|  |
| --- |
| FACULDADE SUMARÉ |
| Modelagem de Dados |
| Banco de Dados |
|  |
| ALEX SANTOS RA: 1622772 ALISON HENRIQUE RA: 1621280 ANDRÉ LUIZ RA: 1622247 ISABEL URÇULA RA: 1622403 VINICIUS SOARES RA: 1610673 |

|  |
| --- |
|  |

São Paulo 2016

Sumário

[APÊNDICE 4](#_Toc468060540)

[1. MODELO RELACIONAL: 6](#_Toc468060541)

[1.1 MODELO DE ENTIDADE RELACIONAL: 7](#_Toc468060542)

[1.2 DIAGRAMA DE ENTIDADE RELACIONAL: 8](#_Toc468060543)

[1.3 MODELO FÍSICO (SCRIPT DE GERAÇÃO): 9](#_Toc468060544)

[2. MODELO ORIENTADO A REDES: 10](#_Toc468060545)

[3. MODELO ORIENTADO A SERVIÇOS: 12](#_Toc468060546)

[3.1 BIG DATA: 12](#_Toc468060547)

[3.2 POR QUE BIG DATA É IMPORTANTE? 13](#_Toc468060548)

[3.3 BUSINESS INTELLIGENCE 13](#_Toc468060549)

[3.4 IMPORTÂNCIA DO BUSINESS INTELLIGENCE PARA EMPRESAS: 14](#_Toc468060550)

[3.5 COM O BI SENDO APLICADO CORRETAMENTE, PODEMOS OBSERVAR OS SEGUINTES BENEFÍCIOS: 14](#_Toc468060551)

[4. BANCO DE DADOS NÃO RELACIONAL 15](#_Toc468060552)

[4.1 ORIGEM 15](#_Toc468060553)

[4.2 CARACTERÍSTICAS DO NOSQL 15](#_Toc468060554)

[4.3 NOSQL: CRESCIMENTO E VANTAGENS. 16](#_Toc468060555)

[4.4 CONCEITOS DE TABELA NOSQL 18](#_Toc468060556)

[4.5 BASE DE DADOS E UTILIZAÇÃO 20](#_Toc468060557)

[5. MODELO ORIENTADO A OBJETOS 21](#_Toc468060558)

[5.1VANTAGENS E DESVANTAGENS DO SGBDOO 21](#_Toc468060559)

[5.2 CONCLUSÃO 22](#_Toc468060560)

[6. CASE HOTELARIA: 23](#_Toc468060561)

[7. SOBRE O PROJETO: 24](#_Toc468060562)

[8. Modelagem do Software: 25](#_Toc468060563)

[8.1 DIAGRAMA DE ENTIDADE RELACIONAL 25](#_Toc468060564)

[8.2 DIAGRAMA DE ENTIDADE RELACIONAL 26](#_Toc468060565)

[8.3 DICIONÁRIO DE DADOS 27](#_Toc468060566)

[8.4 MODELO FÍSICO (SCRIPT) 29](#_Toc468060567)

[9. Sistema: 42](#_Toc468060568)

[9.1 Logotipo: 42](#_Toc468060569)

[9.2 Login: 42](#_Toc468060570)

[9.3 Home: 43](#_Toc468060571)

[9.4 Cadastro de Cliente: 43](#_Toc468060572)

[9.6 Reserva: 44](#_Toc468060573)

[10 Referências Bibliográficas: 45](#_Toc468060574)

# APÊNDICE

Os processos tem se desenvolvido com o tempo, sendo otimizados para obtermos um melhor desempenho e confiabilidade, não tratando-se apenas de processos informáticos, mas sim de qualquer tipo de atividade repetitiva, ou que requer uma certa confiabilidade.

Todo dado pode virar uma informação útil para a empresa, basta abstrairmos de forma correta do mundo real e transformar o dado em algo de essencial importância em nosso cotidiano. Tudo é armazenável, tudo é utilizável.

Em 1960, quando o ser humano parou de armazenar as informações em bancos de dados físicos, como caixas de metal, e nas pastas (Tabelas), os processos começaram a se desenvolver de uma forma mais homogênea e uniforme, pois a probabilidade de erro do preenchimentos de ficha por um humano, era bem maior que de uma máquina. Obviamente isso não foi a panaceia imediata para todos os problemas processuais do antigo cenário da tecnologia. Foi preciso galgar um bom tempo para que a estrutura de dados realmente começasse a ficar confiável e estável. Em 1970, Frank Codd introduziu o modelo de dados relacional, que fazia exatamente todo o sentido: Tudo é relacionado em uma empresa. Um PRODUTO vem do ESTOQUE que vai ser VENDIDO para um CLIENTE, pensando desta forma, começamos a entender que quanto mais bem definido era o relacionamento, melhor era a confiabilidade e solidez. Obviamente isso seria gerado apenas pela criação de um modelo, sempre precisamos de uma linguagem atrelada a um conceito, para que as teorias tomem forma e não fiquem apenas no papel.  Para ser atrelado a proposta do modelo relacional, foi criado a linguagem de consulta estruturada (SQL - Structured Query Language), pela equipe da IBM.  
O SQL foi um grande avanço para a área tecnológica, e é usado até os dias atuais, tendo algumas alterações desde o lançamento.

Por mais ansiosos que a população estivesse para vivenciar os avanços tecnológicos, e “Peripécias” tecnológicas, o banco  de dados é um universo escondido que poucas pessoas tem o conhecimento do poder dele. Se pensarmos que uma aplicação é fantástica, antes precisamos abrir o SGBD\* para  ficar de boca aberta com a sua complexidade e capacidade de cálculos, armazenamento e velocidade.

No fim, uma aplicação serve para mostrar os dados que estão armazenados, e quem faz todos os cálculos e precisa ter a confiabilidade de segurar a persistência dos dados por anos, é o banco. Ao meu ver, o banco de dados é inerente a uma aplicação, como o sistema nervoso para o funcionamento do ser humano.

Existem vários modelos de banco de dados, que seguiram o conceito de redes, posteriormente o relacional criado por codd, orientado a objetos, a serviços… Enfim, existe uma gama de modelos que podemos seguir, mas cabe ao DBA\* utilizar o modelo que mais se enquadra na abstração realizada. É possível termos a mesma confiabilidade em todos os modelos, mas a sua usabilidade é totalmente diferente.

Neste projeto abordaremos os modelos de banco de dados mais relevantes  e uma solução, implementada no modelo relacional.

# MODELO RELACIONAL:

Os princípios do modelo de dados relacional foram descritos por Dr. E. F. Codd em junho de 1970 intitulado de "A Relational Model of Data for Large Shared Data Banks".

Os modelos usados na época eram hierárquicos, rede, estruturas de dados simples.

Um banco de dados relacional nasce de uma abstração do mundo real, que nada mais é que tornar o que é físico em códigos computacionais.

Para efetuarmos uma abstração, precisamos fazer uma análise: O que eu preciso guardar que é de extrema importância? O que eu guardei, eu utilizarei em todo o meu escopo de projeto? Eu preciso guardar algo além do nome disso?  
Bom, em uma situação onde eu tenho uma Papelaria, que vende Produtos a Clientes, eu gero uma Receita, que também pode ser abatida das Despesas.

Esses setores são chamados de Entidades.

Cada um desses setores mencionados tem as suas peculiaridades, que são chamados de Atributos.

Os atribuídos podem ser definidos como fortes ou fracos, como por exemplo: Uma caneta da entidade Produto tem vários atributos, como a sua cor, o seu preço, o seu tamanho. Mas imagina alguém pedir para pegar uma caneta específica no estoque? Existem várias canetas azuis, que custam R$ 2,00, mas isso não determina qual é a caneta específica que foi solicitada. O atributo forte é o Código da caneta, e o atributo fraco são atributos que não definem a sua unicidade.

Os atributos fortes no banco relacional são definidos como Chaves: Existem as chaves primárias que são únicos, e que não se repetem. E as chaves estrangeiras que servem para relacionar as entidades. O relacionamento é algo válido para interligarmos as tabelas e “amarrarmos” os registros.

Um registro aparece horizontalmente em um banco de dados, como uma linha no excel. Os atributos se convertem em colunas, e a tabela é o conjunto dos atributos e dos registros (Tuplas).

