UNIVERSIDADE DO OESTE DE SANTA CATARINA – UNOESC
CAMPUS DE SÃO MIGUEL DO OESTE

APOSTILA DE BANCO DE DADOS

Autoria: Prof. Esp. Roberson Junior Fernandes Alves
Home-page: http://www.unoescsmo.edu.br/~roberson

São Miguel do Oeste(SC), 2010.

SUMÁRIO

1.INTRODUÇÃO

Na década de 90 as organizações dão-se conta que a Informação é um “recurso organizacional”. No entanto, o valor da informação depende de sua correção e disponibilidade, apenas possível com uma gestão adequada.

Conforme Toffler e Toffler(1994, p. 51) “o valor real de companhias como a Compaq ou a Kodak, a Hitachi ou a Siemens, depende mais das ideias, percepções e informações contidas nas cabeças de seus empregados, e nos banco de dados e patentes que estas companhias controlam, do que nos caminhões, linhas de montagem ou outros ativos físicos. [...] o próprio capital está cada vez mais baseado em intangíveis”.

As tecnologias de banco de dados são uma ferramenta essencial à gestão eficaz do recurso chamado *INFORMAÇÃO*.

Dados versus Informação: entendendo as diferenças.

- **Dados** representam fatos do mundo real. Valor de um campo armazenado, matéria-prima para obtenção de informação;
    - *Exemplo: nome, endereço, telefone, e-mail.*
- A **Informação** resulta do processamento dos “dados”, apresentados de forma a permitir interpretação e a fundamentar decisões;
    - *Exemplo: um contato de uma pessoa em uma agenda.*

2.SISTEMAS BASEADOS EM ARQUIVOS

Os primeiros modelos de sistemas, predecessores dos bancos de dados, trabalhavam sobre uma estrutura baseada em arquivos. Em sistemas baseados em arquivos existem vários programas acessando vários arquivos de dados diferentes. Cada programa é responsável por alguma tarefa específica destinada ao usuário final, além disso cada programa define e gerencia seus próprios dados(CONNOLLY; BEGG, 2005). Nesta estrutura não havia um software que gerenciava de forma centralizada o acesso e o tratamento dos dados, por isso estas implementações era feitas em cada programa separadamente a critério do programador.

Alguns dos principais conceitos relacionados a ambientes baseados em arquivos são apresentados no quadro 1:

Conceito	Descrição
Arquivo	Arquivo de dados
Registro	O mesmo que linha ou tupla
Campo	O mesmo que coluna ou atributo

Quadro 1 – Conceitos do ambiente baseado em arquivos.

Fonte: Adaptado de Connolly e Begg(2005).

Um *arquivo* nesta estrutura é uma composição de *registros* que armazenam dados. Cada registro possui um conjunto de *campos*, conectados logicamente.

A figura 1, a seguir, representa de forma geral o funcionamento de sistemas baseados em arquivos:

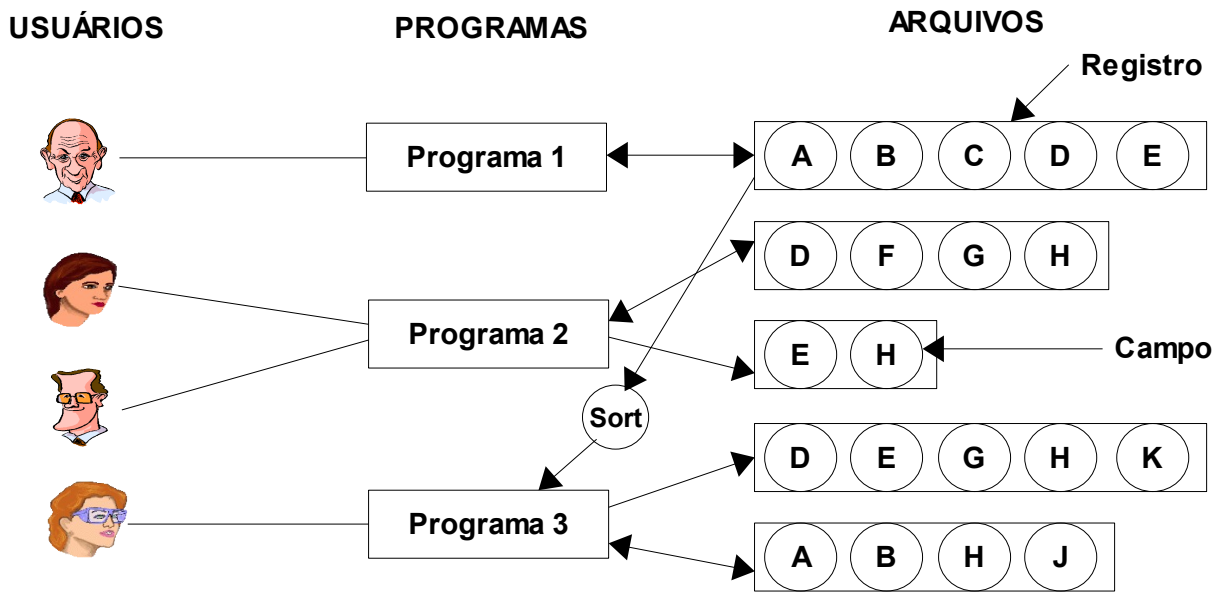


Figura 1 – Funcionamento de um sistema baseado em arquivos.

Os problemas mais sérios apresentados pelos sistemas baseados em arquivos eram:

- *Redundância e inconsistência de dados:* Redundância é a característica onde um elemento de informação aparece duplicado em diversos lugares. Por exemplo, um cliente de uma empresa pode estar cadastrado tanto na base de dados do departamento de crédito assim como no departamento de contabilidade. Esta redundância pode causar a inconsistência, visto que as informações podem ser diferentes para o mesmo cliente;
- *Dificuldade no acesso aos dados:* Suponha que um diretor necessite encontrar os clientes que residem em uma área da cidade onde o CEP seja 12133-433. Se este módulo do sistema de informação não estiver implementado, o diretor terá duas opções: ele pega a lista de clientes e extrai a informação necessária, ou pede para o departamento de informática implementar a rotina necessária para obtenção do relatório;
- *Isolamento de dados:* Uma vez que os dados estão espalhados em diversos arquivos e podem ter formatos diferentes, é difícil escrever novos programas/aplicativos para recuperar os dados adequados;
- *Anomalia de acesso concorrente:* Evitar problemas relacionados a acessos concorrentes em um determinado dado. Por exemplo, se duas pessoas sacarem dinheiro

de uma conta ao mesmo tempo, o resultado das operações concorrentes podem deixar um saldo inconsistente. Considere uma conta com a quantia de R$ 400 e dois saques simultâneo de R$ 100 e R$ 50. Se houver problema de acesso concorrente, a conta pode ficar com valores de R$ 300 ou de R$ 350 ao invés de R$ 250;

- *Segurança:* Cada usuário do sistema poderá ter acesso aos dados relativos a sua função. Por exemplo, um caixa não pode ver dados pertencentes a diretoria;
- *Problemas de Integridade:* Os valores armazenados devem satisfazer certos tipos de restrições de consistência. Por exemplo, um filho de um funcionário não pode ser cadastrado sem que o pai trabalhe na empresa.

Como evolução e para resolver a maior parte destes problemas surgem os bancos de dados.

3.BANCO DE DADOS

Os bancos de dados já são parte essencial da sociedade moderna. Em algum momento em nossas atividades diárias nos deparamos com situações que necessitam de alguma interação com um banco de dados. Como exemplos pode-se citar: quando realizamos o empréstimo de um livro na biblioteca, quando realizamos uma reserva em um hotel, transações bancárias como saques ou depósitos ou mesmo realizar uma compra em uma loja. Estes são alguns exemplos de aplicações que podem ser consideradas tradicionais, onde a maior parte das informações se apresenta, ou em forma de números ou em forma textual.

Com a constante evolução tecnológica novas e interessantes áreas de aplicação tem surgido para banco de dados, como: no armazenamento de imagens, vídeos, sons; informações geográficas como mapas ou imagens de satélite; entre outras.

Buscando definir o conceito de banco de dados, dentre as diversas existentes, **pode se dizer que é um conjunto integrado de dados e/ou elementos de informação que serão compartilhados, e utilizados concorrentemente por múltiplos usuários e/ou programas, para múltiplos objetivos, e com diferentes perspectivas.**

Outra definição interessante, conforme Connolly e Begg(2005), **coleção de dados logicamente relacionados com sua respectiva descrição, desenhados para atender as necessidades da organização.**

Uma das grandes novidades trazida pela tecnologia de Banco de Dados é o que chamamos de: *Data Independence* ou Independência entre Dados e Programas. Para entender melhor vamos conhecer alguns conceitos importantes:

a) *Independência Física:* Do ponto de vista físico, os programas são independentes da estrutura física da informação, ou seja, a estrutura de armazenamento da informação. Alterações na estrutura física dos dados (armazenamento) não implicam

alterações nos programas.

b) *Independência Lógica:* Do ponto de vista lógico, os programas são independentes da estrutura lógica da informação, ou seja, a forma como a informação está organizada. Alterações na estrutura lógica da base de dados(inclusão de novos atributos, novas estruturas) não implicam necessariamente em alterações nos programas.

c) *Integridade:* Os programas são independentes das regras de integridade(coerência) em vigor no banco de dados. Alterações nessas regras de integridade não devem ter implicações nos programas que utilizam a informação.

d) *Distribuição:* Os programas são independentes da localização geográfica da informação. Alterações na localização geográfica da informação e na sua estrutura de distribuição não devem implicar em alterações nos programas.

Algumas grandes vantagens trazidas pelos bancos de dados são:

- *Diminuição de redundância:* é aumentada a integridade da informação e reduzido o espaço ocupado;
- *Compartilhamento de dados:* a ideia é centralizar a informação e permitir o compartilhamento das informações de forma fácil e transparente;
- *Flexibilidade na modificação e manutenção:* possui recursos específicos para manutenção da estrutura interna do banco de dados;
- *Linguagem de interrogação:* introduz o conceito de linguagem de interrogação que permite de forma simples a recuperação e manipulação das informações;
- *Controle centralizado dos dados:* normas, regras de integridade, políticas de segurança e métodos de recuperação residem em um mesmo local.

Algumas operações básicas possíveis de serem realizadas em um banco de dados:

- Adicionar novos arquivos;
- Buscar dados;
- Inserir dados nos arquivos existentes;
- Alterar dados em arquivos existentes;
- Eliminar dados em arquivos existentes;
- Remover arquivos existentes.

No entanto, algumas contrapartidas também foram provenientes do surgimento dos bancos de dados:

- Criação de pontos de conflito que surgiram através do compartilhamento dos dados;
- Necessidade de especialização tecnológica e profissional;
- Criação de motivos para “conflito organizacional”.

Os bancos de dados podem atuar em diferentes níveis. A seguir(figura 2) são apresentados alguns tipos de banco de dados e os níveis de atuação em que podemos encontrá-los:

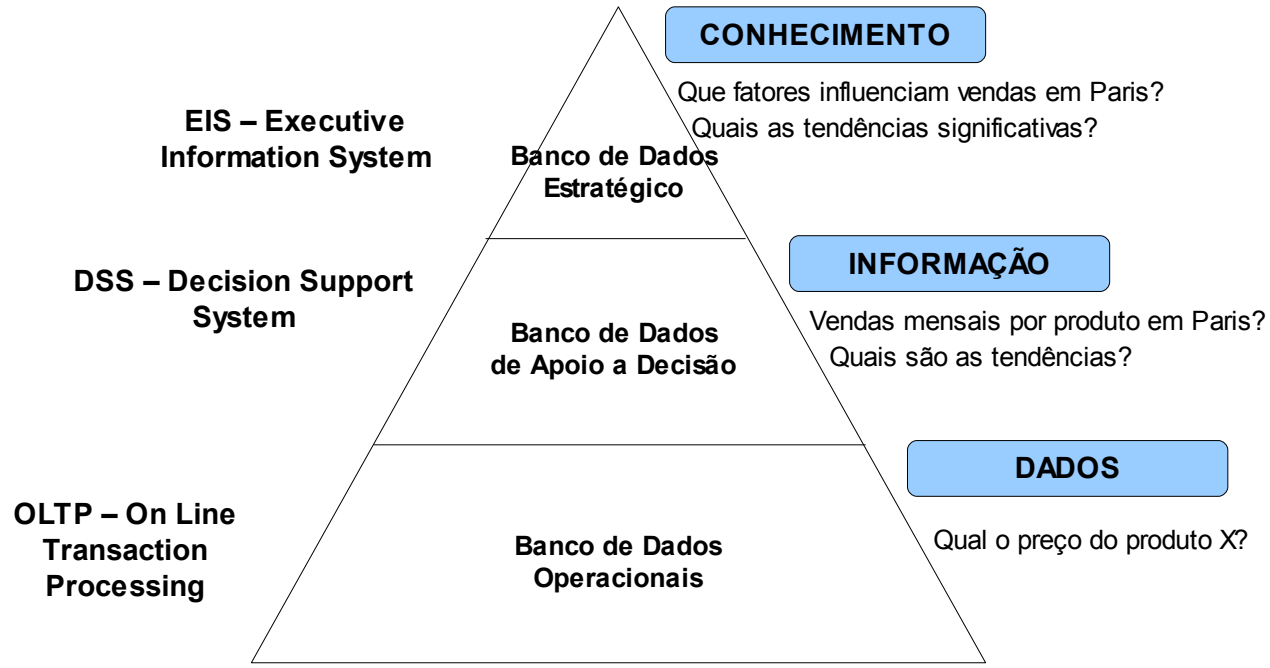


Figura 2 – Tipos de banco de dados e níveis de atuação.
Fonte: Adaptado de Yourdon(1990).

Um dos maiores benefícios dos bancos de dados: promover uma visão abstrata dos dados, ou seja, ocultando detalhes dos mecanismos de armazenamento e manutenção dos dados. Três níveis de abstração principais podem ser percebidos:

- *Nível de visão externo:* Mais alto nível de abstração. Visão de cada usuário. Descreve apenas parte do banco de dados que um dado grupo de usuários tem interesse.
- *Nível lógico(conceitual):* Nível médio de abstração. Visão da comunidade de usuários. Quais dados estão armazenados no banco de dados, e quais os inter-relacionamentos entre eles.
- *Nível físico:* Mais baixo nível de abstração. Como os dados estão de fato armazenados no SGBD.

Para gerenciar os diversos serviços de um banco de dados surgiram os Sistemas Gerenciadores de Banco de Dados(SGBDs).

3.1.SISTEMA GERENCIADOR DE BANCO DE DADOS(SGBD)

Sistema de gerenciamento do banco de dados ou SGBD, também conhecido como DBMS(*DataBase Management System*). O SGBD é o produto de software responsável pelo

gerenciamento do banco de dados.

Seus principais componentes são(figura 3): equipes de desenvolvimento, administradores de dados(DAs), administradores de banco de dados(DBAs), usuários, ferramentas case, interfaces para interação com o usuário, aplicações, dicionário, catálogo e o base de dados em si.

Entendendo melhor os componentes de um SGBD:

- *Catálogo:* contém a definição do banco de dados, descrita em forma de um meta-modelo de dados(específico para cada SGBD), utilizando o mesmo modelo lógico que o usado para a concepção da base de dados. Exemplo: Em um SGBD relacional, o catálogo contém as definições das tabelas, visões, regras de integridades, etc. O catálogo é por vezes designado como “Dicionário de Dados”.
- *Interface de interação com o usuário:* define o modo de interação com a informação contida em um banco de dados. Tal como em um ambiente baseado em arquivos, através de programas escritos especificamente para o efeito. Mas agrega ainda: Linguagens de pesquisa(*Query languages*) e outras interfaces para administração do banco de dados.
- *Administração de Dados:* é a função responsável pela gestão global dos recursos informacionais de uma organização, incluindo a definição e manutenção de conceitos e normas relativas ao(s) modelo(s) de dados/informação da organização.
- *Administração de Banco de Dados:* é a função técnica responsável pelo desenho lógico (em colaboração com as equipes de desenvolvimento) e físico do banco de dados. São ainda da sua responsabilidade a implementação de mecanismos de controle de segurança e de recuperação da base de dados, além da monitoramento do desempenho.
- *Ferramentas Case(Computer Aided Software Engineering)*: é uma aproximação disciplinada e estruturada ao desenvolvimento de sistemas apoiadas por computador.
    - As ferramentas case oferecem:
        - Maior produtividade
        - Melhor qualidade
        - *Reverse engineering* – engenharia reversa
        - Melhor planejamento e controle
        - A criação de um ambiente de desenvolvimento
        - Em todas as fases de desenvolvimento de Sistemas de Informação podemos recorrer a ferramentas CASE.
        - *Upper case*: usadas nas fases de análise e desenvolvimento.
        - *Lower case*: usadas nas fases de concepção (desenho) e construção.

Dados os requisitos de compatibilidade entre as ferramentas usadas nas várias fases, é necessária integração: I-CASE - *Integrated Case*.

Existem várias formas de integração, entre elas:

- Ferramentas de um único fornecedor, cada uma suportando uma atividade;
- Ferramentas heterogêneas que se comunicam através de interfaces e *bridges;*
- Ferramentas heterogêneas que partilham um repositório comum contendo todas as definições, modelos, diagramas, etc;

Funcionalidades importantes de um ambiente *case* integrado:

- O código é gerado automaticamente(*free-bug*);
- Parte do desenho pode ser gerada;
- A consistência dos modelos é assegurada;
- Existe consistência entre o trabalho de cada elemento da equipe;
- A documentação é gerada automaticamente;
- A manutenção é feita por re-geração.

No quadro a seguir são apresentadas algumas ferramentas CASE encontradas no mercado hoje:

Ferramenta	Finalidade	Licença	Link
Oracle Designer	Modelagem de banco de dados	Shareware	http://www.oracle.com
DbDesigner Fork	Modelagem de banco de dados	Free	http://sourceforge.net/projects/dbdesigner-fork/
System Architect	Modelagem de banco de dados	Shareware	http://www.popkin.com/
Microsoft Visio	Modelagem de diversos tipos de diagramas incluindo modelagem de banco de dados e UML	Shareware	http://www.microsoft.com/
Design for Databases	Modelagem de banco de dados	Shareware	http://www.datanamic.com/dezign/index.html
IBM Rational	Ferramenta completa para análise, modelagem, design e construção de aplicações	Shareware	http://www-01.ibm.com/software/br/rational/
Poseidon	Modelagem UML	Shareware	http://www.gentleware.com/
Jude Community	Modelagem UML	Free	http://jude.change-vision.com/jude-web/index.html

StarUML	Modelagem UML	Free	http://staruml.sourceforge.net/
ArgoUML	Modelagem UML	Free - Open Source	http://www.tigris.org/
Mogwai ER Designer	Modelagem de banco de dados	Open Source	http://mogwai.sourceforge.net/?Welcome:ERDesigner_NG
Dia	Similar ao Microsoft Visio. Permite desenhar diagramas incluindo modelagem de banco de dados	Free - Contributions suggested	http://www.gnome.org/projects/dia/
ER/Studio Data Architect	Modelagem de banco de dados	Shareware	http://www.embarcadero.com/products/er-studio-data-architect

- *Dicionário:* base de dados que contém os modelos desenvolvidos usando a ferramenta CASE. Constitui a base para a integração dos modelos construídos durante o ciclo de desenvolvimento de um projeto, sendo a "fonte de informação" que serve de *input* à operação de definição da base de dados e do código das aplicações.

A figura 3, a seguir, ilustra os componentes de um SGBD e a interação entre eles.

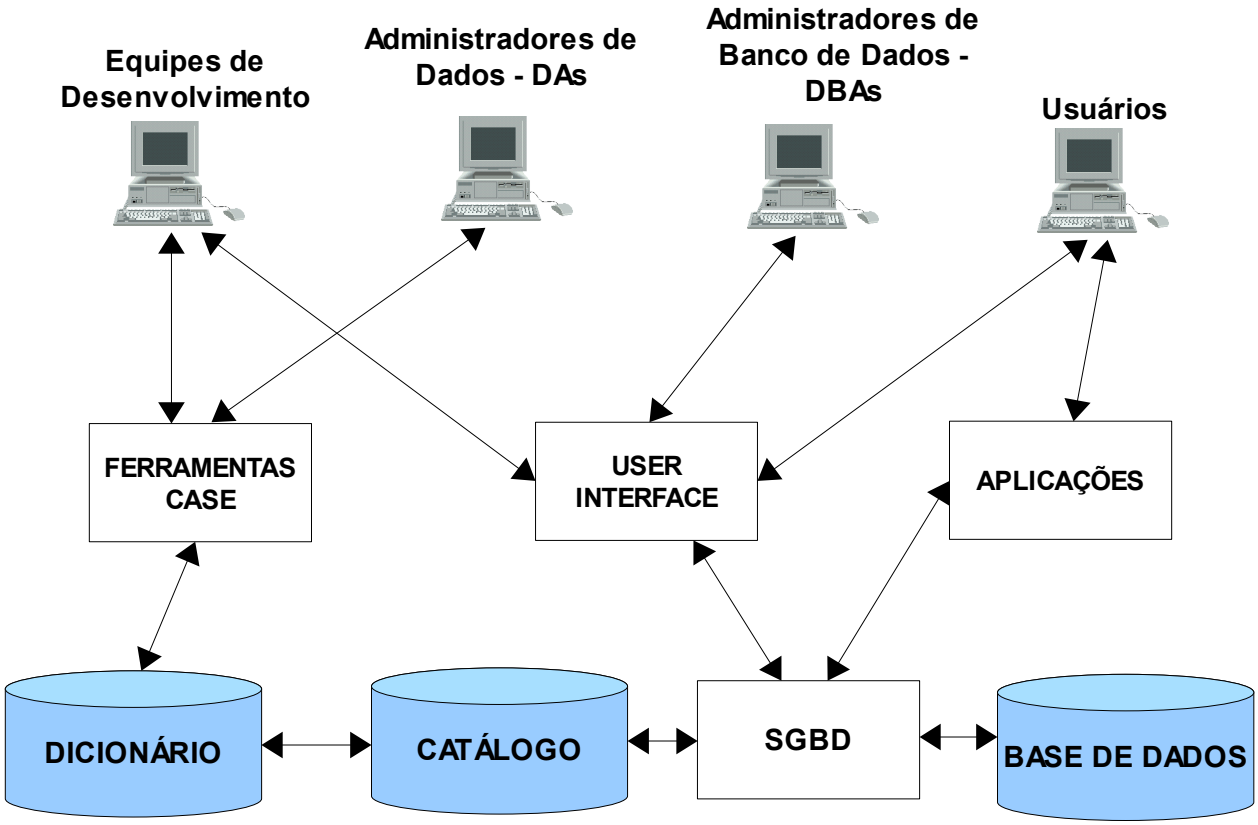


Figura 3 – Componentes de um banco de dados.

Os SGDBS devem apresentar algumas características fundamentais, dentre elas:

- Basear-se em um modelo lógico de dados(rede, hierárquico, relacional, orientados a objetos);
- Gerenciar:
  - Integridade: prover mecanismos para garantir a integridade e o armazenamento dos dados;
  - Transações: permitir um controle eficaz de transações a fim de evitar a entrada de dados inconsistentes;
  - Concorrência: o SGBD já surgiu com a ideia de poder ser acessado por múltiplos usuários simultaneamente, portanto, cabe a ele gerenciar estes acessos concorrentes;
  - Recuperação: garantir a possibilidade de recuperação das informações, seja por controles de tolerância a falha ou mecanismos de *backup*;
  - Disponibilidade: garantir o mínimo de disponibilidade possível para manter o sistema estar *on-line* quando necessário;
  - Controle de acessos: permitir o controle de acesso.
- Assegurar um desempenho adequado.

Exemplos de alguns SGBDs(quadro 2) que podem ser encontrados sob licenças comerciais ou livres:

Comercial	Livre
DB2 - http://www-306.ibm.com/software/data/db2/	MySQL - http://www.mysql.com
Informix - http://www-306.ibm.com/software/data/informix/	Firebird – http://www.firebirdsql.org
Interbase - http://www.codegear.com/br/products/interbase	PostgreSQL – http://www.postgresql.org
Oracle – http://www.oracle.com	Cache - http://www.intersystems.com/cache/index.html
Sybase - http://www.sybase.com.br	HSQLDB - http://hsqldb.org
MsSQL Server - http://www.microsoft.com/brasil/servidores/sql/default.mspx	

Quadro 2 – Alguns SGBDs comerciais e/ou livres.
Fonte: Sites internet.

A seguir são apresentados os modelos lógicos de banco de dados.

3.2.MODELOS LÓGICOS DE BANCO DE DADOS

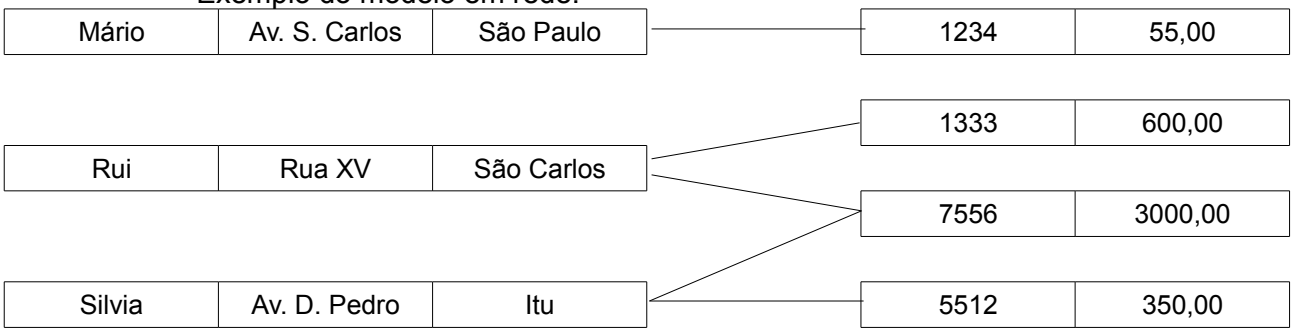
Os modelos lógicos são utilizados para descrever os dados no nível lógico ou conceitual. Alguns modelos lógicos conhecidos:

a) *Modelos Lógicos baseados em registros:* rede, hierárquico e relacional. Nestes modelos o banco de dados é estruturado através de registros de formato fixo de todos os tipos. Possui um número fixo de campos(atributos), e cada campo possui tamanho fixo simplificando a implementação do banco de dados no nível físico.

Descrição dos modelos baseados em registro:

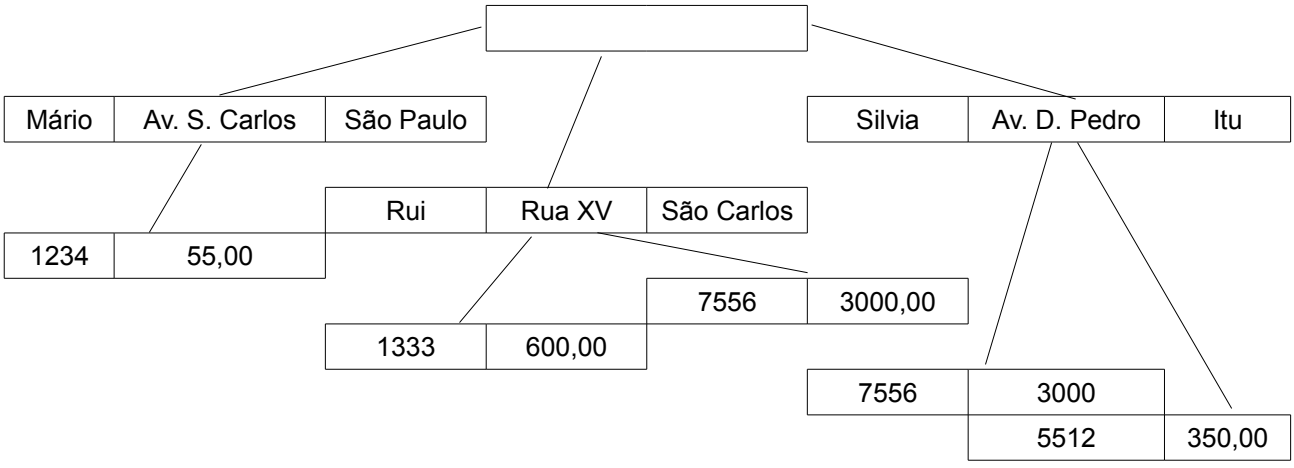
- *Rede:* Neste modelo os registros são representados por um conjunto de registros(como no pascal) e as relações entre estes registros são representadas por *links*(ponteiros). Os registros são organizados no banco de dados por um conjunto arbitrário de grafos[1].

Exemplo do modelo em rede:



- *Hierárquico:* Similar ao modelo em rede por seus dados e suas relações serem representados por registros e *links*. A diferença no modelo hierárquico, é que seus registros estão organizados em árvores ao invés de grafos arbitrários.

Exemplo do modelo hierárquico:



- *Relacional:* Usa um conjunto de tabelas para representar tanto dados como a relação entre eles. Cada tabela possui múltiplas colunas(atributos) e cada uma possui um nome único.

1 Grafo: é um conjunto de pontos, chamados **vértices** (ou nodos ou nós), conectados por linhas, chamadas de **arestas** (ou arcos).

Exemplo de um modelo relacional:

NOMCLI	RUACLI	CIDCLI	NUMCTA
Mário	Av. S. Carlos	São Paulo	1234
Rui	Rua XV	São Carlos	1233
Rui	Rua XV	São Carlos	7556
Silvia	Av. D. Pedro	Itu	5512
Silvia	Av. D. Pedro	Itu	7556

NUMCTA	SALCTA
1234	55,00
1233	600,00
5512	350,00
7556	3000,00

b) *Modelo Orientado a Objetos:* Todos os padrões do modelo orientado a objetos são definidos pela ODMG(*Object Database Management Group* – Fundada em 1991).

Padrões definidos pela ODMG para este modelo:

- Modelo de objetos: um objeto contém valores armazenados em variáveis de instância dentro do objeto. O objeto também contém linhas de código para determinar o comportamento do objeto.
- Linguagem de Definição de Dados – ODL
- Linguagem de Consulta – OQL
- Acoplamento com C++, Java e Smalltalk

c) *Sistemas Objeto-Relacionais:* A área de atuação dos sistemas Objeto-Relacional tenta suprir a dificuldade dos sistemas relacionais convencionais, que é o de representar e manipular dados complexos, visando ser mais representativos em semântica e construções de modelagens. A solução proposta é a adição de facilidades para manusear tais dados utilizando-se das facilidades SQL (*Structured Query Language*) existentes. Para isso, foi necessário adicionar: extensões dos tipos básicos no contexto SQL; representações para objetos complexos no contexto SQL; herança no contexto SQL e sistema para produção de regras.

Diferenças gerais entre os modelos:

- O modelo relacional difere por não utilizar *links* para relacionar os registros;
- Com o não uso de ponteiros é possível o desenvolvimento de fundamentos matemáticos para sua definição;
- Modelos de objetos adicionam recursos como uso de tipos complexos. Exigem que as linguagens suportem o paradigma orientado a objetos.

3.3.GERENCIAMENTO DE TRANSAÇÕES

Muitas operações no banco de dados constituem uma única unidade lógica de trabalho que apresenta as seguintes características(utilizando o modelo ACID):

- *Atomicidade:* executar as operações de forma atômica, ou seja, a operação

deve ser executada por completo ou nada deve ser executado;

- *Consistência:* no banco de dados existem implementações de restrições de consistência. Espera-se que o banco de dados as garanta e preserve;
- *Isolamento:* uma transação deve ser executa como se nenhuma outra estivesse em execução;
- *Durabilidade(persistência):* o efeito de uma transação nunca deve ser perdido, isso depois que a transação tiver sido concluída.

O conceito de transação é: coleção de operações que desempenham uma função lógica única dentro de uma aplicação de um sistema de banco de dados.

3.4.VISÃO GERAL DA ESTRUTURA DO SISTEMA DE BANCO DE DADOS

De forma geral, os componentes funcionais de um sistema de banco de dados podem ser divididos em:

- *Componentes de processamento de consultas:* normalmente o SGBD apresenta um compilador de consultas, responsável por analisar a consulta e gerar um plano de execução para a mesma. Com base neste plano de execução, que envolve também mecanismos de otimização, será executada propriamente a consulta;
- *Componentes de processamento de transações:* é normal agruparmos em um banco de dados uma série de operações dentro de uma única transação. Então cabe ao SGBD garantir as características de uma transação. Algumas das funções do processador de transações: registro de *log*, controle de concorrência e resolução de impasses;
- *Componentes de administração de memória:* os dados de um banco de dados residem normalmente em memória secundária, ou seja, disco magnético. Para carregar os dados para a memória principal o SGBD possui um gerenciador de *buffers*(pequenas regiões na memória com dimensões de página) que executa esta tarefa através do particionamento da memória principal em *buffers*. Informações que podem ser necessárias serem carregadas pelo SGBD para a memória: dados: o conteúdo do banco de dados propriamente dito; metadados: dados que descrevem outros dados; estatísticas: a fim de serem aproveitadas nas consultas; além dos índices;
- *Componentes de gerenciamento do armazenamento:* a parte de armazenamento físico dos dados fica a cargo do SGBD. Estruturas complexas são gerenciadas em segundo plano pelo mesmo, estruturas de dados do tipo: tabela, índice, árvores, etc.

Conhecidos os principais conceitos relacionados a banco de dados e a SGBDs, vamos conhecer os conceitos necessários para a concepção de um sistema de informação utilizando

banco de dados.

4.CONCEPÇÃO DE SISTEMAS DE INFORMAÇÃO

O processo de concepção de um sistema de informação se dá através de algumas etapas. Etapas estas, compostas por:

- Processo de análise dos requisitos organizacionais;
- Desenvolvimento de um modelo conceitual que represente estes requisitos;
- Transformação do modelo conceitual em um modelo lógico e modelo físico, utilizando uma tecnologia de banco de dados.

Entendendo as etapas para concepção de um sistema de informação(figura 4):

1. Análise de requisitos: Identificar e descrever os dados e processos pretendidos pela organização, no que diz respeito a determinado subsistema de informação em estudo.
2. Construção do modelo conceitual: Sintetizar num modelo global as diferentes necessidades dos vários tipos de usuários. Descrever entidades, atributos e associações, independentemente da tecnologia de banco de dados que venha a ser utilizada.
3. Construção do modelo lógico: Transformar o modelo conceitual num modelo lógico dependente da tecnologia de banco de dados a utilizar.
4. Construção do modelo físico: Definir estruturas físicas de dados que sejam mais adequadas para um ambiente tecnológico específico. Estabelecer estratégias de segurança, recuperação e *backup.*

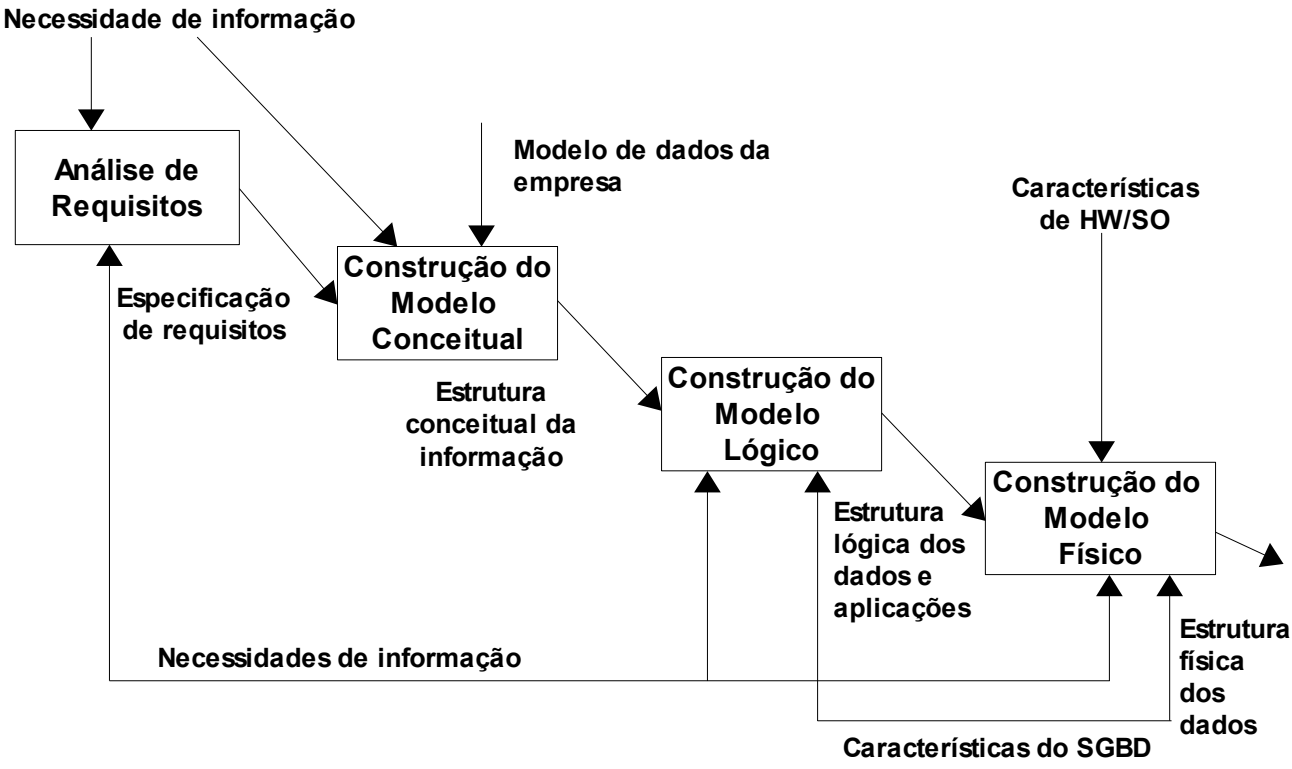


Figura 4 - Etapas para concepção de um Sistema de Informação.

a) Análise de requisitos: Processo de identificação e documentação dos dados, informação e processamentos requeridos pelos usuários finais, de modo a satisfazer as necessidades organizacionais.

