if Tabela("hot_cnpj") = hoteis_nao(y) and
Tabela("qto_numero") = quartos_nao(y) then
                            Tabela.movenext
                            if not Tabela.EOF then
                                y = 0
                            end if
                        end if
```

```
                    next
                end if
                if not Tabela.EOF then
                    hotelcorrente = Tabela("hot_cnpj")
%>
    <tr>
            <td width="100%">
            <p>Nome do Hotel:  <%  = Tabela("hot_nome") %>  </p>
            <p>Endereço:  <% = Tabela("hot_rua") & " / " &
Tabela("hot_bairro") %>  </p>
            <p>Cidade:  <% = Tabela("hot_cidade") & " - " &
Tabela("hot_estado") %>  </p>
            <p>CEP:  <% = Tabela("hot_cep") %>  </p>
            <p>Homepage:  <% = Tabela("hot_homepage") %>  </p>
            <p>Email:  <% = Tabela("hot_email") %>  </p>
<% 'mostrar o telefone
        consulta_tel = "SELECT hot_tel_numero, hot_tel_tipo FROM
hotel_telefone WHERE hot_tel_cnpj = " & hotelcorrente
        Set resultado_tel = Banco.Execute(consulta_tel)
        if not (resultado_tel.BOF AND resultado_tel.EOF) then
             while not resultado_tel.EOF
%>
            <p>Telefone <% = resultado_tel("hot_tel_tipo") %>:  <%
= resultado_tel("hot_tel_numero") %> </p>
<%
                resultado_tel.Movenext
            Wend
        end if 'fim dos telefones
%>
            <table border="1" width="72%" align = center>
            <tr>
                <td>Seleção</td>
                    <td>Tipo do quarto</td>
                <td>Número de pessoas</td>
                <td>Preço</td>
                <td>Descrição</td>
            </tr>
            <tr>
                <td><input type="radio" value="<% = resposta %>"
name="selecao"></td>
                <%
```

```
                    nome = "hot_cnpj" & resposta
                    %>
                    <input type='hidden' name='<% = nome %>'
value='<% = hotelcorrente %>'>
                    <%
                    nome = "numero_quarto" & resposta
                    numeroquarto = Tabela("qto_numero")
                    %>
                    <input type='hidden' name='<% = nome %>'
value='<% = numeroquarto %>'>
                    <%
                    resposta = resposta + 1

                %>
                <td> <%
                    escrita = Tabela("qto_tipo")
                    tipo = Tabela("qto_tipo")
                    Response.write(escrita)
                    %> </td>
                <td> <% escrita = Tabela("qto_nopessoas")
                        nopessoas = Tabela("qto_nopessoas")
                        Response.write(escrita) %> </td>
                <td> <% escrita = Tabela("qto_preco")
                        preco = Tabela("qto_preco")
                        Response.write(escrita) %>,00</td>
                <td> <% = Tabela("qto_descricao") %></td>
            </tr>
<%
            'Aqui vou colocar todos os tipos na tabela do hotel
                flag = 1
                while flag <> 0
                    Tabela.movenext
                    if not Tabela.EOF then
                        for y = 0 to numero_vetor
                            if Tabela("hot_cnpj") = hoteis_nao(y)
and Tabela("qto_numero") = quartos_nao(y) then
                                Tabela.movenext
                                if not Tabela.EOF then
                                    y = 0
                                end if
                            end if
```

```
                    next
                end if
                if not Tabela.EOF then  'para verificar que o
proximo nao e vazio
                    hotel = Tabela("hot_cnpj")
                    if hotel = hotelcorrente then 'continua no
mesmo hotel
                        tipo1 = Tabela("qto_tipo")
                        preco1 = Tabela("qto_preco")
                        nopessoas1 = Tabela("qto_nopessoas")

                        if (CStr(tipo) <> CStr(tipo1)) or
(Ccur(preco) <> Ccur(preco1)) or (CInt(nopessoas) <>
CInt(nopessoas1)) then
%>
                        <tr>
                            <td><input type="radio" value="<%
= resposta %>" name="selecao"></td>
                            <%
                                nome = "hot_cnpj" & resposta
                            %>
                                <input type='hidden' name='<%
= nome %>' value='<% = hotelcorrente %>'>
                            <%
                                nome = "numero_quarto" &
resposta
                                 numeroquarto =
Tabela("qto_numero")
                            %>
                                <input type='hidden' name='<%
= nome %>' value='<% = numeroquarto %>'>
                            <%
                            resposta = resposta + 1

                            %>
                            <td> <%
                                escrita = Tabela("qto_tipo")
                                tipo = Tabela("qto_tipo")
                                Response.write(escrita)
                                 %> </td>
                            <td> <% escrita =
Tabela("qto_nopessoas")
```

```
                                        nopessoas =
Tabela("qto_nopessoas")
                                        Response.write(escrita) %>
</td>
                                <td> <% escrita =
Tabela("qto_preco")
                                         preco =
Tabela("qto_preco")
                                         Response.write(escrita)
%>,00</td>
                               <td> <% = Tabela("qto_descricao")
%></td>