Também precisamos mencionar a Cardinalidade, que podemos limitar a associação dos registros de Um para Um (1:1), de Zero para Um (0:1), Um para Muitos (1:N) e Zero para Muitos (0:N).

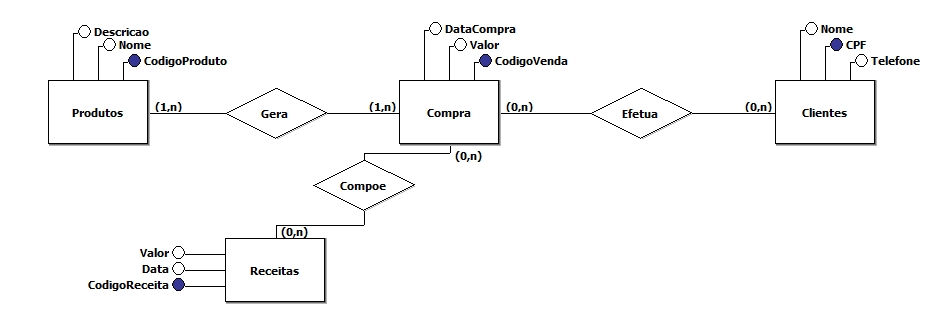
Com o banco de dados relacional, é possível programar a maioria das regras de negócios de diversos ramos de atividades.

O modelo relacional foi explicado com o auxílio de diagramas, que posteriormente foram mais desenvolvidos. Um exemplo dado acima, pode ser facilmente modelado e posteriormente implantado em linguagem SQL. A modelagem é tão importante, pois ela é a documentação de referência para realizar a programação final do banco de dados.  
Existem 3 tipos de diagramas: O lógico, o físico e o Conceitual.

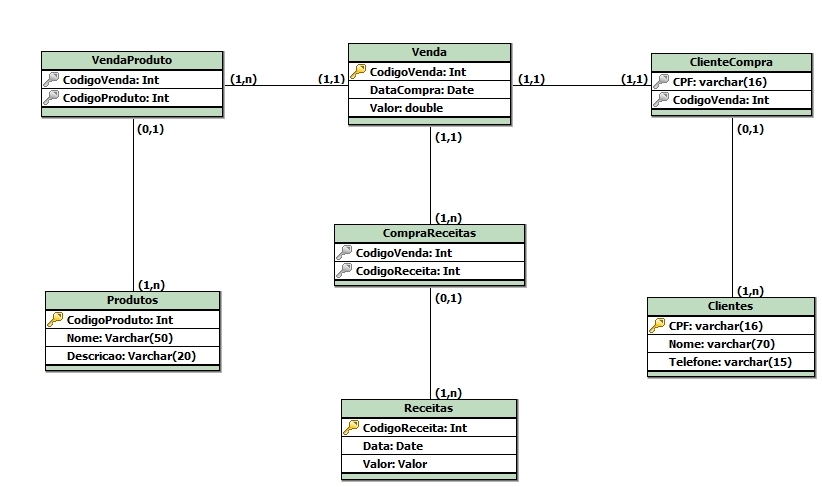
O diagrama conceitual, também conhecido como Modelo de Entidade Relacional (MER), pode ser facilmente identificado pela sua característica, que faz o relacionamento (Losângulo) das entidades (Retângulo), tendo os seus atributos (elipse) fortes destacados e os fracos idênticos aos outros.

Um exemplo prático disso é a modelagem da regra de negócio citada acima:

## 1.1 MODELO DE ENTIDADE RELACIONAL:



## 1.2 DIAGRAMA DE ENTIDADE RELACIONAL:



## 1.3 MODELO FÍSICO (SCRIPT DE GERAÇÃO):

**CREATE** DATABASE PAPELARIA**;**

**USE** PAPELARIA**;**

**CREATE** **TABLE** Receitas **(**

CodigoReceita **Int** **PRIMARY** **KEY,**

**Data** **Date,**

Valor **double**

**)**

**CREATE** **TABLE** ClienteCompra **(**

CPF **varchar(**16**),**

CodigoVenda **Int**

**)**

**CREATE** **TABLE** Venda **(**

CodigoVenda **Int** **PRIMARY** **KEY,**

DataCompra **Date,**

Valor **double**

**)**

**CREATE** **TABLE** VendaProduto **(**

CodigoVenda **Int,**

CodigoProduto **Int,**

**FOREIGN** **KEY(**CodigoVenda**)** **REFERENCES** Venda **(**CodigoVenda**)**

**)**

**CREATE** **TABLE** CompraReceitas **(**

CodigoVenda **Int,**

CodigoReceita **Int,**

**FOREIGN** **KEY(**CodigoVenda**)** **REFERENCES** Venda **(**CodigoVenda**),**

**FOREIGN** **KEY(**CodigoReceita**)** **REFERENCES** Receitas **(**CodigoReceita**)**

**)**

**CREATE** **TABLE** Clientes **(**

CPF **varchar(**16**)** **PRIMARY** **KEY,**

Nome **varchar(**70**),**

Telefone **varchar(**15**)**

**)**

**CREATE** **TABLE** Produtos **(**

CodigoProduto **Int** **PRIMARY** **KEY,**

Nome **Varchar(**50**),**

Descricao **Varchar(**20**)**

**)**

**ALTER** **TABLE** ClienteCompra **ADD** **FOREIGN** **KEY(**CPF**)** **REFERENCES** Clientes **(**CPF**)**

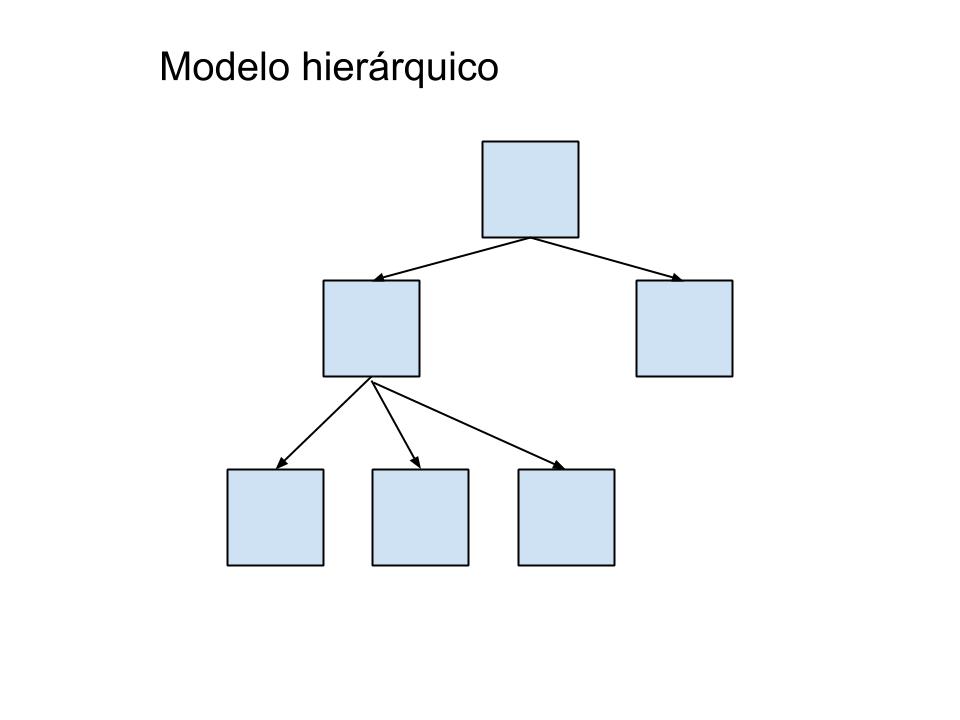
**ALTER** **TABLE** ClienteCompra **ADD** **FOREIGN** **KEY(**CodigoVenda**)** **REFERENCES** Venda **(**CodigoVenda**)**

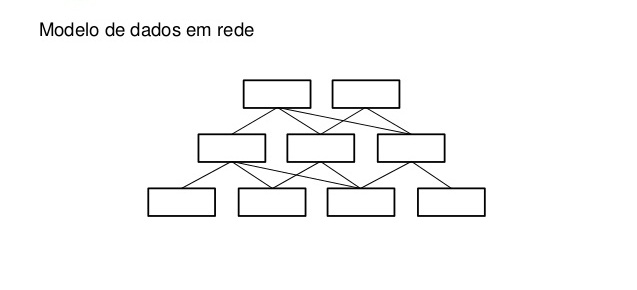
**ALTER** **TABLE** VendaProduto **ADD** **FOREIGN** **KEY(**CodigoProduto**)** **REFERENCES** Produtos **(**CodigoProduto**)**

Já com esse banco de dados básico, podemos inserir e manipular as informações que precisamos para criar uma solução. Obviamente, um banco de uma Papelaria teria muito mais tabelas, e condições.