Durante esta fase , estamos interessados em:

- Informações que descrevam as estruturas de dados(tais como entidades, atributos, associações);
- Informações que descrevam as regras ou restrições que preservam a integridade dos dados;
- Informação acerca dos processamentos operacionais e de gestão necessários ao negócio da organização.

Tarefas na Análise de Requisitos:

- Definir o âmbito do subsistema em estudo;
- Estabelecer a linguagem de comunicação;
- Identificar as necessidades e requisitos dos usuários;
- Identificar requisitos operacionais.

Análise de requisitos: utilizando duas perspectivas.

- *Análise de funções:* Estudo dos dados que fluem no subsistema e os processamentos/atividades que utilizam e/ou transformam esses dados. Normalmente utilizam-se técnicas como o Diagramas de Fluxos de Dados, Dicionário de Dados e Especificação de Processos.
- *Análise de dados:* Levantamento dos subconjuntos de dados requeridos por cada usuário, de modo a este poder realizar as suas tarefas. Nesta perspectiva analisam-se cada uma das "*user views* – visões do usuário", identificando e definindo os respectivos dados.

Análise de requisitos: qual perspectiva adotar?: O ideal será utilizar estas duas perspectivas em complemento uma com a outra para a elaboração da análise de requisitos.

- A análise de funções dá-nos a visão do funcionamento de um sistema como um todo, mas em que não temos detalhes suficientes em relação às estruturas de dados e suas associações;
- Por outro lado, a de análise de dados mostra-nos que estruturas de dados e suas associações são necessárias, dando pouca ênfase aos diferentes funcionamentos do sistema.

Análise de requisitos: Com qual se inicia a análise?

- Se, no subsistema em estudo, os usuários finais conseguem descrever com

certa precisão o seu funcionamento em termos de atividades, então normalmente inicia-se a análise orientadas para as funções.

- Se os usuários estão mais seguros quanto aos dados que utilizam e da sua caracterização, então inicia-se a análise orientada para os dados.

Análise de requisitos pode utilizar-se de alguns métodos e técnicas de levantamento de requisitos, dentre eles: documentação, observação, entrevista e engenharia reversa(*Reverse Engineering*).

Uma vez realizada a análise de requisitos parte-se para a construção do modelo conceitual.

b) Construção do Modelo Conceitual:

- Utilização de uma aproximação específica, adequada a modelação sistematizada dos requisitos;
  - Modelo entidade associação (*Entity* Association: *EA) ou (Entity Relationship*: ER);
  - Modelos orientados a objetos;
- Selecionado o modelo, utiliza-se uma notação diagramática: representação gráfica.

c) Construção do Modelo Lógico: neste ponto são realizadas as tarefas de:

- Transposição do modelo conceitual para um modelo de organização de estruturas(lógicas) de dados
  - Adoção de uma modelo lógico específico;
  - Exemplo: Conversão de diagramas E(ntidade)-A(ssociação) em “tabelas” (ou relações) de um modelo relacional.

d) Construção do Modelo Físico: A gestão e manipulação dos dados do “modelo” físico é feita a partir de funcionalidades oferecidas pelas ferramentas de Administração da Base de Dados contidas no SGBD.

4.1.MODELAGEM DE DADOS E INFORMAÇÃO

Tem a função de identificar, analisar e registrar a política da organização face aos dados referentes ao sistema de informação. A modelagem de informação tem mérito de levantar questões sobre as regras da organização, resultando frequentemente na identificação de novas necessidades de informação ou na resolução de incoerências no atual tratamento da informação.

Modelagem: no sentido lato da informação, o modelo organizacional está embebido no sistema de informação a conceber. Modelagem = Modelação ou Modelização.

Níveis para o Modelo: são apresentados dois.

- Modelo conceitual: representação da realidade, sem atender a quaisquer constrangimentos impostos pelo ambiente de tecnologia;
- Modelos lógico e físico: Adaptação do modelo conceitual às características próprias do sistema tecnológico;
- Estruturas de dados(modelo lógico);
- *Bytes on Disk*(modelo físico).

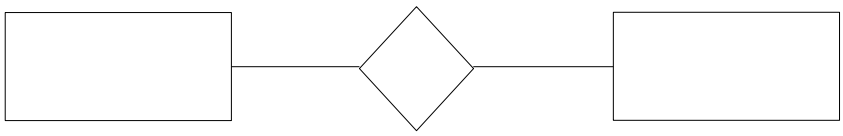
Passo importante do processo de concepção de um sistema de informação, é a adoção de uma metodologia de modelagem adequada.

4.2.METODOLOGIA DE MODELAGEM

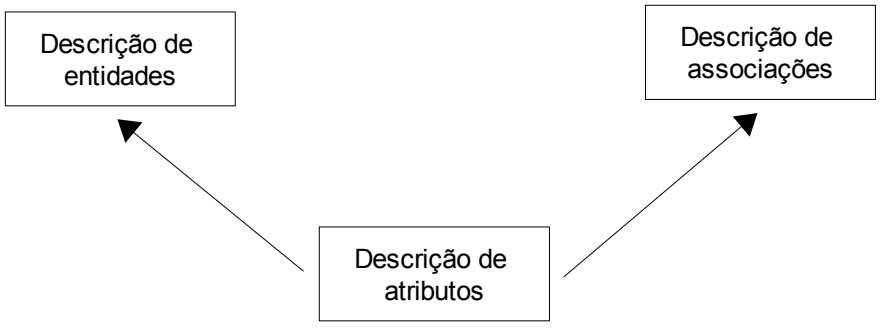
Apresentação do método *Top-Down*. Conforme Chen, o método permite a construção do modelo por fases, partindo dos grandes objetos de informação(entidades) e identificando as suas inter-relações(associações). Este método produz bons elementos de comunicação com os usuários. Não exige esforço inicial de levantamento de níveis mais elementares de informação(atributos), o que é impensável em grandes sistemas de informação.

Especificação do modelo conceitual: pode ser em dois níveis.

Gráfico – Diagrama ER(Entidade-Relacionamento)



Descritivo: especificação para cada componente do modelo.



Etapas do método, propostas por Chen:

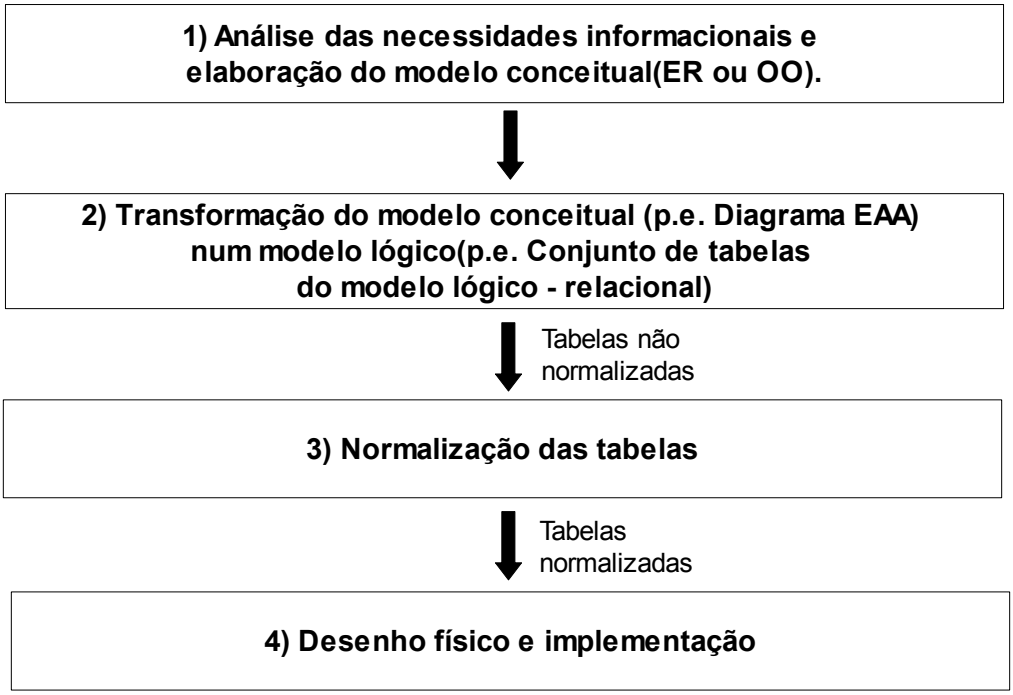


Figura 5 – Etapas do método *top-down*.

Principais conceitos relacionados ao Modelo Entidade-Associação(E-A):

- Básicos
  - Entidade
  - Associação
  - Atributo
  - Entidade Associação
- Extensões
  - Associação Exclusiva
  - Generalização

A representação de um modelo é chamada de esquema do banco de dados. Modelos podem ser presentados de forma gráfica e de forma textual. Na forma textual pode ser utilizada linguagem natural representada por exemplo através de uma gramática BNF(*Backus–Naur Form*, utilizada para descrever gramáticas livres de contexto). Nesta apostila especificamente serão adotadas representações com formas gráficas.

Descrição dos conceitos do Modelo E-A:

Entidade: Qualquer objeto ou conceito com interesse para a organização, sobre o qual se quer guardar informações e que possa ser identificável de forma inequívoca. Manter informações sobre o objeto, armazenadas.

Exemplos: Contrato, Produto, Fornecedor, Cliente, Aluno, Professor, Livro, ..., etc. - Substantivos comuns?

... nome da entidade ...	**representação diagramática ou gráfica**

Atributos: para cada entidade é necessário conhecer as suas propriedades relevantes para o sistema. Um atributo é qualquer propriedade de uma entidade.

Os atributos são sempre elementos atômicos(indivisíveis) de informação e que assumem diferentes valores de um DOMÍNIO. ***Domínio é o conjunto de valores que um atributo pode assumir.***

Exemplos: Número do Cliente, Nome do Cliente, Código do produto, etc.

Tipos de atributos:

- *Identificadores ou chaves candidatas:* São os atributos de uma entidade que identificam uma ocorrência específica dessa entidade, distinguindo-a das restantes – são também chamados de "identificadores de entidade". Para que um atributo seja identificador é necessário que não existam duas ocorrências distintas dessa entidade nas quais o atributo tenha o mesmo valor.

Exemplo: **Qual é o atributo identificador para cliente?**

Número do Cliente	Nome do Cliente	Categoria
1234	Silva	J
1235	Martins	H
1236	Silva	L
1237	Lopes	L

- *Descritores:* São os atributos que apenas descrevem ou caracterizam as ocorrências de uma entidade.

Atributos também pode apresentar cardinalidade. Se a cardinalidade for *(1,1)*, estes

atributos são chamados de **monovalorados**, e a cardinalidade pode ser omitida. Atributos com cardinalidade mínima *0*, são chamados de **opcionais**. Atributos com cardinalidade máxima *n* são chamados de **multivalorados**. O conceito de cardinalidade é melhor explicado mais a frente.

Associação(Relacionamento): As associações representam as interligações relevantes entre as entidades do sistema.

Um associação pode relacionar:

- Duas entidades entre si(binária)
- Várias entidades entre si(complexa)
- Uma entidade consigo própria(unária)

representação diagramática ou gráfica

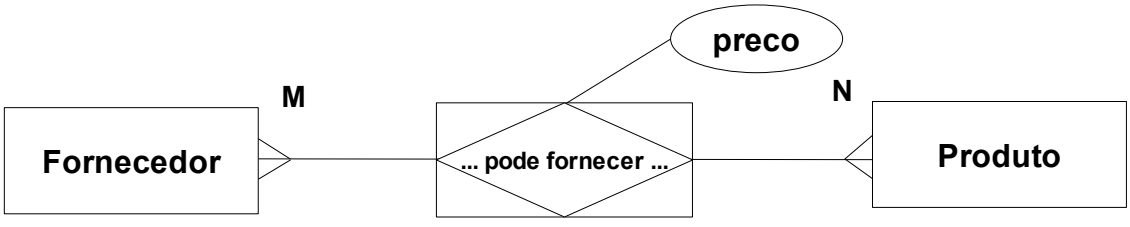
Exemplo de uma associação: Uma associação que relacione EMPREGADO com o DEPARTAMENTO limita-se a indicar:

- Os empregados ligados a um departamentos
- A que departamento está ligado um empregado

Normalmente uma associação limita-se a relacionar entidades entre si, no entanto, podem haver associações com características próprias.

Entidade associativa: Há, no entanto, situações em que as associações têm atributos próprios.

Exemplo: Uma associação que relacione fornecedor com produto, para além de indicar quais os fornecedores de um produto e quais os produtos fornecidos por um fornecedor, pode conter o preço pelo qual um fornecedor fornece produto.



Entidades versus Associações: Para distinguir entidades de associações podemos socorrer-nos do vocabulário.

Geralmente utilizam-se:

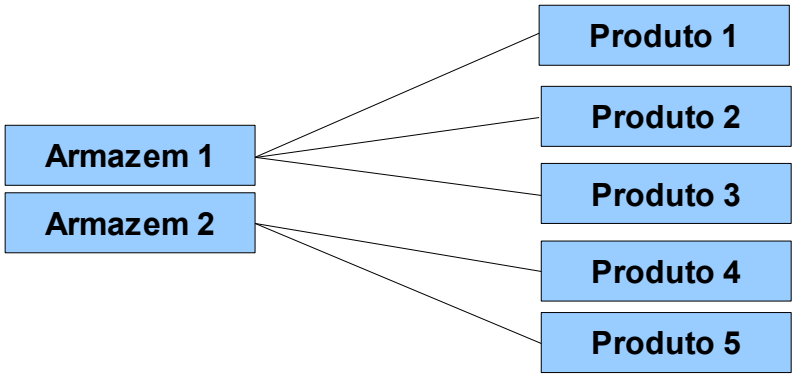
- Substantivos: para referenciar Entidades;
- Verbos: para referenciar Associações.

Na terminologia corrente utiliza-se o termo Entidade(ou Associação):

- Quer para a definição conceitual da entidade(ou associação);
- Quer para referenciar qualquer ocorrência dessa entidade(ou associação).

Entidades e associações: um exemplo.

- 2 entidades, “Armazem” (2 ocorrências) e “Produto” (5 ocorrências)
- 1 associação, “Armazena”, com 5 ocorrências



Propriedades das Associações: *grau de uma associação* ou *cardinalidade*. O grau de uma associação ou cardinalidade se refere ao número de ocorrências de uma entidade em relação a outra. A definição da cardinalidade apresenta o número **mínimo** e **máximo** de ocorrências de entidades associadas.

Para facilitar o entendimento da cardinalidade é muito útil construir um ***diagrama de ocorrências***. Um diagrama de ocorrências é uma representação gráfica, onde: as ocorrências de entidades são representadas por círculos brancos; ocorrências de relacionamentos são representados por círculos negros; ocorrências de entidades participantes de uma ocorrência de relacionamento são indicadas pelas linhas que ligam o círculo negro representativo da ocorrência de relacionamento aos círculos representativos das ocorrências de entidades relacionadas(HEUSER, 2004).

Vamos a um exemplo:

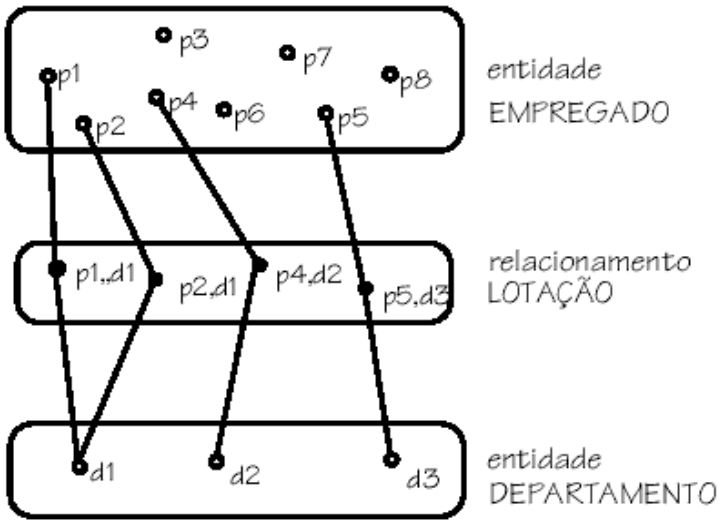


Figura 6 – Exemplo de um diagrama de ocorrências.
Fonte: Heuser(2004).

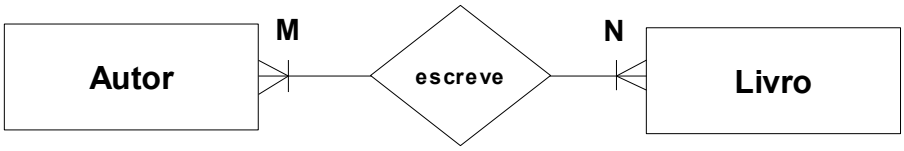
Associações binárias: associam duas entidades entre si.

- Associações 1:1 (1 para 1)
- Associações 1:N (1 para muitos)
- Associações M:N (muitos para muitos)

Associações 1:1: Exemplo: Num curso cada módulo é ministrado por um só instrutor, e cada instrutor ministra apenas um módulo.

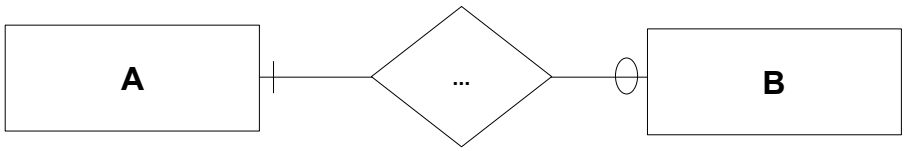
Associações 1:N: Exemplo: Um departamento tem lotados vários empregados(eventualmente só um ou mesmo nenhum), e um empregado está ligado apenas a um departamento(ou nenhum).

Associações M:N: Exemplo: Um livro pode ser escrito por vários autores, assim com um autor pode escrever vários livros.



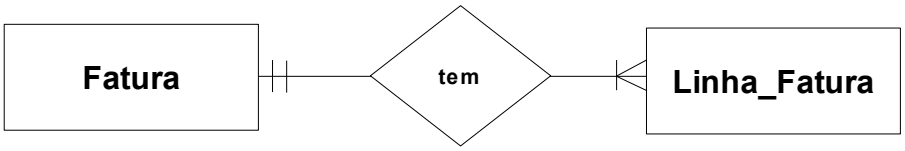
Conectividade de uma Associação: Uma entidade participa numa associação de duas formas:

- *Obrigatória(ou total):* Não pode existir nenhuma ocorrência dessa entidade que não esteja associada a alguma ocorrência de outra entidade que participa na associação.
- *Não obrigatória(ou parcial):* Podem existir ocorrências dessa entidade que não estejam associadas a alguma ocorrência da outra entidade que participa na associação.



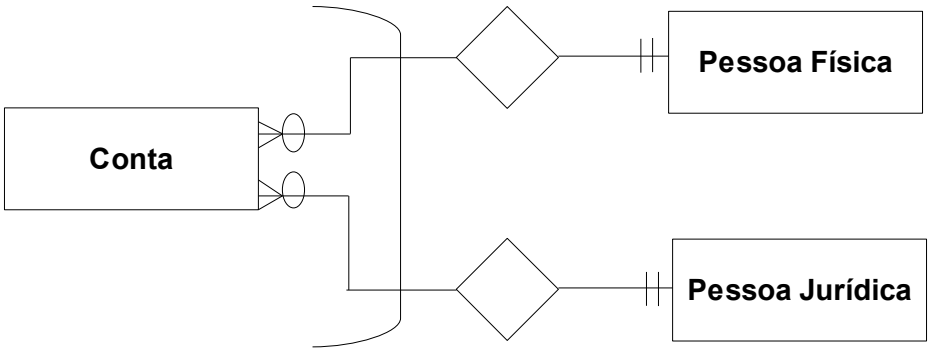
A: Obrigatória - B: Não-obrigatória
(independente do grau)

Exemplos:



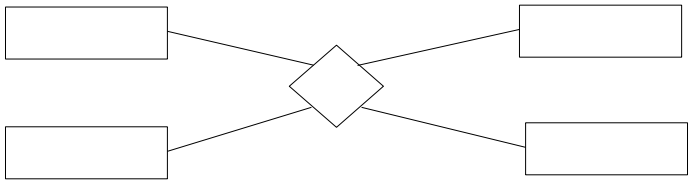
Associações exclusivas: Por vezes, duas ou mais associações de uma mesma entidade são mutuamente exclusivas. Esta situação é representada através do Arco de exclusividade, que unes as associações relevantes para a exclusividade.

Exemplo: Uma conta bancária pode pertencer a uma pessoa física ou a uma pessoa jurídica, mas sempre a uma delas e nunca a ambas.



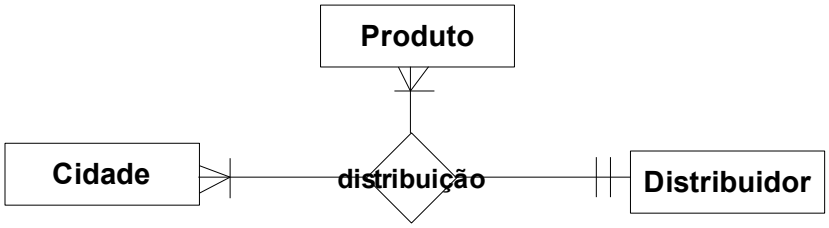
Associações complexas: A abordagem ER permite a construção de relacionamentos de grau maior que 2(ternários, quaternários, etc), portante, associações complexas associam

entre si mais que duas entidades. Devem ser usadas apenas quando o conceito inerente à associação não possa ser representado por um conjunto de associações binárias entras as entidades envolvidas.

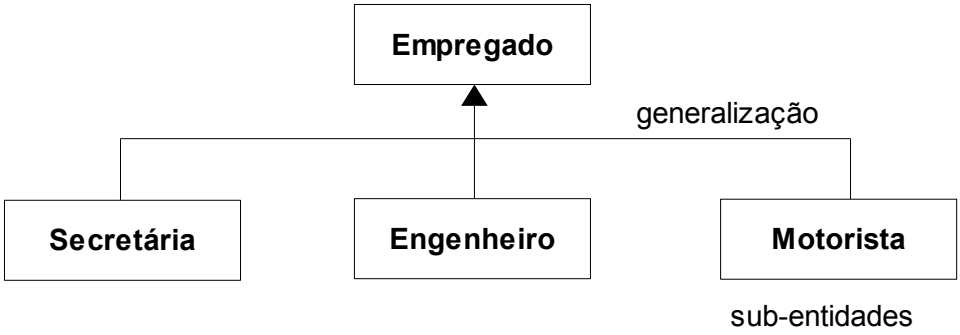


Como definir o grau de uma associação complexa?: Fixa-se cada par de entidades e analisa-se qual o número de ocorrências da terceira entidade. No exemplo:

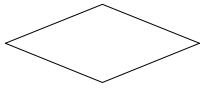
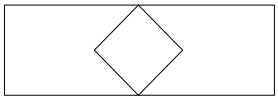
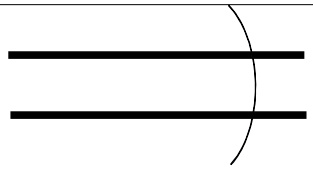
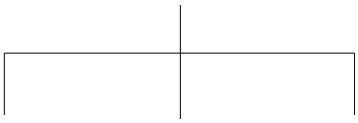
- para o par Produto,Cidade só pode existir uma ocorrência de Distribuidor
- para o par Cidade,Distribuidor podem existir n ocorrências de Produto
- para o par Produto,Distribuidor podem existir n ocorrências de Cidade

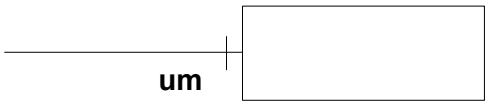
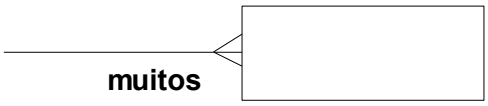
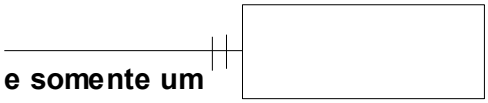
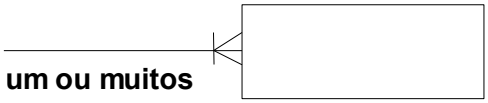
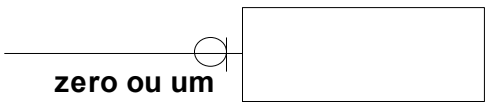
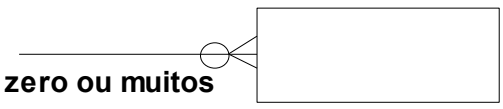


Generalização: Uma entidade E é a generalização das entidades E1...En, se cada ocorrência de E1...En é também ocorrência de E. As entidades E1...En podem ser entendidas como entidades especializadas de E.



No quadro a seguir são apresentadas notações diagramáticas adotadas neste material.

Notação Diagramática Geral	
Conceito	**Representação Gráfica**
Entidade	
Associação	
Entidade Associativa	
Atributo	
Chave Candidata ou Identificador	
Associação Exclusiva	
Generalização	

Notação Diagramática nas Associações(notação "pé-de-galinha" da engenharia de informações)	
	
	
	

Notação alternativa para as Associações(notação Chen)	
1 — um	N — muitos
1 — um e somente um	N — um ou muitos
1 — zero ou um	N — zero ou muitos

O próximo passo de modelagem é a seleção de um modelo lógico de banco de dados e a conversão do modelo conceitual para o modelo escolhido. Nesta apostila o modelo lógico a ser trabalhado é o relacional. A seguir são apresentadas mais características deste modelo.

4.3.MODELO RELACIONAL

O modelo relacional tornou-se disponível na década de 70. Desde então este modelo tem sido implementado em um grande número de SGBDs comerciais e livres, como: DB2, Informix(IBM), Oracle, SQL Server, PostgreSQL, entre outros. Neste capítulo serão explicados os principais conceitos relacionados ao modelo relacional.

Níveis de abstração em um Sistema de Banco de Dados(figura 6): antes de chegarmos a um modelo mais completo, muitos modelos parciais ou intermediários poderão surgir.

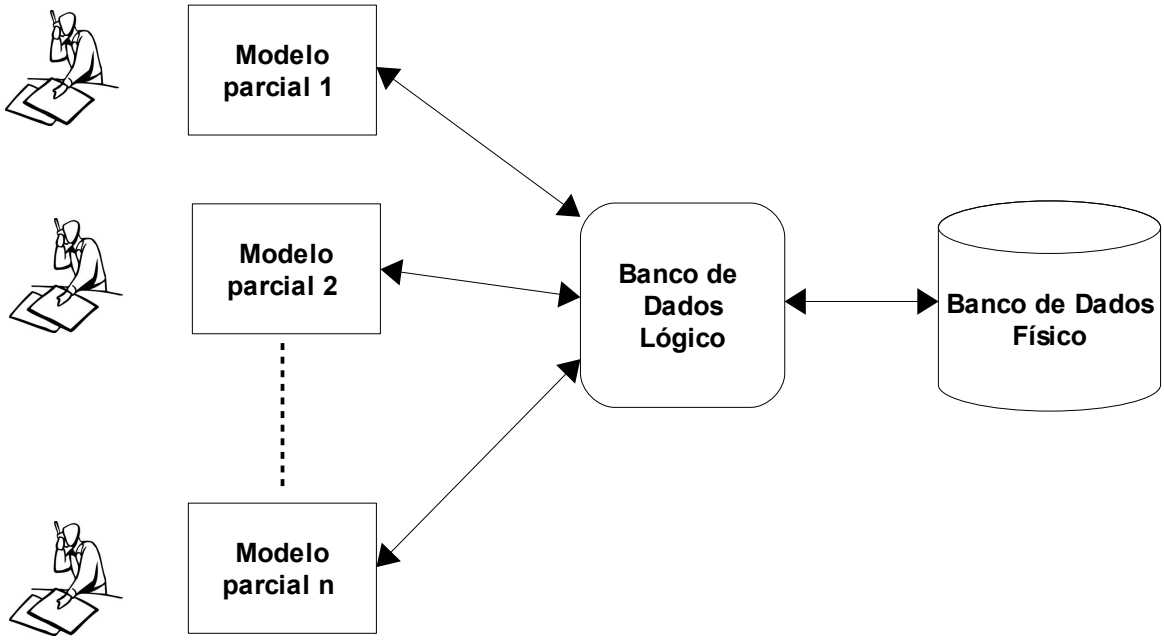
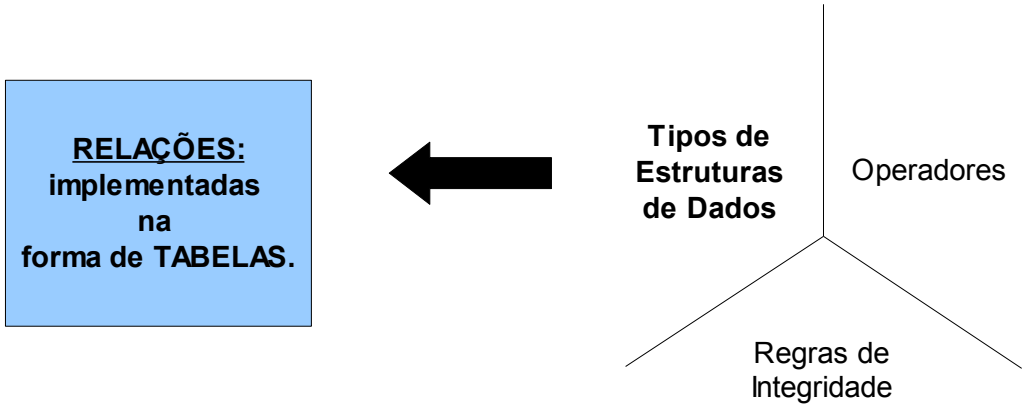


Figura 6 - Níveis de abstração em um Sistema de Banco de Dados.

Como é formado o modelo de dados(figura 7): além das estruturas de dados são disponibilizadas regras de integridade: para garantir a integridade das informações; e operadores que auxiliarão na manipulação da informações.

Figura 7 – Estrutura do modelo de dados relacional.

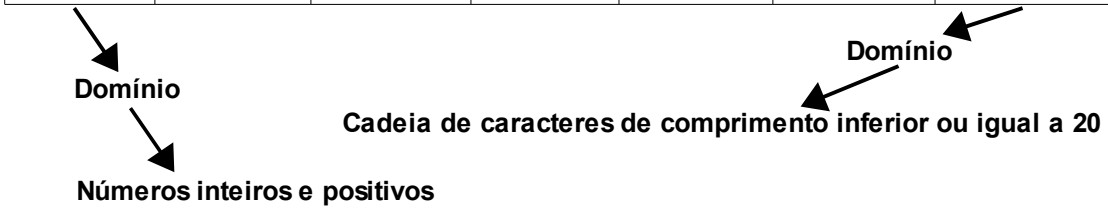
A estruturas básicas do modelo relacional são as **relações** ou **tabelas*:***



O que é uma relação ou tabela? É um conjunto não ordenado de **linhas** ou **tuplas**.

Exemplo de uma Relação(EMPREGADO):

NUMEMP	NOMEMP	TELEMP	CATEMP	SALEMP	COMEMP	FUNEMP
10	Antunes	12345678	B	100	15	Analista
20	Bento	12345668	A	250	50	Diretor
30	Correia	12345679	B	60,50		Porteiro
40	Dias	12345670	C	90,50		Programador
50	Edmundo	12345688	B	120	12.5	Contador



Alguns conceitos importantes sobre uma relação:

a) *Grau*: número de colunas da relação.
b) *Atributo*: ou colunas. Propriedades particulares que a descrevem.
c) *Esquema da relação*: definição da relação

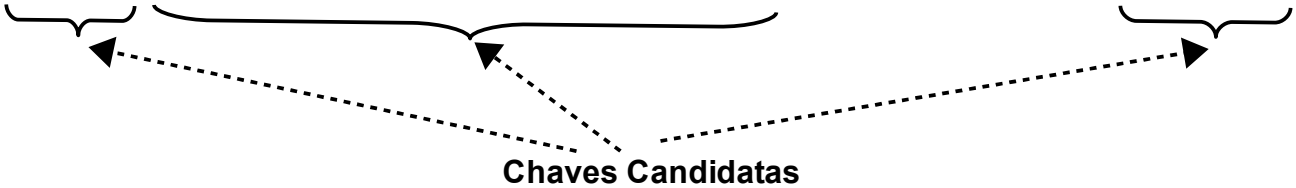
   Exemplo:

   - Empregado(<u>numemp</u>, nomemp, telemp, catemp, salemp, comemp, funemp)

d) *Esquema relacional:* Definição de um Banco de Dados relacional sob a forma de um conjunto de esquemas de relação.
e) Chave candidata(C*andidate Key*): é considerada a chave candidata de uma relação. Define o conjunto mínimo de atributos que identificam cada ocorrência da relação. Não existem duas linhas da relação com o mesmo conjunto de valores neste atributo(ou atributos).

Relação CLIENTE

NUMCLI	NOMCLI	RUACLI	CIDCLI	PAIS	TIPCLI	NUMCON
10	Antônio Abreu	Alberto Antunes	Andorra	Andorra	1	123.456.789
20	Bernardo Bento	Beta Brás	Bruxelas	Bélgica	2	789.123.654
30	Carlos Castro	Clara Campos	Camberra	Canadá	1	987.456.321
40	Daniel Dantas	Dominicana	Camberra	Canadá	2	789.098.543
50	Manuel Matos	Marco Moita	Maputo	Moçambique	1	765.098.345



h) Chave primária(*Primary Key):* Chave primária da relação -> Entre as chaves candidatas de uma relação, escolhe-se uma para ser a chave efetiva da mesma(a que for mais útil no sistema em questão). A essa chave, dá-se o nome de chave primária.

i) Chave estrangeira(*Foreign Key*): Chave estrangeira de uma relação -> Em algumas relações, temos um atributo(ou conjunto de atributos) cujas ocorrências são referências à chave primária de uma outra relação. A esses atributos damos o nome de Chaves Estrangeiras.

Exemplo de chave estrangeira:

NUMEMP	NOMEMP	CODCAT
01	João	C3
02	José	C1
03	Gil	C2
04	Júlio	C1

CODCAT	DESCAT	SALBSE
C1	Diretor	300,00
C2	Técnico	200,00
C3	Administrador	120,00
C4	Auxiliar	90,00

Chave Estrangeira

h) *Domínio*: representa o conjunto de valores atômicos que um atributo pode conter.

i) *Definição de Banco de Dados Relacional:* Conjunto de relações, cujo conteúdo varia ao longo do tempo.