                            </tr>
<%
                            end if 'fim do if dos quartos
                        else
                            flag = 0
                        end if ' fim do if do hotel
                     else
                        flag = 0
                    end if 'fim da verificação do eof
                Wend 'fim do do-while com flag = 0
%>
            </table>
            </td>
    </tr>
<%
                else 'se não há resultados porque todos estão
reservados
                imprimir_botao = 0
%>
</table>
</form>
<p align="center"><font face="Comix Regular Caps" size="5">Nenhum
resultado foi encontrado.</font></p>
<%
            end if 'do tabela.eof também
            Wend 'fim do while do tabela.eof
            if imprimir_botao = 1 then
%>
    </table>
```

```
    <input type='hidden' name='qtdrespostas' value='<%
Response.write(resposta) %>'>
    <input type='hidden' name='data_entrada' value='<%
Response.write(data_entrada) %>'>
    <input type='hidden' name='data_saida' value='<%
Response.write(data_saida) %>'>
    <p align="center"><input type="submit" value="Reservar"
name="B1"></p>
    </form>
<%
            end if
        else ' não há resultados
%>
<p align="center"><font face="Comix Regular Caps" size="5">Nenhum
resultado foi encontrado.</font></p>
<%
        end if
    else
%>
<p align="center"><font face="Comix Regular Caps"
size="5">Verifique as datas de entrada e de saida</font></p>
<%
    end if'da data
%>
</body>
</html>
```

4 Arquivo confirma_hotel.asp

```
<html>
<head>
<title>Reserva do quarto</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="6">Reserva do quarto</font></p>
<form method="POST" action="confirma_reserva.asp">
<%
    Dim Banco, Tabela
    Set Banco = Server.CreateObject("ADODB.connection")
    Banco.open "hotel", "root", ""
    selecao = Request("selecao")
```

```
    qtdrespostas = Request("qtdrespostas")
       for i = 0 to (qtdrespostas - 1) step 1
              if Cint(selecao) = Cint(i) then
            nome = "hot_cnpj" & i
            cnpjhotel = Request(nome)
            nome = "numero_quarto" & i
            noquarto = Request(nome)
                    consulta = "SELECT hot_nome, hot_cidade,
hot_estado, qto_tipo, qto_preco, qto_nopessoas, qto_descricao FROM
hotel, quarto WHERE hot_cnpj = " & cnpjhotel & " AND hot_cnpj =
qto_hot_cnpj AND qto_numero = " & noquarto
            Set Tabela = Banco.Execute(consulta)
%>
       <p align="center"><font face="Times New Roman"
size="4">Voce selecionou um quarto <% = Tabela("qto_tipo") %> do
hotel <% = Tabela("hot_nome") %> da cidade de <% =
Tabela("hot_cidade") %> (<% = Tabela("hot_estado") %>) para <% =
Tabela("qto_nopessoas") %> pessoas. O quarto possui <% =
Tabela("qto_descricao") %> e o preço será R$ <% =
Tabela("qto_preco") %>,00 reais. Para confirmar a reserva do
quarto, você deverá preencher os campos abaixo.</font></p>
<p align="center"> </p>
<%
             i = (qtdrespostas - 1) 'para parar o loop
            data_entrada = Request("data_entrada")
            data_saida = Request("data_saida")
%>
                    <input type='hidden' name='cnpjhotel'
value='<% Response.write(cnpjhotel) %>'>
                    <input type='hidden' name='noquarto' value='<%
Response.write(noquarto) %>'>
                    <input type='hidden' name='data_entrada'
value='<% Response.write(data_entrada) %>'>
                    <input type='hidden' name='data_saida'
value='<% Response.write(data_saida) %>'>
<%
             end if
       Next 'fim do loop
%>
<table border="0" width="88%" height="132" cellspacing="3">
       <tr>
       <td width="55%" height="24"><font face="Verdana"><font
size="4">CPF:</font>
       </td>
```

```
      <td width="55%" height="24"><input type="text" name="cpf"
size="12">
      </td>
      </tr>
      <tr>
      <td width="55%" height="24"><font face="Verdana"><font
size="4">Senha:</font>
      </td>
      <td width="55%" height="24"> <input type="password"
name="senha" size="12">
      </td>
      </tr>
</table>
<p align="center"><input type="submit" value="Confirmar reserva"
name="B1"></p>
</form>
</body>
</html>
```

5 Arquivo confirma_reserva.asp

```
<html>
<head>
<title>Reserva do quarto</title>
</head>
<body topmargin="0" leftmargin="0" bgcolor="#FFFFFF">
<p align="center"><font color="#000080" face="Comix Regular Caps"
size="6">Reserva do quarto</font></p>
<p align="center"> </p>

<%
    Dim Banco, Tabela
    Set Banco = Server.CreateObject("ADODB.connection")
    Banco.open "hotel", "root", ""
    cpf = Request("cpf")
    consulta = "SELECT cl_senha FROM cliente WHERE cl_cpf = " &
cpf
    Set Tabela = Banco.Execute(consulta)
    senha = Request("senha")
       if senha = Tabela("cl_senha") then
        cnpjhotel = Request("cnpjhotel")
        noquarto = Request("noquarto")
```