# MODELO ORIENTADO A REDES:

Charles Bachman foi quem iniciou os trabalhos utilizando este modelo no ano de 1964. O Banco de dados de REDE é muito similar ao modelo hierárquicos, pois também possui uma estrutura hierárquica, porém possuem suas diferenças. O banco de dados de rede possui uma estrutura que nos lembram uma teia de aranha, enquanto  o modelo hierárquico está mais para uma árvore de cabeça para baixo.





Esse modelo fez cair o conceito de hierarquia, e aceitando que apenas um registro pudesse interagir com muitas associações. Em banco de dados de rede, podemos denominar membro como “criança” e os proprietários de “pais” (ou dono). O diferencial está em que cada membro pode ter mais de um proprietário, e com essa questão possuímos conexões mais complexas, tendo muita utilidade em sistemas de mainframe. Os  Bancos de dados de rede são considerados flexíveis por conseguirem efetuar mais conexões entre diferentes tipos de dados. Esse modelo possui uma estrutura mais completa, composto por propriedades básicas de registros, conjuntos de ocorrência, e tem a DDL  e a DML como linguagens de definição, que são compostas pela estrutura formada por entidade, atributos, registro e ocorrência do registro.

# MODELO ORIENTADO A SERVIÇOS:

## 3.1 BIG DATA:

Big Data é o termo que descreve o imenso conjunto de dados estruturados e não estruturados, que precisam de ferramentas para que possam ser organizadas, e assim encontradas, analisadas e aproveitadas o mais rápido possível.

O Big Data é baseado em 5 V's:

VELOCIDADE**.** Os dados são gerados em uma grande velocidade, e devem ser organizados e executados em uma grande velocidade.

VOLUME.A quantidade de dados . Big Data necessita do processamento de grandes volumes de dados desconhecidos, como publicações em redes socias, troca de e-mails, tráfego de dados em linhas telefônicas e muito mais. O trabalho de big data é converter esses dados em informações importantes.

VARIEDADE.Em todos os tipos de formatos os dados são gerados, desde dados estruturados que são armazenados em bancos de dados,  vídeos, áudios, textos não estruturados, transações financeiras e etc.

VERACIDADE. Os dados precisam ser confiáveis, precisam ser autênticos.  
Com isso a análise e estatística de grandes volumes de dados é possível compensar as informações incorretas.

VALOR. De nada adianta possuir acesso a uma grande quantidade de dados se eles não são relevantes e não geram valor.  
Com todas as combinações citadas acima o big data consegue obter valor em seus dados, e garantir um retorno satisfatório.

Então entendemos que Big Data  apenas define as informações de uma natureza específica, Big Data não nos informa como utilizar as informações, com que destreza ela deve ser trabalhada, ou que tratamentos estatísticos ela deve receber. Big Data não é uma solução, Big Data define uma  problemática.

## 3.2 POR QUE BIG DATA É IMPORTANTE?

A importância do Big Data não está focado na quantia de dados que possue, a questão fundamental é, o que você faz com esses dados. Você pode analisar quaisquer dados de fontes para encontrar respostas que permitam 1 -  redução de custos, 2 - redução de tempo, 3 - desenvolvimento de novos produtos, 4 -  decisões mais coerentes. Quando você junta o big data com a potência do analytics, você pode tratar de tarefas interligadas a negócios, como:

Analisar a causa raiz de falhas, problemas e defeitos em tempo quase real;

Gerar cupons no ponto de venda com base em hábitos de compra dos clientes;

Recalcular carteiras de risco inteiras, em questão de minutos;

Detectar Fraudes antes que elas afetem sua organização.

## 3.3 BUSINESS INTELLIGENCE

Em um mercado mutável, com empresas tendo que se adaptar e se atualizar constantemente, o Business Intelligence facilita a interpretação de dados brutos da empresa, em informações relevantes e significativas. é um processo de organização, coleta, análise, monitoramento e compartilhamento das informações, que são necessárias para a gestão de negócios.  é uma maneira de ajudar os gestores a saírem do campo da intuição, partindo para a tomada de decisões assertivas com dados reais, assim gerando sucesso e valores para a empresa.

As empresas que trabalham com BI, têm a capacidade de auxiliar na organização de um grande numero de dados desestruturados. Essa organização é capaz de ajudar a empresa a criar novas estrátegias para seu negócio.

## 3.4 IMPORTÂNCIA DO BUSINESS INTELLIGENCE PARA EMPRESAS:

O Necessário é atender e resolver casos em processos de negócios já existentes na empresa, seja qual for, desde os processos mais comuns (recebimentos, pagamentos, contabilidade, compras, etc) até os mais complexos. Acontece de  muitos projetos falharem, ou simplesmente morrerem, principalmente por não estarem focados.

Sim, os projetos de BI tem uma alta taxa de mortalidade, acontece de projetos de milhões de reais serem simplesmente descontinuados.

## 3.5 COM O BI SENDO APLICADO CORRETAMENTE, PODEMOS OBSERVAR OS SEGUINTES BENEFÍCIOS:

* Facilidade em identificar tendências.
* Antecipação dos riscos.
* Acesso e distribuição de informações de forma abrangente.
* Ofertas e análise de dados mais rápidas e com maior flexibilidade.

# BANCO DE DADOS NÃO RELACIONAL

## 4.1 ORIGEM

O Banco de Dados não relacional (NOSQL – Not Only SQL) surgiu em meados de 1998 em meio a uma solução de banco de dados na qual não possuía uma interface comum na época, a do SQL, porém a arquitetura ainda era baseada na mesma, isso fez com que esse termo fosse uma alternativa real ao modelo relacional.

  Um fato interessante é que com o surgimento do NOSQL, o avanço da web deu um passo gigantesco, isso porque com a quantidade de serviços online como rede sociais e streaming, surgiu um novo conceito que seria a “Web Semântica ou Web 3.0”, e uma forma de armazenamento de fácil acesso e sem restrições ajudou muito esse processo que abrangia uma organização personalizada, com pesquisas inteligentes e sites mais estruturados.

## 4.2 CARACTERÍSTICAS DO NOSQL

São diversas as características do NOSQL, e entre elas a principal seria a capacidade de ser altamente escalável, sendo assim, facilmente distribuído em caso de grandes ou imensos volumes de dados (BIG DATA). Outra característica inquestionável seria o custo x beneficio, já que a maioria dos bancos de dados NOSQL são gratuitos, evitando gastos com licenças.

## 4.3 NOSQL: CRESCIMENTO E VANTAGENS.

Um dos fatores que ajudaram o crescimento do NOSQL foi saber que a ferramenta possui diversas  variedades de tecnologia no campo de banco de dados e o surgimento de tal foi como uma resposta para o avanço da tecnologia que atualmente conta com milhões de usuários ativos mundialmente (Ex: Web 3.0 ou Web Semântica), realizando cadastros, atualizações e diversas atividades que utilizam uma base de dados e essa alta demanda só poderia ser controlada de forma não relacional.

 Mas afinal, quais seriam as vantagens do NOSQL para uma base de dados relacional?