Exemplo de um Banco de Dados Relacional:

FATURA

NUMFAT	DATEMI	DATPAG	NUMCLI
01	01/01/1989	15/01/1989	10
02	12/02/2004	25/04/2004	20
03	15/03/2001	20/03/2001	20

PRODUTO

CODPRO	NOMPRO	PRECO	QTDPRO
01	Lápis	100	2543
02	Caneta	150	1321
03	Régua	500	354

CLIENTE

NUMCLI	NOMCLI	RUACLI	CIDCLI	PAIS	TIPCLI
10	Antônio Abreu	Alberto Antunes	Andorra	Andorra	1
20	Bernardo Bento	Beta Brás	Bruxelas	Bélgica	2
30	Carlos Castro	Clara Campos	Camberra	Canadá	1
40	Daniel Dantas	Dominicana	Camberra	Canadá	2

FATURA_PRODUTO

NUMFAT	CODPRO	QTDFAT
01	01	30
01	02	15
01	03	23
02	01	40
02	02	10
02	03	15
03	01	25

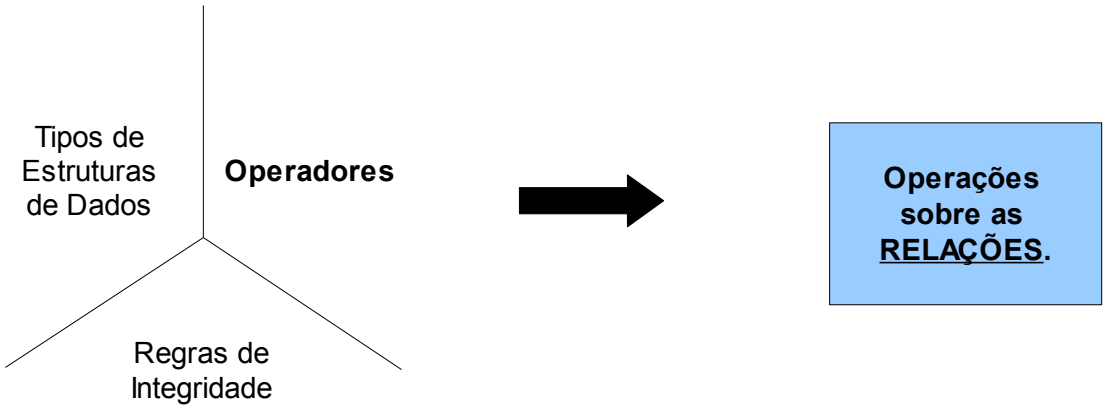
Linhas de fatura 

03	02	60
03	03	10

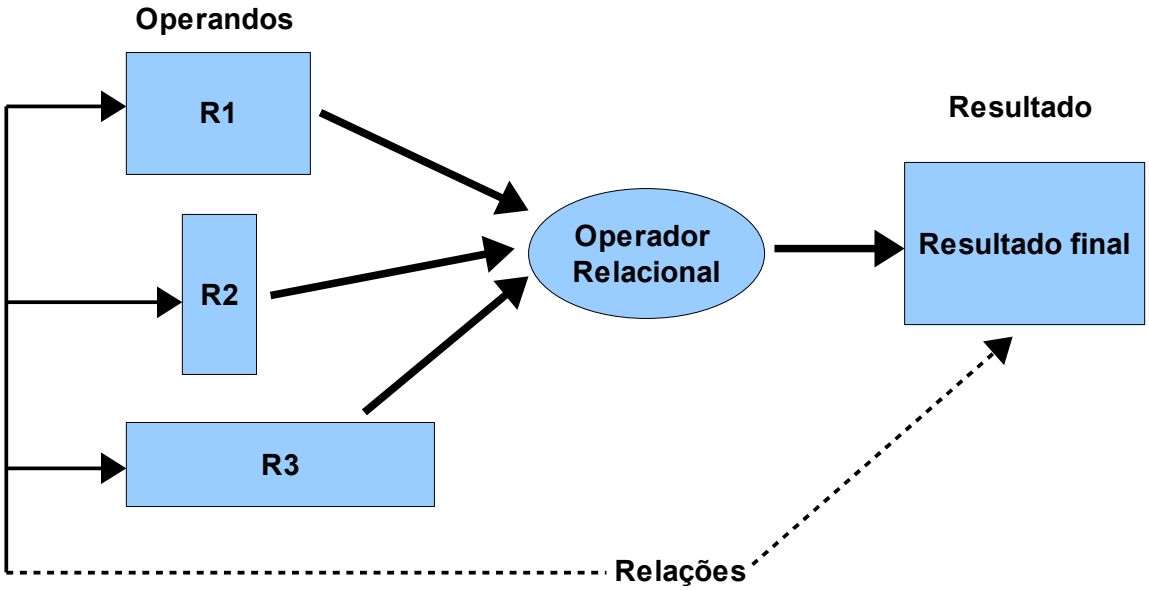
Esquema relacional do banco apresentado anteriormente:

CLIENTE(numcli, nomcli, ruacli, cidcli, pais, tipcli)
FATURA(numfat, datadm, datpag, numcli)
PRODUTO(codpro, nompro, preco, qtdpro)
FATURA_PRODUTO(numfat, codpro, qtdfat)

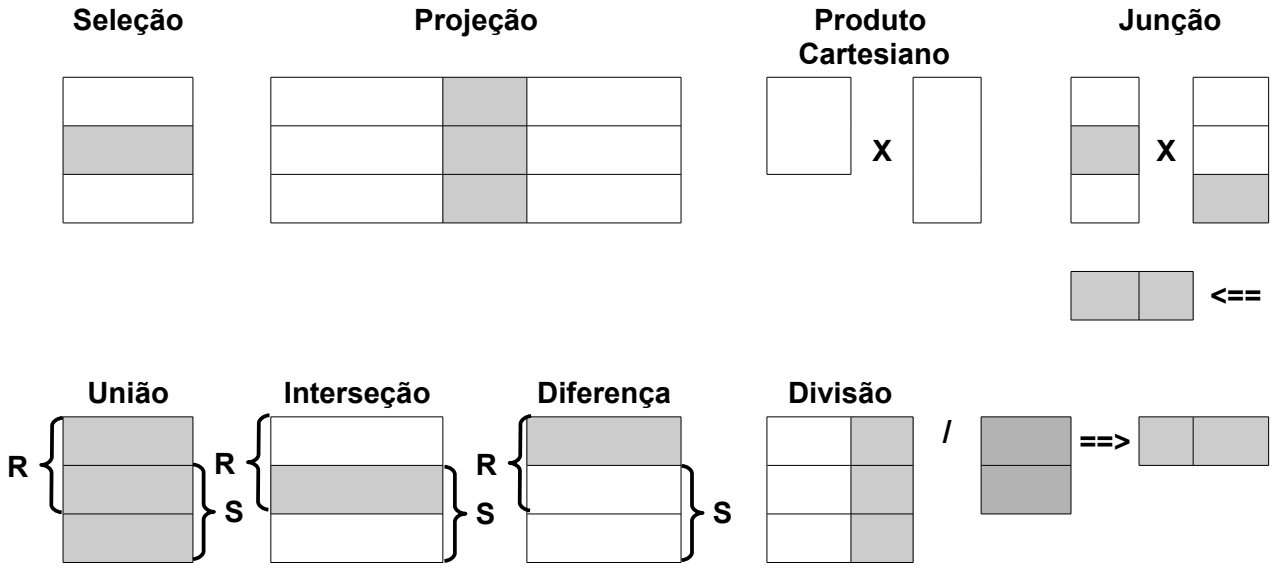
Operadores do Modelo Relacional: operadores utilizados para manipular as informações contidas nas relações.



Comportamento dos operadores relacionais:



Resumo do funcionamento das operações de álgebra relacional: seleção, projeção, produto cartesiano, junção, união, interseção, divisão, diferença.

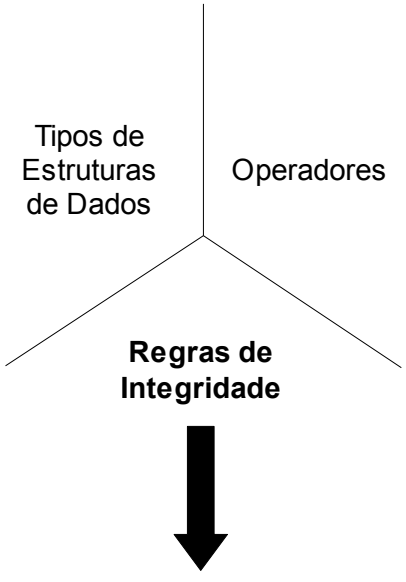


Conceito de valor NULL: Numa relação podemos, em determinado momento, ter um atributo cujo valor é desconhecido ou não é aplicável no momento. Diz-se então que esse elemento tem o valor NULL.

Características dos valores NULL:

- Independentes de domínio: inteiro, real, caracter, data, etc.
- Não comparáveis entre si: não é possível dizer que um valor NULL é igual a outro valor NULL.

Regras de Integridade do Modelo Relacional: um banco de dados relacional deve prover recursos para permitir a integridade das informações.



Integridade das Entidades: também chamado de integridade de identidade. Diz que

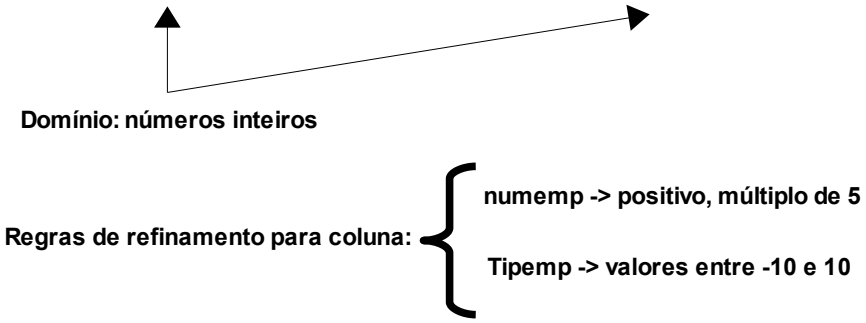
nenhum componente da chave primária de uma relação poderá em algum momento ter valor NULL, em qualquer um de seus atributos.

Integridade de Domínio: Cada coluna de uma relação tem um domínio, isto é, um conjunto(limitado ou não) de valores possíveis. Em todas as linhas dessa relação, o valor dessa coluna terá que pertencer sempre a esse domínio.

Integridade de Coluna: refinamento da integridade de domínio.

Exemplo:

EMPREGADO(numemp, nomemp, ruaemp, salemp, tipoemp)



Integridade Referencial: Numa relação, qualquer ocorrência de uma chave estrangeira deverá obrigatoriamente existir como ocorrência da chave primária da relação à qual se refere.

Integridade Definida pelo Usuário: Qualquer outra regra a que as ocorrências de um determinado banco de dados deverá obedecer e que não é abrangida pelos tipos mencionados anteriormente.

Apresentadas as características do modelo lógico relacional, vamos conhecer o que um SGBD relacional tem a oferecer.

4.4.SGBD RELACIONAL

O que faz de um SGBD relacional? Segundo Codd(1985b) um SGBD pode ser considerado minimamente relacional se satisfaz as seguintes condições:

- Estrutura: Ao usuário são apresentadas apenas tabelas e nada mais do que tabelas.
- Manipulação: Suporta operações de restrição, projeção e junção natural, sem necessidade e definição de caminhos físicos de acesso para estas operações.

A partir da definição de Codd, Date(2004) sugere uma classificação para SGBDs relacionais:

1. Tabular: Suporta a estrutura tabular, mas não os operadores relacionais;
2. Minimamente relacional: Suporta a estrutura tabular e os operadores de restrição, projeção e junção(apenas estes);

3. Relacional: Suporta a estrutura tabular e todos os operadores da álgebra relacional;

4. Completamente relacional: Suporta todos os aspectos do modelo relacional(estrutura tabular e domínios, todos os operadores da álgebra relacional e a integridade de entidades e referencial).

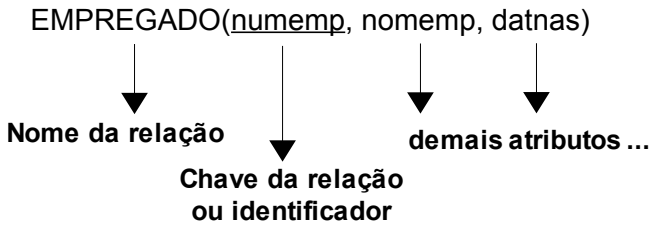
Segundo um artigo posterior de Codd(1985a) um SGBD completamente relacional(*fully relational*) se gerencia o banco de dados utilizando apenas e só as possibilidades oferecidas pelo modelo relacional. Assim sendo, o SGBD relacional, tem de satisfazer as seguintes 12 regras:

1. Informação em tabelas
2. Garantia de acesso
3. Tratamento sistemático de valores nulos
4. Catálogo *on-line* e ativo baseado no modelo relacional
5. Sublinguagem de manipulação de dados compreensível
6. Atualização de *views*
7. Insert, Update e Delete de alto nível: tratamento de conjuntos
8. Independência física dos dados
9. Independência lógica dos dados
10. Independência da integridade
11. Independência da distribuição
12. Não subversão: instruções de baixo nível não podem quebrar regras de integridade.

Conhecidos os principais conceitos do modelo relacional e dos SGBDs relacionais, é o momento de converter o modelo conceitual para um modelo relacional.

4.5.TRANSFORMAÇÃO DO MODELO CONCEITUAL EM MODELO RELACIONAL

Representação de uma relação:



Como se dá o processo de transformação/conversão: regras gerais.

- **Entidades:** Cada entidade dá normalmente origem a uma relação com:

- Identificador da entidade: chave da relação;
- Descritores da entidade: outros atributos da relação;
- Identificadores de outras entidades que eventualmente lhe estejam associadas: Chaves estrangeiras;

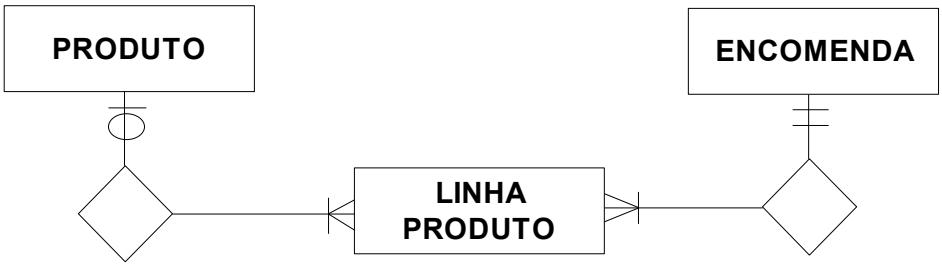
- **Associações:**
  - Associações 1:1 e associações 1:N são representadas normalmente pela adição de novos atributos (chaves estrangeiras) às relações existentes;
  - Associações M:N dão origem a uma nova relação.

Regras específicas de transformação:

- **Entidades Associativas:** as entidades associativas podem ser refinadas em novas entidades.

Antes de refinado

Depois de refinado



- **Associações Binárias:**
  - ***Associações 1:1***

1. Caso: Ambas as entidades são obrigatórias. Basta uma tabela para representar esta situação

- EMPREGADO_CARRO(<u>numemp</u>, nomemp, ..., <u>matricula</u>, marca)
- Chaves candidatas: numemp, matricula

Nota: Analisar criteriosamente esta situação: muito limitativa.

2. Caso: apenas uma das entidades é obrigatória. Para representar está situação são necessárias 2 tabelas, uma para cada entidade. Colocar o identificador da entidade não obrigatória na tabela correspondente à entidade obrigatória.

- EMPREGADO(<u>numemp</u>, nomemp, ...)
- CARRO(<u>matricula</u>, marca, ..., numemp)

3. Caso: Nenhuma das entidades é obrigatória. Situação representada por 3 tabelas, 1 por entidade e 1 para a associação.

- EMPREGADO(<u>numemp</u>, nomemp, ...)
- CARRO(<u>matricula</u>, marca, ...)
- EMPREGADO_CARRO(<u>numemp</u>, <u>matricula</u>)
    - *Chaves candidatas: numemp, matricula*

Nota: Esta situação será normalmente representada por apenas duas tabelas. É necessário ter em conta a proporção do número de ocorrências da associação / número de ocorrências das tabelas associadas.

- **Associações 1:N**

1. Caso: Entidade do lado "N" obrigatória. Esta situação é representada por 2 tabelas, uma para cada entidade.

- DEPARTAMENTO(<u>coddep</u>, nomdep, ...)
- EMPREGADO(<u>numemp</u>, nomemp, ..., coddep)

2. Caso: Entidade do lado "N" não-obrigatória. Esta situação pode ser representada por três tabelas, uma para cada entidade e uma para a associação.

- DEPARTAMENTO(coddep, nomdep, ...)
- EMPREGADO(numemp, nomemp, ...)
- EMPREGADO_DEPARTAMENTO(numemp, coddep)

Nota: Esta situação será normalmente representada por 2 tabelas. É necessário ter em conta a proporção do número de ocorrências da associação / número de ocorrências da tabela EMPREGADO.

- **Associações M:N**

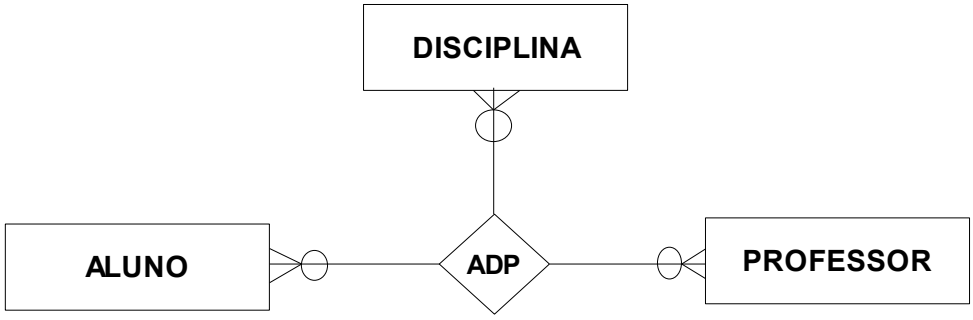
Esta situação é sempre representada por três tabelas, uma para cada entidade e uma para a associação.

- FORNECEDOR(codfor, nomfor, ...)
- PRODUTO(codpro, nompro, ...)
- FORNECEDOR_PRODUTO(codfor, codpro)

- **Associações Complexas**

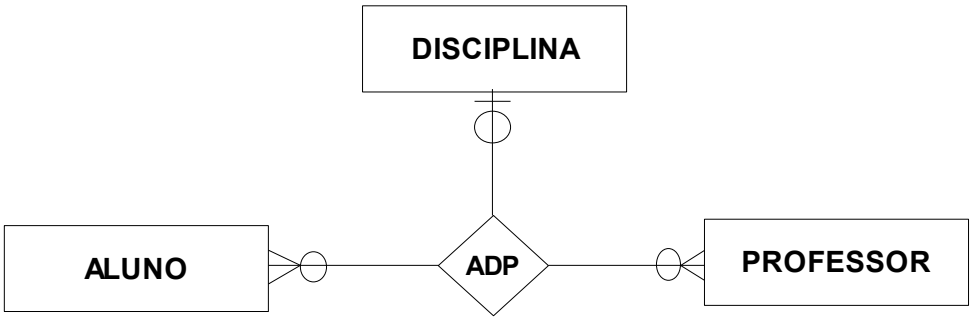
Esta situação é representada por quatro tabelas, uma para cada entidade e uma para a associação.

- DISCIPLINA(coddis, nomdis, ...)
- ALUNO(codalu, nomalu, ...)
- PROFESSOR(codpro, nompro,...)
- ADP(coddis, codalu, codpro)



Nota: A chave da associação, neste caso concreto, é a concatenação das três chaves e entidades presentes. Pode não ser assim, pois este fato depende do grau da associação.

- DISCIPLINA(coddis, nomdis, ...)
- ALUNO(codalu, nomalu, ...)
- PROFESSOR(codpro, nompro,...)
- ADP(coddis, codalu, codpro)

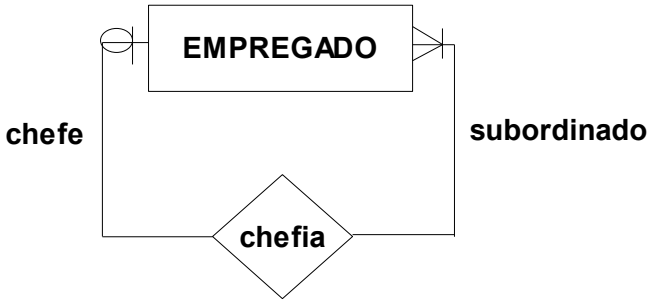


- **Associações Unárias 1:N**

Lado "N" obrigatório. Esta situação é representado por uma única tabela.

- EMPREGADO(numemp, nomemp, ..., numchf)

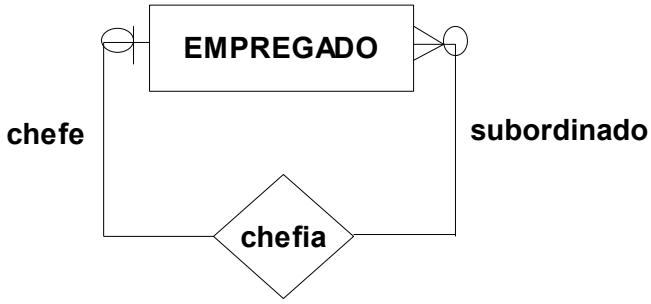
Nota: Neste caso, é necessário ter uma atenção que o chefe máximo é chefe dele próprio.



Lado "N" não é obrigatório. Esta situação é representada por duas tabelas, uma para a entidade, outra para a associação.

- EMPREGADO(numemp, nomemp, ...)
- CHEFIA(numemp, numchf)

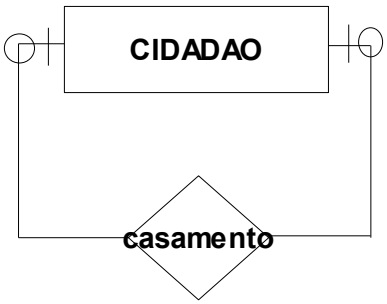
Nota: Esta situação será normalmente representada por apenas uma tabela. É necessário ter em conta a proporção do número de ocorrências da associação / número de ocorrências da tabela EMPREGADO.



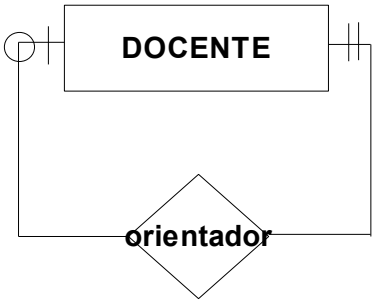
- **Associações Unárias 1:1**

Nenhum dos lados é obrigatório. Esta situação é representada por duas tabelas, uma para a entidade e uma para a associação.

- CIDADAO(numcid, nomcid, ...)
- CASAMENTO(numcid_marido, numcid_esposa)



Um dos lados é obrigatório. Esta é uma situação representada por uma única tabela.

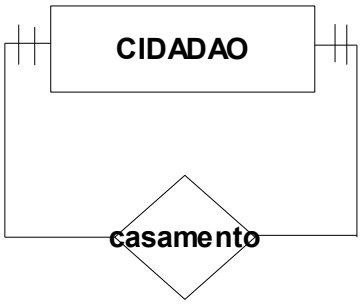


Ambos os lados são obrigatórios. Considere-se o registro de casamentos de uma igreja. Esta situação é representada por uma única tabela.

- CASAMENTO(numcid_marido, nome_marido, ..., numcid_esposa, nome_esposa)

\

- Chaves candidatas: numcid_marido, numcid_esposa.

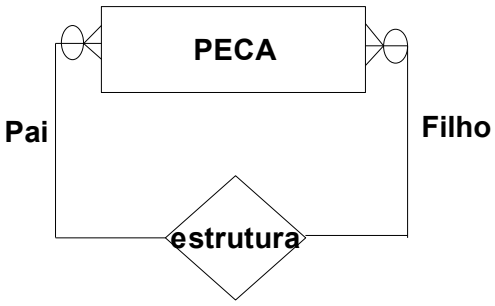


Nota: Analisar criteriosamente esta situação: muito limitativa.

- **Associações Unárias M:N**

  Esta situação é representada por 2 tabelas, 1 para a entidade, 1 para a associação.

  - PECA(codpec, ...)
  - ESTRUTURA(codpecpai, codpecfil)



Finalizado o processo de transformação do modelo conceitual em um modelo relacional passa-se então para o processo de normalização.

4.6.NORMALIZAÇÃO

Por que Normalizar? Após a construção do modelo conceitual dos dados é feita a transformação para um modelo lógico(esquema relacional). O esquema relacional obtido representa a estrutura de informação de um modelo natural e completo. **Mas terá o mínimo de redundância possível?**

A normalização tem por objetivo avaliar a qualidade do esquema relacional e transformá-lo(em caso de necessidade) num esquema relacional equivalente, menos redundante e mais acessível.

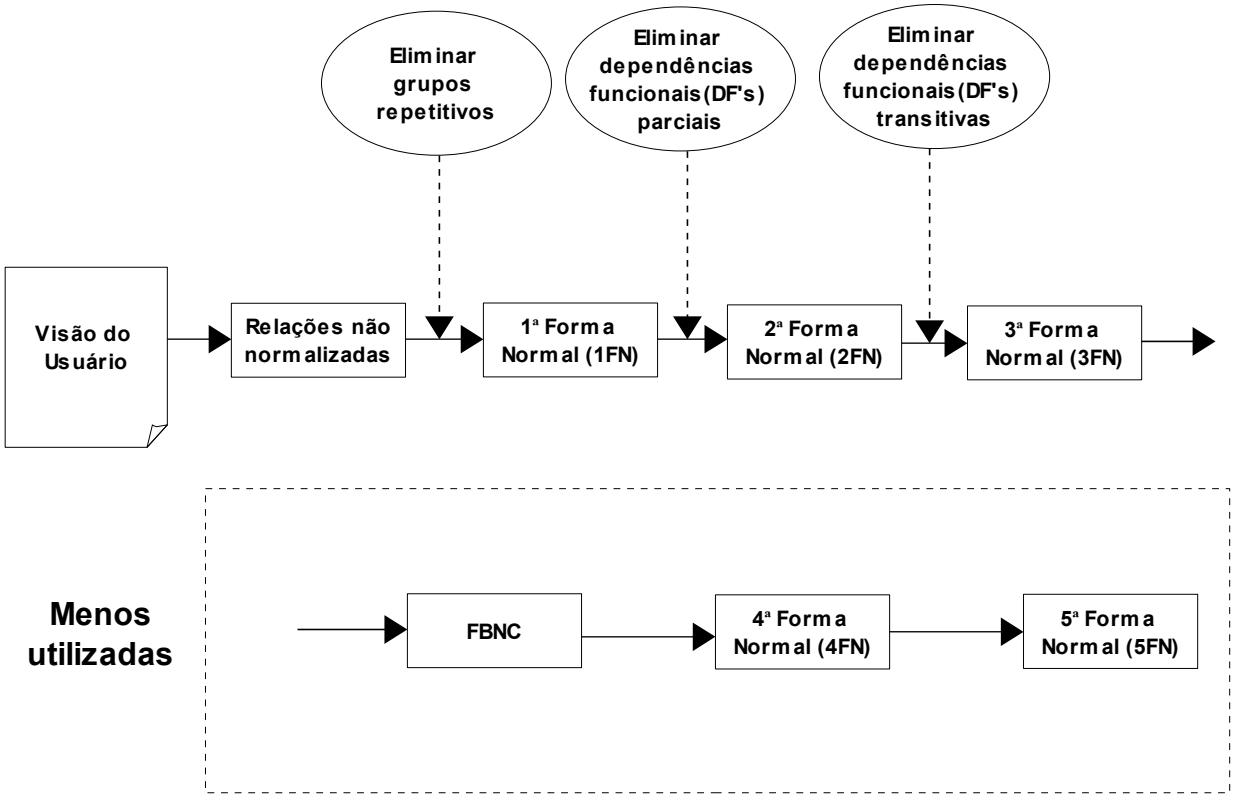


Figura 7 - Etapas do Processo de Normalização.

Dependência Funcional: Em uma tabela relacional diz-se que uma coluna C2 depende funcionalmente de uma coluna C1(ou que uma coluna C1 determina a coluna C2), quando todas as linhas da tabela, para cada valor de C1 que aparece na tabela, aparece o mesmo valor de C2.

Por vezes dois atributos(ou dois grupos de atributos) estão intrinsecamente ligados entre si.

Exemplo:

Num determinada instante, em qualquer ponto da base de dados que figurem estes 2 atributos, a um mesmo número de cliente corresponderá necessariamente o mesmo nome. O inverso não pode ser verdade.

Diz-se então que: Nome do Cliente depende do Número do Cliente

ou que: Número do cliente determina Nome do Cliente.

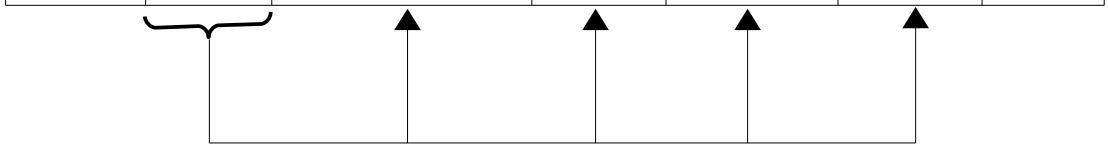
Dependências Funcionais Parciais: Uma dependência funcional parcial ocorre quando uma coluna depende apenas de parte de uma chave primária composta.

Exemplo de dependência parcial:

CHAVE

Relação NOTAS

NUMNOT	CODDIS	NOMDIS	CODPRO	NOMPRO	GRAPRO	NOTA
21934	04	Álgebra	21	Gil Algébrico	PA	15
21934	14	Análise de Sistemas	87	Ana Listada	PC	12
21934	23	Programação Linear	43	Plíneo	AS	16
42346	08	Topologia	32	Topo Lógico	AE	10
42346	04	Álgebra	21	Gil Algébrico	PA	12
42346	12	Geometria	21	Gil Algébrico	PA	18
42346	16	Lógica	32	Topo Lógico	AE	13
54323	04	Álgebra	21	Gil Algébrico	PA	11
54323	08	Topologia	32	Topo Lógico	AE	10



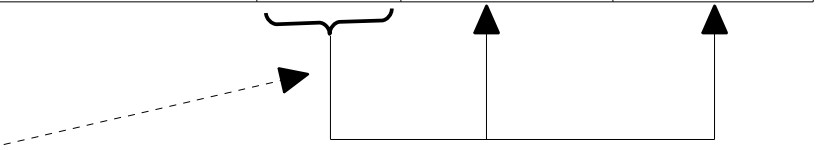
Obs.: Os atributos **NOMDIS, CODPRO, NOMPRO e GRAPRO** dependem apenas do atributo **CODDIS** (que está estritamente contido na chave da relação).

Dependências Funcionais Transitivas: Uma dependência funcional transitiva ocorre quando uma coluna, além de depender da chave primária da tabela, também depende de outra coluna ou conjunto de colunas da tabela que não são chave.

CHAVE

Relação DISCIPLINAS

CODDIS	NOMDIS	CODPRO	NOMPRO	GRAPRO
04	Álgebra	21	Gil Algébrico	PA
14	Análise de Sistemas	87	Ana Listada	PC
23	Programação Linear	43	Plíneo	AS
08	Topologia	32	Topo Lógico	AE
12	Geometria	21	Gil Algébrico	PA
16	Lógica	32	Topo Lógico	AE



Obs.: Os atributos **NOMPRO e GRAPRO** dependem do atributo **CODPRO** (que não é a chave da relação) e portanto **CODDIS -> CODPRO, CODDIS -> NOMPRO e CODDIS -> GRAPRO** não são

diretas.

Novas relações:

Relação PROFESSORES

CODPRO	NOMPRO	GRAPRO
21	Gil Algébrico	PA
87	Ana Listada	PC
43	Plíneo	AS
32	Topo Lógico	AE

Relação DISCIPLINAS

CODDIS	NOMDIS	CODPRO
04	Álgebra	21
14	Análise de Sistemas	87
23	Programação Linear	43
08	Topologia	32
12	Geometria	21
16	Lógica	32

O processo de normalização é composto pelas chamadas formas normais. A seguir as mesmas são descritas com maior detalhamento.

4.7.FORMAS NORMAIS

1) Primeira Forma Normal(1FN): A primeira forma normal resulta na eliminação de grupos repetitivos de atributos em uma entidade, logo, dizemos que uma entidade está na primeira formal normal, quando não possuir atributos com cardinalidade N.

Primeira FN: A aplicação da 1FN consiste em:

- Eliminar da Entidade os atributos com cardinalidade N, criando uma entidade para os mesmos, conforme seu agrupamento.
- Definir como chave primária desta nova entidade, a chave da entidade origem mais um atributo da nova entidade.

Exemplo:

CLIENTE	
1 - 1	CODCLI
0 - 1	NOMCLI
1 - 1	CEPCLI
1 - 1	CIDCLI

0 - N	NUMDEPENDENTES
0 - N	DATNASDEPENDENTES

CLIENTE	
1 - 1	CODCLI
0 - 1	NOMCLI
1 - 1	CEPCLI
1 - 1	CIDCLI

DEPENDENTE	
1 - 1	CODCLI
1 - 1	NUMDEPENDENTE
1 – 1	DATNASDEPENDENTE

2) Segunda Forma Normal(2FN): A 2FN resulta na eliminação de atributos que não dependam dos dois ou mais atributos da chave primária. Podemos observar então que aplica-se a 2FN, apenas em entidades que possuem chave primária composta, ou seja, entidades com apenas um atributo na chave primária já estão na 2FN.

Se não aplicarmos a 2FN para eliminarmos esta anomalia, os atributos que dependem apenas de parte da chave primária se repetirão N vezes na entidade, podendo gerar inconsistências nos dados armazenados.

A aplicação da 2FN consiste em:

- Eliminar da entidade os atributos que não dependam de todos os atributos da chave primária;
- Criar uma nova entidade com os atributos excluídos, mais os atributos da chave primária que causam a dependência dos atributos excluídos na entidade origem;
- Definir como chave primária desta nova entidade, os atributos herdados da chave primária da entidade origem.

3) Terceira Forma Normal(3FN): Resulta na eliminação de atributos que dependam de outros atributos que não façam parte da chave primária. Uma entidade está na 3FN quando todos os atributos da entidade dependam única e exclusivamente dos atributos da chave primária. Chamamos de anomalia de dependência transitiva, e quando não solucionada, os atributos com dependência transitiva irão se repetir na entidade, podendo gerar inconsistências nos dados armazenados.

A aplicação da 3FN consiste em:

- Eliminar da entidade os atributos que dependam de atributos que não pertencem a chave primária;
- Criar uma nova entidade com os atributos excluídos mais o atributo causador da dependência transitiva dos atributos excluídos;
- Definir como chave primária desta nova entidade, o atributo herdado da entidade de origem e que causou a dependência transitiva.

Exemplo:

CLIENTE	
1 - 1	CODCLI
0 - 1	NOMCLI
1 - 1	CEPCLI
1 - 1	CIDCLI

CLIENTE	
1 - 1	CODCLI
0 - 1	NOMCLI
1 - 1	CEPCLI
1 - 1	CODCID

CIDADE	
1 - 1	CODCID
1 - 1	NOMCID

Conclusão sobre Normalização: Uma entidade estará normalizada se estiver atendendo até a 3FN, ou seja, uma entidade estará normalizada se:

- Não possuir grupos repetitivos de atributos (1FN);
- Os atributos dependem de todos e unicamente de todos os atributos da chave primária(2FN e 3FN);

Um modelo de dados está normalizado se todas as suas entidades estão normalizadas até a 3 FN.

Outras Formas Normais:

- Boyce-Codd(FBNC): extensão a 3FN. Uma entidade está na FBNC se e somente se todos os determinantes forem chaves candidatas;
- 4FN: Aplica-se a atributos multivalorados. Uma entidade está na 4FN quando está na 3FN e não possui mais do que um fato a respeito de uma entidade descrita;
- 5FN: Uma entidade está na 5FN quando está na 4FN e quando seu conteúdo puder ser reconstruído a partir de diversas entidades menores que não possuam a mesma chave primária.

Desnormalização: A normalização ajuda a preservar a consistência dos dados, porém, requer navegação através de várias tabelas para composição da informação.

Quando desnormalizar?

- Observando o imperativo desempenho: *Data Warehouse*;
- Aumento ou redução da granularidade dos dados;
- Cálculos numerosos e complexos;
- BI(*Business Intelligence*): para poder modelar vários níveis de visão da informação.

Na sequência serão apresentadas algumas linguagens(incluindo as formais) para expressar consultas em banco de dados.

5.CÁLCULO RELACIONAL

O cálculo relacional é uma linguagem considerada não procedural. Nela os usuários definem as consultas em termos de "*O QUE*" ele quer sem descrever "*COMO*" obtê-la.

O cálculo relacional se divide em duas categorias: cálculo relacional de tupla e cálculo relacional de domínio.

1. **Cálculo relacional de tupla:** o objetivo é encontrar tuplas para cada vez que o predicado informado é verdadeiro. São utilizadas **variáveis de tupla**.

Expressão geral: { t | P(t) } = que significa o conjunto de todas as tuplas t, tal que o predicado P seja verdadeiro para t.

2. **Cálculo relacional de domínio:** Nesta forma são utilizadas variáveis de domínio que tomam valores do domínio de um atributo, em vez de valores da tupla.

Uma expressão neste cálculo tem a forma: {<x1, x2, ... , xn > | P(x1, x2, ..., xn)} = onde x1, x2, ... , xn representam variáveis de domínio e P representa uma fórmula composta de átomos, como no cálculo relacional de tupla.

Em seguida é apresentada a álgebra relacional.

6.ÁLGEBRA RELACIONAL

Antes de entrar diretamente no assunto relacionado à linguagem SQL é importante conhecer a álgebra relacional, que é considerada uma linguagem formal de interrogação. A álgebra relacional é uma linguagem procedural.

As operações da álgebra relacional se baseiam na teoria de conjuntos, onde os conjuntos são representados pelas relações ou tabelas. Dentro da álgebra relacional encontramos algumas operações consideradas fundamentais, como: projeção, seleção, produto cartesiano, união e diferença.

Além das operações fundamentais, existem ainda outras, como: interseção, ligação(junção normal e natural) e divisão.

A álgebra relacional compõe-se destes operadores para realizar manipulações de dados em um banco de dados relacional.