```
        data_entrada = Request("data_entrada")
        data_saida = Request("data_saida")
       insercao = "INSERT INTO reserva VALUES (" & cpf & ", " &
cnpjhotel & ", " & noquarto & ", '" & data_entrada & "', '" &
data_saida & "')"
        Set Tabela = Banco.Execute(insercao)
%>
<p align="center"><font face="Times New Roman" size="5">Reserva
realizada com sucesso.
</font></p>
<%
    else
%>
<p align="center"><font face="Times New Roman" size="5">A senha
está incorreta.
</font></p>
<%
       end if
%>
</body>
</html>
```

ANEXO E

1 Listagem completa de comandos do Oracle8i

```
CREATE TYPE "VANESSA"."ENDERECO_CLI" AS OBJECT (
"CL_RUA" VARCHAR2(50),
"CL_BAIRRO" VARCHAR2(25),
"CL_CIDADE" VARCHAR2(20),
"CL_ESTADO" CHAR(2),
"CL_CEP" CHAR(9),
"CL_PAIS" VARCHAR2(20))

CREATE TYPE "VANESSA"."TELEFONE" AS OBJECT (
“TEL_NUMERO" CHAR(15),
"TEL_TIPO" VARCHAR2(15))

CREATE TYPE "VANESSA"."TELS" AS VARRAY (5) OF VANESSA.TELEFONE

CREATE TYPE "VANESSA"."DEPENDENTE" AS OBJECT (
"DEP_NOME" VARCHAR2(60),
"DEP_DTNASC" DATE,
"DEP_RELACAO" VARCHAR2(40))

CREATE TYPE "VANESSA"."REFDEPS" AS TABLE OF
“VANESSA"."DEPENDENTE"

CREATE TYPE "VANESSA"."CLIENTE" AS OBJECT (
"CL_NOME" VARCHAR2(60),
"CL_CPF" INTEGER,
"CL_END" "VANESSA"."ENDERECO_CLI",
"CL_DTNASC" DATE,
"CL_SENHA" VARCHAR2(20),
"CL_TELS" "VANESSA"."TELS",
"CL_DEPS" "VANESSA"."REFDEPS")

CREATE TABLE "VANESSA".CLIENTES OF "VANESSA"."CLIENTE" (
"CL_NOME" NOT NULL,
"CL_CPF" NOT NULL,
```

```
"CL_SENHA" NOT NULL,
PRIMARY KEY("CL_CPF"))
NESTED TABLE "CL_DEPS" STORE AS tab_deps

CREATE TYPE "VANESSA"."ENDERECO_HOT" AS OBJECT (
"HOT_RUA" VARCHAR2(50),
"HOT_BAIRRO" VARCHAR2(25),
"HOT_CIDADE" VARCHAR2(20),
"HOT_ESTADO" CHAR(2),
"HOT_CEP" VARCHAR2(9))

CREATE TYPE "VANESSA"."HOTEL" AS OBJECT (
"HOT_NOME" VARCHAR2(50),
"HOT_END" "VANESSA"."ENDERECO_HOT",
"HOT_CNPJ" INTEGER,
"HOT_HOMEPAGE" VARCHAR2(50),
"HOT_EMAIL" VARCHAR2(50),
"HOT_TELS" "VANESSA"."TELS" )

CREATE TABLE "VANESSA".HOTEIS OF "VANESSA"."HOTEL" (
"HOT_NOME" NOT NULL,
"HOT_CNPJ" NOT NULL,
PRIMARY KEY("HOT_CNPJ"))

CREATE OR REPLACE TYPE "VANESSA"."QUARTO" AS OBJECT (
"QTO_HOTEL" REF "VANESSA"."HOTEL",
"QTO_NUMERO" INTEGER,
"QTO_TIPO" VARCHAR2(25),
"QTO_PRECO" DECIMAL(9),
"QTO_NOPESSOAS" INTEGER,
"QTO_DESCRICAO" VARCHAR2(200))

CREATE TABLE "VANESSA".QUARTOS OF "VANESSA"."QUARTO"(
"QTO_HOTEL" NOT NULL,
"QTO_NUMERO" NOT NULL,
PRIMARY KEY("QTO_HOTEL","QTO_CODIGO"))
```

```
CREATE OR REPLACE TYPE "VANESSA"."RESERVA" AS OBJECT (
"RES_CLIENTE" REF "VANESSA"."CLIENTE",
"RES_QUARTO" REF "VANESSA"."QUARTO",
"RES_ENTRADA" DATE,
"RES_SAIDA" DATE)

CREATE TABLE "VANESSA".RESERVAS OF "VANESSA"."RESERVA" (
"RES_CLIENTE" NOT NULL,
"RES_QUARTO" NOT NULL,
"RES_ENTRADA" NOT NULL,
"RES_SAIDA" NOT NULL,
"RES_CODIGO" NOT NULL,
PRIMARY KEY("RES_CODIGO"))
```