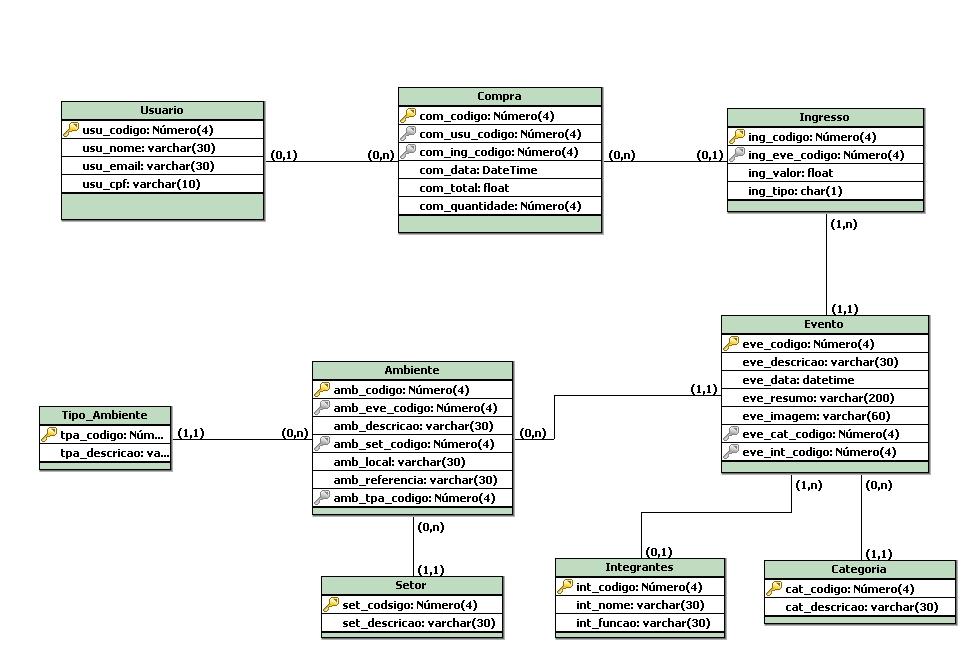
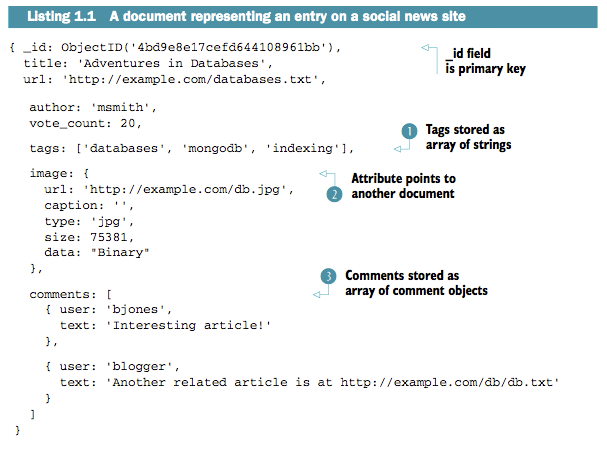
*Figura 3.1*

FIGURA 3.1**:** Nessa imagem vemos um modelo de base de dados relacional e podemos concluir que tudo é registrado e guardado em tabelas, com regras específicas e que uma tabela se relaciona com a outra para agilidade e praticidade no armazenamento e busca de informações no banco de dados.



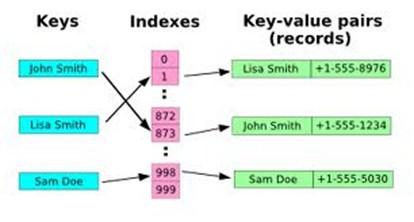
*Figura 3.2*

FIGURA 3.2: Já nesta imagem, verificamos que a base de dados não relacional não possui regras, isso porque o NOSQL visa administrar altos volumes de informações, e isso é feito com a criação de entidades e tags, onde a entidade criada pode ser constantemente alterada e mesmo assim com acesso mais rápido tanto para leitura quanto para escrita e ser “reconhecida” com uma entidade única.

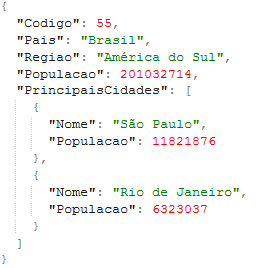
De fato as duas bases de dados são ótimas ferramentas tanto que são usadas em grandes empresas mundialmente falando, porém cada um têm sua especialidade, e nisso a base de dados não relacional se sobressai, isso pelo fato de ser escalável, gratuito e open source, o que nos leva a evitar gastos com hardware (Cloud) e licenças.

## 4.4 CONCEITOS DE TABELA NOSQL

* Sistemas baseados em armazenamento chave valor**:** Situação baseada na existência de uma coleção de chaves únicas e de valores e são associadas ás chaves enquanto um único indíce é utilizado para encontrar esses valores.



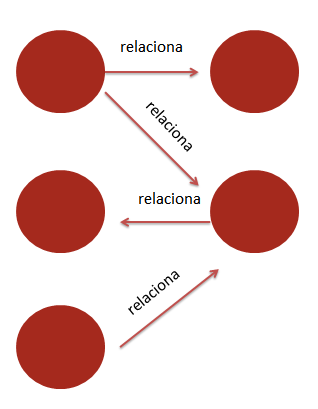
* Sistemas orientados a documentos:Toda entrada no banco de dados é referente a um documento que é decomposto por um identificador e um valor, sendo possível criar mais identificadores e valores dentro desse mesmo documento. São dados estruturados de uma forma encadeada, sendo assim coleções, datas, informações e etc …



* Sistemas orientados a coluna:Utiliza-se de tabelas para representação de entidades, e os dados são gravados em disco, agrupados por colunas, o que reduz o tempo de leitura e escrita em disco isso porque o banco de dados irá sempre buscar a associação com um ID Virtual.



* Sistemas baseados em grafos:È a forma mais escalável de uma base de dados não relacional pois toda sua busca é feita por base de relações entre os dados, bastante utilizado em sites como Twitter onde, por exemplo, se têm a opção de ser seguido e seguir de volta algum usuário.



## 4.5 BASE DE DADOS E UTILIZAÇÃO

Atualmente no mercado, encontramos várias bases de dados NOSQL, e como comentado anteriormente muitas empresas no mercado de TI já utilizam, e, sem dúvida, por conta de uma gigantesca quantidade de dados e os mesmos devem sempre estar acessíveis e de forma rápida, vejamos alguns exemplos de empresas e base de dados que são utilizadas:

* **Cassandra:** Facebook, Twitter. IBM e Netflix;
* **Hadoop:** Yahoo;
* **MongoDB:** Engine Yard;
* **Dynamo:** Amazon;
* **BigTable:** Google.

# MODELO ORIENTADO A OBJETOS

O desenvolvimento dos Sistemas de Gerenciamento de Banco de Dados Orientado a Objetos, teve seu nascimento na combinação de ideias dos modelos de dados tradicionais e também de linguagens de programação orientada a objetos.No modelo orientado a objetos, a noção de objeto é usada no nível lógico que possue traçosbem distintos do que é visto nas linguagens de programações convencionais, como operadores de manipulação de estruturas, gerenciamento de armazenamento, tratamento de integridade e persistência dos dados.

Esse modelo tem um importante intuito nos SGBDs, pois eles são mais convenientes para lidar comdados mais complexos e que representam de fato as aplicações gráficas, de hipertexto ( internet), de vídeo, imagem e som, programas e simulações. Por possuírem uma maior natureza conceitual, e por estarem de acordo com tendências em linguagens de programação e engenharia de software de uma melhor forma no contexto de orientação a objetos.

## 5.1VANTAGENS E DESVANTAGENS DO SGBDOO

VANTAGENS

* A capacidade de armazenamento de objetos;
* Um maior desempenho das consultas por não possuir chaves primárias e nem mesmo estrangeiras;
* Os objetos se comunicam através de mensagens;
* Reuso de código;

DESVANTAGENS

* Falta de padronização nas linguagens de manipulação de dados;
* Alto custo na aquisição de novas tecnologias;
* Maior tempo de aprendizagem e adaptação ao novo ambiente*;*

## 5.2 CONCLUSÃO

Por motivo de desenvolvimento, linguagem de programação, e informações entre banco de dados e desenvolvimento, necessidade de adaptação e grande escalabilidade, o BDOOé a melhor opção para armazenamento de informações, quando se utiliza o padrão orientado  a objetos para aplicações em desenvolvimento.