Além disso a álgebra relacional é a base lógica para a especificação da linguagem SQL, a estrutura de um comando SQL pode ser melhor compreendida tendo-se conhecimento destes conceitos até porque cada comando SQL pode ser desmembrado e representado através de uma expressão em álgebra relacional.

O poder da álgebra relacional também se encontra no fato de que cada operação tem como entrada um ou mais conjuntos, e como resultado obtém-se um novo conjunto. Partindo-se desta ideia é possível se encadear diversas operações relacionais em uma mesma expressão algébrica onde o resultado de uma operação sobre um conjunto alimenta a próxima operação. Isso aumenta muito o poder desta linguagem de consulta.

Alguns conceitos importantes envolvidos na álgebra relacional:

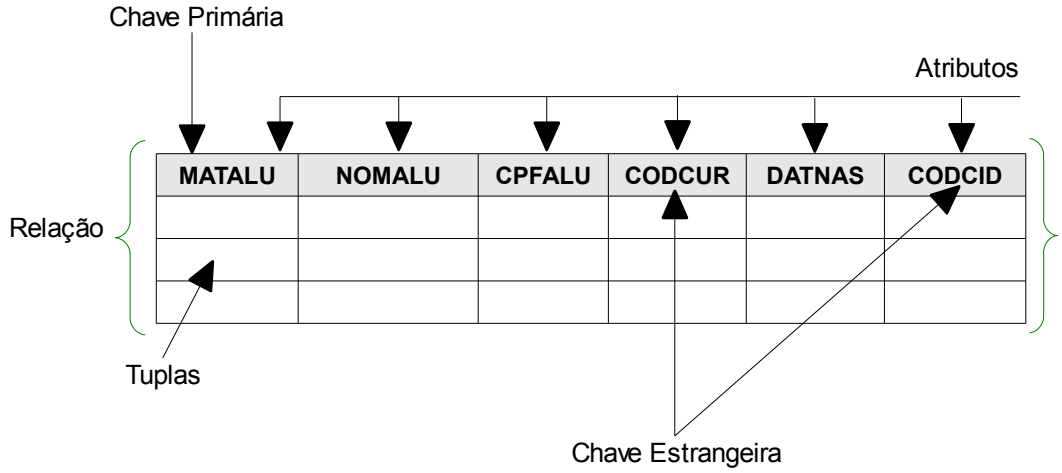
a) **Relação:** representada por por uma tabela de duas dimensões(linhas e colunas);

b) **Tupla:** corresponde a uma linha da relação;

c) **Atributo:** corresponde às colunas da relação;

d) **Chave primária:** conjuntos de atributos que identificam univocamente cada tupla da relação;

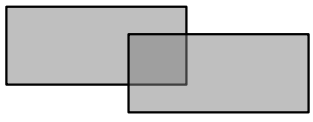
e) **Chave estrangeira:** atributo de uma relação que é chave primária de outra relação.



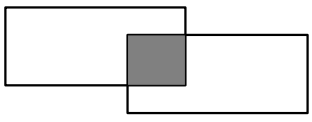
A álgebra relacional apresenta um conjunto de operações aplicáveis sobre modelos relacionais de dados. Estas operações podem ser agrupadas em duas categorias principais:

a) Operadores tradicionais: herdados da matemática, mais precisamente da teoria de conjuntos.

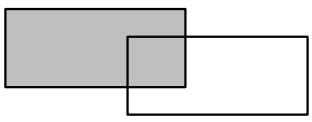
- União



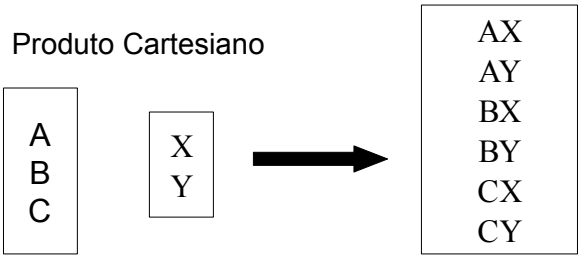
- Interseção



- Diferença



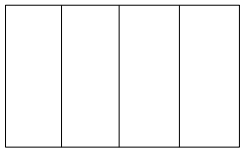
- Produto Cartesiano



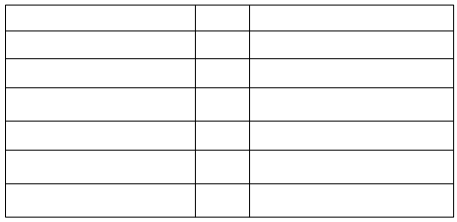
b) Operadores relacionais: criados propriamente para a álgebra relacional.

- Seleção

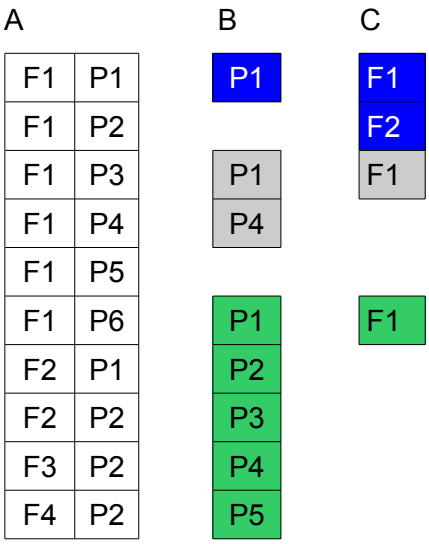
- Projeção



- Junção



- Divisão



Cada uma das operações mencionadas anteriormente será melhor explanada na sequência e para exemplificação tanto da álgebra relacional quanto da SQL serão utilizados os modelos de dados apresentados a seguir(Modelo 01 e Modelo 02).

MODELO DE DADOS 01

Tabela Cargo

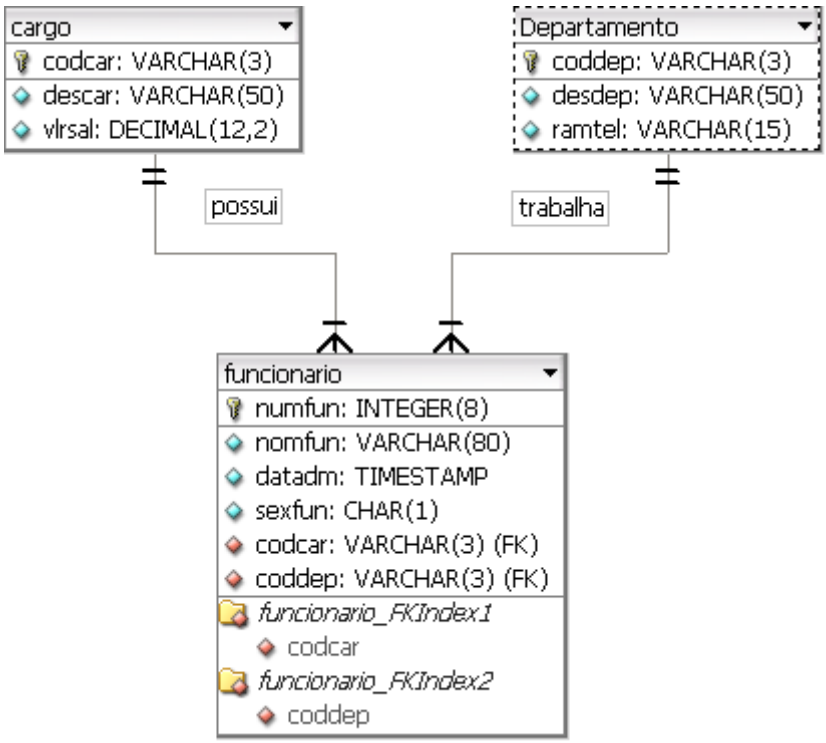
CODCAR	DESCAR	VLRSAL
C1	Aux. Vendas	350,00
C3	Vendedor	800,00
C7	Diretor	2500,00
C2	Vigia	400,00
C5	Gerente	1000,00
C4	Aux. Cobrança	250,00

Tabela Departamento

CODDEP	DESDEP	RAMTEL
D1	Assist. Técnica	2246
D2	Estoque	2589
D3	Administração	2772
D4	Segurança	1810
D5	Vendas	2599
D6	Cobrança	2688

Tabela Funcionario

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
101	Renato da Silva	10/08/2003	M	C3	D5
104	Carlos Pereira	02/03/2004	M	C4	D6
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1
121	Antônio Machado	01/12/2001	M	C3	D5
195	Larissa Silva	15/01/2007	F	C1	D5
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6
148	Cláudia de Souza	06/01/2008	F	C4	D3
115	João Alves	15/10/2003	M	C3	D5
22	Marcos Coimbra	10/02/2000	M	C2	D4



MODELO DE DADOS 02

Tabela Usuario

CODUSU	NOMUSU	ENDUSU	CEPUSU
01	Pedro da Silva	Rua Ab	890
02	Renata de Souza	Rua Cal	980
03	Casemiro Alves	Rua Ypr	765
04	Aristides da Cunha	Rua Hbc	546
05	Rafael dos Santos	Rua Ab	890
06	Jonny English	Rua Rtf	809

Tabela Livro

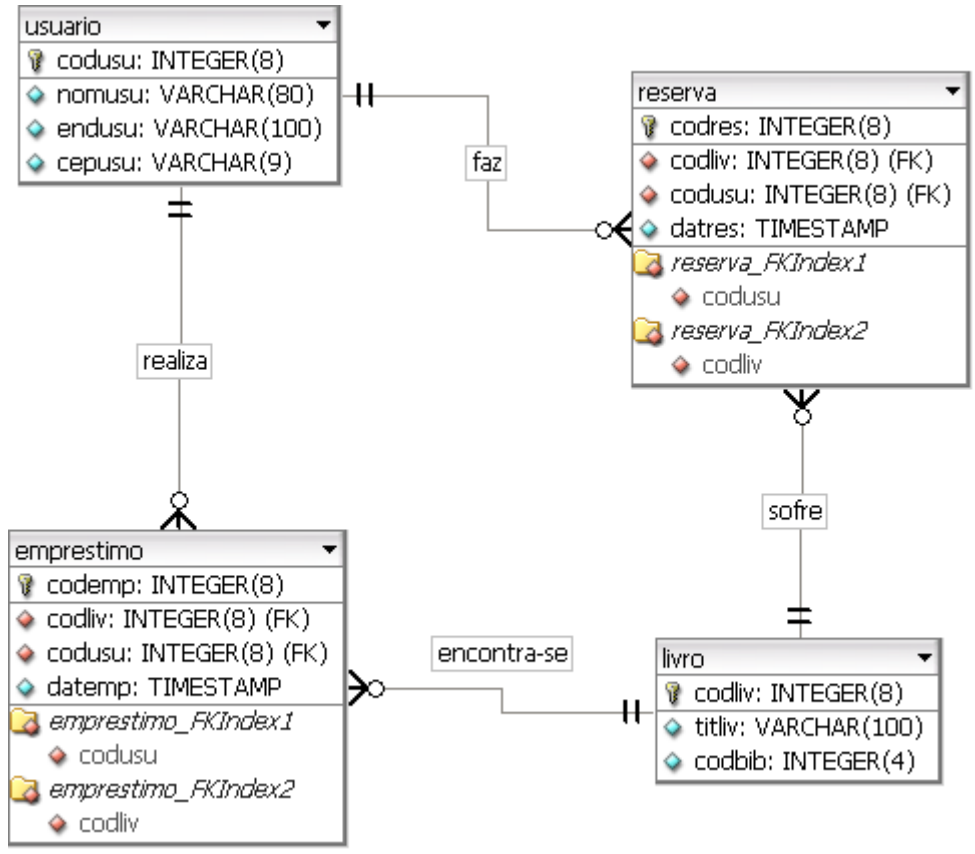
CODLIV	TITLIV	CODBIB
1001	LIVRO 01	1
1002	LIVRO 02	1
1003	LIVRO 03	2
1004	LIVRO 04	3
1005	LIVRO 05	3
1006	LIVRO 06	4

Tabela Emprestimo

CODEMP	CODLIV	CODUSU	DATEMP
20002	1001	01	13/01/2008
20005	1002	02	13/01/2008
20006	1004	01	14/01/2008
20007	1005	06	14/01/2008
20008	1001	05	02/02/2008
20010	1006	06	05/02/2008
20001	1005	04	12/01/2008
20009	1003	03	04/02/2008
20011	1002	01	04/03/2008

Tabela Reserva

CODRES	CODLIV	CODUSU	DATRES
30001	1002	02	11/01/2008
30002	1002	03	13/01/2008
30003	1002	05	23/01/2008
30004	1003	04	01/02/2008
30005	1004	03	01/02/2008
30006	1001	01	02/02/2008
30007	1005	01	03/02/2008
30008	1006	06	04/02/2008
30009	1006	02	07/02/2008
30010	1005	04	07/02/2008



6.1.OPERAÇÃO DE PROJEÇÃO(UNÁRIA)

A operação é representada pelo símbolo π(letra PI do alfabeto grego). É uma operação usada para construir um subconjunto vertical de uma tabela(ou relação), cujas linhas satisfaçam uma determinada condição que nada mais é que uma lista de colunas. A operação de projeção pode ser também entendida como um filtro de colunas(ou atributos) de uma determinada tabela.

Esta operação atua somente sobre uma tabela por isso é considerada **unária**.

Sua sintaxe é:

$$\pi_{\text{nome coluna}_1, \text{ nome coluna}_2, ..., \text{ nome coluna}_n}(\text{NOME TABELA})$$

NOME TABELA = Conjunto de entrada para a operação
COLUNAS = Argumentos da operação

A operação de projeção age sobre somente uma tabela ou tabela resultante de alguma outra operação da álgebra relacional.

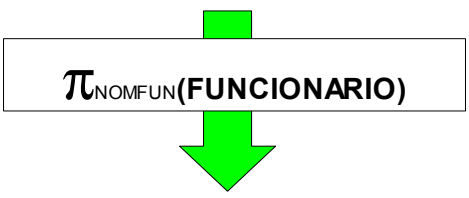
Exemplos de aplicação da operação de projeção:

E-01) Realizar a projeção da coluna *NOMFUN* da tabela *FUNCIONARIO*.

Expressão: $\pi_{\text{NOMFUN}}(\text{FUNCIONARIO})$

resolvendo ...

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
101	Renato da Silva	10/08/2003	M	C3	D5
104	Carlos Pereira	02/03/2004	M	C4	D6
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1
121	Antônio Machado	01/12/2001	M	C3	D5
195	Larissa Silva	15/01/2007	F	C1	D5
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6
148	Cláudia de Souza	06/01/2008	F	C4	D3
115	João Alves	15/10/2003	M	C3	D5
22	Marcos Coimbra	10/02/2000	M	C2	D4



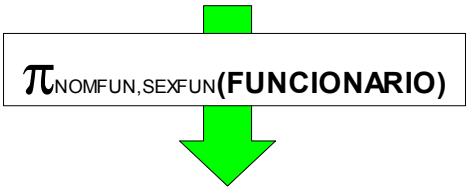
NOMFUN
Renato da Silva
Carlos Pereira
Pedro Sérgio Doto
Antônio Machado
Larissa Silva
Rejane da Cunha
Marcos Vinicios da Penha
Cláudia de Souza
João Alves
Marcos Coimbra

E-02) Realizar a projeção da coluna *DESFUN* e *SEXFUN* da tabela *FUNCIONARIO*.

Expressão: $\pi_{\text{NOMFUN,SEXFUN}}$**(FUNCIONARIO)**

resolvendo:

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
101	Renato da Silva	10/08/2003	M	C3	D5
104	Carlos Pereira	02/03/2004	M	C4	D6
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1
121	Antônio Machado	01/12/2001	M	C3	D5
195	Larissa Silva	15/01/2007	F	C1	D5
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6
148	Cláudia de Souza	06/01/2008	F	C4	D3
115	João Alves	15/10/2003	M	C3	D5
22	Marcos Coimbra	10/02/2000	M	C2	D4



NOMFUN	SEXFUN
Renato da Silva	M
Carlos Pereira	M
Pedro Sérgio Doto	M
Antônio Machado	M
Larissa Silva	F
Rejane da Cunha	F

Marcos Vinicios da Penha	M
Cláudia de Souza	F
João Alves	M
Marcos Coimbra	M

Com base nos exemplos apresentados percebe-se que a operação de projeção gera apenas uma nova visão da tabela fornecida como entrada, mostrando somente as colunas desejadas.

É muito importante ressaltar que as colunas informadas como argumentos da operação de projeção devem existir na tabela fornecida como entrada.

Em muitos casos faz-se necessária, além da projeção de somente algumas colunas de uma tabela, a recuperação de somente um número limitado de ocorrências(linhas) da tabela de entrada. Por exemplo se quisermos recuperar somente os funcionários do departamento D6. Para isso utiliza-se a operação de seleção, apresentada a seguir.

6.2.OPERAÇÃO DE SELEÇÃO(UNÁRIA)

A operação é representada pelo símbolo σ (letra SIGMA do alfabeto grego).

É uma operação usada para construir um subconjunto horizontal de uma tabela(ou relação), cujas linhas satisfaçam uma determinada predicado ou condição. Em outras palavras, esta operação trabalha sobre uma tabela de entrada e com base em uma condição(chamado de **predicado**) retorna uma tabela semelhante em termos de estrutura mas apenas com as ocorrências(linhas) que atendam a condição estabelecida. A tabela retornada é um subconjunto da tabela fornecida como entrada.

Sua sintaxe é:

$\sigma_{\textbf{predicado(NOME TABELA)}}$

NOME TABELA = Conjunto de Entrada para a operação
PREDICADO = Predicado da operação(condição para a operação)

A operação de seleção também é uma operação de filtragem, mas neste caso são filtradas as linhas e não as colunas da tabela de entrada.

A operação de seleção também é uma operação **unária**.

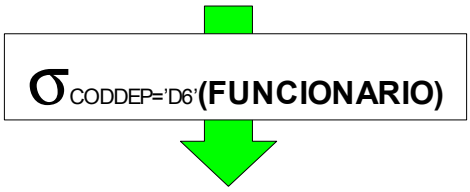
Exemplos de aplicação da operação de seleção:

E-03) Selecionar/exibir/mostrar os funcionários que trabalham no departamento *D6*(Coluna *CODDEP*) da tabela *FUNCIONARIO*.

Expressão: $\sigma_{CODDEP='D6'}$**(FUNCIONARIO)**

resolvendo ...

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
101	Renato da Silva	10/08/2003	M	C3	D5
104	Carlos Pereira	02/03/2004	M	C4	D6
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1
121	Antônio Machado	01/12/2001	M	C3	D5
195	Larissa Silva	15/01/2007	F	C1	D5
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6
148	Cláudia de Souza	06/01/2008	F	C4	D3
115	João Alves	15/10/2003	M	C3	D5
22	Marcos Coimbra	10/02/2000	M	C2	D4



NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
104	Carlos Pereira	02/03/2004	M	C4	D6
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6

Obs.: Neste primeiro exemplo podemos observar que em termos de colunas a estrutura após a aplicação da operação de seleção permanece a mesma. Somente as linhas que atendem o predicado informado como argumento são retornadas.

E-04) Projetar a coluna *NOMFUN*, *DATADM* da tabela *FUNCIONARIO* somente dos funcionários que são do departamento *D6*.

Expressão: $\pi_{NOMFUN,\ DATADM}(\sigma_{CODDEP='D6'}$**(FUNCIONARIO))**

Neste segundo exemplo queremos retornar somente os funcionários do departamento *D6* e projetar(exibir) somente as colunas *NOMFUN* e *DATADM*. Este caso já apresenta uma regra importante que deve ser respeitada quando operadores relacionais tiverem que ser aninhados. Para podermos projetar(exibir) somente as colunas desejadas de uma determinada tabela

devemos primeiro realizar a operação de seleção.

Este é um caso onde pode-se mostrar claramente como se dá a solução de cada operação bem como porque a álgebra relacional é considerada uma linguagem formal.

Então vamos resolver este exemplo:

Funcionários – Tabela Original

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
101	Renato da Silva	10/08/2003	M	C3	D5
104	Carlos Pereira	02/03/2004	M	C4	D6
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1
121	Antônio Machado	01/12/2001	M	C3	D5
195	Larissa Silva	15/01/2007	F	C1	D5
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6
148	Cláudia de Souza	06/01/2008	F	C4	D3
115	João Alves	15/10/2003	M	C3	D5
22	Marcos Coimbra	10/02/2000	M	C2	D4

Aplicando-se a operação de seleção para os funcionários do departamento D6, temos como resultado a tabela abaixo(tabela intermediária):

Tabela Intermediária

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
104	Carlos Pereira	02/03/2004	M	C4	D6
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6

A tabela intermediária resultante da operação de seleção agora servirá de entrada para a operação de projeção e o resultado é:

Tabela Final

NOMFUN	DATADM
Carlos Pereira	02/03/2004
Rejane da Cunha	12/01/2005
Marcos Vinicios da Penha	29/06/2003

Agora sim atingimos nosso objetivo, a tabela final representa o resultado da aplicação da nossa operação de seleção e de nossa operação de projeção.

Pode surgir a dúvida de porque por exemplo a coluna *CODDEP* da tabela *FUNCIONARIO* não apareceu em nossa tabela final.

É importante lembrar que somente as colunas fornecidas como argumentos para a operação de projeção serão exibidas ao final.

A formação de um predicado(condição) para a operação de seleção pode se utilizar ainda dos operadores booleanos(lógicos): **AND** ou **^**, **OR** ou **v** e **NOT**. Através destes podemos conectar várias expressões a serem testadas em nossa seleção, formando um único predicado.

Além disso ainda podemos utilizar os operadores de comparação apresentados a seguir:

Operador de Comparação	Descrição
=	Igual
<>	Menor ou Maior que(Diferente)
>	Maior que
<	Menor que
>=	Maior ou igual que
<=	Menor ou igual que

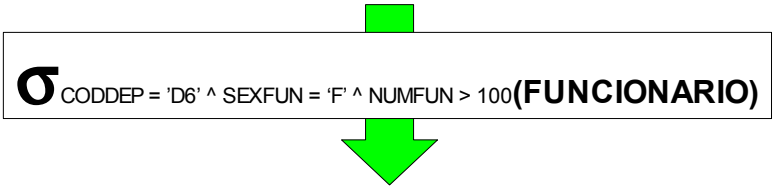
Um exemplo de uma expressão algébrica relacional utilizando os operadores lógicos e booleanos.

E-05) Selecionar/exibir/mostrar os funcionários que trabalham no departamento *D6*, que sejam do sexo Feminino(F) e cujo código do funcionário seja maior que 100.

Expressão: $\sigma_{\text{CODDEP = 'D6' ^ SEXFUN = 'F' ^ NUMFUN > 100}}(\textbf{FUNCIONARIO})$

resolvendo ...

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
101	Renato da Silva	10/08/2003	M	C3	D5
104	Carlos Pereira	02/03/2004	M	C4	D6
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1
121	Antônio Machado	01/12/2001	M	C3	D5
195	Larissa Silva	15/01/2007	F	C1	D5
139	Rejane da Cunha	12/01/2005	F	C4	D6
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6
148	Cláudia de Souza	06/01/2008	F	C4	D3
115	João Alves	15/10/2003	M	C3	D5
22	Marcos Coimbra	10/02/2000	M	C2	D4



NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP
139	Rejane da Cunha	12/01/2005	F	C4	D6

Para avançarmos mais em nosso estudo imaginemos uma situação onde queiramos selecionar somente os funcionários do departamento *D6* da tabela *FUNCIONARIO*, mas além de exibir as colunas *NOMFUN* e *DATADM* queremos também exibir a descrição *DESDEP* do departamento da tabela *DEPARTAMENTO*. Neste caso já temos mais de uma tabela em questão e quem pode nos ajudar a conseguir alcançar o resultado esperado é a operação chamada de produto cartesiano, descrita a seguir.

6.3.PRODUTO CARTESIANO(BINÁRIA)

A operação é representada pelo símbolo **X**.

O produto cartesiano de duas tabelas(relações) A e B é o conjunto de todas as linhas **t** originadas da concatenação das linhas(tuplas) **a** pertencentes a A e das linhas **b** pertencentes a B.

A operação de produto cartesiano é considerada binária, por ser aplicada sobre duas tabelas e o resultado de sua aplicação é uma terceira tabela com a combinação de todos os elementos de ambas as tabelas que serviram de entrada.

Um exemplo para explicar isso é: tendo-se uma tabela A, com 3 colunas e 6 linhas, e uma tabela B, com 4 colunas e 5 linhas tem-se após a aplicação do produto cartesiano:

3 colunas(tabela A) + 4 colunas(tabela B) = 7 colunas(nova tabela)
6 linhas(tabela A) X 5 linhas(tabela B) = 30 linhas(nova tabela)

Em termos práticos o que acontece na aplicação desta operação é a multiplicação de cada elemento(linhas) da primeira tabela por todos os elementos da segunda tabela. Não há um uso prático para este tipo de operação, dificilmente teremos interesse em saber todas as combinações possíveis entre duas tabelas, além disso esta é uma operação bastante “pesada”. Imagine uma tabela com 30 colunas e 3000 linhas multiplicada por uma tabela com 23 colunas e 25000 linhas, esta operação exigiria uma grande quantidade de recurso do SGBD para poder ser processada.

Mas apesar disso esta é uma das formas mais básicas que permite realizar a junção de tabelas.

Sua sintaxe é:

TABELA A X TABELA B

Vejamos exemplos da aplicação do produto cartesiano.

E-06) Aplicando o produto cartesiano entre as tabelas *FUNCIONARIO* e *CARGO*.

Expressão: **FUNCIONARIO** $\mathbf{X}$ **CARGO**

resolvendo ...

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP	CODCAR	DESCAR	VLRSAL
101	Renato da Silva	10/08/2003	M	C3	D5	C1	Aux. Vendas	350,00
101	Renato da Silva	10/08/2003	M	C3	D5	C3	Vendedor	800,00
101	Renato da Silva	10/08/2003	M	C3	D5	C7	Diretor	2500,00
101	Renato da Silva	10/08/2003	M	C3	D5	C2	Vigia	400,00
101	Renato da Silva	10/08/2003	M	C3	D5	C5	Gerente	1000,00
101	Renato da Silva	10/08/2003	M	C3	D5	C4	Aux. Cobrança	250,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C1	Aux. Vendas	350,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C3	Vendedor	800,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C7	Diretor	2500,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C2	Vigia	400,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C5	Gerente	1000,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C4	Aux. Cobrança	250,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C1	Aux. Vendas	350,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C3	Vendedor	800,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C7	Diretor	2500,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C2	Vigia	400,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C5	Gerente	1000,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C4	Aux. Cobrança	250,00
...	...	...	...	...	...	...	...	...

Acima é apresentado um resultado parcial da aplicação da operação de produto cartesiano.

Agora conhecendo um pouco da operação produto cartesiano vamos tentar resolver a seguinte expressão algébrica:

Expressão: $\pi_{\text{NOMFUN, DATADM}}(\sigma_{\text{DESCAR='Vigia'}}(\textbf{FUNCIONARIO } \mathbf{X} \textbf{ CARGO}))$

Nesta expressão acima queremos recuperar o nome e a data de admissão dos funcionários cujo cargo é Vigia. Para resolvermos esta expressão aplicamos inicialmente a operação de produto cartesiano, depois a operação de seleção e finalmente a operação de projeção.

Neste exemplo percebemos a integração destas operações da álgebra relacional.

Vamos ver o resultado parcial da operação:

NUMFUN	NOMFUN	DATADM	SEXFUN	CODCAR	CODDEP	CODCAR	DESCAR	VLRSAL
101	Renato da Silva	10/08/2003	M	C3	D5	C2	Vigia	400,00
104	Carlos Pereira	02/03/2004	M	C4	D6	C2	Vigia	400,00
134	Pedro Sérgio Doto	23/05/2003	M	C5	D1	C2	Vigia	400,00
121	Antônio Machado	01/12/2001	M	C3	D5	C2	Vigia	400,00
195	Larissa Silva	15/01/2007	F	C1	D5	C2	Vigia	400,00
139	Rejane da Cunha	12/01/2005	F	C4	D6	C2	Vigia	400,00
123	Marcos Vinicios da Penha	29/06/2003	M	C7	D6	C2	Vigia	400,00
148	Cláudia de Souza	06/01/2008	F	C4	D3	C2	Vigia	400,00
115	João Alves	15/10/2003	M	C3	D5	C2	Vigia	400,00
22	Marcos Coimbra	10/02/2000	M	C2	D4	C2	Vigia	400,00

=

NOMFUN	DATADM
Renato da Silva	10/08/2003
Carlos Pereira	02/03/2004
Pedro Sérgio Doto	23/05/2003
Antônio Machado	01/12/2001
Larissa Silva	15/01/2007
Rejane da Cunha	12/01/2005
Marcos Vinicios da Penha	29/06/2003
Cláudia de Souza	06/01/2008
João Alves	15/10/2003
Marcos Coimbra	10/02/2000

Bom, vamos ver se atingimos nosso resultado final esperado. Observando-se o que foi retornado, acredito que não, porque todos os funcionários foram retornados como tendo o cargo de vigia e sabemos que isso não é verdade, portanto existe algum erro em nossa expressão.

Para conseguirmos atender a ideia inicial de nossa expressão precisamos especificar explicitamente como um funcionário está ligado ao seu cargo, ou seja, qual cargo efetivamente ele possui.

Então para resolvermos precisamos fazer a ligação entre as duas tabelas de uma maneira que se possa identificar exatamente qual é o cargo de um determinado funcionário, e para fazer isso vamos utilizar o conceito de Chave Estrangeira. Observando-se as tabelas percebe-se que é possível ainda se cruzar as colunas *CODCAR* da tabela *FUNCIONARIO* e *CODCAR* da tabela *CARGO*.

Neste caso teremos dois predicados ou condições que deverão ser atendidas, e somente os elementos das tabelas que os atenderem deverão ser retornados.

Para resolver este caso poderemos utilizar o operador booleano **AND** ou **^**.

Expressão: πNOMFUN, DATADM(σDESCAR='Vigia' ^ FUNCIONARIO.CODCAR = CARGO.CODCAR(**FUNCIONARIO** X **CARGO))**

Agora sim, aplicando-se esta expressão algébrica relacional obteremos o resultado que desejamos.

Tabela resultado

NOMFUN	DATADM
Marcos Coimbra	10/02/2000

A próxima operação que vamos conhecer é a operação de renomear.

6.4.OPERAÇÃO DE RENOMEAR

Esta operação é representada pelo símbolo ρ(letra RHO do alfabeto grego).

A operação de renomear é utilizada em casos onde necessitamos renomear uma tabela, caso esta tabela apareça mais de uma vez em uma consulta. Podemos utilizá-la também para identificar resultados de expressões em álgebra que ainda não possuem nome definido.

Sua sintaxe é:

ρ**<novo nome>(TABELA)**

Vamos a um exemplo com esta operação: Quais são os funcionários que trabalham no mesmo departamento da 'Larissa Silva'.

Expressão na sua forma geral:

πFUNCIONARIO2.NOMFUN, FUNCIONARIO2.CODDEP(σFUNCIONARIO2.CODDEP = FUNCIONARIO.CODDEP((σNOMFUN = 'Larissa Silva'(**FUNCIONARIO**)) X (ρFUNCIONARIO2(**FUNCIONARIO**)))

Vamos resolver a primeira situação que temos: qual é afinal o departamento da

Larissa?

Expressão: $\pi_{\text{CODDEP}}(\sigma_{\text{NOMFUN = 'Larissa Silva'}}(\text{FUNCIONARIO})$

resultando em ...

CODDEP
D5

Então no relacionamento com a tabela *FUNCIONARIO2* serão retornados somente os funcionários do departamento 'D5'.

Por fim precisamos apenas projetar as colunas do funcionário que desejamos apresentar na tela.

Que seria? ...

NOMFUN	CODDEP
Renato da Silva	D5
Antônio Machado	D5
Larissa Silva	D5
João Alves	D5

Vamos agora conhecer a operação binária de união.

6.5.OPERAÇÃO DE UNIÃO(BINÁRIA)

Esta operação é representada pelo símbolo $\cup$.

A união de duas tabelas(relações) A e B é o conjunto de todas as linhas(tuplas) pertencentes a A mais as linhas(tuplas) pertencentes a relação B.

A operação de união também é uma operação binária, pois precisa de duas tabelas que serão unidas e o resultado produzido será uma tabela com o mesmo número de colunas das tabelas de entrada. **No resultado da união de duas tabelas os registros repetidos são retornados apenas uma única vez.**

Outro ponto importante que deve ser observado é que a união de duas tabelas só é possível se ambas possuírem o mesmo número de colunas e seus domínios(tipos) devem ser equivalentes, ou seja, as tabelas devem ser compatíveis.

Sua sintaxe é:

TABELA A $\cup$ TABELA B

Agora utilizando o segundo modelo de dados vamos tentar resolver o seguinte problema: recuperar o nome de todos os usuários que possuem empréstimo **OU** reserva de livros.

Bem para reduzirmos nosso trabalho vamos recuperar os dados para o livro 1005(CODLIV = 1005).

Vamos resolver a primeira situação que temos:

Expressão: $\pi_{\text{NOMUSU}}(\sigma_{\text{CODLIV = 1005 ^ EMPRESTIMO.CODUSU = USUARIO.CODUSU}}(\text{USUARIO} \times \text{EMPRESTIMO}))$

Neste caso vamos recuperar todos os usuários que fizeram empréstimo do livro 1005. Aplicando a operação de seleção:

CODUSU	NOMUSU	ENDUSU	CEPUSU	CODEMP	CODLIV	CODUSU	DATEMP
04	Aristides da Cunha	Rua Hbc	546	20001	1005	04	12/01/2008
06	Jonny English	Rua Rtf	809	20007	1005	06	14/01/2008

Devido a projeção aplicada nosso resultado final é:

NOMUSU
Aristides da Cunha
Jonny English

Agora vamos a segunda parte, saber quem já fez reservas do livro 1005(CODLIV = 1005):

Expressão: $\pi_{\text{NOMUSU}}(\sigma_{\text{CODLIV = 1005 ^ RESERVA.CODUSU = USUARIO.CODUSU}}(\text{USUARIO} \times \text{RESERVA}))$

Neste caso vamos recuperar todos os usuários que fizeram reserva do livro 1005. Aplicando a operação de seleção:

CODUSU	NOMUSU	ENDUSU	CEPUSU	CODRES	CODLIV	CODUSU	DATRES
01	Pedro da Silva	Rua Ab	890	30007	1005	01	03/02/2008
04	Aristides da Cunha	Rua Hbc	546	30010	1005	04	07/02/2008

Devido a projeção aplicada nosso resultado final é:

NOMUSU
Pedro da Silva
Aristides da Cunha

Agora já sabemos quem já realizou empréstimos ou já realizou reservas. Percebam que o objetivo de realizar as operações separadamente é gerar duas tabelas compatíveis que servirão de entrada para a operação de união.

Expressão:

$$\pi_{NOMUSU}(\sigma_{CODLIV = 1005 \wedge EMPRESTIMO.CODUSU = USUARIO.CODUSU}(\textbf{USUARIO} \times \textbf{EMPRESTIMO}))$$

$$\textbf{U}$$

$$\pi_{NOMUSU}(\sigma_{CODLIV = 1005 \wedge RESERVA.CODUSU = USUARIO.CODUSU}(\textbf{USUARIO} \times \textbf{RESERVA}))$$

Resultados parciais de nossas expressões apurados anteriormente:

1)

NOMUSU
Aristides da Cunha
Jonny English

2)

NOMUSU
Pedro da Silva
Aristides da Cunha

Aplicando-se a operação de união obteremos o seguinte resultado:

NOMUSU
Aristides da Cunha
Jonny English
Pedro da Silva

Para uma operação de união entre duas tabelas ser possível deve-ser observar as seguintes regras:

- As tabelas A e B precisam ter o mesmo número de colunas;
- Estas colunas precisam ter o mesmo domínio, ou seja, ser do mesmo tipo; e
- As colunas devem estar na mesma ordem ou sequência.

No exemplo abaixo a aplicação da operação de união não funcionaria:

CODUSU	NOMUSU	ENDUSU	CEPUSU
01	Pedro da Silva	Rua Ab	890
04	Aristides da Cunha	Rua Hbc	546

CEPUSU	NOMUSU	CODUSU	ENDUSU
890	Pedro da Silva	01	Rua Ab
546	Aristides da Cunha	04	Rua Hbc

A seguir vamos conhecer a operação de interseção.

6.6.OPERAÇÃO DE INTERSEÇÃO(BINÁRIA)

A operação é representada pelo símbolo ∩.

A interseção de duas tabelas(relações) A e B é o conjunto de todas as linhas(tupla) pertencentes A e também pertencentes a B.

Esta é uma outra operação típica de conjuntos. É considerada binária já que para sua resolução são necessárias serem informadas como entrada duas tabelas e produz como resultado outra tabela com todos os elementos, **sem repetição**, que são comuns as duas tabelas de entrada.

Da mesma forma como acontece na operação de união as tabelas devem ser compatíveis.

Sua sintaxe é:

TABELA A ∩ TABELA B

Algumas combinações possíveis que podem fornecer o mesmo resultado da aplicação da operação de interseção são:

a) Tabela A ∩ Tabela B = Tabela A − (Tabela A − Tabela B)

b) = (Tabela A ∪ Tabela B) − (Tabela A − Tabela B) − (Tabela B − Tabela A)

c) = (Tabela A ∪ Tabela B) − ((Tabela A − Tabela B) ∪ (Tabela B − Tabela A))

Agora vamos exercitar a operação de interseção. Utilizando o mesmo modelo da operação anterior vamos tentar descobrir quais usuários(nome) emprestaram e reservaram o livro 1005(CODLIV = 1005).

Para este caso teremos que utilizar o operador booleano **AND** já que queremos que ambas as condições sejam satisfeitas.

Nossa expressão é representada abaixo:

Expressão:

$$\pi_{\text{NOMUSU}}(\sigma_{\text{CODLIV = 1005 ^ EMPRESTIMO.CODUSU = USUARIO.CODUSU}}(\textbf{USUARIO} \times \textbf{EMPRESTIMO}))$$

$$\cap$$

$$\pi_{\text{NOMUSU}}(\sigma_{\text{CODLIV = 1005 ^ RESERVA.CODUSU = USUARIO.CODUSU}}(\textbf{USUARIO} \times \textbf{RESERVA}))$$

Vamos aos resultados parciais de nossas expressões:

1)

NOMUSU
Aristides da Cunha
Jonny English

2)

NOMUSU
Pedro da Silva
Aristides da Cunha

Agora só precisamos verificar quais são as linhas comuns entre as duas tabelas, e como resultado final obtemos:

NOMUSU
Aristides da Cunha

Partiremos agora para o estudo de uma outra operação importante, a operação de diferença.

6.7.OPERAÇÃO DE DIFERENÇA(BINÁRIA)

A operação é representada pelo símbolo −.

A diferença de duas tabelas(relações) A e B é o conjunto de todas as linhas pertencentes a tabela A e não pertencentes a tabela B.

Esta operação tem a função de mostrar quais linhas existem na primeira tabela da operação e não existem na segunda tabela da operação. Também é uma operação binária por necessitar de duas tabelas de entrada.

Sua sintaxe é:

TABELA A − TABELA B

Utilizando as tabelas do modelo anterior da operação de interseção vamos agora ver como funciona a operação de diferença.

Nosso objetivo é encontrar todos os usuários que possuem empréstimo, mas que não tenham nenhuma reserva do livro 1005(CODLIV = 1005).

Expressão:

$$\pi_{\text{NOMUSU}}(\sigma_{\text{CODLIV = 1005 ^ EMPRESTIMO.CODUSU = USUARIO.CODUSU}}(\textbf{USUARIO} \times \textbf{EMPRESTIMO}))$$

$$-$$

$$\pi_{\text{NOMUSU}}(\sigma_{\text{CODLIV = 1005 ^ RESERVA.CODUSU = USUARIO.CODUSU}}(\textbf{USUARIO} \times \textbf{RESERVA}))$$

O resultado de nossa primeira expressão é:

NOMUSU
Aristides da Cunha
Jonny English

Estes usuários possuem empréstimos do livro 1005.

A segunda expressão resulta em:

NOMUSU
Pedro da Silva
Aristides da Cunha

Observando-se a regra da diferença obtêm-se como resultado:

NOMUSU
Jonny English

Este usuário tem empréstimo do livro 1005, mas não tem reserva do mesmo.

Como base no que foi apresentado podemos concluir que as expressões a seguir são muito diferentes:

Expressão 1: Tabela A – Tabela B

é diferente de

Expressão 2: Tabela B – Tabela A

A inversão das tabelas como mostrado no segundo caso nos levaria a um erro no resultado obtido que seria igual a:

NOMUSU
Pedro da Silva

Percebe-se no resultado obtido, que o usuário não possui empréstimos do livro 1005.

Vamos conhecer em seguida a operação de junção.

6.8.OPERAÇÃO DE JUNÇÃO(BINÁRIA)

A operação é representada pelo símbolo ⋈.

De duas tabelas(relações) *R1* e *R2*, que possuem um atributo em comum *D*, é o subconjunto do produto cartesiano das duas relações, cujos valores dos elementos dos atributos comuns sejam iguais nas duas tabelas.

Na tabela resultante elimina-se a repetição da coluna *D*. Normalmente a relação é feita através de chaves associadas entre duas tabelas. Esta também é uma operação binária.

Esta é uma operação de uso bastante comum, sua diferença com relação ao produto cartesiano é que o predicado de ligação(relação) e a junção das tabelas já é aplicado diretamente sobre a operação de seleção.

Sua sintaxe é:

TABELA A ⋈ A.CHAVE1 = B.CHAVE2TABELA B

A.CHAVE1 = B.CHAVE2 : predicado da seleção diretamente sobre o produto cartesiano

No resultado de uma operação de junção todas as colunas da primeira e da segunda tabela são recuperadas ocasionando a duplicação por exemplo da coluna que é a chave de ligação entre as duas tabelas.

Existem alguns tipos de junção, uma delas é chamada de *junção natural*, através deste tipo de junção o predicado(condição) não é informado e a recuperação é feita através do método de comparação de todas as colunas que são idênticas entre as duas tabelas. **Este tipo de junção não é recomendada se necessitamos combinar somente uma ou algumas colunas**.

Para o nosso caso utilizaremos a junção normal onde informamos o predicado de comparação e a combinação entre as duas tabelas.

Utilizando nosso modelo anterior vamos descobrir o seguinte: os usuários que

realizaram empréstimos e vamos exibir o cep de sua cidade.

Vamos representar nossa expressão:

Expressão: $\pi_{\text{NOMUSU, CEPUSU}}$**(USUARIO** $\bowtie_{\text{USUARIO.CODUSU = EMPRESTIMO.CODUSU}}$ **EMPRESTIMO)**

Para este caso poderíamos até utilizar uma junção natural, porque em comum as tabelas possuem somente uma coluna(*CODUSU*).

A representação da aplicação de numa junção natural ficaria assim:

Expressão: **USUARIO** $\bowtie$ **EMPRESTIMO**

Bem, nosso primeiro passo então é realizar a junção das tabelas, cujo resultado é:

CODUSU	NOMUSU	ENDUSU	CEPUSU	CODEMP	CODLIV	CODUSU	DATEMP
01	Pedro da Silva	Rua Ab	890	20011	1002	01	04/03/2008
01	Pedro da Silva	Rua Ab	890	20006	1004	01	14/01/2008
01	Pedro da Silva	Rua Ab	890	20002	1001	01	13/01/2008
02	Renata de Souza	Rua Cal	980	20005	1002	02	13/01/2008
03	Casemiro Alves	Rua Ypr	765	20009	1003	03	04/02/2008
04	Aristides da Cunha	Rua Hbc	546	20001	1005	04	12/01/2008
05	Rafael dos Santos	Rua Ab	890	20008	1001	05	02/02/2008
06	Jonny English	Rua Rtf	809	20010	1006	06	05/02/2008
06	Jonny English	Rua Rtf	809	20007	1005	06	14/01/2008

Realizando agora a projeção das colunas desejadas:

Expressão: $\pi_{\text{NOMUSU, CEPUSU}}$**(TABELA RESULTADO)**

ou com uma junção natural:

Expressão: $\pi_{\text{NOMUSU, CEPUSU}}$**(TABELA RESULTADO)**

resultaria em:

NOMUSU	CEPUSU
Pedro da Silva	890
Pedro da Silva	890
Pedro da Silva	890
Renata de Souza	980
Casemiro Alves	765
Aristides da Cunha	546

Rafael dos Santos	890
Jonny English	809
Jonny English	809

A próxima operação que vamos conhecer é a de divisão.

6.9.OPERAÇÃO DE DIVISÃO(BINÁRIA)

A operação é representada pelo símbolo / ou ÷.

Seja A uma tabela binária com os atributos **x** e **y** e B uma tabela unária com o atributo **z**, com **y** e **z** definidos sobre o mesmo domínio. Definimos que a operação de divisão, como sendo o conjunto dos elementos **x** com os pares(x,y) pertencentes a A para todos os valores pertencentes a B.

Vamos adotar a barra "/" para representar as operações de divisão.

Sua sintaxe é:

TABELA A / TABELA B

Para entendermos como funciona esta operação vamos tentar resolver a seguinte situação: recuperar todos os usuários que já emprestaram todos os livros da Biblioteca 1(CODBIB = 1).

Vamos começar então resolvendo nosso problema: precisamos descobrir quais são os livros da biblioteca 1.

Expressão: $\pi_{CODLIV}(\sigma_{CODBIB = 1}(\mathbf{LIVRO}))$

Como resultado obtemos a tabela:

CODLIV
1001
1002

Vamos encontrar agora todos os usuários que já emprestaram algum livro:

Expressão: $\pi_{\text{NOMUSU, CODLIV}}$**(USUARIO** $\bowtie$ **EMPRESTIMO)**

resultando em:

NOMUSU	CODLIV
Pedro da Silva	1002
Pedro da Silva	1004
Pedro da Silva	1001
Renata de Souza	1002
Casemiro Alves	1003
Aristides da Cunha	1005
Rafael dos Santos	1001
Jonny English	1006
Jonny English	1005

Agora vamos localizar cada um dos usuários que possuem todos os livros da biblioteca 2:

Expressão: *UsuarioLivro* <-- $\pi_{\text{NOMUSU, CODLIV}}$**(USUARIO** $\bowtie$ **EMPRESTIMO)**

$/$

LivroBiblioteca1 <-- π_{CODLIV}($\sigma_{\text{CODBIB = 1}}$**(LIVRO))**

Percebam que o atributo comum as duas tabelas é o *CODLIV*. E o que estamos procurando aqui são todos os conjuntos de linhas em UsuarioLivro cujos valores dos atributos comuns são iguais a todos os que aparecem em LivroBiblioteca1.

Temos, portanto, nosso divisor, tabela A, e nosso dividendo, tabela B, e procuramos nosso quociente:

NOMUSU	CODLIV
Pedro da Silva	1002
Pedro da Silva	1004
Pedro da Silva	1001
Renata de Souza	1002
Casemiro Alves	1003
Aristides da Cunha	1005

$/$

CODLIV
1001
1002

Rafael dos Santos	1001
Jonny English	1006
Jonny English	1005

Como resultado final obtemos:

NOMUSU
Pedro da Silva

Vamos partir agora para a parte de comandos SQL, mas antes disso é apresentada a seguir um quadro para consulta das operações de álgebra vistas até aqui, isto facilitará a busca pelas operações.

Símbolo	Operação	Sintaxe
π	Projeção	$\pi_{COLUNA_1, COLUNA_2, ..., COLUNA_N}$**(NOME TABELA)**
σ	Seleção	$\sigma_{PREDICADO}$**(NOME TABELA)**
$\cup$	União	Tabela A **U** Tabela B
$\cap$	Interseção	Tabela A $\cap$ Tabela B
-	Diferença	Tabela A - Tabela B
X	Produto Cartesiano	Tabela A X Tabela B
$\bowtie$	Junção	Tabela A $\bowtie$ Tabela B
/	Divisão	Tabela A / Tabela B
ρ	Renomeação	ρ<novo nome>(Tabela)

Na sequência da álgebra relacional, vamos conhecer a linguagem SQL.

7. SQL – STRUCTURED QUERY LANGUAGE

O surgimento da SQL se deu após o surgimento do modelo relacional. O modelo relacional conhecido hoje foi apresentado no final dos anos 60 pelo matemático Edgar Frank Codd tendo sua estrutura baseada em um modelo matemático.

O berço da SQL foi o surgimento do primeiro sistema gerenciador de banco de dados chamado System R. O projeto foi desenvolvido em 1973 pela IBM e provou que o modelo matemático relacional era viável.

Em 1974 surge a SEQUEL(*Structured English Query Language*), desenvolvida por Don Chamberlin e outros pesquisadores nos laboratórios da IBM, com o objetivo de expressar as operações do modelo relacional em um formato de linguagem declarativa utilizando-se de palavras comuns da língua inglesa. A linguagem sofreu revisões e alterações e teve sua sigla modificada posteriormente para SQL, mas sua pronúncia em inglês continua sendo *sequel*.

Na SQL o desenvolvedor diz "o que" ele deseja e não se preocupa "como" isso será executado. Por esta característica e também por ser declarativa seu surgimento foi uma grande novidade na época já que os SGBDs(Sistema Gerenciador de Banco de Dados) utilizados eram baseados nos modelos lógicos hierárquico ou em rede.

Apesar das pesquisas iniciais sobre SGBDR(Sistema Gerenciador de Banco de Dados Relacional) terem sido realizadas pela IBM, o primeiro banco de dados relacional comercial a surgir foi o ORACLE em 1979, desenvolvido na época pela Rational Software, hoje ORACLE Corporation. A IBM entrou no mercado em 1981 com o SQL/DS e consolidou-se em 1983 com o DB2.

No inicio dos anos 90 a maior parte dos SGBDRs começaram a integrar a SQL a suas implementações e com isso percebeu-se a necessidade e importância de se padronizar o uso da SQL a fim de permitir portabilidade entre os diversos produtos que surgiram.

Em 1986 a ANSI(*American National Standards Institute*) publicou a primeira versão do padrão SQL. Este padrão ficou conhecido também como SQL-86. No ano seguinte esta norma foi publicada como sendo internacional pela ISO(*International Organization for Standardization*). Em 1989 saiu uma nova publicação da norma abrangendo novos recursos.

Algumas versões da SQL publicadas: SQL-92(1992, também conhecida como SQL-2), SQL:1999(1999, também conhecida como SQL3), SQL:2003(2003) e a SQL:2006(2006).

A SQL pode ser utilizada em diversos enfoques dentro de uma estrutura de banco de dados relacional, como:

a) **Linguagem Interativa de Consulta (*Query AdHoc*):** permite a criação de consultas sem a necessidade da criação de programas específicos;

b) **Linguagem de Programação para acesso a banco de dados:** permite embutir comandos SQL em aplicações para acesso a dados armazenados;

c) **Linguagem de Administração de banco de dados:** as tarefas do DBA podem ser executadas através de comandos SQL;

d) **Linguagem cliente/servidor:** comandos SQL embutidos em aplicações acessando um único local na rede onde os dados estão armazenados;

e) **Linguagem para banco de dados distribuído:** a SQL pode auxiliar na

distribuição dos dados entre nós conectados a uma rede;

f) **Caminho de acesso a outros bancos de dados em diferentes máquinas:** a SQL auxiliar na conversão entre diferentes produtos de banco de dados distribuídos em diversas máquinas.

A SQL também apresenta algumas vantagens e desvantagens. Como vantagens diretas temos:

- ✔ **Independência de fabricante:** a SQL é um padrão adotado pela maior parte dos SGBDs, e para os que ainda não a usam logo o estarão. Por isso não existem grandes preocupações quando se precisa trocar o banco de dados em uso.
- ✔ **Portabilidade entre computadores:** Pode ser utilizada em computadores pessoais, *workstations* ou computadores de grande porte.
- ✔ **Redução dos custos com treinamento:** A reciclagem da equipe de desenvolvimento não se faz necessária já que as aplicações pode ser mudadas entre os diversos ambientes.
- ✔ **Inglês estruturado de alto nível:** o SQL é de fácil entendimento, pois é formado por um conjunto simples de palavras do inglês.
- ✔ **Consulta Interativa:** a SQL provê de forma rápida respostas ao usuário, tanto para questões mais simples como mais complexas.
- ✔ **Múltiplas visões dos dados:** Diferentes visões podem ser exibidas a diferentes usuários.
- ✔ **Definição dinâmica dos dados:** com uma grande flexibilidade pode-se alterar, incluir e excluir as estruturas dos dados armazenados.

Algumas desvantagens:

- ✗ Inibição da criatividade devido à padronização;
- ✗ Falta de ortogonalidade nas expressões;
- ✗ Falta de algumas funções;
- ✗ Erros(campos null, violação de chave primária, cláusula FROM, etc).

Características da linguagem SQL:

- Linguagem não procedimental em que se específica O QUÊ e não COMO
  - Existe uma clara abstração da estrutura física dos dados, não sendo necessário especificar caminhos de acesso nem algoritmos de pesquisa física;
- Operações sobre estruturas lógicas
  - As operação são efetuadas sobre conjuntos de dados(tabelas) não sendo necessário(nem possível) manipular-se dados linha-a-linha;

A linguagem SQL pode ser dividida nas seguintes partes ou componentes:

1) ***DDL (Data Definition Language*/Linguagem de Definição de Dados)**. A SQL DDL fornece comandos para definição e modificação de esquemas de relação, remoção de relações e criação de índices. Os principais comandos que fazem parte da DDL são: CREATE, ALTER, DROP.

2) ***DML (Data Manipulation Language*/Linguagem de Manipulação de Dados)**. A SQL DML inclui uma linguagem de consulta baseada na álgebra relacional e no cálculo relacional. Compreende também comandos para inserir, remover e modificar informações em um banco de dados. Os comandos básicos da DML são: SELECT, INSERT, UPDATE e DELETE.

3) ***DCL (Data Control Language*/Linguagem de Controle de Dados)**. É o conjunto de comandos que fazem o cadastramento de usuários e determina seu nível de acesso aos objetos do banco de dados. Os principais comandos são: CREATE USER, GRANT, REVOKE.

4) ***TML (Transaction Manipulation Language*/Linguagem de Manipulação de Transações)**. A SQL inclui comandos para especificação do início e fim das transações. Diversas implementações permitem o trancamento explícito de dados para o controle de concorrência (COMMIT, ROLLBACK, SAVEPOINT).

A seguir serão apresentados mais detalhadamente cada um dos componentes da SQL. É importante estar atento ao quadro de padrão de sintaxe adotado na apresentação dos comandos.

SQL – Padrão de Sintaxe	
Convenção	**Descrição**
Palavras em negrito	Representam palavras-chave e dados em um comando. Devem aparecer exatamente como mostrados em negrito.
{ }	As chaves indicam a distinção entre diferentes seções de um comando. Quando as chaves são seguidas de um “*”(asterisco), elas podem ser repetir 0 ou mais vezes no comando. Se as chaves forem seguidas de um “+”(mais) então a seção deve ocorrer ao menos uma vez.
[ ]	Os colchetes indicam que a seção é opcional
( )	Parênteses em negrito indicam que a estrutura deve aparecer como é mostrada. Parênteses que não estão em negrito indicam ordem lógica de avaliação das expressões.
...	O pontilhado indica que a seção pode ser repetida n vezes.
|	A barra vertical indica o valor “ou”.

Antes de entrarmos diretamente nos comandos, segue uma lista de objetos que podem ser encontrados em um banco de dados:

Objeto	Descrição
Espaços de tabela	Também chamados *Tablespaces*. São espaços lógicos para dividir os dados. São úteis quando desejamos separar determinados dados que geram muita fragmentação. Fragmentação para o SGBD, assim como para o Sistema Operacional, é sinônimo de lentidão na recuperação das informações.
Esquemas	Também chamados de *Schemas*. São divisões lógicas no SGBD utilizadas para separar os objetos. Muitas vezes a ideia de *Schema* se confunde com a ideia de usuário.
Funções	*Também chamadas de Functions*. Subalgoritmo implementado utilizando linguagem específica do SGBD em uso. Lembrando que função retorna valor.
Gatilhos	Também chamados de *Triggers*. Subalgoritmo implementado para ser disparado quando determinado eventos ocorrem no banco de dados. Um exemplo de uso de *triggers* é para auditoria, podemos disparar um evento de *log* todas vez que um registro for excluído de uma tabela. As *triggers* podem ser utilizadas *também* para manter a integridade da base de dados.
Índices	Também chamado de *Index*. Utilizado pra indexar informações no banco de dados. Muito utilizado para melhora de desempenho de consultas ao banco de dados.
Permissões	Também chamado de *Privileges*. São lista de permissões possíveis de serem concedidas em determinado SGBD. Existem permissões comuns a todos os SGBDs e também existem permissões específicas.
Procedimentos	Também chamados de *Procedures*. Subalgoritmo implementado utilizando linguagem específica do SGBD em uso. Lembrando que procedimento não retorna valor.
Usuários	Também chamados de *Users*. São os usuários propriamente ditos que poderão acessar o SGBD. É de responsabilidade do DBA gerenciar os usuários.
Visões	Também chamados de *Views*. São conhecidas como tabelas virtuais já que se apresentam em forma de tabela quando consultadas, no entanto, não mantém persistidos os dados consultas. As *views* nada mais são que consultas salvas no banco de dados.
Arquivos de Dados	Também chamados de *Datafiles*. É onde ficam armazenados os dados fisicamente. Seu gerenciamento é de responsabilidade do SGBD e não podem ser acessados sem passar pelo SGBD.

7.1. DML *(Data Manipulation Language/*Linguagem de Manipulação de Dados)

Comando: SELECT

Função: Selecionar(buscar, recuperar) dados de uma ou mais tabelas do banco de dados.

Sintaxe:

```
SELECT [EXPRESSÃO] [ tabela.coluna [AS] rotulo {, tabela.coluna }* ]
FROM tabela {, tabela }* [[AS] rotulo]
[ WHERE condição de busca ]
[ GROUP BY coluna {, coluna }* ]
[ HAVING condição ]
[ ORDER BY coluna {, coluna }* [ASC | DESC] ]
;
```

Opções		
EXPRESSÃO	**ALL**	Retorna todos os registros
	DISTINCT	Retorna os registros de forma distinta(sem repetição)
OPERADORES	**=**	Igual a
	!=, OU <>, OU ^=	Não igual a
	>	Maior que
	>=	Maior ou Igual a
	<	Menor que
	<=	Menor ou igual a
	BETWEEN condição	Entre dois valores
	IN(lista de valores)	Dentro da lista de valores
	LIKE	Que seja igual aos caracteres da amostra. Junto ao comando like podem ser utilizados os caracteres especiais(ou coringas) “_” e “%'
	IS NULL	Testa se é um valor nulo
	EXISTS	Se utiliza em comparações de verdade/falso para determinar se a subconsulta devolve algum registro.
	ANY	Utilizado para recuperar registros da consulta principal, que satisfaçam a comparação com qualquer outro registro recuperado na subconsulta.
	SOME	Sinônimo do comando ANY.
Obs.: Para a forma negativa pode ser utilizado o operador 'NOT'. Ex.: Not between, Is not null, Not like, Not exists, etc.		

E-07) Selecionar o código e nome dos funcionários do departamento 5.

SELECT NUMFUN, NOMFUN

FROM FUNCIONARIO

WHERE CODDEP = 'D5';

E-08) Selecionar o código, nome do funcionário e o nome do cargo do mesmo. Ordenar os dados pelo nome do funcionário.

SELECT NUMFUN, NOMFUN, DESCAR

FROM FUNCIONARIO, CARGO

WHERE FUNCIONARIO.CODCAR = CARGO.CODCAR

ORDER BY NOMFUN;

E-09) Recuperar usuários que realizaram pelo menos 1 empréstimo(Usando Exists).

SELECT CODUSU, NOMUSU

FROM USUARIO

WHERE EXISTS (**SELECT** 'X' **FROM** EMPRESTIMO **WHERE**

EMPRESTIMO.CODUSU = USUARIO.CODUSU);

E-10) Retornar todos os livros cujo código seja maior que o código de livros emprestados no dia '01/03/2008'.

SELECT CODLIV, TITLIV

FROM LIVRO

WHERE CODLIV > **ANY**(**SELECT** CODLIV **FROM** EMPRESTIMO **WHERE** DATEMP = '01/03/2008');

E-11) Retornar o código e nome dos usuários cujo cep é 890 e seu código é maior que 3.

SELECT CODUSU, NOMUSU

FROM USUARIO

WHERE CEPUSU = 890 **AND** CODUSU > 3;

Funções Aritméticas		
Função	**Sintaxe**	**Descrição**
ABS	ABS(balance)	Retorna o valor absoluto
ROUND	ROUND(valor, casas decimais)	Arredonda o valor para o número de casas especificado
TRUNC	TRUNC(valor, casas decimais)	Trunca o valor para o número de casas especificado
CEIL	CEIL(valor)	Retorna o menor inteiro maior ou igual ao valor
FLOOR	FLOOR(valor)	Retorna o maior inteiro menor ou igual ao valor
MOD	MOD(valor1, valor2)	Retorna o resto da divisão
POWER	POWER(valor, expoente)	Retorna um numero elevado a outro
SIGN	SIGN(valor)	Se valor maior que 0 retorna +1 Se valor menor que 0 retorna -1 Se valor igual a 0 retorna 0
SQRT	SQRT(valor)	Retorna a raiz quadrada do valor

Operador Lógico ou Booleano	**Descrição**
AND	E
OR	OU
NOT	NÃO(Negação)

Operador de Comparação	**Descrição**
=	Igual
<>	Menor ou Maior que(Diferente)
>	Maior que
<	Menor que
>=	Maior ou igual que
<=	Menor ou igual que

Operadores Aritméticos	
Operador	**Função**

+	Soma
-	Subtração
*	Multiplicação
/	Divisão
Precedência(prioridade): *, /, +, - da esquerda para direta	

Funções de Agregação		
Função	**Sintaxe**	**Descrição**
MAX	MAX(valor)	Retorna o maior valor
MIN	MIN(valor)	Retorna o menor valor
SUM	SUM(valor)	Retorna a soma dos valores
AVG	AVG(valor)	Retorna a média dos valores
COUNT	COUNT(valor | *)	Conta a quantidade de valores

Agrupamento: O *GROUP BY* é utilizado agregado as funções de agregação para realizar sumarização de dados, pode ser também utilizado para eliminar duplicidade de tuplas.

Filtro de Agrupamento: A expressão *HAVING* pode ser utilizada quando existe uma expressão *GROUP BY*, funcionando como um filtro para o agrupamento. Resumidamente, o *HAVING* é o *WHERE* do *GROUP BY*. Através do *HAVING* pode-se selecionar apenas os agrupamentos desejados que atendam determinadas condições. Os campos que fazem o filtro do *HAVING* não precisam necessariamente estar selecionados.

E-12) Número de funcionários por departamento, mas apenas para os departamentos com mais de 5 funcionários.

```
SELECT CODDEP, COUNT (*)
FROM EMPREGADO
GROUP BY CODDEP
HAVING COUNT (*) > 5;
```

E-13) Média salarial dos departamentos que tenham mais de 2 funcionários.

```
SELECT CODDEP, AVG(SALFUN)
FROM FUNCIONARIO
GROUP BY CODDEP
HAVING COUNT (*) > 2;
```

Como limitar o número de linhas retornadas por uma consulta?
R.: No caso do PostgreSQL podem ser utilizadas as cláusulas LIMIT e OFFSET. A

cláusula LIMIT permite indicar o número de linhas a serem exibidas seguindo a sintaxe: LIMIT { número de linhas | ALL }. O OFFSET indica o início da contagem de linhas. Um exemplo:

```
SELECT * FROM funcionario
LIMIT 10
OFFSET 2;
```

Naturalmente em um modelo relacional ocorrem situações onde existe a necessidade de relacionar mais de uma tabela em uma consulta SQL. Para isso utilizamos o conceito de junção. Estas junções podem ser realizadas implicitamente através do uso do operador *WHERE*, ou explicitamente através dos operadores de junção. Estes operadores de junção são explicados mais detalhadamente no quadro a seguir.

DICAS SQL: JUNÇÕES SQL E *SUBQUERYS*

JUNÇÕES SQL: As junções são operações que permitem consultar dados de várias tabelas. *Existem vários tipos de junções, dentre elas podem ser destacadas: self join, natural join, inner join, left join, right join, right outer join, left outer join , full outer join e o cross join.* Alguns dos exemplos de junções são acompanhados de um diagrama *Venn*(ilustrações utilizadas para representar conjuntos, relações matemáticas ou relações lógicas).

Para explicar o funcionamento as junções vamos basear os exemplos no seguinte esquema relacional: uma empresa que possui vários funcionários, que por sua vez residem em uma única cidade.

EMPRESA(CODEMP, NOMEMP, CEPEND)

CIDADE(CODCID, NOMCID)

FUNCIONARIO(CODFUN, NOMFUN, CODEMP, CODSUP, CODCID)

Sintaxe aplicável no comando SELECT: [INNER | NATURAL | {{ LEFT | RIGHT | FULL } [OUTER]} | [CROSS] JOIN tabela [ON (predicado de relacionamento)].

- ***Join*:** Implementação mais simples de junção. Permite recuperar dados de uma associação binária. Neste caso são recuperadas apenas as tuplas que satisfazem a condição de junção. Similar a utilização do INNER JOIN.

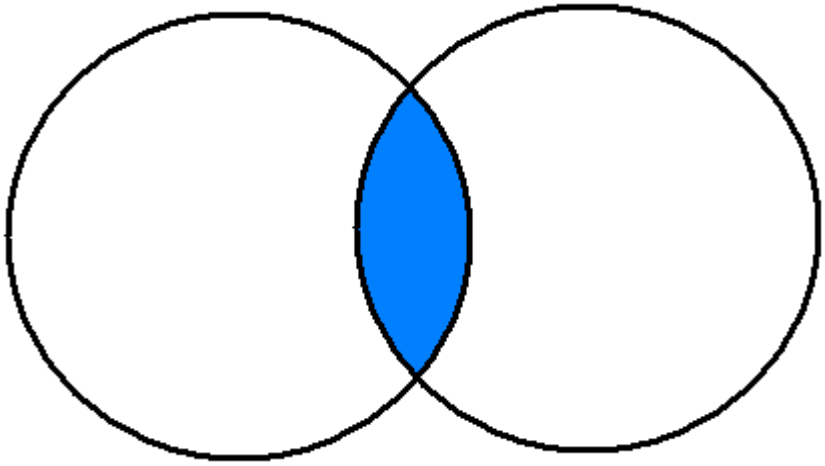
  Exemplo: Nome da cidade de cada funcionário?

  SELECT F.*, C.NOMCID
  FROM FUNCIONARIO F
  JOIN CIDADE C **ON** F.CODCID = C.CODCID;

  Utilizando a cláusula WHERE(o mesmo resultado):

  SELECT F.*, C.NOMCID
  FROM FUNCIONARIO F, CIDADE C
  WHERE F.CODCID = C.CODCID;

- *Representação Gráfica:*



FUNCIONARIO / CIDADE

- ***Self Join:*** Também chamado de auto-relacionamento, ou seja, uma tabela se relaciona com ela mesma. É através deste tipo de junção que podemos recuperar dados de associações unárias. Este comando também só recupera as tuplas que atenderem a condição de junção.
  Exemplo: Um funcionário possui um supervisor, que também é um funcionário. Então para responder a pergunta : O nome do supervisor de cada funcionário?
  SELECT F1.NOMFUN FUNC, F2.NOMFUN SUPER
  FROM FUNCIONARIO F1, FUNCIONARIO F2
  WHERE F1.CODSUP = F2.CODFUN;

- ***Natural Join:*** Realiza junções naturais entre duas tabelas, ou seja, relaciona as colunas com mesmo nome sem necessidade de informar a condição de relacionamento. Só retorna as tuplas que satisfazem a condição de junção.
  Exemplo: Qual o nome da empresa funcionário?
  SELECT F.*, E.NOMEMP
  FROM FUNCIONARIO F
  NATURAL JOIN EMPRESA;
  Obs.: Como a coluna em comum entre as duas tabelas é CODEMP então o comando relacionará estas duas colunas automaticamente para realizar a junção.

- ***Cross Join:*** Implementa o produto cartesiano entre duas tabelas. Se resume a combinar todas as linhas das duas tabelas realizando uma operação de multiplicação entre elas. Neste caso não existe condição de junção.

  Exemplo:

  SELECT F.*, C.*

  FROM FUNCIONARIO F

  CROSS JOIN CIDADE C;

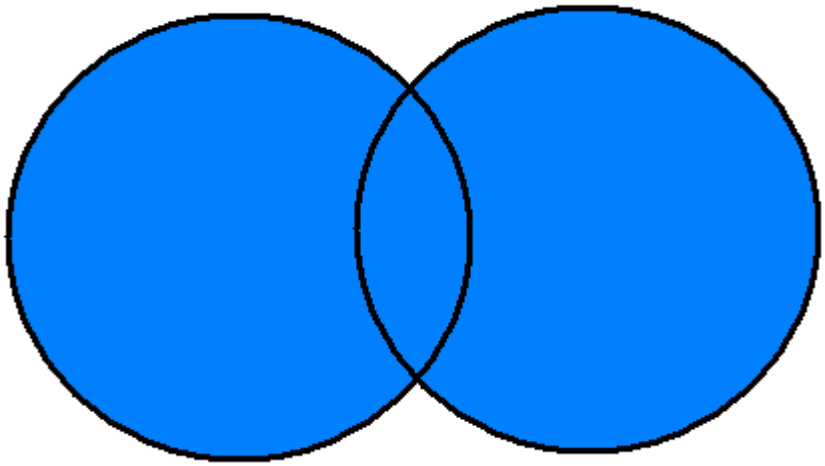
  Outro exemplo utilizando a cláusula FROM:

  SELECT F.*, C.*

  FROM FUNCIONARIO F, CIDADE C;

  Obs.: Uma das operações mais custosas para o SGBD executar.

  Representação Gráfica:



FUNCIONARIO / CIDADE

- ***Left Join:*** retorna todas as tuplas que satisfazem a condição de junção mais as tuplas da tabela a esquerda da junção.

  Exemplo: Retorna todos os funcionários com o respectivo nome da cidade, mesmo que o funcionário não possua cidade.

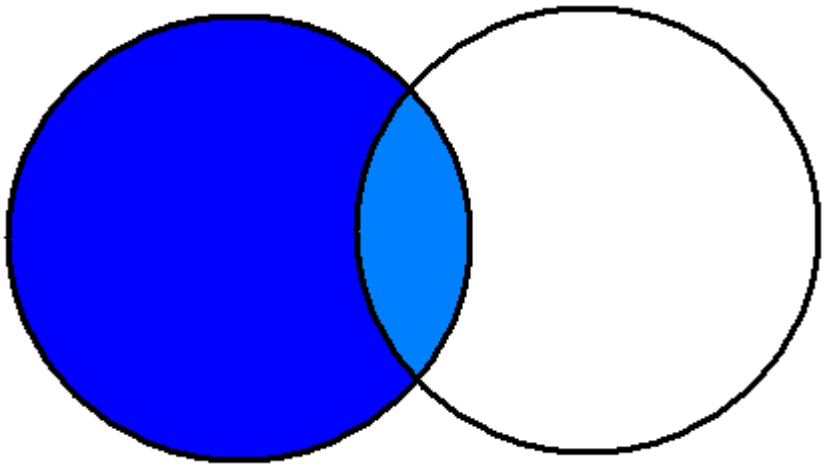
  SELECT F.NUMFUN. F.NOMFUN, C.NOMCID

  FROM FUNCIONARIO F **LEFT JOIN** CIDADE C

  ON F.CODCID = C.CODCID;

  Percebam que a tabela FUNCIONARIO esta a esquerda da junção.

Representação gráfica:



FUNCIONARIO / CIDADE

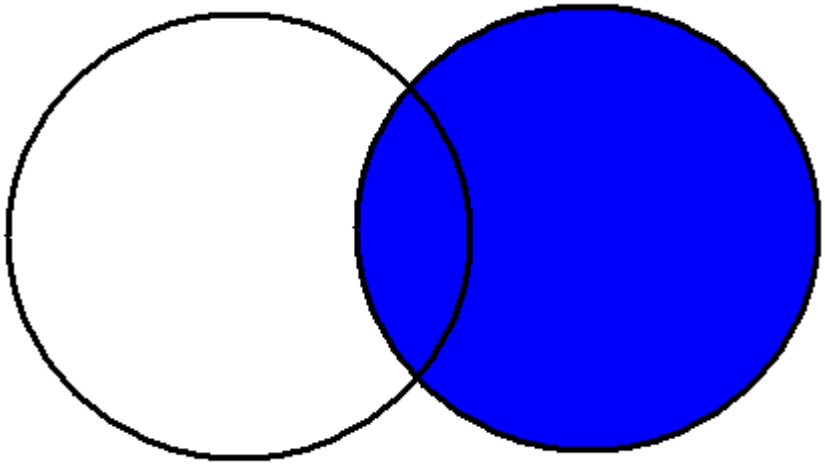
- ***Right Join:*** retorna todas as tuplas que satisfazem a condição de junção mais as tuplas da tabela a direita da junção.

  Exemplo: Retorna todos os funcionários com o respectivo nome da cidade, mais as cidades que não possuem nenhum funcionário residindo nela.

  SELECT F.NUMFUN. F.NOMFUN, C.NOMCID
  FROM FUNCIONARIO F **RIGHT JOIN** CIDADE C
  ON F.CODCID = C.CODCID;

  Percebam que a tabela CIDADE esta a direta da junção.

  Representação gráfica:



FUNCIONARIO / CIDADE

- ***Outer Join:*** Recuperamos todas as tuplas que não satisfazem a condição de junção de uma das tabelas. Este comando tem variações de sintaxe conforme o banco de

dados utilizado.