Os bancos de dados orientados a objetos ainda possuem uma certa dificuldade em questão de aprendizagem, sendo caracterizado pelo seu complicado manuseio, porém estão sendo bem recebidos, em consequência do crescimento da multimídia e internet..

# CASE HOTELARIA:

Diversos estabelecimentos do ramo de hotelaria fazem o seu gerenciamento por meio de planilhas em excel, ou em bloco de notas. Analisando esse processo, criamos uma solução, que envolve um banco de dados relacional e um sistema para gerenciar os cadastros e efetuar reservas, para evitar Overbookings e perda de dados de clientes.

Atualmente, qualquer perda de informação pode gerar um prejuizo imenso para qualquer empresa, devido e esse perigo que ninguém mais quer correr, mais estabelecimentos tem informatizado os seus processos. Isso não só traz um ganho de estabilidade na segurança, mas também, na otimização dos processos.

Um atendente que precisa atender uma ligação, ver disponibilidade em uma planilha em excel, colher todos os dados do cliete, informar valores, e anotar a data da reserva, e logo após enviar um voucher com uma confirmação. Quanto tempo isso demora, e qual a probabilidade de ocorrer um erro? Bem, essa resposta fica por conta de quem passa por isso.

Um sistema bem programado, pode validar os dados, reaproveitar as informações já inseridas no passado, efetuar cálculos, realizar relatórios, e inclusive gerar o voucher em PDF. Além de mais confiável, ele pode ter diversos recursos como backup, anotações e relatórios que demoravam dias para ser feitos, que o sistema gera em segundos.

No quesito segurança, podemos criar camadas para que um usuário seja diferenciado do administrador,liberando ou restringindo funções para cada nível de permissão, log de sistema para não ter aquela famosa frase: “Não foi eu que fiz, foi o fulano de tal…”.

No quesito estética e usabilidade, parece que não altera muito na produtividade, mas diversos designers trabalham para um site, ou um sistema ficar com uma usabilidade boa, com o dimensionamento e pociosionamento dos elementos na tela. Não creio que muitas pessoas gostam de passar horas na frente do Excel, e uma interface gráfica bem elaborada prende as pessoas a não se incomodarem tanto de ter que usar aquilo em sua jornada de trabalho. Visto pela evolução que teve após a criação do Cipper.

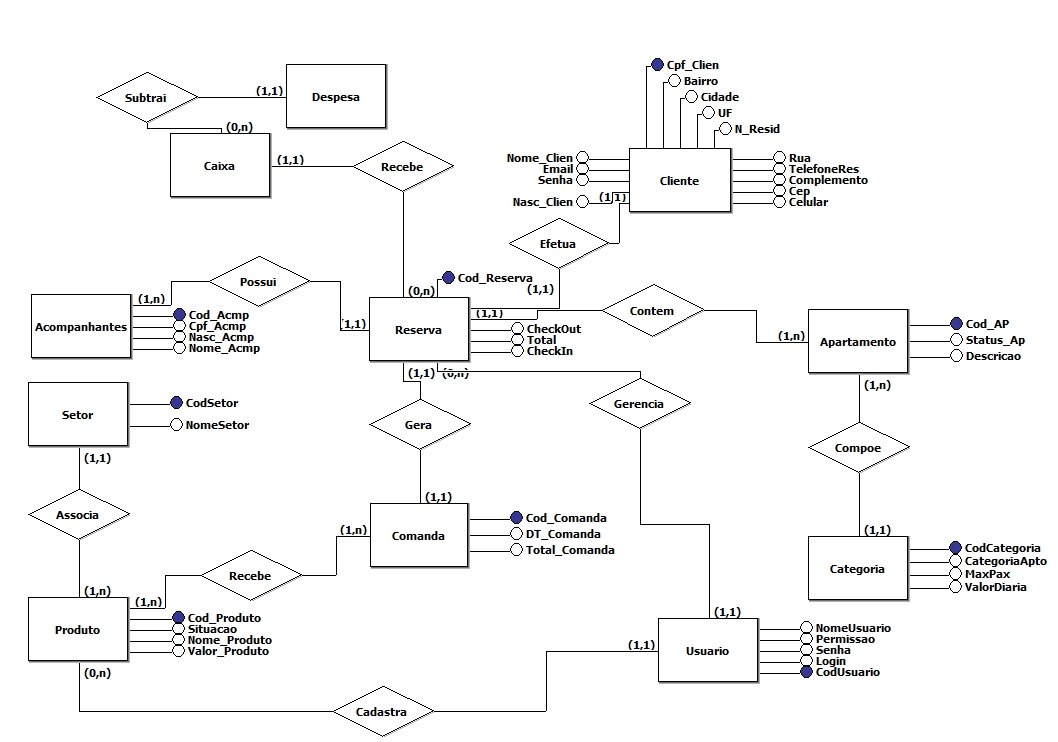
# SOBRE O PROJETO:

Um hotel é composto de várias áreas, mas o foco dele pode não ser determinado apenas com a hospedagem. Existem hotéis Spa’s, com salões de eventos, com restaurante aberto ao público, clubes, enfim… Analisando esse caso, precisamos determinar que pode haver vários setores atrelados ao produto final que é a hospedagem, como a cozinha, itens vendidos na recepção, ou até mesmo serviços, como locação de bicicletas, massagens e etc.

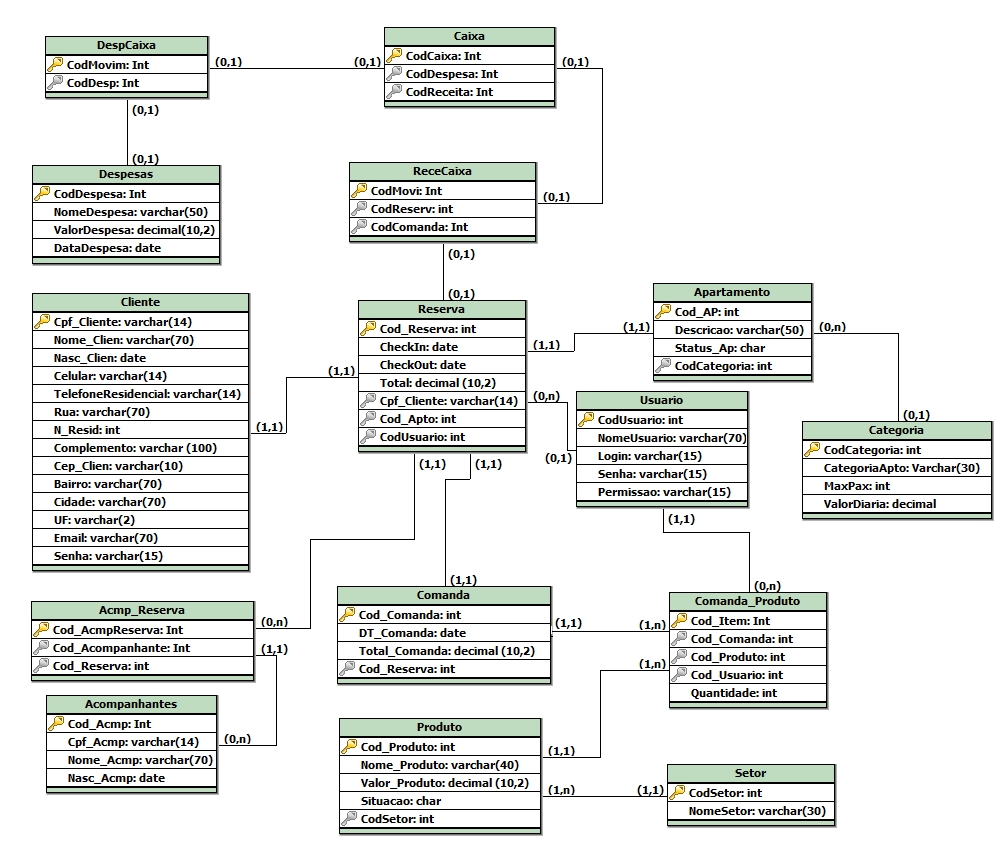
A modelagem foi baseada de uma forma micro, podendo ser expandida posteriormente e adequada para diversas categorias do ramo da hotelaria.