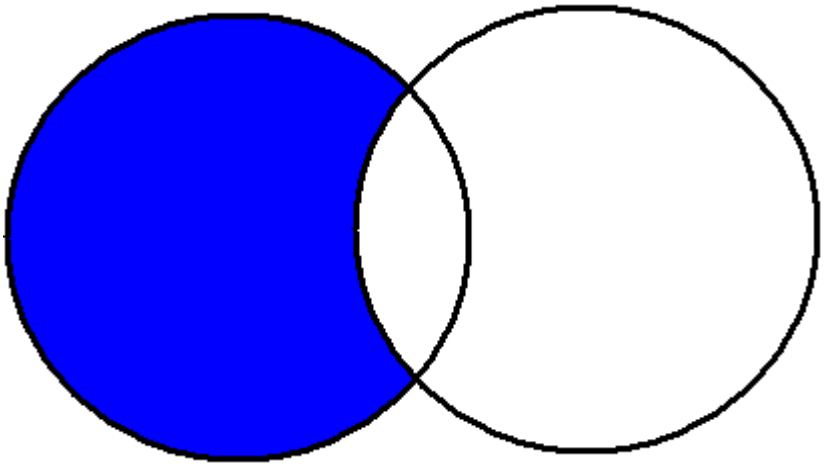
- ***Left Outer Join:*** retorna somente os registros da esquerda que não existem na tabela da direita.

  Exemplo: Selecione todos os funcionários que não possuem cidade.

  SELECT NUMFUN, NOMFUN

  FROM FUNCIONARIO F **LEFT OUTER JOIN** CIDADE C **ON** F.CODCID = C.CODCID;

  Representação gráfica:



FUNCIONARIO / CIDADE

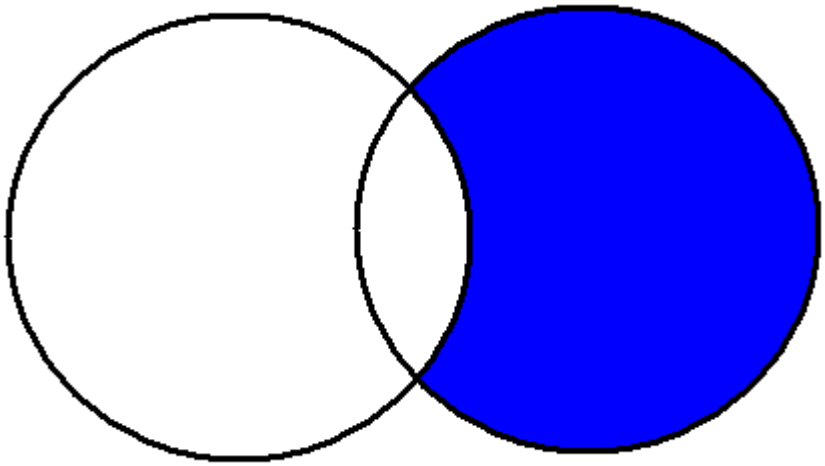
- ***Right Outer Join:*** retorna somente os registros da direita que não existem na tabela da esquerda.

  Exemplo: Selecione todas as cidades sem nenhum funcionário vinculado.

  SELECT C.CODCID, C.NOMCID

  FROM FUNCIONARIO F RIGHT OUTER **JOIN** CIDADE C **ON** F.CODCID = C.CODCID;

  Representação gráfica:



FUNCIONARIO / CIDADE

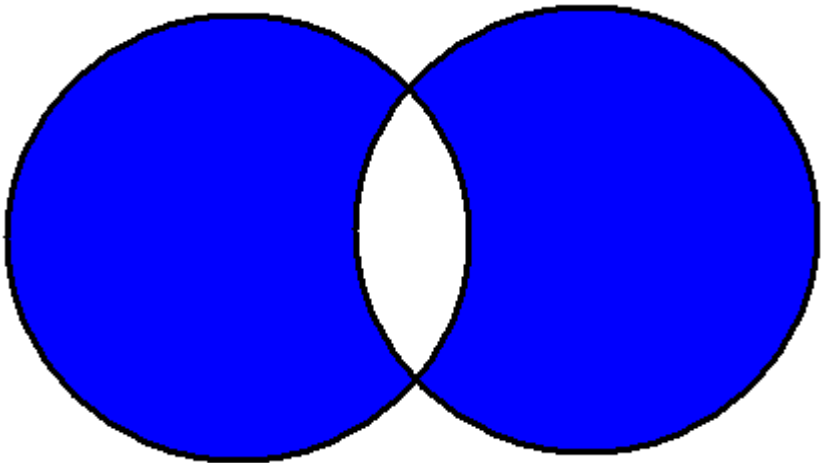
- ***Full Outer Join:*** Combina os resultados da aplicação do LEFT e do RIGHT OUTER JOIN.

  Exemplo: Todos os funcionários e cidades que não estão vinculados, ou seja, funcionários sem cidade e cidades sem funcionários.

  SELECT F.NUMFUN, F.NOMFUN, C.CODCID, C.NOMCID

  FROM FUNCIONARIO F **FULL OUTER JOIN** CIDADE C **ON** F.CODCID = C.CODCID;

  Representação gráfica:



FUNCIONARIO / CIDADE

SUBQUERYS: Uma instrução SELECT trabalha sobre predicados lógicos que precisam ser atendidos para que determinadas tuplas sejam recuperadas. Uma *subquery* nada mais é que uma instrução SELECT embutida em outra instrução SELECT.

Exemplos:

1) Recuperar a todos os funcionários da cidade 10?

SELECT NUMFUN, NOMFUN

FROM FUNCIONARIO

WHERE CODCID = (**SELECT** CODCID **FROM** CIDADE **WHERE** CODCID = 10);

A *subquery* neste caso gera uma lista de somente 1 valor, a cidade de código 10, e isso é usado na condição de busca da cláusula WHERE.

2) Todas cidades que possuam mais de dois funcionários vinculados.

SELECT C.CODCID, C.NOMCID

FROM CIDADE C

WHERE (**SELECT** COUNT(*) **FROM** FUNCIONARIO F **WHERE** F.CODCID = C.CODCID) > 1;

3) Todos funcionários das cidades 'SMOESTE', 'MARAVILHA' E 'PINHALZINHO'.

SELECT NUMFUN, NOMFUN

FROM FUNCIONARIO

WHERE CODCID **IN** (**SELECT** CODCID **FROM** CIDADE **WHERE** NOMCID **IN** ('SMOESTE','MARAVILHA','PINHALZINHO'));

Quadro Comparativo dos Operadores de Álgebra Relacional com os de SQL	
Projeção	SELECT coluna1, coluna2, ..., colunaN
Seleção	SELECT + WHERE
União	UNION
Interseção	INTERSECT
Diferença	EXCEPT | MINUS
Junção	JOIN
Produto Cartesiano	CROSS JOIN | FROM tabela1, tabela2, ..., tabelaN

Comando: INSERT INTO

Função: Inserir(incluir, adicionar) novas tuplas(linhas ou registros) em uma tabela(ou relação).

Sintaxe:

INSERT INTO tabela [**(**coluna { , coluna }* **)**] **VALUES (**valor { , valor }***)**;

TIPOS DE DADOS DAS TABELAS:

Este comando exige que sejam conhecidos os tipos de dados de cada coluna na qual deseja-se inserir um valor. Para conhecer os tipos de dados de uma tabela precisamos utilizar um comando para recuperar sua estrutura ou os metadados[2] armazenados no dicionário do banco de dados em uso. Isto pode variar entre bancos de dados diferentes.

Alguns exemplos para mostrar a estrutura de uma tabela a partir do dicionário do BD:

Oracle:

```
DESC TABELA;
```

Sybase/SQL Server:

```
SP_COLUMNS TABELA;
```

Mysql:

```
DESC TABELA;
```

Firebird/Interbase:

```
SELECT
RDB$RELATION_FIELDS.RDB$FIELD_NAME AS COLUMN_NAME,
RDB$TYPES.RDB$TYPE_NAME AS DATA_TYPE
FROM RDB$RELATIONS
INNER JOIN RDB$RELATION_FIELDS ON
RDB$RELATIONS.RDB$RELATION_NAME =
RDB$RELATION_FIELDS.RDB$RELATION_NAME
LEFT JOIN RDB$FIELDS ON RDB$RELATION_FIELDS.RDB$FIELD_SOURCE =
RDB$FIELDS.RDB$FIELD_NAME
LEFT JOIN RDB$TYPES ON RDB$FIELDS.RDB$FIELD_TYPE =
RDB$TYPES.RDB$TYPE
WHERE RDB$RELATIONS.RDB$RELATION_NAME = ‘TABELA’ AND
RDB$RELATIONS.RDB$SYSTEM_FLAG = 0 AND
RDB$TYPES.RDB$FIELD_NAME='RDB$FIELD_TYPE';
```

Postgresql:

```
SELECT N.NSPNAME AS ESQUEMA, C.RELNAME AS TABELA, A.ATTNAME AS CAMPO,
FORMAT_TYPE(T.OID, NULL) AS TIPO_DE_DADO
FROM PG_NAMESPACE N, PG_CLASS C,
    PG_ATTRIBUTE A, PG_TYPE T
WHERE N.OID = C.RELNAMESPACE
  AND C.RELKIND = 'R'     -- NO INDICES
  AND N.NSPNAME NOT LIKE 'PG\\_%' -- NO CATALOGS
  AND N.NSPNAME != 'INFORMATION_SCHEMA' -- NO INFORMATION_SCHEMA
  AND A.ATTNUM > 0        -- NO SYSTEM ATT'S
  AND NOT A.ATTISDROPPED   -- NO DROPPED COLUMNS
  AND A.ATTRELID = C.OID
  AND A.ATTTYPID = T.OID
  AND C.RELNAME = 'TABELA';
```

2 **Metadado:** Também conhecido como metainformação. Em resumo são dados capazes de descrever outros dados.

Exemplos utilizando o comando INSERT:

E-14) Inserção de um funcionário.

INSERT INTO FUNCIONARIO **VALUES** (1000, 'MARINALDO JOHN COIMBRA', '20/01/2005', 'M', 'C1', 'D2');

E-15) Inserção de um funcionário(indicando as colunas).

INSERT INTO FUNCIONARIO (NUMFUN, NOMFUN, DATADM, SEXFUN, CODCAR, CODDEP) **VALUES** (1000, 'MARINALDO JOHN COIMBRA', '20/01/2005', 'M', 'C1', 'D2');

Existe uma variação do comando INSERT que pode ser utilizada para inserir um número determinado de linhas baseado em um SELECT.

INSERT INTO tabela [**(**coluna { , coluna }* **)**] **SELECT** corpo_select;

E-16) Inserindo os dados da tabela funcionário em uma tabela cópia.

INSERT INTO FUNCIONARIO_COPIA **SELECT** * **FROM** FUNCIONARIO;

É possível inserir mais de uma tupla no mesmo comando insert?
R.: No PostgreSQL é possível utilizando a sintaxe conforme o exemplo: inserindo dois funcionários na tabela FUNCIONARIO.

INSERT INTO FUNCIONARIO (NUMFUN, NOMFUN, DATADM, SEXFUN, CODCAR, CODDEP)
VALUES
(1000, 'MARINALDO JOHN COIMBRA', '20/01/2005', 'M', 'C1', 'D2'),
(1001, 'JOÃO DA SILVA', '20/01/2009', 'M', 'C1', 'D2');

Comando: UPDATE

Função: Atualizar(modificar, alterar) valores de colunas de uma tabela(ou relação).

Sintaxe:

UPDATE tabela **SET** lista de atribuições [**WHERE** condição];
Lista de atribuições: coluna = expressão(separados por vírgula)

E-17) Atualizando o cargo do funcionário 101 para C5.

UPDATE FUNCIONARIO
SET CODCAR = 'C5'
WHERE NUMFUN = 101;

E-18) Atualizando o departamento do funcionário 104 para D5 e o cargo para C1.

UPDATE FUNCIONARIO
SET CODDEP = 'D5', CODCAR = 'C1'

WHERE NUMFUN = 104;

É possível realizar junções de tabelas no comando update?
R.: Isso na verdade depende do SGBD. Normalmente se temos necessidade de realizar um update que dependa da junção de várias tabelas, são necessárias serem utilizadas *subquerys* para resolver. O PostgreSQL entretanto suporta a cláusula FROM no update.
Exemplo: Atualizando o nome de cada funcionário para o formato: nome atual + ' - ' + descrição do seu cargo.
UPDATE FUNCIONARIO **SET** NOMFUN = NOMFUN || ' - ' || CARGO.DESCAR
FROM CARGO
WHERE CARGO.CODCAR = FUNCIONARIO.CODCAR;

Comando: DELETE

Função: Excluir(eliminar, apagar) uma linha(tupla) de uma tabela(relação).

Sintaxe:

DELETE FROM tabela **[WHERE** condição**];**

E-19) Excluindo o funcionário 102.

DELETE FROM FUNCIONARIO
WHERE NUMFUN = 102**;**

E-20) Excluindo todos os funcionários cujo código seja maior que 104 e o sexo seja masculino.

DELETE FROM FUNCIONARIO
WHERE NUMFUN > 104 **AND**
SEXFUN = 'M';

É possível realizar junções de tabelas no comando delete?
R.: Normalmente se temos necessidade de realizar um delete que dependa da junção de várias tabelas, são necessárias serem utilizadas *subquerys* para resolver. O PostgreSQL entretanto suporta a cláusula USING no delete.
Exemplo: Excluindo empréstimos de usuários cujo cep é 890.
DELETE FROM EMPRESTIMO
USING USUARIO
WHERE USUARIO.CODUSU = EMPRESTIMO.CODUSU **AND**
USUARIO.CEPUSU = '890';

7.2.DDL (*Data Definition Language*/Linguagem de Definição de Dados)

Comando: CREATE DATABASE

Função: Criar uma base de dados.

Sintaxe:

```
CREATE DATABASE nome_base [ [WITH] [ OWNER = proprietario]
[TEMPLATE = modelo] [ENCODING = 'codificação'] [TABLESPACE =
espaço de tabela] ]
```

E-21) Criar uma base de dados chamada BDAULA tendo como proprietário o usuário aluno.

CREATE DATABASE BDAULA **WITH OWNER** = aluno **ENCODING** = 'latin1';

Comando: CREATE SCHEMA

Função: Criar um novo esquema. Um esquema é um *namespace* ou uma espécie de diretório onde poderão ser criados os objetos de um banco de dados. Um banco de dados pode possuir vários esquemas.

Sintaxe:

```
CREATE SCHEMA nome_esquema;
```

E-22) Criar um esquema chamado ADM.

CREATE SCHEMA ADM;

Comando: CREATE TABLE

Função: Utilizado para a criação/definição de tabelas(relações).

Sintaxe:

```
CREATE TABLE nome_tabela (
{(nome_coluna tipo_de_dado [DEFAULT valor_padrão] [NOT NULL | NULL]
[PRIMARY KEY] [restrição_coluna]) [restrição_tabela] }
);
restrição_coluna:
CONSTRAINT nome_restrição { DEFAULT valor_padrão | NOT NULL |
```

NULL | UNIQUE | PRIMARY KEY | CHECK (condição) **| REFERENCES** [tabela(nome_coluna)] [**ON DELETE** ação] [**ON UPDATE** ação]}
restrição_tabela:
CONSTRAINT nome_restrição { **NOT NULL | NULL | UNIQUE** (nome_coluna [,...]) **| PRIMARY KEY** (nome_coluna [,...]) **| CHECK** (condição) **| FOREIGN KEY** (nome_coluna [,...]) **REFERENCES** tabela[(nome_coluna [,...])]] [**ON DELETE** ação] [**ON UPDATE** ação] }

Sintaxe alternativa:

CREATE TABLE nome_tabela **AS** comando_select;

QUADRO DE ITENS DA SINTAXE	
Nome	**Descrição**
NOT NULL OU NULL	Indica se a coluna pode ou não receber valores nulos, ou seja, ficar sem valor.
CONSTRAINT	Define uma restrição. Esta restrição pode ser a nível de tabela ou a nível de coluna.
UNIQUE	Define um índice *unique*, ou seja, único sobre a coluna ou colunas desejadas. Significa que nesta coluna valores não poderão ser repetidos. No entanto, esta tipo de índice aceita valores nulos.
PRIMARY KEY	Define a coluna ou colunas chave primária. Uma chave primária só pode ser definida sobre colunas NOT NULL e cujo conjunto de valores não apresentem repetição.
FOREIGN KEY	Define uma coluna ou colunas chave estrangeira. Através desta opção se relacional duas tabelas no modelo relacional. Somente colunas pertencente a chave primária de uma tabela podem se tornar chave estrangeira em outra tabela(Integridade Referencial).
REFERENCES	É parte integrante da definição de uma chave estrangeira. Através desta opção definem-se as colunas e o nome da tabela de origem da chave estrangeira.
DEFAULT	Permite definir um valor padrão para uma determinada coluna. A coluna assumirá este valor quando realizado um comando se inserção de nova linha e a coluna não possuir valor.
CHECK	É um tipo de restrição. Permite definir restrições aos valores informados para determinada coluna. Baseia-se em um teste lógico.
ON UPDATE E ON DELETE	Também pode compor a definição das chave estrangeiras. Através destas opções são definidos os comportamentos assumidos por registros “filhos” em caso de atualização ou exclusão dos registros “pais”. Tipos de Ações: RESTRICT ou NO ACTION: ação padrão. Proíbe a exclusão de registros pais sem antes serem excluídos os registros filhos. CASCADE: Replicação a atualização do registro “pai” para todos os registros “filhos”. Exclui os registros “filhos” quando o registro “pai” for excluído. SET NULL: Colunas de registros filhos são alterados para NULL. SET DEFAULT: Altera o valor *default* caso o mesmo exista.

Obs.: A DEFINIÇÃO DE CONSTRAINTS É OPCIONAL. É RECOMENDADO SUA UTILIZAÇÃO PARA FACILITAR POSSÍVEIS MANUTENÇÕES POSTERIORES.

A seguir são apresentados os tipos de dados básicos dos principais SGBDs livres:

Tipos de Dados do Firebird		
Categoria	**Tipo de Dado/Sintaxe**	**Descrição**
Alfa	CHAR(n)	Tamanho fixo. Completado com espaços em branco.
	VARCHAR(n)	Tamanho variável.
Numérico	INTEGER	Inteiro. Intervalo suportado: -2147483648 até + 2147483647.
	DECIMAL(TD,DD)	Números reais. O número de dígitos total(TD – total digits) indica a quantidade de dígitos do valor como um todo, já o número de dígitos decimais(DD – decimal digits) indica o número de casas decimais após o separador decimal. Um inteiro é um valor decimal com DD = 0. Ex.: 23,4567 = Decimal(6,4).
	NUMERIC(TD,DD)	Números reais. O número de dígitos total(TD – total digits) indica a quantidade de dígitos do valor como um todo, já o número de dígitos decimais(DD – decimal digits) indica o número de casas decimais após o separador decimal. Um inteiro é um valor decimal com DD = 0. Ex.: 23,4567 = Numeric(6,4).
Data/Hora	TIMESTAMP	Data e hora.
	DATE	Somente data.
	TIME	Somente hora.
Binário	BLOB	Armazena valores binários como: imagens, som, texto, etc.

Tipos de Dados do MySQL		
Categoria	**Tipo de Dado/Sintaxe**	**Descrição**
Alfa	CHAR(n)	Tamanho fixo. Completado com espaços em branco.
	VARCHAR(n)	Tamanho variável.
	TEXT	Tamanho variável. Recomendado para textos longos.
Numérico	INTEGER	Inteiro. Intervalo suportado: -2147483648 até + 2147483647.
	DECIMAL(TD,DD)	Números reais. O número de dígitos total(TD – *total digits*) indica a quantidade de dígitos do valor como um todo, já o número de dígitos decimais(DD – *decimal digits*) indica o número de casas decimais após o separador decimal. Um inteiro é um valor decimal com DD = 0. Ex.: 23,4567 = Decimal(6,4).
	NUMERIC(TD,DD)	Números reais. O número de dígitos total(TD – *total digits*) indica a quantidade de dígitos do valor como um todo, já o número de dígitos decimais(DD – *decimal digits*) indica o número de casas decimais após o separador decimal. Um inteiro é um valor decimal com DD = 0. Ex.: 23,4567 = Numeric(6,4).
	AUTO_INCREMENT	Inteiro auto-incremento. Intervalo suportado: 1 até 2147483647.
Lógicos	BOOLEAN	Valores lógicos: 1 – true /0 – false .
Data/Hora	TIMESTAMP	Data e hora.
	DATE	Somente data.
	TIME	Somente hora.

Binário	BLOB	Armazena valores binários como: imagens, som, texto, etc.

Tipos de Dados do PostgreSQL		
Categoria	**Tipo de Dado/Sintaxe**	**Descrição**
Alfa	CHAR(n)	Tamanho fixo. Completado com espaços em branco.
	VARCHAR(n)	Tamanho variável.
	TEXT	Tamanho variável. Recomendado para textos longos.
Numérico	INTEGER	Inteiro. Intervalo suportado: -2147483648 até + 2147483647.
	DECIMAL(TD,DD)	Números reais. O número de dígitos total(TD – *total digits*) indica a quantidade de dígitos do valor como um todo, já o número de dígitos decimais(DD – *decimal digits*) indica o número de casas decimais após o separador decimal. Um inteiro é um valor decimal com DD = 0. Ex.: 23,4567 = Decimal(6,4).
	NUMERIC(TD,DD)	Números reais. O número de dígitos total(TD – *total digits*) indica a quantidade de dígitos do valor como um todo, já o número de dígitos decimais(DD – *decimal digits*) indica o número de casas decimais após o separador decimal. Um inteiro é um valor decimal com DD = 0. Ex.: 23,4567 = Decimal(6,4).
	SERIAL	Inteiro auto-incremento. Intervalo suportado: 1 até 2147483647.
Lógicos	BOOLEAN	Valores lógicos: true/false.
Data/Hora	TIMESTAMP	Data e hora.
	DATE	Somente data.
	TIME	Somente hora.
Binário	BYTEA	Armazena valores binários como: imagens, som, texto, etc.

Seguem vários exemplos de criação de tabelas:

E-23) Criação da tabela DEPARTAMENTO, com campos NOT NULL e a CHAVE PRIMÁRIA.

```
CREATE TABLE DEPARTAMENTO (
   CODDEP INTEGER NOT NULL PRIMARY KEY,
   NOMDEP VARCHAR(50) NOT NULL,
   RAMTEL VARCHAR(15)
);
```

E-24) Criação da tabela DEPARTAMENTO, com campos NOT NULL e utilizando restrição de coluna.

```
CREATE TABLE DEPARTAMENTO (
   CODDEP INTEGER NOT NULL CONSTRAINT DEP_PK PRIMARY KEY,
   NOMDEP VARCHAR(50) NOT NULL,
   RAMTEL VARCHAR(15)
);
```

E-25) Criação da tabela DEPARTAMENTO, com campos NOT NULL e utilizando restrição de tabela.

```
CREATE TABLE DEPARTAMENTO (
```

```
    CODDEP INTEGER NOT NULL,
    NOMDEP VARCHAR(50) NOT NULL,
    RAMTEL VARCHAR(15),
    CONSTRAINT DEP_PK PRIMARY KEY(CODDEP)
);
```

E-26) Criação da tabela FUNCIONARIO, com campos NOT NULL, uma restrição CHECK e utilizando restrição de coluna para chave primária e restrição de coluna para chave estrangeira.

```
CREATE TABLE FUNCIONARIO(
    NUMFUN INTEGER CONSTRAINT FUN_PK PRIMARY KEY,
    NOMFUN VARCHAR(80) NOT NULL,
    DATADM TIMESTAMP NOT NULL,
    SEXFUN NOT NULL CONSTRAINT FUN_SEXFUN_CK CHECK(SEXFUN IN ('F', 'M')),
    CODCAR INTEGER CONSTRAINT FUN_CODCAR_FK REFERENCES CARGO(CODCAR),
    CODDEP INTEGER CONSTRAINT FUN_CODDEP_FK REFERENCES DEPARTAMENTO(CODDEP)
);
```

E-27) Criação da tabela FUNCIONARIO, com campos NOT NULL, uma restrição CHECK e utilizando restrição de tabela para chave primária e restrição de tabela para chave estrangeira.

```
CREATE TABLE FUNCIONARIO(
    NUMFUN INTEGER NOT NULL,
    NOMFUN VARCHAR(80) NOT NULL,
    DATADM TIMESTAMP NOT NULL,
    SEXFUN NOT NULL,
    CODCAR INTEGER,
    CODDEP INTEGER,
    CONSTRAINT FUN_PK PRIMARY KEY(NUMFUN),
    CONSTRAINT FUN_SEXFUN_CK CHECK(SEXFUN IN ('F', 'M')),
    CONSTRAINT FUN_CODCAR_FK FOREIGN KEY(CODCAR) REFERENCES CARGO(CODCAR),
    CONSTRAINT FUN_CODDEP_FK FOREIGN KEY(CODDEP) REFERENCES DEPARTAMENTO(CODDEP)
);
```

E-28) Criação da tabela FUNCIONARIO utilizando as opções ON DELETE e ON UPDATE.

```
CREATE TABLE FUNCIONARIO(
    NUMFUN INTEGER NOT NULL,
    NOMFUN VARCHAR(80) NOT NULL,
    DATADM TIMESTAMP NOT NULL,
    SEXFUN NOT NULL,
    CODCAR INTEGER,
    CODDEP INTEGER,
    CONSTRAINT FUN_PK PRIMARY KEY(NUMFUN),
    CONSTRAINT FUN_SEXFUN_CK CHECK(SEXFUN IN ('F', 'M')),
```

CONSTRAINT FUN_CODCAR_FK **FOREIGN KEY**(CODCAR) **REFERENCES** CARGO(CODCAR) **ON DELETE** RESTRICT **ON UPDATE** CASCADE**,**
CONSTRAINT FUN_CODDEP_FK **FOREIGN KEY**(CODDEP) **REFERENCES** DEPARTAMENTO(CODDEP) **ON DELETE** RESTRICT **ON UPDATE** CASCADE
);

E-29) Criação da tabela FUNCIONARIO com uma coluna com valor DEFAULT.

CREATE TABLE FUNCIONARIO(
NUMFUN **INTEGER NOT NULL**,
NOMFUN **VARCHAR(80) NOT NULL**,
DATADM **TIMESTAMP DEFAULT** CURRENT_TIMESTAMP **NOT NULL**,
SEXFUN **NOT NULL,**
CODCAR **INTEGER,**
CODDEP **INTEGER,**
CONSTRAINT FUN_PK **PRIMARY KEY**(NUMFUN),
CONSTRAINT FUN_SEXFUN_CK **CHECK**(SEXFUN **IN** ('F', 'M')),
CONSTRAINT FUN_CODCAR_FK **FOREIGN KEY**(CODCAR) **REFERENCES** CARGO(CODCAR)**,**
CONSTRAINT FUN_CODDEP_FK **FOREIGN KEY**(CODDEP) **REFERENCES** DEPARTAMENTO(CODDEP)
);

E-30) Criação da tabela FUNCIONARIO_COPIA somente com os dados das funcionários do sexo feminino.

CREATE TABLE FUNCIONARIO_COPIA **AS SELECT** * **FROM** FUNCIONARIO **WHERE** SEXFUN = 'F';

Comando: ALTER TABLE

Função: Utilizado para alterar a estrutura de uma tabela.

Sintaxe:

```
ALTER TABLE nome_tabela ADD nome_coluna tipo_de_dado;
ALTER TABLE nome_tabela ADD definição_restrição;
ALTER TABLE nome_tabela DROP nome_coluna;
ALTER TABLE nome_tabela DROP CONSTRAINT nome_restrição;
ALTER TABLE nome_tabela RENAME [COLUMN] nome_coluna TO [COLUMN]  novo_nome_coluna;
ALTER TABLE nome_tabela ALTER COLUMN nome_coluna TYPE novo_tipo_dado;
```

E-31) Adicionar uma coluna de salário na tabela FUNCIONARIO.

ALTER TABLE FUNCIONARIO **ADD** SALFUN **DECIMAL(12,2) NOT NULL**;

E-32) Adicionar uma restrição CHECK a coluna SALFUN.

ALTER TABLE FUNCIONARIO **ADD CONSTRAINT** FUN_SALFUN_CK **CHECK**(SALFUN > 0);

E-33) Excluir uma restrição da tabela FUNCIONARIO.

ALTER TABLE FUNCIONARIO **DROP CONSTRAINT** FUN_SALFUN_CK;

E-34) Excluir a coluna SALFUN da tabela EMPREGADO.

ALTER TABLE FUNCIONARIO **DROP** SALFUN;

Comando: DROP TABLE

Função: Utilizado para excluir tabelas.

Sintaxe:

```
DROP TABLE nome_tabela;
```

E-35) Excluindo a tabela FUNCIONARIO.

DROP TABLE FUNCIONARIO;

Comando: CREATE INDEX

Função: Utilizado para a criação/definição de índices.

Sintaxe:

```
CREATE [UNIQUE] INDEX nome_indice ON nome_tabela(nome_coluna[,...]) [ASC | DESC];
```

O que é um índice e por que criar?

Os índices são estruturas físicas de banco de dados criadas para otimizar a performance no acesso aos dados de uma tabela. Os comandos **SELECT** que envolvem a cláusula **ORDER BY**, ficam mais rápidos após a criação de índices sobre os campos que fazem parte ordenação. Na hora de criar um índice deve-se analisar quais campos da tabela participam das cláusulas **WHERE** e **ORDER BY** em comandos como **SELECT**, **UPDATE** e **DELETE**. No entanto, o uso excessivo de índices pode ser prejudicial à performance, pois todo comando que atualiza a tabela origem pode gerar uma alteração no índice.

Os índices geralmente são definidos em conjunto, pelo DBA e pelo DA ou analista de sistemas, que são os indivíduos que conhecem quais são as consultas mais críticas e que envolvem o tratamento de um maior volume de dados. Os índices podem ou não ser únicos (**UNIQUE**). O índice é único quando não se permite repetições de valores nos registros que compõem o índice. Assim sendo, uma chave primária é um índice **UNIQUE**.

Uma forma de avaliar se é uma boa ideia criar um índice sobre determinadas colunas de uma tabela, é utilizar a seguinte fórmula: número de registros distintos/número de registros

total; esta fórmula vai gerar a seletividade do índice, quanto mais próximo de 1 mais indicada é a criação do índice.

E-36) Criando um índice sobre a coluna nome do funcionário.

CREATE INDEX FUN_NOMFUN_SK **ON** FUNCIONARIO(NOMFUN);

E-37) Criando um índice único sobre a coluna nome do departamento.

CREATE UNIQUE INDEX DEP_NOMDEP_UK **ON** DEPARTAMENTO(NOMDEP);

E-38) Criando um índice sobre as colunas nome e cargo do empregado.

CREATE INDEX FUN_NOMFUN_CODCAR_SK **ON** FUNCIONARIO(NOMFUN, CODCAR);

Comando: DROP INDEX

Função: Utilizado para excluir índices.

Sintaxe:

```
DROP INDEX nome_indice;
```

E-39) Apagando um índice sobre a coluna nome do funcionário.

DROP INDEX FUN_NOMFUN_SK**;**

E-40) Apagando um índice único sobre a coluna nome do departamento.

DROP INDEX DEP_NOMDEP_UK;

Comando: CREATE VIEW

Função: Utilizado para a criação/definição de visões.

Sintaxe:

```
CREATE VIEW nome_visao[(nome_coluna[,...])] AS comando_select;
```

O que é uma visão e por que utilizar?

As visões são estruturas físicas de banco de dados úteis para filtrar e formatar a apresentação de determinados dados. Uma visão também é conhecida como uma tabela virtual, sua estrutura é gerada com base em uma instrução **SELECT**.

Os dados de uma visão são atualizados dinamicamente, cada acesso a mesma faz com que os dados exibidos sejam re-consultados.

E-41) Uma visão sobre a tabela FUNCIONARIO utilizando somente algumas colunas.

```
CREATE VIEW FUN_VW AS
SELECT NUMFUN AS CODIGO, NOMFUN AS NOME
FROM FUNCIONARIO
```

ORDER BY NOMFUN;

E-42) Uma visão sobre a tabela DEPARTAMENTO pegando somente departamentos cujo número do ramal é maior que 2000.

```
CREATE VIEW DEP_VW AS
SELECT CODDEP AS CODIGO, NOMDEP AS NOME
FROM DEPARTAMENTO
WHERE RAMTEL > 2000
ORDER BY NOMDEP;
```

E-43) Uma visão sobre a tabela DEPARTAMENTO pegando somente departamentos cujo número do ramal é maior que 2000. Com opção de definir as colunas.

```
CREATE VIEW DEP_VW(CODIGO, NOME)  AS
SELECT CODDEP, NOMDEP
FROM DEPARTAMENTO
WHERE RAMTEL > 2000
ORDER BY NOMDEP;
```

Comando: DROP VIEW

Função: Utilizado para excluir visões.

Sintaxe:

```
DROP VIEW nome_visao;
```

E-44) Apagar a visão criada sobre a tabela FUNCIONARIO.

DROP VIEW FUN_VW;

E-45) Apagar a visão criada sobre a tabela DEPARTAMENTO.

DROP VIEW DEP_VW;

7.3.DCL *(Data Control Language/*Linguagem de Controle de Dados)

CONTROLE DE USUÁRIOS E PERMISSÕES DE ACESSO

CRIAÇÃO DE USUÁRIOS/PAPÉIS

Para acessar uma base de dados criada, é necessário também informar um usuário e uma senha. Este usuário deverá ter permissão de acesso ao servidor e a base de dados desejada. Dois aspectos precisam ser observados: *Autenticar o usuário* e *Autorizar o Usuário*.

Autenticar o usuário: Significa que é necessário criar um usuário e definir uma senha para o mesmo. Uma vez o usuário criado ele poderá fazer seu "*login*" no servidor de banco de

dados.

Como criar um usuário?

Para a criação de um usuário o processo é relativamente simples. Geralmente é feito através do comando CREATE USER.

Comando: CREATE USER

Função: Criar um usuário.

Sintaxe:

CREATE USER nome_usuario **[[WITH] [SUPERUSER | NOSUPERUSER | LOGIN | NOLOGIN | CREATEDB | NOCREATEDB | CREATEROLE | NOCREATEROLE | CREATEUSER | NOCREATEUSER** | **PASSWORD** 'password' **]]**;

QUADRO DE OPÇÕES	
Opção	**Descrição**
WITH	Permite criar o usuário com as opções apresentadas na sequência.
SUPERUSER / NOSUPERUSER	Indica se o usuário será ou não superusuário.
LOGIN / NOLOGIN	Indica se o usuário pode ou não realizar o login no servidor.
CREATEDB / NOCREATEDB	Indica se o usuário poderá ou não criar uma base de dados.
CREATEROLE / NOCREATEROLE	Indica se o usuário poderá ou não criar papéis(ou grupos).
CREATEUSER / NOCREATEUSER	Indica se o usuário poderá ou não criar outros usuários.
PASSWORD	Indica a senha do usuário.
Obs.: É importante ressaltar que as opções do comando CREATE USER podem variar de um SGBD para outro, é sempre importante observar a documentação do SGBD para verificar qual a melhor forma de utilizar o comando.	

E-46) Criar um usuário chamado ALUNO com a senha '123456'.

CREATE USER ALUNO **WITH PASSWORD** '123456';

Em caso de necessidade de se alterar informações em relação ao usuário criado pode ser utilizada a instrução a seguir.

Comando: ALTER USER

Função: Alterar informações de um usuário criado.

Sintaxe:

ALTER USER nome_usuario **[[WITH] [SUPERUSER | NOSUPERUSER | LOGIN | NOLOGIN | CREATEDB | NOCREATEDB | CREATEROLE |**

NOCREATEROLE | CREATEUSER | NOCREATEUSER | PASSWORD 'password'**]]**;

E-47) Alterar a senha do usuário ALUNO para '654321'.

ALTER USER ALUNO **WITH PASSWORD '**654321';

Como excluir um usuário?

Apesar de existirem opções para realizar o bloqueio dos usuários no SGBD, existe a opção de excluir um usuário.

Comando: DROP USER

Função: Exclui um usuário.

Sintaxe:

DROP USER nome_usuario;

E-48) Excluir o usuário ALUNO.

DROP USER ALUNO;

Obs.: Se o usuário possuir objetos dependentes, como: bases de dados, tabelas, etc; será necessário excluir as mesmas antes de excluir o usuário.

Autorizar o usuário: Significa que é necessário definir o que o usuário pode ou não fazer quando estiver conectado na base de dados, ou seja, definir as permissões do usuário.

As permissões determinarão as ações que um usuário pode executar sobre os objetos de uma base de dados. Estas permissões abrangem ações como SELECT(leitura) até ações como INSERT(escrita) na base de dados, entre outras.

Como definir permissões para um usuário?

Uma vez criado o usuário, torna-se necessário conceder as permissões ao mesmo, ou seja, dizer o que ele pode fazer ou acessar quando estiver conectado a base de dados.

Comando: GRANT

Função: Concede permissões para um usuário.

Sintaxe:

GRANT { select | insert | update | delete | references | trigger | all [privileges] **} ON** { nome_tabela [, ...] } **TO {**nome_usuario | public | group} **[WITH GRANT OPTION]**;

Variação da sintaxe:

GRANT group [, ...] **TO** usuario [, ...] **[WITH ADMIN OPTION]**;

QUADRO DE OPÇÕES	
Opção	**Descrição**
SELECT | INSERT | DELETE | UPDATE	Concede as permissões de ler, inserir, excluir ou atualizar informações em determinada tabela da base de dados.
REFERENCES	Concede permissão para que sejam criadas chaves estrangeiras sobre determinada tabela da base de dados.
TRIGGER	Concede permissão de criação de gatilhos(*triggers*), procedimento e funções dentro da base de dados.
ALL PRIVILEGES	Concede todas as permissões.
PUBLIC	Grupo público, uma vez concedida uma permissão para este grupo, todos(usuários e grupos) terão acesso a mesma.
GROUP	Um grupo qualquer criado na base de dados.
WITH GRANT OPTION	Além de conceder a permissão permite que o usuário ou grupo passe a mesma adiante.

E-49) Conceder permissão de leitura e inserção na tabela PESSOA para o usuário ALUNO.

GRANT SELECT, INSERT **ON** PESSOA **TO** ALUNO;

Como remover permissões de um usuário?

Se necessário podemos também revogar as permissões de um usuário, tudo isso varia de acordo com a política de acesso ao servidor de banco de dados que foi definida.

Comando: REVOKE

Função: Revoga permissões de um usuário.

Sintaxe:

REVOKE { select | insert | update | delete | references | trigger | all [privileges] **}** **ON** { nome_tabela [, ...] } **FROM {** nome_usuário | public | group };

E-50) Revogar as permissões anteriormente concedidas ao usuário ALUNO.

REVOKE SELECT, INSERT **ON** PESSOA **FROM** ALUNO;

CRIAÇÃO DE GRUPOS

Para facilitar a concessão e organização das permissões em uma base de dados

existe ainda a possibilidade de serem definidos grupos, chamados de “GROUPs”. Nestes grupos poderão ser definidas as regras gerais de autorização. Por fim basta conceder um grupo para o usuário e este poderá herdar todas as permissões pertencentes ao grupo.

Como criar um grupo?

Comando: CREATE GROUP

Função: Criar um grupo.

Sintaxe:

```
CREATE GROUP nome_grupo;
```

Como apagar um grupo?

Comando: DROP GROUP

Função: Apagar um grupo.

Sintaxe:

```
DROP GROUP nome_grupo;
```

E-51) Criar dois grupos, um chamado CONSULTA e outro chamado OPERADOR, conceder as permissões necessárias.

CREATE GROUP CONSULTA;
CREATE GROUP OPERADOR;
GRANT SELECT **ON** EMPREGADO **TO** CONSULTA;
GRANT SELECT, INSERT, DELETE, UPDATE **ON** EMPREGADO **TO** OPERADOR;

Adicionando o usuário ALUNO ao grupo CONSULTA:
GRANT CONSULTA **TO** ALUNO;

Adicionando o usuário GERENTE ao grupo OPERADOR:
GRANT OPERADOR **TO** GERENTE;

7.4.TML *(Transaction Manipulation Language/*Linguagem de Manipulação de Transações)

O controle de transações é atividade fundamental para garantir a consistência dos

dados bem como sua possível recuperação após uma falha na sua manipulação. A maior parte dos bancos de dados oferece duas formas de iniciar uma transação: *implícita* – iniciada quando uma operação DDL ou DML é executada na base de dados; *explícita* – quando uma transação é explicitamente iniciada pelo usuário que está interagindo com a base de dados.

É importante verificar qual modelo de controle de concorrência é utilizado pelo banco de dados. Um modelo que alguns bancos de dados utilizam é o MVCC (*Multiversion Concurrency Control*).

Comando: BEGIN | START

Função: Iniciar explicitamente um processo de transação.

Sintaxe:

```
[BEGIN WORK | TRANSACTION] | [START TRANSACTION]
```

Comando: COMMIT

Função: Finalizar uma transação confirmando as alterações feitas em uma base de dados.

Sintaxe:

```
COMMIT [WORK] | [TRANSACTION];
```

Comando: ROLLBACK

Função: Finalizar uma transação desfazendo as alterações efetuadas em uma base de dados.

Sintaxe:

```
ROLLBACK [WORK] | [TRANSACTION];
```

Níveis de Isolamento de uma Transação SQL

Nível Isolamento	Dirty Read	Nonrepeatable Read	Phantom read
Read uncommited	Possível	Possível	Possível
Read commited	Impossível	Possível	Possível
Repeatable read	Impossível	Impossível	Possível
Serializable	Impossível	Impossível	Impossível

Fenômenos indesejáveis relacionados a transação

Dirty read: Uma transação lê dados “não-comitados” de uma transação concorrente.

Nonrepeatable read: Uma transação re-lê dados já lidos e detecta que os dados sofreram alteração.

Phantom read: Uma transação re-executa uma *query* retornando um conjunto de registros que satisfazem uma determinada condição e detecta que o conjunto de registros

retornados foi alterado por outra transação recentemente “comitada”.

8.BACKUP E RESTORE

Uma prática fundamental no uso de banco de dados é a estratégia adotada para efetuar *backup (*cópia de segurança) das informações.

A informação armazenada no banco de dados tem um valor altíssimo que só pode ser dimensionado quando da perda da mesma.

Falhas de hardware ou software, quedas de energia e até mesmo usuários maliciosos podem causar a perda destas informações.

É recomendada a adoção de uma boa estratégia de *backup* das informações armazenadas em um banco de dados. Este *backup* pode ser feito de várias formas e abordagens, o importante é montar uma estratégia compatível com o banco de dados em uso e que forneça a condição, em caso de perda de alguma informação, de uma rápida recuperação da mesma.

É preciso ressaltar também que uma estratégia de *backup* não consiste apenas em copiar as informações, porque caso as mesmas tenham sido copiadas e não possam ser recuperadas de nada vale o *backup*.

Quase todo banco de dados implementa pelo menos três tipos de *backup*, que podem ser classificados como: *backup* lógico(cópia dos dados), *backup* físico(cópia física dos arquivos de dados) e logs de alterações(também chamado de modo de arquivamento - PITR).

Tipos de Backup:

Backup Lógico(Logical Backup): O backup lógico consiste em uma cópia das informações(dados) de um banco de dados. Alguns bancos de dados armazenam os dados de um backup lógico em arquivos texto e outros em arquivos binários.

A disposição dos dados no arquivo também depende do banco de dados, o mais comum é os dados serem gerados em formato de *script* incluindo os comandos DML e DDL necessários para retorná-los a uma base de dados.

Backup Físico(File System Level Backup): O *backup* físico consiste em uma cópia física dos arquivos que compõe a base de dados do SGBD. Em alguns casos além da cópia física dos arquivos de dados faz-se necessária a cópia de arquivos de configuração do SGBD para que os arquivos de dados possam novamente ser montados na mesma ou em outra base de dados.

Logs de Alterações(Continuous Archiving(Archive Mode) – Point-in-Time Recovery(PITR)): Este é um tipo de *backup* que na verdade se comporta como um *log*. Todas as alterações em uma base de dados são armazenadas em um *log*, o tempo de vida deste *log* é determinado pelo DBA(*DataBase Administrator*). Quando uma falha ocorre no banco de dados o DBA pode recuperar as informações de um determinado momento a partir destes logs. O

diferencial deste backup é que o mesmo permite a recuperação de informações perdidas em faixas menores de tempo, normalmente não permitidas pelas outras técnicas de backup.

O modo do backup em um servidor ainda pode ser realizado em dois modos:

a) Modo On-line(Hot Backup): Neste modo o backup pode ser realizado com o servidor de banco de dados ativo. Obs.: Este tipo de backup pode deixar lentas demais atividades realizadas no servidor de banco de dados.

b) Modo Off-line(Cold Backup): Neste modo de backup o servidor de banco de dados deve ser parado ou deverão ser evitadas conexões ao servidor para que o backup possa ser realizado.

Backup no PostgreSQL

Como qualquer outro banco de dados o PostgreSQL também permite e necessita que seus dados sejam “backupeados” regularmente a fim de evitar perda de informações.

O PostgreSQL suporta fundamentalmente três tipos de *backup* diferentes:

- SQL dump: permite a criação de um arquivo texto com os comandos SQL necessários para restaurar uma base dados. Para isso é utilizado um aplicativo cliente chamado PG_DUMP(http://www.postgresql.org/docs/8.4/interactive/app-pgdump.html). Este aplicativo é executa em modo caractere. Este tipo de *backup* pode ser considerado um *backup* lógico.

  Algumas das sintaxes possíveis para o aplicativo PG_DUMP:

  - pg_dump nome_base_de_dados > arquivo_saida(backup sendo realizado localmente no servidor);
  - pg_dump -h servidor base_de_dados > arquivo_saida(backup sendo realizado de um servidor remoto).

  Existe ainda um aplicativo cliente chamado PG_DUMPALL que pode ser utilizado para realizar um backup completo de todas as bases de dados residentes no servidor, sem a necessidade de informar base a base.

  Para restaurar dados de um backup realizados pelo PG_DUMP ou PG_DUMPALL pode ser utilizar o aplicativo cliente PSQL(http://www.postgresql.org/docs/8.4/interactive/app-psql.html) ou PG_RESTORE(http://www.postgresql.org/docs/8.4/interactive/app-pgrestore.html).

  Exemplo:

  - psql nome_base_de_dados < arquivo_entrada;
  - pg_restore -d nome_base_de_dados arquivo_entrada;

- *File system level backup*: Neste caso estamos realizando um backup físico. Em resumo o que fazemos neste tipo de backup é fazer uma cópia de toda o diretório que

contém os dados do PostgreSQL(normalmente chamada de data, residindo dentro da pasta de instalação do SGBD). Devem ser copiados também os arquivos de configuração.

- *Continuous archiving*: Nesta modalidade de backup o PostgreSQL permite que sejam registrados logs de todas as alterações realizadas no banco de dados. Caso ocorram problemas durante alguma operação nos dados através deste tipo de backup podemos recuperar as informações até determinado ponto no tempo, retornando a base de dados a um estado consistente.

9.ESTUDOS DE CASO PARA MODELAGEM

1. **CADASTRO NACIONAL DE VEÍCULOS:** Você apresentará um modelo de dados para o cadastro nacional de veículos. Sabe-se que:
   - O veículo possui sempre uma placa única em todo o país;
   - O veículo possui sempre um responsável legal por ele. É necessário manter o histórico desta responsabilidade (propriedade);
   - O veículo pertence sempre a uma categoria;
   - O veículo é sempre de uma marca e de um modelo e possui ano de fabricação.

2. **BIBLIOTECA:** A Srta. Doralice Sampaio é a coordenadora de uma Biblioteca de uma universidade brasileira. Ultimamente esta universidade vem recebendo muitos microcomputadores para uso de seus alunos e, como há tantos disponíveis, o diretor da escola acenou com a possibilidade de instalar alguns deles na biblioteca. Com estes microcomputadores instalados na biblioteca percebeu-se que os processos realizados atualmente na biblioteca poderiam ser automatizados através de um sistema de informação. A biblioteca possui uma funcionária para comprar livros das editoras e livrarias. Quando o livro é recebido pela biblioteca ele é tombado, classificado por assunto e catalogado, sendo em seguida liberado para empréstimos ou consulta. O acervo fica aberto para que os leitores possam ir até as estantes escolher o livro desejado. Ao sair, uma atendente anota o empréstimo. Os livros emprestados podem ser renovados uma quantidade ilimitada de vezes, a menos que alguns livros sejam muito consultados ou então sofram alguma reserva. Livros usados como texto básico em cursos durante o período letivo, não podem ser emprestados, devendo ser consultados nas dependências da biblioteca. Os leitores devem se cadastrar como usuários da biblioteca, e só podem ser alunos, professores e funcionários. Os leitores têm prazo fixo de tempo para empréstimo. Existe a possibilidade de visitantes utilizarem a biblioteca, mas os mesmos também deverão ser cadastrados, receberão uma carteirinha de uso temporário e não poderão realizar empréstimos de livros. O leitor pode se desligar espontaneamente da biblioteca, cancelando sua inscrição ou pode ser desligado a revelia, depois de um período sem utilizar a biblioteca. Isto ocorre com frequência com os alunos que se formam ou mudam de cidade. Para leitores não familiarizados com os termos técnicos usados no ambiente de uma biblioteca, esclarecemos que o termo “tombar” significa registrar o livro no acervo na biblioteca,

atribuindo-lhe um número sequencial e único, e o termo “classificar” significa atribuir um número de código para o livro, geralmente por tipo de assunto. Este código será usado para localizar os livros nas estantes.

3. **CONTROLE DE ESTOQUE:** Uma empresa do comércio varejista, deseja fazer o controle de estoque de seu estabelecimento. Para facilitar a administração do seu estoque, a organização criou uma estrutura de almoxarifados, onde um produto pode ser estocado em vários almoxarifados e um almoxarifado pode conter vários produtos. A reposição de estoque acontece quando os produtos adquiridos de um fornecedor chegam com sua respectiva nota fiscal de compra. Já a baixa do estoque se dá quando ocorre a emissão de uma nota fiscal de venda para um determinado cliente. Além disso, deseja-se classificar os produtos em linhas a serem determinadas pelo usuário de acordo com a sua necessidade. Elabore um DER que contemple os dados necessários a este controle. Represente as chaves primárias e estrangeiras, os principais atributos, as integridades referenciais e a obrigatoriedade/opcionalidade dos relacionamentos.

4. **POSTOS DE COMBUSTÍVEL:** A rede de postos de combustível “BOA ESTRADA” contratou a sua companhia para fazer um sistema que ajude a controlar o estoque de combustíveis do posto. Os combustíveis podem ser álcool, diesel, gasolina, aditivados ou não. Cada combustível tem um estoque mínimo aceitável. Os combustíveis são guardados em tanques. Cada tanque é destinado para um tipo específico de combustível, mas pode existir mais de um tanque para o mesmo tipo de combustível. Cada tanque suporta um volume específico de combustível. . As bombas podem ter um ou mais bicos e cada bico pode estar ligado a um tanque. Cada bico da bomba tem um contador que determina a quantidade de litros vendida desde sua instalação. Ao fechar o posto, o dono deve inserir no sistema a contagem marcada em cada bico bomba (que indicará o consumo diário da bomba). Caso algum combustível fique abaixo do estoque mínimo, deve ser gerado um pedido de compra para a empresa distribuidora. Quando o caminhão de distribuição chega, o dono deve colocar no sistema os dados referentes a quantos litros foram colocados em cada tanque (e data). Quando o dono solicitar, o sistema deve gerar uma declaração de venda de combustível, listando todas as vendas desde a última declaração. Esta declaração é padronizada e inclui uma série de impostos com alíquotas diferentes para cada combustível. Quando precisar, o dono deve ser capaz de alterar os impostos ou as alíquotas referentes ao relatório anterior. Os diversos combustíveis comercializados pela empresa possuem um identificador, nome comercial e número de octanas(Octanagem é o índice de resistência à detonação da gasolina). Cada posto tem um identificador único, bem como um nome e morada. As bombas possuem um número que é um atributo com o número sequencial da bomba. Os preços de venda ao público de cada tipo de combustível, para cada posto, em vigor a partir da data DataIn até a data DataFim. Cada venda de combustível a um cliente possui um identificador único, sendo registada com o identificador do posto, o número da bomba, o identificador do combustível, a data e hora da venda e quantidade vendida. Cada uma destas bombas possui sempre várias pistolas numeradas sequencialmente. Uma pistola retira o combustível de um e um só depósito. Naturalmente, não se instalam depósitos

para combustíveis que não podem ser fornecidos pelas pistolas existentes.

5. **CONTROLE DE OCORRÊNCIAS DE UMA DELEGACIA:** Deseja-se informatizar serviços oferecidos por uma delegacia. Para isso foi criado o sistema “DELEGA VIRTUAL”. Detre as atividades informatizadas estão o registro de ocorrências e demais controles internos de uma delegacia. De forma geral o objetivo é necessário saber os dados das vítimas e dos criminosos para dar seguimento aos documentos processuais, bem como armazenar informações sobre cada crime ocorrido. Um criminoso comete um crime contra um ou mais vítimas, e uma vítima pode sofrer um crime por um ou vários criminosos. Um crime é cometido em um município de um estado, e em um crime pode é utilizado um tipo de arma. Além disso, deseja-se ainda permitir o registro on-line de: acidentes de trânsito sem vítima, aviso de desaparecimento ou localização de pessoa, extravio (perda de documentos pessoais ou celulares, por exemplo) e furto simples (quando não há vestígios, como sinais de arrombamento, impressões digitais ou marcas de escalada de muro), solicitação de cópia de registro de ocorrência e crimes eletrônicos. O sistema ainda viabilizará uma maneira fácil de automatizar o acompanhamento desses procedimentos agilizando a manipulação dos dados. A delegacia possui um pátio para guardar os veículos apreendidos devido a infrações de transito. Para cada veículo apreendido deve-se guardar os dados do policial responsável pela apreensão (nome e CPF), assim como os dados do veículo (ano de fabricação, marca, modelo e RANAVAM) e os dados do proprietário do veículo (nome e CPF). Para cada veículo apreendido deve-se armazenar as infrações que levaram a apreensão do veículo. A delegacia possui também o serviço de denúncias, para cada denúncia deve ser armazenado a data da denúncia, o motivo da denúncia, a pessoa que foi denunciada (se houver) e se a denuncia foi confirmada ou não.

6. **FLORICULTURA:** A X.P.T.O LTDA criou a FLOWERNET, uma rede que tem como objetivo atender todo o mercado nacional no que diz respeito à venda e entrega de flores. Através desta rede, um cliente pode fazer uma compra de flores em Belo Horizonte e pedir para a entrega ser feita em Fortaleza. Para isso a X.P.T.O firmou convênio com várias floriculturas em várias cidades do Brasil. Uma floricultura pode atender várias cidades de uma região. O pedido do cliente, que pode possuir vários tipos de flores, é cadastrado e repassado para uma das floriculturas conveniadas que atendem a cidade, na qual será entregue o pedido. Além das atividades já mencionadas pretende-se ainda:
   a) Inicialmente, deseja manter um cadastro de todos os seus clientes, mantendo informações como: CPF, RG, nome, telefone e endereço;
   b) Deseja também manter um cadastro contendo informações sobre os produtos(flores) que vende, tais como: nome do produto, tipo (flor, vaso, planta,...), preço e quantidade em estoque;
   c) Quando um cliente faz uma compra, a mesma é armazenada, mantendo informação sobre o cliente que fez a compra, a data da compra, o valor total e os produtos comprados;
   d) É importante controlar os tipos de flores;
   e) Deve-se controlar o cadastro das floriculturas e os respectivas cidades e estados onde se localizam;

f) Deve-se controlar os dados dos funcionários das floriculturas;
g) O cliente pode fazer seu pedido de forma on-line ou via telefone;
h) O vendedor que atendeu o pedido do cliente por telefone recebe 5% de comissão;
i) Controle também os fornecedores de flores e tipos;

A XPTO ainda mantém um cadastro das possíveis empresas que podem realizar a entrega das flores bem como onde estão localizadas. Os dados destas empresas são fornecidos pelas floriculturas conveniadas..

7. **APURAÇÃO ELEITORAL:** Para facilitar o processamento da apuração eleitoral da eleição municipal a ser realizada nesse ano, o TRE (Tribunal Regional Eleitoral) resolveu informatizar esse processo. Sabe-se que cada localidade é dividida em várias zonas eleitorais que, por sua vez, são divididas em várias seções nas quais os eleitores estão vinculados. O candidato a um cargo público deve estar vinculado a um único partido político. Vale ressaltar que, segundo a legislação vigente, o voto é secreto. Elabore um DER que represente os dados referentes a esse processo, juntamente com os principais atributos, chaves (primárias e estrangeiras), integridades referenciais e a opcionalidade/obrigatoriedade dos relacionamentos.

8. **CONCURSO PÚBLICO:** Uma organização deseja implementar o procedimento de concurso público para tornar transparente o seu processo de seleção de pessoal. Esta organização possui vários departamentos, que por sua vez, possui vários cargos. O mesmo cargo pode estar vinculado à vários departamentos. Um concurso público é realizado para vários cargos, e um cargo pode ser oferecido em vários concursos. O candidato faz inscrição em somente um cargo oferecido em um concurso público. O concurso tem várias etapas, que tem a participação de vários candidatos. O candidato obtém a nota em cada etapa que participa. A etapa pode ser classificatória ou eliminatória. Elabore um DER que represente os dados necessários a este controle, juntamente com as Integridades Referenciais e a Opcionalidade/Obrigatoriedade dos relacionamentos.

9. **LOJA DE DISCOS:** Deseja-se representar os dados necessários para o controle de um conjunto de discos/cds. Uma música pode estar presente em vários discos/cds, que por sua vez é exclusivo de uma gravadora. Uma música pode ter vários compositores e vários interpretes, mas só é classificada em um estilo. Elabore um DER que represente os dados referentes a esse controle, juntamente com os principais atributos, chaves (primárias e estrangeiras), integridades referenciais e a opcionalidade/obrigatoriedade dos relacionamentos.

10. **FATURA DE CONTA DE TELEFONE:** Uma empresa prestadora de serviços de Telecomunicações deseja informatizar a emissão de sua fatura de serviços. Os principais dados desta fatura são: número do telefone, data de vencimento, data de emissão, código do cliente, nome e endereço do cliente, mês de referência dos serviços e, um espaço reservado para no máximo 10 serviços, onde cada um tem data e hora, tipo do serviço, descrição do tipo do serviço, tempo de utilização, telefone chamado, valor unitário, valor da utilização e o percentual de ICMS do tipo do serviço. Além disso

tem-se o total da fatura. Normalize a entidade Fatura Serviço até a 3FN, passando pela 1FN e 2FN. Represente os principais atributos, chaves, Irs, obrigatoriedade e opcionalidade.

11. **CONTROLE DE EQUIPAMENTOS:** Uma empresa deseja controlar os seus equipamentos de informática. Existem vários tipos de equipamento, tais como: CPU, impressoras, modem, etc. Cada equipamento está fisicamente em um departamento da empresa. Somente o empregado responsável pelo departamento, o chefe, pode solicitar a compra de um novo equipamento. Deseja-se também controlar as manutenções realizadas em cada equipamento. Elabore um DER que represente esses dados, juntamente com os principais atributos, chaves (primárias e estrangeiras), integridades referenciais e a opcionalidade/obrigatoriedade dos relacionamentos.

12. **CONTROLE DE HOSPEDAGEM E UTILIZAÇÃO DE SERVIÇOS DE UMA REDE DE HOTÉIS:** Uma rede de hotéis necessita mapear/modelar os dados necessários ao seu controle de hospedagem e a utilização de serviços pelos hóspedes. Sabe-se que, ao solicitar uma reserva, os dados do cliente são devidamente cadastrados, inclusive o tipo de convênio que está sendo utilizado. Ao dar entrada no balcão de atendimento, o cliente passa assumir a condição de hóspede. Vários quartos podem estar relacionados com o mesmo hóspede, como, por exemplo, no caso de viagem em família. Mesmo assim a empresa deseja guardar informações de quais pessoas encontram-se em cada acomodação. Os quartos são classificados como de luxo e *standard*, e podem ser ocupados por vários hóspedes em períodos distintos. O serviço de copa do hotel registra os pedidos de itens do cardápio em função do relacionamento entre hóspede e quarto. O mesmo ocorre para os demais serviços do hotel, como eventos/passeios, lavanderia, sauna, etc. Elabore um DER que represente esses dados, juntamente com os principais atributos, chaves (primárias e estrangeiras), integridades referenciais e a opcionalidade/obrigatoriedade dos relacionamentos.

13. **DIÁRIO DE CLASSE:** Uma universidade deseja informatizar o controle de frequência de seus alunos. Este controle é realizado através dos diários de classe de cada turma. Os principais dados do diário de classe são: sigla e nome do curso, matrícula e nome do professor, código e nome da turma de uma disciplina, período e turno da turma, um espaço reservado para os alunos da turma (um aluno pode fazer parte de várias turmas), o mês e o ano de referência do diário de classe. Além disto, cada aluno/turma está relacionado com os dias de cada mês de referência do diário de classe, ou seja, com os dias que acontece as aulas da turma e para os quais serão registrados as frequências dos alunos (P - presente e F - falta). Elabore um DER na 3FN que represente os dados necessários a esse controle. Represente os principais atributos, chaves, Irs, obrigatoriedade e opcionalidade.

14. **CAMPEONATO DE FÓRMULA I:** A Federação Internacional de Automobilismo deseja controlar os dados referentes aos Campeonatos Mundiais de Fórmula I. As equipes, de diversos países, participam dos campeonatos. Uma equipe possui vários pilotos, e exige a sua fidelidade. Cada Grande Prêmio (GP) de um campeonato é realizado em um país diferente, e tem a participação de

vários pilotos. A pontuação é obtida por um piloto em cada GP que participa. Deseja-se controlar também a nacionalidade dos pilotos. Elabore um DER que contemple os dados necessários a este controle. Represente as chaves primárias e estrangeiras, os principais atributos e a obrigatoriedade/opcionalidade dos relacionamentos.

15. **PLANO DE SAÚDE:** Uma empresa, que gerencia um plano de saúde, deseja construir um sistema para facilitar e agilizar o gerenciamento das internações, seja em enfermaria ou em apartamento do tipo *standard*, realizadas por seus associados (usuários), bem como o controle dos pagamentos que devem ser realizados aos hospitais credenciados junto à empresa. O controle de mensalidades dos associados não faz parte do escopo do sistema. A empresa só trabalha com um tipo de plano de saúde. Só se deseja controlar as diárias de internações.

16. **ESTACIONAMENTO UNIVERSITÁRIO:** Uma Universidade deseja construir um sistema para facilitar e agilizar o controle de acesso de veículos aos seus cinco estacionamentos para prover mais segurança e comodidade para os seus usuários. O sistema deverá permitir que se cadastre todos os tipos de usuários (alunos, professores e funcionários), que receberão um cartão com um código de barra para sua identificação. Cada usuário poderá solicitar o cadastramento de vários veículos com os quais utiliza os estacionamentos da universidade. Ao chegar a qualquer portão de acesso à universidade, o vigilante irá informar a placa do veículo e o usuário deverá passar o cartão magnético em um leitor de código de barras, e com isso, o sistema irá identificar se o veículo está relacionado com a identificação do usuário. Ao sair, o usuário simplesmente passará o seu cartão em outra leitora de código de barras. O visitante (usuário não cadastrado) deverá pegar um cartão especial com os vigilantes. Através desses procedimentos, o sistema poderá fornecer dados de ocupação de cada estacionamento, além de permitir a consulta de quais os veículos estão, ou estiveram, dentro da universidade em um determinado dia e horário.

17. **LOJA DE MATERIAIS DE CONSTRUÇÃO:** Uma organização que atua no ramo de vendas de materiais de construção deseja montar um banco de dados para emissão de faturas em suas lojas, gerenciando a comissão de cada empregado. Sabe-se:
    - A empresa possui diversas lojas;
    - Um empregado pertence sempre a uma loja mas já pode ter trabalhado em outras lojas da rede;

    Uma nota fiscal é composta de dados genéricos (número da nota fiscal, nome do cliente, data de emissão, valor total da Nota fiscal, nome do empregado responsável pela venda) e dados do detalhe da venda (nome do material vendido, quantidade deste material, valor unitário, valor total do item de material vendido).

16. **CONTROLE DE EVENTOS:** Uma empresa, chamada EVENTO MANIA, especializada em eventos necessita informatizar o controle de seus eventos. Tendo-se em vista a longa experiência da empresa no ramo de organização de eventos a mesma definiu os seguintes requisitos a serem

atendidos pelo sistema:

- Controlar o cadastro de participantes e palestrantes. É importante manter dados completos de ambos já que ao final serão emitidos certificados e declarações para os mesmos. Deve ser cadastrada a foto do participante e do palestrante;
- O sistema deverá permitir o cadastro dos eventos. Vários eventos podem estar ocorrendo simultaneamente em lugares diferentes;
- O sistema deverá também permitir o cadastro da programação completa de cada evento, programação esta que pode ser composta por: palestras ou oficinas;
- O sistema deve controlar o processo de inscrição no evento bem como as formas de pagamento para o mesmo;
- Durante o evento o sistema deve controlar o acesso ao evento. Não será permitida a entrada de participantes que não estão inscritos ou que não quitaram o pagamento da inscrição. No entanto, é possível realizar inscrições de cortesia;
- É importante controlar o choque de horário na inscrição no evento, como assim, um participante não pode estar inscrito para programações diferentes, no mesmo horário, no mesmo evento;
- Uma programação tem um limite de participantes, isso é definido em função do local onde será realizada a mesma, local este que pode ser: uma sala, um auditório ou laboratório. Estes locais possuem limite de capacidade e portanto o sistema não deve permitir que esta capacidade seja excedida;
- No cadastro do palestrante ele podem informar um resumo da palestra e também os recursos necessários para sua apresentação dentro de uma programação;
- Uma mesma programação pode apresentar mais de um palestrante;
- Alguns relatórios importantes para este sistema:
    - Certificado ou declaração de participação contemplando: as programações, horas, palestrante, dados pessoais do participante;
    - Lista de inscrições: pagas, não pagas, cortesia, inscritos presentes;
    - Lista de inscritos presentes;
    - Crachás para o controle de acesso;
    - Quadro com a programação completa do evento;
    - Cidade com mais inscritos e cidade com menos inscritos;
    - Evento, número de inscritos. Este relatório pode ser dividido em inscritos pagantes, não pagantes, presentes e não presentes.

17. **AGÊNCIA DE TURISMO:** Uma empresa de turismo necessita de uma aplicação que favoreça o cadastramento de seus clientes e o controle dos serviços utilizados por eles. Esta empresa trabalha com diversos fornecedores que oferecem produtos distintos, como hotéis, passagens aéreas, aluguel de veículo, etc.. Um mesmo fornecedor pode oferecer mais de um tipo de serviço. Quando um cliente compra um serviço desta empresa, é emitida uma nota de pedido relacionando todos os produtos que envolveram a transação. Cada serviço possui uma modalidade de pagamento, e na negociação, o atendente da agência de turismo poderá ou não unificar a modalidade de pagamento

para todo o pedido. A aplicação encomendada irá controlar as ordens de pagamento para os fornecedores destes serviços, assim como, a cobrança aos clientes. Esta ultima é feita via banco para o qual a agência de turismo deverá repassar as informações do cliente, da data de vencimento e do valor a ser cobrado. Este banco irá retornar a agência a informação dos pagamentos efetuados, quando, então, será dado baixa no arquivo de contas-a-receber.

18. **IMOBILIÁRIA:** Uma imobiliária com especialidade em aluguéis deseja desenvolver uma aplicação para controle do recebimento e repasse dos aluguéis a seus clientes. Esta empresa cadastra todos os seus clientes e os inquilinos destes. Ela necessita guardar informações dos imóveis sob sua responsabilidade e dos contratos de aluguel por ela efetuados. Para facilidade de controle nos pagamentos dos aluguéis, estes serão feitos via sistema de cobrança bancária. O banco credenciado, após a cobrança, irá depositar automaticamente na conta da imobiliária o valor de 5% relativo à comissão de seus serviços e o restante, na conta do proprietário do imóvel, em conta/banco/agência indicada por ele. O sistema mensalmente repassará ao banco de cobrança informações para tal. Os contratos podem ser cancelados a qualquer instante, dentro da legislação vigente. Quinzenalmente o sistema deverá emitir uma relação dos inquilinos em atraso.

19. **EMPRESA:** As seguintes características são representadas na base de dados. A empresa é organizada em departamentos, cada qual tendo um nome, um número de identificação e um empregado responsável (gerente) pelo mesmo. Guarda-se como informação a data a partir da qual o empregado assumiu a gerência do departamento. Cada departamento pode estar presente em diversas localidades do país. Um departamento controla um certo número de projetos, cada qual tendo um nome, um número de identificação e uma localidade única. Os departamentos têm vários empregados, para os quais guarda-se os respectivos nomes, número de matrícula na empresa, endereço residencial, sexo, data de nascimento e salário. Todo empregado tem um empregado que é seu superior hierárquico (supervisor direto) dentro do departamento, informação que também deve ser mantida no banco de dados. Um empregado é alocado a um único departamento mas pode estar alocado a mais de um projeto, estes não necessariamente controlados pelo mesmo departamento. Controla-se o total de horas semanais em que um empregado trabalha em cada projeto. Para fim de controle de seguro de saúde de cada empregado, deve-se manter informações dos nomes, sexo, data de nascimento e grau de parentesco de seus dependentes.

20. **REVENDA DE AUTOMÓVEIS:** Na relação AUTOMOVEIS cada automóvel é identificado por um código nacional, de acordo com o nome do fabricante e modelo do carro. Os preços de tabela são determinados também pelo ano do carro. Apenas revendedoras autorizadas cadastradas na relação REVENDEDORAS podem vender os carros no mercado. Estas podem estar em diversas cidades e estados do país e seu CGC identifica revendedoras. Consta também o nome da revendedora e a identidade de seu proprietário. Os consumidores têm identidade única em território nacional e são cadastrados na relação CONSUMIDORES contendo também nome e sobrenome. Cada negócio efetuado é registrado na tabela NEGOCIOS com detalhamento de data e preço pago, além da

identidade do comprador, a identificação da Revenda (ou CGC da revendedora) e do código e ano do automóvel. Supõe- e que um mesmo consumidor não compra um automóvel de características idênticas em uma mesma revendedora em datas diferentes. Por fim, a relação GARAGEM determina quais automóveis as revendedoras têm a intenção (se de fabricante exclusivo, se apenas carro do ano etc.) de negociar e qual seu potencial de vendas a cada momento, isto é, quantidade de carros na garagem que podem ser negociados.

21. **LOCADORA DE CARROS:** Uma locadora aluga carros aos clientes previamente cadastrados. Caso o cliente não esteja cadastrado, esta atividade é realizada. Caso um carro, disponível, seja escolhido pelo cliente este é alugado, sendo registrada a data inicial junto ao aluguel. Para que o cliente possa alugar um carro, este não pode estar com dívida pendente. Os carros são descritos pela placa, ano, modelo, descrição, km, preço por km, situação(disponível, etc), taxa diária, observações (informações gerais) e sua imagem. Os clientes são cadastrados pelo seu cpf, nome, endereço, telefone e dívida(reservado para registrar pagamentos pendentes). Quando o cliente devolve o carro, a situação do carro é mudada para “disponível”, o km é atualizado com o km atual do carro e um recibo é emitido, baseado nos KMs rodados e nos dias em que ficou com o carro. Ainda na atividade de devolução, caso o cliente não possa pagar, a dívida do aluguel é registrada junto ao cliente. O Cliente pode a qualquer momento pagar sua dívida e o gerente pode solicitar relatórios sobre as informações da locadora. Principais requisitos operacionais que devem ser atendidos pelo modelo conceitual:
    - Controlar o aluguel e devolução dos carros;
    - Controlar o pagamento de aluguel dos carros;
    - Manter cadastros de clientes e carros;
    - Emitir relatórios e permitir consultas; e
    - Controlar a dívida dos clientes.

22. **ADMINISTRAÇÃO DE CIRURGIAS:** Para este problema foi definido um conjunto de informações obtidas em uma reunião para definir o que o sistema deve controlar e realizar na visão dos usuários administradores. Em um hospital existem diversas salas em seu centro cirúrgico. As salas se cirurgia possuem recursos para grupos de especialidades médicas, sendo apropriadas para cirurgias de uma especialidade. Existem salas que se prestam somente a cirurgias de uma única especialidade, pois estão dotadas de equipamentos apropriados a essa especialidade médica. Os médicos que realizam cirurgias no hospital em seus pacientes agendam as cirurgias conforme a disponibilidade das salas, informando datas, hora inicial e hora final prevista, junto à operadora do sistema. Os horários para controle das salas são modulados de 30 em 30 minutos, de forma a manter-se uma quebra equilibrada da distribuição de horários. Quando uma cirurgia é encerrada, é informado ao controle do Centro Cirúrgico para a preparação e higienização para a próxima ocupação. Uma sala de cirurgia não pode ser utilizada simultaneamente por mais de uma cirurgia, pois somente existe um leito cirúrgico em cada uma. Um médico não pode ter cirurgias coincidentes

em data e horário, mesmo que em salas diferentes. Uma cirurgia somente deve ser realizada em uma sala apropriada para a sua especialidade. Admitem-se urgências, porém devem ser destacadas para utilizarem salas que não as específicas da especialidade. Urgência é considerada uma qualificação da cirurgia, mas não tem restrição de especialidade, podendo utilizar qualquer uma das salas. Toda cirurgia tem somente um médico responsável, que devemos registrar para o controle do sistema. Os medicamentos, materiais e remédios consumidos pela cirurgia devem ser computados para cobrança posterior. O paciente é identificado pelo hospital, inclusive com o seu leito de internação e datas. Não existe no sistema nenhum controle pré-cirúrgico. O sistema deve controlar cirurgias marcadas, assim como as já realizadas. Para o Centro Cirúrgico especialidade é um dado do tipo: Cardiorrespiratória, Nefrologia, Ginecologia e Obstetrícia, Transplantes, Gastroenterologia, Oftalmologia, Traumatologia, Cirurgia Plástica, Oncologia, etc. Os médicos são todos registrados como habilitados no hospital, suas especialidades também controladas, podendo realizar cirurgias em somente uma especialidade. São considerados materiais gastos em uma cirurgia as agulhas de sutura, algodões, sangue, gaze. Como medicamentos temos produtos como anestésicos em geral , antissépticos, soro, etc., e remédios diversos com aplicação intravenosa.