# Modelagem do Software:

## 8.1 DIAGRAMA DE ENTIDADE RELACIONAL



## 8.2 DIAGRAMA DE ENTIDADE RELACIONAL



## 8.3 DICIONÁRIO DE DADOS





## 8.4 MODELO FÍSICO (SCRIPT)

**DROP** **DATABASE** **IF** **EXISTS** hotel**;**

**CREATE** **DATABASE** hotel**;**

**USE** hotel**;**

**CREATE** **TABLE** Setor **(**

CodSetor **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

NomeSetor **varchar(**30**)** **not** **null**

**);**

**CREATE** **TABLE** Produto **(**

Cod\_Produto **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

Nome\_Produto **varchar(**40**)** **not** **null,**

Valor\_Produto **decimal** **(**10**,**2**)** **not** **null,**

Situacao **char** **not** **null,**

CodSetor **int** unsigned **not** **null,**

**Foreign** **key** **(**CodSetor**)** **References** Setor **(**CodSetor**)**

**);**

**CREATE** **TABLE** Acompanhantes **(**

Cod\_Acmp **int** unsigned **Primary** **Key** auto\_increment**,**

Cpf\_Acmp **varchar(**14**)** **unique** **not** **null,**

Nome\_Acmp **varchar(**70**)** **not** **null,**

Nasc\_Acmp **date** **not** **null**

**);**

**CREATE** **TABLE** Categoria **(**

CodCategoria **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

CategoriaApto **varchar(**30**)** **not** **null,**

MaxPax **int** unsigned **not** **null,**

ValorDiaria **decimal** **(**10**,**2**)** **not** **null**

**);**

**CREATE** **TABLE** Apartamento **(**

Cod\_AP **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

Descricao **varchar(**50**)** **not** **null,**

Status\_Ap **char** **not** **null,**

Cod\_Categoria **int** unsigned **not** **null,**

**Foreign** **key** **(**Cod\_Categoria**)** **References** Categoria **(**CodCategoria**)**

**);**

**CREATE** **TABLE** Usuario **(**

CodUsuario **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

NomeUsuario **varchar(**70**)** **not** **null,**

Login **varchar(**15**)** **not** **null** **unique,**

Senha **varchar(**15**)** **not** **null,**

Permissao **varchar(**15**)** **not** **null**

**);**

**CREATE** **TABLE** Cliente **(**

Cpf\_Cliente **varchar(**14**)** **PRIMARY** **KEY,**

Nome\_Clien **varchar(**70**)** **not** **null,**

Nasc\_Clien **date** **not** **null,**

Celular **varchar(**16**)** **not** **null,**

TelefoneResidencial **varchar(**16**)** **not** **null,**

Rua **varchar(**70**),**

N\_Resid **int** unsigned **not** **null,**

Complemento **varchar** **(**100**),**

Cep\_Clien **varchar(**10**)** **not** **null,**

Bairro **varchar(**70**),**

Cidade **varchar(**70**),**

UF **varchar(**2**),**

Email **varchar(**100**)** **unique,**

Senha **varchar(**15**)**

**);**

**CREATE** **TABLE** Reserva **(**

Cod\_Reserva **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

CheckIn **date** **not** **null,**

CheckOut **date** **not** **null,**

Total **decimal** **(**10**,**2**)** **not** **null,**

Cpf\_Cliente **varchar(**14**)** **not** **null,**

Cod\_Apto **int** unsigned **not** **null,**

CodUsuario **int** unsigned **not** **null,**

**Foreign** **key** **(**Cpf\_Cliente**)** **References** Cliente **(**Cpf\_Cliente**),**

**Foreign** **key** **(**Cod\_Apto**)** **References** Apartamento **(**Cod\_AP**),**

**Foreign** **key** **(**CodUsuario**)** **References** Usuario **(**CodUsuario**)**

**);**

**CREATE** **TABLE** Comanda **(**

Cod\_Comanda **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

DT\_Comanda **date** **not** **null,**

Total\_Comanda **decimal** **(**10**,**2**)** **not** **null,**

Cod\_Reserva **int** unsigned **,**

**FOREIGN** **KEY(**Cod\_Reserva**)** **REFERENCES** Reserva **(**Cod\_Reserva**)**

**);**

**CREATE** **TABLE** Despesas**(**

CodDespesa **int** unsigned **primary** **key** auto\_increment**,**

NomeDespesa **varchar** **(**50**)** **not** **null,**

ValorDespesa **decimal** **(**10**,**2**)** **not** **null,**

DataDespesa **date** **not** **null**

**);**

**CREATE** **TABLE** DespCaixa**(**

CodMovim **int** unsigned **primary** **key** auto\_increment**,**

CodDep **int** unsigned**,**

**Foreign** **key** **(**CodDep**)** **References** Despesas **(**CodDespesa**)**

**);**

**CREATE** **TABLE** ReceCaixa**(**

CodMovim **int** unsigned **primary** **key** auto\_increment**,**

CodReserva **int** unsigned**,**

CodComanda **int** unsigned**,**

**Foreign** **key** **(**CodReserva**)** **References** Reserva **(**Cod\_Reserva**),**

**Foreign** **key** **(**CodComanda**)** **References** Comanda **(**Cod\_Comanda**)**

**);**

**CREATE** **TABLE** Caixa**(**

CodCaixa **int** unsigned **primary** **key** auto\_increment**,**

CodDespesa **int** unsigned**,**

CodReceita **int** unsigned**,**

**Foreign** **key** **(**CodDespesa**)** **References** DespCaixa **(**CodMovim**),**

**Foreign** **key** **(**CodReceita**)** **References** ReceCaixa **(**CodMovim**)**

**);**

**CREATE** **TABLE** Comanda\_Produto **(**

Cod\_Item **int** unsigned **PRIMARY** **KEY** auto\_increment**,**

Cod\_Comanda **int** unsigned **not** **null,**

Cod\_Produto **int** unsigned **not** **null,**

Cod\_Usuario **int** unsigned **not** **null,**

Quantidade **int** **not** **null,**

**FOREIGN** **KEY(**Cod\_Comanda**)** **REFERENCES** Comanda **(**Cod\_Comanda**),**

**FOREIGN** **KEY(**Cod\_Produto**)** **REFERENCES** Produto **(**Cod\_Produto**),**

**FOREIGN** **KEY(**Cod\_Usuario**)** **REFERENCES** Usuario **(**CodUsuario**)**

**);**

**CREATE** **TABLE** Acmp\_Reserva **(**

Cod\_AcmpReserva **int** unsigned **primary** **key** auto\_increment**,**

Cod\_Reserva **int** unsigned **,**

Cod\_Acmp **int** unsigned **not** **null,**

**FOREIGN** **KEY(**Cod\_Reserva**)** **REFERENCES** Reserva **(**Cod\_Reserva**),**

**FOREIGN** **KEY(**Cod\_Acmp**)** **REFERENCES** Acompanhantes **(**Cod\_Acmp**)**

**);**

-- USER STORORED PROCEDURES -----------------------------------

-- CLIENTES-----------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_CadastrarCliente**(**

P\_Cpf\_Cliente **varchar(**14**),**

P\_Nome\_Clien **varchar(**70**),**

P\_Nasc\_Clien **date,**

P\_Celular **varchar(**16**),**

P\_TelefoneResidencial **varchar(**16**),**

P\_Rua **varchar(**70**),**

P\_N\_Resid **int,**

P\_Complemento **varchar** **(**100**),**

P\_Cep\_Clien **varchar(**10**),**

P\_Bairro **varchar(**70**),**

P\_Cidade **varchar(**70**),**

P\_UF **varchar(**2**),**

P\_Email **varchar(**100**),**

P\_Senha **varchar(**15**)**

**)**

**begin**

**insert** **into** Cliente **(**Cpf\_Cliente**,** Nome\_Clien**,** Nasc\_Clien**,** Celular**,** TelefoneResidencial**,** Rua**,** N\_Resid**,** Complemento**,** Cep\_Clien**,**

Bairro**,** Cidade**,** UF**,** Email**,** Senha **)**

**Values** **(**P\_Cpf\_Cliente**,** P\_Nome\_Clien**,** P\_Nasc\_Clien**,** P\_Celular**,** P\_TelefoneResidencial**,** P\_Rua**,** P\_N\_Resid**,** P\_Complemento**,** P\_Cep\_Clien**,**

P\_Bairro**,** P\_Cidade**,** P\_UF**,** P\_Email**,** P\_Senha**);**

**END**$$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_AlteraCliente**(**

P\_Cpf\_Cliente **varchar(**14**),**

P\_Nome\_Clien **varchar(**70**),**

P\_Nasc\_Clien **date,**

P\_Celular **varchar(**14**),**

P\_TelefoneResidencial **varchar(**14**),**

P\_Rua **varchar(**70**),**

P\_N\_Resid **int,**

P\_Complemento **varchar** **(**100**),**

P\_Cep\_Clien **varchar(**10**),**

P\_Bairro **varchar(**70**),**

P\_Cidade **varchar(**70**),**

P\_UF **varchar(**2**),**

P\_Email **varchar(**100**),**

P\_Senha **varchar(**15**),**

p\_Cpf\_ClienRecebido **varchar(**14**)**

**)**

**begin**

**update** Cliente **set** Cpf\_Cliente **=** P\_Cpf\_Cliente**,** Nome\_Clien **=** P\_Nome\_Clien**,** Nasc\_Clien **=** P\_Nasc\_Clien**,** Celular **=** P\_Celular**,**

TelefoneResidencial **=** P\_TelefoneResidencial**,** Rua **=** P\_Rua**,** N\_Resid **=** P\_N\_Resid**,** Complemento **=** P\_Complemento**,** Cep\_Clien **=** P\_Cep\_Clien**,**

Bairro **=** P\_Bairro**,** Cidade **=** P\_Cidade**,** UF **=** P\_UF**,** Email **=** P\_Email**,** Senha **=** P\_Senha **where** Cpf\_Clien **=** p\_Cpf\_ClienRecebido**;**

**END**$$

DELIMITER **;**

-- Categoria -----------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_InsereCategoria**(**

P\_CategoriaApto **varchar(**30**)** **,**

P\_MaxPax **int** unsigned**,**

P\_ValorDiaria **decimal** **(**10**,**2**)**

**)**

**begin**

**Insert** **into** Categoria **(**CategoriaApto**,** MaxPax**,** ValorDiaria **)** **values** **(**P\_CategoriaApto**,** P\_MaxPax**,** P\_ValorDiaria**);**

**END**$$

DELIMITER **;**

-- ------------------------------------------

Delimiter $$

**create** **procedure** JAVA\_USP\_AlteraCategoria**(**

P\_CodCategoria **int** unsigned**,**

P\_CategoriaApto **varchar(**30**),**

P\_MaxPax **int** unsigned**,**

P\_ValorDiaria **decimal** **(**10**,**2**)**

**)**

**begin**

**update** Categoria **set** CategoriaApto **=** P\_CategoriaApartamento**,** MaxPax **=** P\_MaxPax**,** ValorDiaria **=** P\_ValorDiaria **where** CodCategoria **=** P\_CodCategoria**;**

**END**$$

Delimiter**;**

-- APARTAMENTO -----------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_InsereApartamento**(**

P\_Descricao **varchar(**50**),**

P\_Status **char,**

P\_CodCategoria **int**

**)**

**begin**

**Insert** **into** apartamento **(**Descricao**,** Status\_Ap**,** Cod\_Categoria **)** **values** **(** P\_Descricao**,** P\_Status**,**P\_CodCategoria**);**

**END**$$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_AlteraApartamento**(**

P\_Cod\_AP **int,**

P\_Descricao **varchar(**50**),**

P\_Status **char,**

P\_CodCategoria **int**

**)**

**begin**

**update** apartamento **set** Descricao **=** Descricao**,** Status\_Ap **=** P\_Status **where** Cod\_AP **=** P\_Cod\_AP **;**

**END**$$

DELIMITER **;**

-- SETOR ---------------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_InsereSetor**(**

P\_NomeSetor **varchar(**30**)**

**)**

**begin**

**Insert** **into** Setor **(**NomeSetor**)** **values** **(**P\_NomeSetor**);**

**END**$$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_AlteraSetor**(**

P\_CodSetor **int** unsigned**,**

P\_NomeSetor **varchar(**30**)**

**)**

**begin**

**update** Setor **set** NomeSetor **=** P\_NomeSetor **where** CodSetor **=** P\_CodSetor **;**

**END**$$

DELIMITER **;**

-- PRODUTOS-----------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_InsereProduto**(**

P\_Nome\_Produto **varchar(**40**),**

P\_Valor\_Produto **decimal** **(**10**,**2**)** **,**

P\_Situacao **char,**

P\_CodSetor **int** unsigned

**)**

**begin**

**Insert** **into** Produto **(**Nome\_Produto**,** Valor\_Produto**,** Situacao**,** CodSetor**)** **Values** **(**P\_Nome\_Produto**,** P\_Valor\_Produto**,** P\_Situacao**,** P\_CodSetor**);**

**END**$$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_AlteraProduto**(**

P\_Cod\_Produto **int** unsigned **,**

P\_Nome\_Produto **varchar(**40**),**

P\_Valor\_Produto **decimal** **(**10**,**2**)** **,**

P\_Situacao **char,**

P\_CodSetor **int** unsigned

**)**

**begin**

**update** Produto **set** Nome\_Produto **=** P\_Nome\_Produto**,** Valor\_Produto **=** P\_Valor\_Produto**,** Situacao **=** P\_Situacao**,** CodSetor **=** P\_CodSetor

**where** Cod\_Produto **=** P\_Cod\_Produto**;**

**END**$$

DELIMITER **;**

-- RESERVAS e Abre COMANDAS--------------------------

DELIMITER $$

**create** **procedure** JAVA\_USP\_INSERERESERVACOMANDA**(**

P\_CheckIn **date** **,**

P\_CheckOut **date** **,**

P\_Total **decimal** **(**10**,**2**)** **,**

P\_Cpf\_Cliente **varchar(**14**)** **,**

P\_Cod\_Apto **int** unsigned **,**

P\_CodUsuario **int** unsigned **,**

Total\_Comanda **decimal** **(**10**,**2**)**

**)**

**begin**

**START** **TRANSACTION;**

**insert** **into** Reserva **(**CheckIn**,** CheckOut**,** Total**,** Cpf\_Cliente**,** Cod\_Apto**,** CodUsuario**)**

**values** **(**P\_CheckIn**,** P\_CheckOut**,** P\_Total**,** P\_Cpf\_Cliente**,** P\_Cod\_Apto**,** P\_CodUsuario**);**

**insert** **into** comanda **(**DT\_Comanda**,** Total\_Comanda**,** Cod\_Reserva **)**

**values** **(**P\_CheckIn**,** Total\_Comanda**,** LAST\_INSERT\_ID**());**

**COMMIT;**

**END**$$

DELIMITER **;**

-- ACOMPANHANTES-----------------------------------

DELIMITER $$

**create** **procedure** USP\_JAVA\_INSEREACOMPANHANTE**(**

P\_Cpf\_Acmp **varchar(**14**),**

P\_Nome\_Acmp **varchar(**70**)** **,**

P\_Nasc\_Acmp **date**

**)**

**begin**

**insert** **into** Acompanhantes **(**Cpf\_Acmp**,** Nome\_Acmp**,** Nasc\_Acmp**)** **Values** **(**P\_Cpf\_Acmp**,** P\_Nome\_Acmp**,** P\_Nasc\_Acmp**);**

**END** $$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**create** **procedure** USP\_JAVA\_ALTERAACOMPANHANTE**(**

P\_Cpf\_Acmp **varchar(**14**),**

P\_Nome\_Acmp **varchar(**70**)** **,**

P\_Nasc\_Acmp **date** **,**

P\_Cod\_Acmp **int**

**)**

**begin**

**update** Acompanhantes **set** Cpf\_Acmp **=** P\_Cpf\_Acmp**,** Nome\_Acmp **=** P\_Nome\_Acmp**,** Nasc\_Acmp **=** P\_Nasc\_Acmp **where** Cod\_Acmp **=** P\_Cod\_Acmp **;**

**END** $$

DELIMITER **;**

-- USUARIO-------------------------------------

DELIMITER $$

**Create** **Procedure** USP\_JAVA\_InsereUsuario**(**

P\_NomeUsuario **varchar(**70**),**

P\_Login **varchar(**15**),**

P\_Senha **varchar(**15**),**

P\_Permissao **varchar(**15**)**

**)**

**begin**

**insert** **into** Usuario **(**NomeUsuario**,** Login**,** Senha**,** Permissao **)** **Values** **(**P\_NomeUsuario**,** P\_Login**,** P\_Senha**,** P\_Permissao**);**

**end**$$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**Create** **Procedure** USP\_JAVA\_AlteraUsuario**(**

P\_NomeUsuario **varchar(**70**),**

P\_Login **varchar(**15**),**

P\_Senha **varchar(**15**),**

P\_Permissao **varchar(**15**),**

P\_CodUsuario **int**

**)**

**begin**

**update** Usuario **set** NomeUsuario **=** P\_NomeUsuario**,** Login **=** P\_Login**,** Senha **=** P\_Senha**,** Permissao **=** P\_Permissao **where** CodUsuario **=** P\_CodUsuario**;**

**end**$$

DELIMITER **;**

-- ACMP\_RESERVA--------------------------------

DELIMITER $$

**Create** **Procedure** USP\_JAVA\_InsereAcomp\_Reserva**(**

P\_Cod\_Reserva **int** **,**

P\_Cod\_Acmp **int**

**)**

**begin**

**insert** **into** Acmp\_Reserva **(**Cod\_Reserva**,** Cod\_Acmp**)** **Values** **(**P\_Cod\_Reserva**,** P\_Cod\_Acmp**);**

**END** $$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**Create** **Procedure** USP\_JAVA\_DeletaAcomp\_Reserva**(**

P\_Cod\_AcmpReserva **int**

**)**

**begin**

**delete** **from** Acmp\_Reserva **where** Cod\_AcmpReserva **=** P\_Cod\_AcmpReserva**;**

**END** $$

DELIMITER **;**

-- Comanda\_Produto--------------------------------

DELIMITER $$

**Create** **Procedure** USP\_JAVA\_InsereComanda\_Produto**(**

P\_Cod\_Comanda **int,**

P\_Cod\_Produto **int,**

P\_Cod\_Usuario **int,**

P\_Quantidade **int**

**)**

**begin**

**insert** **into** Comanda\_Produto **(**Cod\_Comanda**,** Cod\_Produto**,** Cod\_Usuario**,** Quantidade**)** **VALUES** **(**P\_Cod\_Comanda**,** P\_Cod\_Produto**,** P\_Cod\_Usuario**,** P\_Quantidade**);**

**END** $$

DELIMITER **;**

-- ------------------------------------------

DELIMITER $$

**Create** **Procedure** USP\_JAVA\_DeletaComanda\_Produto**(**

P\_Cod\_Item **int**

**)**

**begin**

**delete** **from** Comanda\_Produto **where** Cod\_Item **=** P\_Cod\_Item **;**

**END** $$

DELIMITER **;**

-- INSERTS-------------------------------------------

**call** JAVA\_USP\_CadastrarCliente**(**'399.305.868-22'**,** 'Alex Santos'**,** '1993-01-19'**,** '(11) 96695-3355'**,** '(11) 2258-2212'**,**'Marconi'**,** 107**,** ''**,**

'02.645-000'**,** 'Centro'**,** 'São Paulo'**,** 'SP'**,** 'alexsantosinformatica@gmail.com'**,**'senha123' **);**

**call** JAVA\_USP\_InsereCategoria **(**'Single'**,** 1**,** 100.00**);**

**call** JAVA\_USP\_InsereCategoria **(**'Duplo'**,** 2**,** 200.00**);**

**call** JAVA\_USP\_InsereCategoria **(**'Triplo'**,** 3**,** 300.00**);**

**call** JAVA\_USP\_InsereCategoria **(**'Quádruplo'**,** 4**,** 400.00**);**

**call** JAVA\_USP\_InsereCategoria **(**'Quíntuplo'**,** 5**,** 500.00**);**

**call** JAVA\_USP\_InsereCategoria **(**'Especial'**,** 6**,** 600.00**);**

**call** JAVA\_USP\_InsereApartamento **(**'Single com vista para o Mar'**,** 'D'**,** 1**);**

**call** JAVA\_USP\_InsereApartamento **(**'Duplo com vista para o Mar'**,** 'D'**,** 2**);**

**call** JAVA\_USP\_InsereApartamento **(**'Triplo com vista para o Mar'**,** 'M'**,** 3**);**

**call** JAVA\_USP\_InsereApartamento **(**'Quádruplo com vista para o Mar'**,** 'D'**,** 4**);**

**call** JAVA\_USP\_InsereApartamento **(**'Quíntuplo com vista para o Mar'**,** 'D'**,** 5**);**

**call** JAVA\_USP\_InsereApartamento **(**'Especial com vista para o Mar'**,** 'D'**,** 6**);**

**call** JAVA\_USP\_InsereSetor **(**'Bebidas'**);**

**call** JAVA\_USP\_InsereSetor **(**'Cozinha'**);**

**call** JAVA\_USP\_InsereSetor **(**'Mercadinho'**);**

**call** JAVA\_USP\_InsereSetor **(**'Serviços'**);**

**call** JAVA\_USP\_InsereProduto **(**'Coca Cola'**,** 3.50**,** 'A'**,** 1**);**

**call** JAVA\_USP\_InsereProduto**(**'Prato Executivo'**,** 14.00**,** 'A'**,** 2**);**

**call** JAVA\_USP\_InsereProduto**(**'Escova de Dente'**,** 2.00**,** 'A'**,** 3**);**

**call** JAVA\_USP\_InsereProduto**(**'Massagem'**,** 35.00**,** 'A'**,** 4**);**

**call** JAVA\_USP\_INSERERESERVACOMANDA**(**'2000-01-01'**,**'2000-01-01'**,** 0.00**,** '399.305.868-22'**,** 1**,** 1**,** 0.00**);**

**call** USP\_JAVA\_INSEREACOMPANHANTE **(**'350.007.518-50'**,** 'Flávia Santos'**,** '1986-04-18'**);**

**call** USP\_JAVA\_InsereUsuario **(**'Alison Henrique'**,** 'Alison'**,** '123'**,** 'Adm'**);**

**call** USP\_JAVA\_InsereAcomp\_Reserva**(**1**,**1**);**

**call** USP\_JAVA\_InsereComanda\_Produto **(**1**,**1**,**1**,**3**);**

-- ---------

# Sistema:

## 9.1 Logotipo:



## 9.2 Login:

## C:\Users\Alex\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Login.png

## 9.3 Home:

## 

## 9.4 Cadastro de Cliente:

## 

## 9.6 Reserva:

## 

# 10 Referências Bibliográficas:

DEVMEDIA.

<http://www.devmedia.com.br/bancos-de-dados-orientados-a-objetos-sql-magazine-78/17717>

BATEBYTE.

[*http://www.batebyte.pr.gov.br/modules/conteudo/conteudo.php?conteudo=560*](http://www.batebyte.pr.gov.br/modules/conteudo/conteudo.php?conteudo=560)

SCRIBD

<http://www.pt.scribd.com/document/33325995/Banco-de-Dados-Orientado-a-Objetos>

WIKIPÉDIA

<https://pt.wikipedia.org/wiki/SQL>

DATASCIENCEACADEMY

<http://datascienceacademy.com.br/blog/2016/quando-utilizar-rdbms-ou-nosql/>

SLIDESHARE

<http://pt.slideshare.net/alexculpado/jose-alexandrerdbm-sxnosql>

DOCPLAYER

<http://docplayer.com.br/1152433-Carlos-filipe-marques-teixeira-junior-guilherme-rangel-ferreira-relacionais-e-nosql-para-dados-de-proveniencia-em-workflows-cientificos.html>

SLIDESHARE

<http://pt.slideshare.net/Celio12/trabalho-no-sql-aricelio-de-souza>

SLIDEPLAYER

<http://slideplayer.com.br/slide/295205/>

IMASTERS

<http://imasters.com.br/banco-de-dados/graphdb-series-o-que-e-um-banco-de-dados-de-grafos/?trace=1519021197&source=single>