23. **CONTROLE DE CINEMAS:** Um cinema chamado CINELEGAL pretende se informatizar. Como requisitos foram elencados os seguintes:
    - O cinema possui várias salas de cinema, as quais exibem filmes em horários diversos;
    - O cinema tem interesse em saber quais filmes estão atualmente em cartaz, em que salas e em que horários;
    - Cada sala possui um nome (único) e capacidade (número de lugares);
    - Os filmes são caracterizados por seu nome em português, nome na língua original (se estrangeiro), diretor, ano de lançamento, tipo, sinopse e imagens de divulgação. É importante saber se o filme é legendado ou dublado;
    - Não existem dois filmes com o mesmo nome (em português) e ano de lançamento. Eventualmente, podem existir para o filme premiações ou indicações para premiação (e.g. Palma de Ouro em 1987, Oscar de melhor atriz em 89, indicado para melhor filme estrangeiro em 1996), e esta informação é usada para divulgação dos filmes;
    - Uma exibição de filme ocorre em uma dada sala e horário. Um mesmo filme pode ser exibido na mesma sala, em vários horários. Para filmes muito procurados, o cinema pode ter exibição simultâneas em várias salas (em horários simultâneos ou não).
    - Filmes diferentes podem passar na mesma sala, desde que obviamente não no mesmo horário. O cinema só trabalha com horários fixos de filmes, os quais atualmente são: 16:00, 17:00, 18:00, 19:30, 20:00, 22:00, 24:00. A cada um destes horários está vinculado um conjunto de funcionários responsáveis pelo bom andamento das atividades do cinema naquele horário, e que desempenham uma função (ex: caixa, balas, lanterninha, bilheteiro);
    - Cada funcionário é caracterizado pelo número da carteira da trabalho (único), nome, data de admissão e salário. Para maior satisfação dos funcionários, existe um rodízio das funções conforme o horário (ex: um mesmo funcionário pode ser caixa no horário das

16:00, e baleiro no horário das 21:00). Todo horário tem pelo menos três funcionários alocados.

24. **OFICINA MECÂNICA:** Em uma oficina mecânica chamada “VRUM” especializada em veículos, existe a necessidade de se implementar um banco de dados(sistema) para organizar os veículos deixados na oficina para que seja feito o orçamento de serviço e a futura execução. O dono da oficina gostaria que existisse um cadastro dos clientes da oficina e um cadastro de todos os veículos deixados na oficina pelos clientes. Além disto, ele gostaria de armazenar os orçamentos de cada veículo.

Seguindo as orientações do dono da oficina, podemos definir alguns requisitos a serem atendidos:

a) O cadastro do cliente deve conter o CPF, o nome, endereço e o telefone de cada cliente;
b) O orçamento de cada veículo é composto por um número que o identifica unicamente, a data do orçamento e deve pertencer a um e apenas um veículo. Cada orçamento é composto por 1 ou mais serviços. Além disto, um serviço pode estar presente em mais de um orçamento;
c) O orçamento entregue para o cliente contém também o valor de cada serviço e o valor total do orçamento. O valor de cada serviço, é calculado consultando-se uma tabela de referência de mão-de-obra. O valor de cada peça necessária à execução do serviço também é computado;
d) Aprovado o orçamento, cada veículo é designado a uma equipe de mecânicos que identifica os serviços a serem executados e preenche uma ordem de serviço (OS) e prevê uma data de entrega;
e) Os veículos possuem código, placa, ano de fabricação, descrição, marca e modelo;
f) Cada mecânico possui código, nome, endereço e especialidade;
g) Cada OS possui um número, uma data de emissão, um valor e uma data para conclusão dos trabalhos. Uma OS pode ser composta de vários itens (serviços) e um mesmo serviço pode constar em várias ordens de serviço. Uma OS pode envolver vários tipos de peças e um mesmo tipo de peça pode ser necessária em várias ordens de serviço.
h) Os clientes trazem seus veículos à oficina mecânica para serem consertados ou para passarem por revisões periódicas. As revisões periódicas também podem gerar um orçamento e uma ordem de serviço para execução dos reparos;
i) Pode existir ordens de serviço que não foram originadas a partir de orçamentos prévios.

25. **EMPRESA DE MANUFATURA:** Uma empresa de manufatura funciona em um esquema de projetos, nos quais são alocados seus empregados, com um certo percentual de dedicação. Administrativamente os empregados estão lotados em departamentos e podem gerenciar um ou mais projetos que por sua vez são gerenciados por somente um empregado. As peças utilizadas nos projetos são armazenadas nos vários armazéns da empresa. É necessário controlar-se, também , a composição das peças. A empresa mantém um controle do fornecimento efetivo de

peças feito aos projetos pelos fornecedores, e um controle do fornecimento potencial das peças de cada um dos fornecedores.

26. **CONTROLE DE ESTÁGIOS:** A universidade UNIMUNDO pretende informatizar o processo de gestão das disciplinas de estágio ou trabalho de conclusão de curso(adotaremos o termo estágio). Esta etapa é essencial para a formação acadêmica e possui diretrizes bem definidas para sua execução. Com base nestas diretrizes pode-se definir os seguintes requisitos a serem atendidos pelo sistema:
    - Controlar o cadastro dos alunos orientandos, professores orientadores e dos coordenadores de estágio;
    - Permitir o controle em fases do processo de estágio, ou seja, algumas disciplinas de estágio podem ser divididas em 1, 2 ou mais fases ou etapas, somente ao final destas fases a disciplina poderá ser considerada como completada;
    - Permitir o cadastro dos trabalhos: título, tema, curso, etc.;
    - Armazenar e controlar os mais diversos tipos de documentos pertinentes ao estágio. Estes documentos contemplam versões eletrônicas de: artigos, trabalhos, outros formulários;
    - Permitir o registro informatizado do andamento do trabalho, encontros realizados, etc;
    - Permitir a configuração de calendário para as atividades do estágio durante as fases previstas;
    - Permitir a configuração das bancas de estágio;
    - Permitir e controlar o lançamento de notas de avaliação dos trabalhos;
    - O coordenador de estágio pode através do sistema controlar a publicação do mesmo para posterior consulta de forma on-line por parte dos demais alunos e professores.

27. **PIZZA ON-LINE:** Pretende-se desenvolver um sistema de informação para apoiar a gestão de encomendas de um grupo de pizzarias – PizzaInHouse. O sistema irá prestar serviços de atendimento, acompanhamento de clientes e de encomendas. Para além das pizzarias (lojas), o grupo tem uma Central de Atendimento aos Clientes (CAC) que disponibiliza um serviço de encomendas de produtos por telefone e pela Internet. Para o efeito, se o cliente contatar por telefone deve indicar a sua localização. Caso se encontre na área de cobertura do grupo, a CAC efetua a encomenda e envia-o para a pizzaria mais próxima da zona onde o cliente se encontra. Se o pedido for feito pela Internet, o utilizador tem que efetuar primeiro um pré-registo (indicando login, password, morada, telefone e email). Em seguida recebe via email um código de acesso para poder ativar os serviços disponibilizados pela pizzaria. Após a ativação dos serviços, o código de acesso não será jamais utilizado. Assim que o cliente receber o código de acesso, pode efetuar encomendas através do seu login e password. A pizzaria tem um catálogo que consiste numa listagem de produtos por código, nome, descrição e preço. Os produtos disponíveis são bebidas, saladas, entradas, pizzas, massas e sobremesas. Uma pizza é constituída por uma pizza base (Margarita) à qual se podem adicionar até 10 ingredientes. Existem também pizzas com os ingredientes pré-definidos (Vegetariana, Tropical). A pizzaria é composta pelas seguintes áreas:

restaurante, balcão, entregas, cozinha e armazém. É no restaurante que estão as mesas. As mesas (numeradas) podem ser reservadas pelos clientes, sendo a reserva referenciada pelo número da mesa e pela data. Um cliente pode reservar várias mesas. Na pizzaria, o pedido de uma encomenda pode ser feito ao balcão ou à mesa, havendo em qualquer dos casos um terminal do sistema com um ecrã táctil disponibilizado para introduzir o pedido. Os funcionários distribuem-se pelas seguintes categorias: gestor de loja, empregado de mesa, gestor de encomendas, estafeta. Todos têm um número e podem trabalhar em várias lojas, sendo importante saber a data e a duração do contrato em cada loja. Cada loja é caracterizada por código, nome, zona de influencia. O procedimento para encomendar é idêntico, quer o pedido seja feito por telefone, pela Internet ou localmente. Quando uma pizzaria recebe uma encomenda, adiciona-a à fila de espera de encomendas a satisfazer. Assim que a encomenda começa a ser confeccionada, o seu estado passa a “processo”. Logo que todos itens da encomenda estejam prontos para serem entregues, esta é entregue ao cliente pelo empregado de mesa ou pelo estafeta, dependendo do tipo da encomenda. O estado da encomenda passa então a ”caminho”. Assim que o cliente recebe a encomenda, é gerada uma fatura que é caracterizada por um número, data, itens de produto com valor unitário e valor total. Nessa altura, o estado da encomenda passa a “entregue” e toda a informação referente à encomenda e respectiva fatura é arquivada.

28. **CLINICA MÉDICA:** A clinica médica “BEM-ESTAR” resolveu informatizar suas atividades. Nesta clínica trabalham médicos e existem pacientes internados. Cada médico é identificado pelo seu CRM, possui um nome e recebe um salário na clínica. Um médico tem formação em diversas especialidades (ortopedia, traumatologia, etc), mas só exerce uma delas na clínica. Para todo paciente internado na clínica são cadastrados alguns dados pessoais: nome, RG, CPF, endereço, telefone(s) para contato e data do nascimento. Um paciente tem sempre um determinado médico como responsável (com um horário de visita diário predeterminado), porém vários outros médicos podem participar do seu tratamento. Pacientes estão sempre internados em quartos individuais, que são identificados por um número e ficam em um andar da clínica. Além disso, os médicos podem atender diversos convênios. Para cada convênio, é necessário identificar os médicos associados, além de algumas informações, como o seu código, razão social, telefone para atendimento e endereço. Na recepção da clínica, são registradas as consultas. Cada consulta está associada a um paciente e a um único médico, além de um código, descrição, data e hora, medicação e o diagnóstico. Em relação aos pacientes, é necessário registrar seu código, nome, endereço, telefone e data de nascimento. O sistema ainda deve controlar: a alteração da senha de acesso; o agendamento de consultas; o cadastro dos medicamentos; o cadastro de exames complementares; a geração de receitas; a geração de laudos; a consulta do histórico dos pacientes (prontuário); a atualização do prontuário; o cadastro de profissionais médicos; o cadastro de planos de saúde conveniados; o cadastro dos quartos; em quais quartos o paciente está internado; e o cadastro dos funcionários e cargos da clínica.

29. **ORGANIZAÇÃO DE CONFERÊNCIAS:** A universidade UNIFANTASIA deseja informatizar a parte

de conferências organizadas pela mesma. Esta universidade costuma organizar variados tipos de conferências e com variadas dimensões. O workflow(fluxo de dados) de uma conferência deste nível prevê as etapas de: submissão, avaliação, publicação e posterior apresentação em evento, dos artigos. Para atender estas necessidades o sistema deverá apresentar os seguintes requisitos:

- Permitir o cadastro de autores(codigo, nome, instituição de ensino, endereço completo, e-mail, login, senha, foto do autor);
- Permitir que os autores possam submeter de forma on-line seus artigos;
- O artigo deverá contemplar(autor, título, área de conhecimento, resumo);
- Após a submissão o artigo será revisado e avaliado. O autor poderá acompanhar o processo de avaliação de seus artigos também de forma on-line;
- Para a avaliação/revisão dos artigos são definidos profissionais especializados da própria universidade. São necessários serem definidos 3 avaliadores/revisores;
- A avaliação do artigo não resulta em uma nota, mas sim em um parecer descritivo dos revisores;
- Emitir avisos nas diversas situações do processo de gestão: revisores que se estão atrasados no processo de revisão, autores que ainda não submeteram a versão final do seu artigo, etc.;
- Permitir a criação de sessões, permitir a organização de horários para apresentação dos artigos, e definir os artigos que irão compor cada uma delas;
- Permitir a todos os utilizadores do sistema um simples e eficaz processo de inscrição na conferência, prevendo os tipos de participantes: aluno, professor, funcionários e comunidade em geral. Deve ser cadastrada também uma foto do participante para posterior emissão no crachá de acesso a conferência;
- Não esquecer que uma inscrição é paga e que o inscrito pode escolher qual apresentação de artigo deseja ver;
- É importante controlar a situação do artigo, isso porque o mesmo pode ser aprovado ou não para a conferência.

30. **CLINICA VETERINÁRIA/PET-SHOP:** A clínica veterinária “CAOSARADO” deseja informatizar suas atividades. Consultados, o dono da clínica e alguns funcionários, descreveram suas atividades na clínica assim:
    a) Um cliente primeiramente se dirige à Clínica onde marca uma consulta com a secretária, fornecendo suas informações pessoais e do animal que deseja tratar. Se o cliente ou o animal ainda não estiverem cadastrados no sistema ou possuam algum dado que precise ser atualizado, a secretária deverá atualizar seus cadastros;
    b) Em cada sessão de tratamento (uma sessão equivale a uma consulta), o cliente deve informar os sintomas aparentes do animal e estes devem ser registrados. Um tratamento pode ser encerrado em apenas uma consulta, quando se tratar de algo simples ou pode se arrastar por muitas sessões dependendo do diagnóstico do médico-veterinário;
    c) Durante uma sessão o veterinário pode marcar exames para o animal, a serem trazidos na

sessão seguinte. O pedido dos exames, bem como seus resultados devem ser registrados no histórico de tratamentos do animal;

d) Após cada sessão, o histórico da consulta deve ser atualizado e gera-se uma conta a receber a ser paga pelo cliente. A manutenção das consultas é responsabilidade exclusiva do médico-veterinário que a realizou;
e) É responsabilidade da secretária manter atualizados os cadastros de clientes, animais, médicos e espécies;
f) É importante controlar os dados dos funcionários da clinica, como: médicos, secretária e demais funcionários;
g) O sistema deve controlar de forma geral: o cadastro de animais, clientes, raças, espécies, vacinas e vermífugos, registros de consulta(agendamento), vacinação e vermifugação(combate à verminose).

31. **VIDEO LOCADORA:** A locadora "VIDEOPLAY" trabalha com os seguintes mídias: fitas de video, DVDs e discos *blu-ray*. O cliente da video locadora é previamente cadastrado pelo *nome*, *endereço*, *telefone* por um número de *celular* e *CPF*. Cada cliente "pode" *locar* uma ou mais fitas. Cada *locação* registrará a *data*, a *hora*, o *histórico* sobre a situação das fitas locadas e o *valor* da locação. As fitas disponíveis na locadora são identificadas por um *código da fita* (que é obrigatoriamente atribuído de forma manual assim que a fita é comprada, é um número de cinco dígitos sequêncial único), o *título* (que pode se repetir. Ex: Titanic (antigo) Titanic (moderno)), a *sinopse*, a *duração* e a *quantidade*. Uma fita "pode" ser *locada* por um ou mais clientes. Uma fita "pode" *ser* uma e somente uma comédia. As comédias são registradas pela sua *trilha sonora, diretor* e por ser ou não um filme *premiado*; Uma comédia "deve" *ser* uma e somente uma fita na locadora. Cadastramos também o distribuidor de fitas para a locadora. Cada distribuidor "pode" *ter* uma ou mais fitas associadas ao seu registro. O distribuidor é registrado pela *razão social* (pode ser repetida para distribuidores de cidades diferentes), *endereço*, *telefone*, um nome de *contato* e *CNPJ*. Uma fita "deve" *ter* um e somente um distribuidor registrado no sistema.

32. HELP DESK: Uma empresa de suporte a microinformática pretende informatizar o processo de atendimento a seus clientes. Para isso a mesma elencou as principais necessidades que precisariam ser atendidas e controladas por este sistema de informação. Dentre estas necessidades estão:

a) O sistema deve permitir o controle e cadastro de usuários, atendentes e técnicos;
b) O sistema deverá possibilitar cadastro de chamados pelo atendente e enviar um e-mail atribuindo o próximo chamado da fila ao técnico que estiver disponível;
c) Quando o atendente der por concluído o chamado o sistema deve enviar um e-mail com um link para que o usuário confirme a resolução do chamado. E um e-mail para o atendente que abriu o chamado;
d) O sistema deve fechar o chamado e arquivá-lo quando o técnico encerre a última tarefa do chamado;

e) Os usuários e técnicos devem poder consultar o status dos seus chamados;
f) Chamados relacionados a hardware também serão controlados, para isso o sistema deverá permitir ao técnico manter um cadastro de hardware (computadores, monitores, scanners e impressoras etc). Além disso o cadastro de peças disponíveis no estoque também deverá ser mantido;
g) O sistema deverá vincular as peças de estoque ao equipamento onde foram instaladas;
h) Com as soluções propostas para os chamados será construída uma base de conhecimento, ou seja, deverá ser permitida a consulta a esta base caso já existam problemas similares;
i) O sistema deve possibilitar ao atendente classificar o chamado como problema de hardware, software, equipamento terceirizado, servidores, entre outros. E conforme o tipo criar uma tarefa para o técnico responsável pelo tipo de atendimento;
j) O sistema deverá permitir a emissão de uma série de relatórios tanto para o uso operacional quanto para os gestores do processo.

33. **CAMPEONATO DE FUTEBOL:** No campeonato de futebol da 1ª divisão participam equipes, compostas por jogadores. Cada equipe tem um treinador e um dirigente. As equipes jogam entre si, em jogos sobre os quais se mantém informações sobre a data, resultado final e numero de espectadores. Os jogos são arbitrados por um árbitro principal e dois auxiliares. Os jogos ocorrem nos estádios das diversas equipes, identificados por nome, número de lugares e localização. Todas a pessoas acima mencionadas são identificadas pelo nome e data de nascimento. Além disto, para cada jogador é mantida a data fim de contrato com a equipe para o mesmo quem joga. Para cada dirigente, mantém-se a data final do mandato, e para cada árbitro a data de início das atividades na 1ª divisão.

34. **LIVRARIA:** Uma livraria mantém o cadastro de livros disponíveis para a venda. Para cada livro são armazenados código, nome, língua e ano em que foi escrito. Para os autores é mantido igualmente um cadastro que inclui nome, data de nascimento, pais de nascimento e uma breve nota biográfica. Cada livro pode ter vários autores e para um mesmo autor podem existir vários livros cadastrados. Um autor pode estar incluído no cadastro ainda quando não exista um livro seu para venda. As editoras são incluídas no cadastro a partir do seu nome, endereço, telefone. Uma editora pode estar cadastrada mesmo quando não existam livros editados por ela em venda. Para um mesmo livro podem existir várias edições realizadas por editoras diferentes ou em anos diferentes . Cada edição tem um código (ISBN) , preço, ano, número de páginas e quantidade em estoque. Considere que um livro pode ser cadastrado se existe pelo menos uma edição do mesmo para venda.

35. **AGÊNCIA DE FINANCIAMENTO:** Uma agência de financiamento de projetos de pesquisa deseja criar um sistema de banco de dados para gerenciar seu funcionamento. Para cada projeto são cadastrados: um código interno, título, duração do projeto, instituição onde será realizado e área de pesquisa. As áreas de pesquisa estão predefinidas é para cada uma delas são cadastrados código, nome, descrição e um índice que indica sua relevância econômica. Para cada pesquisador solicitante são cadastrados: RG, CPF, nome, sexo, data de nascimento, grau cientifico e instituição

onde foi alcançado esse título. Note-se que um mesmo pesquisador pode ter vários projetos em análise. Um pesquisador é cadastrado no sistema unicamente quando o primeiro dos seus projetos é submetido. A agência recebe os projetos submetidos pelos pesquisadores e associa cada um destes a um assessor que deve aprovar ou não o financiamento. Para estes assessores são cadastrados: RG, CPF, nome, sexo, data de nascimento, grau científico, instituição onde trabalha e as áreas nas quais tem capacidade de avaliar projetos. Estas áreas de pesquisa devem ser definidas dentre a lista de áreas predefinidas antes mencionadas. Um assessor pode ser cadastrado mesmo sem ter analisado nenhum projeto. Quando um projeto é enviado a um assessor para análise é cadastrada a data deste envio. Posteriormente, quando o assessor retorna sua avaliação são também cadastrados a data de resposta e o resultado de aprovação ou não do projeto.

36. **ACADEMIA:** Uma academia de ginástica deseja manter um controle do seu funcionamento. Os alunos são organizados em turmas associadas a um tipo específico de atividade. As informações sobre uma turma são número de alunos, horário da aula, duração da aula, data inicial, data final e tipo de atividade. Cada turma é orientada por um único instrutor para o qual são cadastrados RG, nome, data de nascimento, titulação e todos os telefones possíveis para sua localização. Um instrutor pode orientar várias turmas que podem ser de diferentes atividades. Os dados cadastrados dos alunos são: código de matricula , data de matrícula, nome, endereço, telefone, data de nascimento, altura e peso. Um aluno pode estar matriculado em várias turmas se deseja realizar atividades diferentes e para cada matrícula é mantido um registro das ausências do aluno. Para cada turma existe um aluno monitor que auxilia o instrutor da turma, sendo que um aluno pode ser monitor no máximo em uma turma.

37. **AUTO-ESCOLA:** Construa o modelo de um sistema que permita controlar e automatizar os processos de sua auto-escola desde cadastros e matrículas a acompanhamentos de aulas práticas e teóricas permitindo acompanhar alunos, instrutores e veículos. O sistema deverá emitir relatórios de todas as movimentações de aulas práticas e teóricas por alunos, acompanhamentos dos resultados dos exames junto ao Detran. Também deverá ser possível obter relatórios de movimentações financeiras além de poder acompanhar custos por veículo ou por aluno, entre outros inúmeros relatórios e estatísticas emitidos pelo sistema. O sistema deverá ainda controlar os gastos da Auto Escola, o consumo de combustível, emissão de Certificados, Controle de Contas a Pagar e Receber, Controle de Funcionários e a Lista de chamada.

38. **SALÃO DE BELEZA:** O FashionHair é um salão de beleza, cabelereiros e estética que deseja informatizar suas atividades. Para isso o sistema deverá controlar: cadastro de funcionários, controle das porcentagens dos funcionários, cadastro de clientes, marcação de horários (agenda), registro dos serviços, comissões recebidas pelos funcionários e pelo salão, fluxo de caixa, cadastro de produtos, além de emitir importantes relatórios como: ficha de controle de cliente, listagem por cliente e por funcionario, listagem de entrada e saida no estoque e lista de agendamentos por dia ou

semana.

39. **EMPRESA DE BUFFET:** Elabore um modelo para uma empresa de buffet esperando-se com isso que a empresa tenha um crescimento na produtividade e qualidade de seus serviços e produtos, através de um atendimento personalizado ao cliente, provocando indiscutivelmente o aumento da lucratividade. O sistema deverá contemplar as seguintes funcionalidades:
    - Cadastro de clientes(Nome, Endereço, Telefone, RG, CPF, E-mail, Local de Trabalho, Setor, Profissão, Cargo, Data de Nascimento);
    - Cadastro de Funcionários(Nome, Endereço, Telefone, RG, CPF, E-mail, Local de Trabalho, Setor, Profissão, Cargo, Data de Nascimento, entre outros dados);
    - Cadastro de Fornecedores(Nome, Endereço, Telefone, RG, CPF, E-mail, Local de Trabalho, Setor, Profissão, Cargo, Data de Nascimento, entre outros dados);
    - Cadastro de produtos: Armazena os dados dos Produtos (materiais) que o buffet deseja ter dados em estoque;
    - Cadastro de tipos de festa: nascimento, aniversário, casamento, formatura, bodas, reencontros familiares e confraternizações diversas;
    - Registro de vendas: vendas realizadas, orçamentos e contratos;
    - Controle dos atendimentos realizados aos clientes;
    - Relatórios como: contratos, orçamentos, recibos, etc.

40. **ESCOLA DE CURSOS:** Desenvolva um sistema para uma escola de cursos que permita os seguintes controles:
    - Cadastrar alunos, professores, cursos, séries, funcionários, turmas, horários, notas;
    - Controlar os pagamentos e mensalidades dos alunos;
    - Agendamento das atividades dos alunos;
    - Controlar a movimentação da ficha do aluno;
    - Emitir relatórios como: lista de alunos por turma, mensalidades a receber, aniversariantes, diário de classe do professor, etc.

41. **AGÊNCIA DE EMPREGOS E RECRUTAMENTO:** Recrutamento, seleção e fornecimento de mão-de-obra temporária ou efetiva é um serviço cuja demanda está em crescimento com a tendência da terceirização das funções da área pessoal nas empresas. E o agenciamento de empregos é um desses serviços que, em razão, da nova realidade empresarial, tende a se ampliar. Motivado por esta nova perspectiva de negócios desenvolva um modelo para uma agência de empregos e recrutamento que permita:
    - Cadastro de Candidatos;
    - Cadastro de Currículos para Candidatos;
    - Cadastro de Anunciantes;
    - Cadastro de Anúncios;

- Categorias de Vagas;
- Controle de estágios;
- Acompanhamento de entrevistas e testes realizados;
- Emissão de diversos relatórios.

42. **EMPRESA DE FORMATURAS:** A empresa FORME-SE BEM quer um sistema para informatizar suas atividades na área de gestão de formaturas. Com o sistema a empresa poderá proporcionar maior organização, aumentando o controle de contas a pagar e a receber e gerenciando todo o movimento do seu negócio. Com o sistema, será possível registrar o cadastro de clientes com os contratos e serviços/oferecidos e até os relatórios financeiros ligados a eles. Algumas funcionalidades do sistema:
    - Cadastro de clientes;
    - Contratos;
    - Serviços;
    - Fornecedores;
    - Movimento caixa e bancário;
    - Contas pagar e a receber;
    - Agendas de compromissos;
    - Categorias e kits de produtos.

43. **PADARIA:** Desenvolva o modelo de um sistema para uma padaria. O sistema deverá registrar a compra, baixar todos os estoques, registrar quem fez as vendas e o caixa, quanto foi fornecido para pagar, suportar o fornecimento de tickets (alimentação, restaurante, etc.), fornecer cupom para simples conferencia, e fornecer o vale-compras. Além disso o sistema ainda deverá ter como funcionalidades o controle de clientes, fornecedores, produtos, produção, produção programada, receitas, funcionários, promoções, vendas, compras, notas fiscais, caixa, contas a receber, contas a pagar, ponto de venda, consultas e estatísticas variadas, múltiplas formas de pagamento nas vendas (incluindo fiado), status de clientes.

44. **PAPELARIA:** Modele um sistema para uma papelaria prevendo os seguintes requisitos:
    - Clientes (com relatórios diversos, impressões da ficha, dados de compras);
    - Financeiros (conta a pagar, conta a receber controle de cheques, relatórios diversos sobre o financeiro e gráfico mensal);
    - Fornecedores (com tipo de saída, mercadorias ou manutenção);
    - Formas de pagamento (formulário que permite criar a sua própria forma de fechamento de venda. Ex: a vista, cartão, crediário, cheque, etc);
    - Produtos (formulário completo com diversos relatórios e filtros, controle sobre todo o movimento de estoque. Ex: entrada de mercadoria, estorno, alteração de preços ou quantidade, controle de margem de lucro, margem de compra e fotos em produtos, cadastro com leitor de código de barra);

- Vendedores (cadastro completo dos vendedores);
- Transportadora (cadastro completo);
- Empresa (cadastro de dados da sua empresa, não precisa nos enviar seus dados e pode ser modificada a hora que for preciso sem nosso auxilio);
- Tipo pagamento (podemos criar nossas próprias formas de recebimento de contas, tornando ágil e automática o parcelamento e gerar contas e carnês).

45. **AGÊNCIA DE TELEMENSAGENS:** Modele um sistema para uma agência de telemensagens observando os seguintes requisitos:
    - Cadastro de CEP da Região;
    - Cadastro de clientes e de mensagens;
    - Controlar o pedido do cliente, efetua todo o processamento, desde o envio da mensagem até a cobrança do serviço e o respectivo pagamento;
    - Diversos relatórios: telemensagens a enviar, telemensagens a enviadas, serviços a entregar, serviços entregues, serviços a executar, produtos a entregar e entregues, vendas por cliente, vendas por vendedor, comissão por vendedor, vendas por período, vendas a receber, estoque, aniversariantes do mês.

46. **CORRETORA DE SEGUROS:** Modele um sistema para gerenciamento de corretoras de seguros que tenha como foco principal o crescimento das vendas, redução dos custos e organização da empresa. Além do controle de apólices de todos os ramos de seguros e controle de qualquer tipo de comissão. A corretora poderá atuar em qualquer especialidade como: vida, saúde, automóvel, incêndio, previdência privada, DPVAT, garantia, imobiliário, frotas. Alguns requisitos que o sistema modelado deverá contemplar:
    - Cadastro de Clientes (prospecções, segurados e ex-segurados);
    - Controle de relacionamento com clientes campanhas, eventos e ações (CRM);
    - Montagem de questionários de pesquisa de satisfação de clientes;
    - Estatísticas de controle de resposta dos questionários de pesquisas;
    - Mala-direta (cartas e etiquetas) para clientes;
    - Controle de acompanhamento de vendas (orçamentos);
    - Controle de propostas enviadas e pendentes de emissão;
    - Controle de Apólices, Endossos e Faturas mensais emitidas;
    - Controle dos dados dos itens e objetivos segurados;
    - Dados de seguros - Auto, Patrimoniais, Vida e Saúde;
    - Controle de questionário de perfil ou declaração de saúde;
    - Cadastramento e impressão de texto de cláusulas e condições gerais;
    - Controle de movimentação de segurados de vida e saúde;
    - Controle de apólices a renovar, renovadas e não renovadas;
    - Controle de prêmios e comissões pendentes;
    - Controle de administração de sinistros;

- Controle de recebimento e remesse de documentos para regulação de sinistro;
- Cadastramento de prestadores de serviços de atendimento de sinistro;
- Controle de redes Credenciadas de seguros de saúde.

47. **AEROCLUBE:** Num aeroclube, estão inscritos pilotos, instrutores e alunos de pilotagem. Todos sócios (inscritos) são identificados pelo número de matrícula, e caracterizados por nome, endereço e idade. Os pilotos possuem um número de brevê (único). Os instrutores são pilotos com formação adicional de instrutor, e deve ser registrado o nome do curso, a data de obtenção do diploma, bem como a instituição. Para os alunos de pilotagem, guarda-se o registros de todas suas saídas para contabilização de horas para obtenção do brevê. Para cada saída registra-se a data, instrutor, hora de saída de de chegada, bem como o parecer do instrutor sobre o voo. A escola só ministra cursos básicos, e portanto não há professores que são alunos de cursos avançados. Para emissão do brevê, é necessário que o aluno comprove ter o número de horas mínimo de voo, bem como apresente os pareceres dos instrutores sobre as habilidades desenvolvidas a cada aula prática.

48. **COMPANHIA DE TRANSPORTE:** Uma companhia de transporte é responsável por reservas de uma cadeia de varejo e entrega de remessas de armazéns para depósitos da empresas. Armazéns e depósitos são identificados por números e atualmente existem 6 localizações de armazéns e 45 de depósitos.
    - Um caminhão pode carregar várias remessas durante uma viagem e levar remessas para múltiplos depósitos (sai de um armazém origem e tem vários depósitos destino);
    - Uma viagem é identificada por um número. Será necessário manter informações sobre peso e volume da viagem;
    - Cada remessa é identificada pelo número da remessa e inclui dado sobre volume, peso e destino da remessa;
    - O caminhão é identificado pelo código da licença e tem diferentes capacidades para volume e peso que eles podem carregar.

    A companhia de caminhões atualmente tem 150 caminhões e um caminhão faz de 3 a 4 viagens por semana.

49. **COMPANHIA AÉREA:** Considere a descrição de um sistema de venda de passagens aéreas dada abaixo e desenhe um diagrama ER.
    - Para um passageiro são registrados o número do documento de identidade e o seu nome;
    - Um avião é caracterizado por um número de série único e por um modelo;
    - Um assento é uma posição única em um avião, identificada por um código. Um assento é da classe econômica ou executiva;
    - Um avião possui vários assentos e pode ser usado em vários voos;
    - Um voo é identificado por um número e utiliza: um avião, um piloto, um aeroporto de partida e outro de chegada;
    - Um passageiro pode reservar um assento em um voo.

50. **FARMÁCIA:** As farmácias “A BARATEIRA” compõem uma rede de atendimento ao cliente que prima pela qualidade e bom serviço. A rede disponibiliza farmacêuticos 24hs e também atendimento por telefone, com tele-entrega de medicamentos para toda a cidade. A rede “A BARATEIRA” de farmácias tem um sistema que automatiza de maneira precária suas tarefas diárias. Foram feitas consultorias com especialistas do mercado que sugeriram a implantação de um sistema informatizado que venha atender eficazmente toda rede. A seguir são descritos problemas e deficiências que devem ser resolvidos pelo novo sistema:
    1. Melhorar a forma com que os produtos são registrados. Com o advento dos genéricos, a base de dados deve primar pelo controle do princípio ativo do remédio e depois as demais características. O registro do nome fantasia do remédio também é válido. Há clientes que preferem a marca e não o remédio genérico;
    2. Essa melhora no registro das drogas também visa influenciar o controle do estoque. O controle deixará de ser por remédio e passará a ser por princípio ativo. Quando um tipo de princípio ativo (nome genérico) atingir o estoque mínimo deve ser disparado o procedimento de compras. Na chegada de uma Nota Fiscal (NF) de compra os produtos adquiridos são somados a quantidade em estoque. Se tiver um novo item este deve ser cadastrado antes de ser contabilizado no estoque;
    3. É importante a cotação em pelo menos três indústrias farmacêuticas que produzem a mesma droga antes de adquiri-la;
    4. Antes da compra as cotações devem ser avaliadas. A compra é feita com a indústria farmacêutica que tiver menor preço;
    5. “A BARATEIRA” também necessita de um controle mais rígido dos medicamentos que entram em promoção. Hoje não há este controle, fazendo com que o mesmo medicamento entre em promoção várias vezes seguidas, gerando prejuízos a empresa. Um medicamento não pode entrar em promoção se houver um outro medicamento com o mesmo princípio ativo já em promoção. Ainda, o valor de venda da promoção não pode ser menor que o valor de compra do remédio acrescido de 10%;
    6. Outro ponto que precisa ser melhorado está relacionado a gerência das vendas. Para o controle ideal das vendas o futuro sistema deverá disponibilizar os seguintes itens:
        a) Registro da venda e retirada da nota fiscal para o cliente;
        b) Cálculo da comissão do balconista que atendeu o cliente. Cada balconista recebe da farmácia um salário mais comissão. Se for venda por telefone ganha a comissão quem atendeu o telefone;
        c) Troca de produtos. “A BARATEIRA” não permite a devolução de dinheiro. O cliente terá a opção de pegar um produto no mesmo valor ou levar para casa um vale no valor da devolução;
        d) Visualização do estoque de cada filial da rede, permitindo que a partir de um princípio ativo seja visualizado todos os remédios genéricos correspondentes;
        e) Controle do histórico de vendas de cada vendedor;

f) O estoque de remédios controlados (tarja preta) só podem ser atualizado pelos farmacêuticos. Caso outro funcionário tente acessar esse estoque, é necessário guardar essa informação para posterior consulta pelo administrador do sistema;
g) Os clientes da farmácia devem poder consultar o preço dos remédios diretamente via os terminais disponíveis em algumas filiais da farmácia.

10.LISTA DE EXERCÍCIOS – TEORIA RELACIONAL

1. Explique o que significa ser um “Administrador de Banco de Dados” e quais são as responsabilidades inerentes ao cargo.
2. Apresente a descrição das siglas SGBD e DBMS. Conceitue o termo SGBD.
3. Relacione as principais características de um SGBD.
4. O que significa construir um **“modelo conceitual”** no desenvolvimento de um sistema de informação?
5. O que são metadados?
6. Cite alguns problemas típicos apresentados por ambientes que utilizavam arquivos para o armazenamento das informações.
7. Cite pelo menos três SGBDs de “mercado”(utilizados comercialmente) e pesquise algumas informações relacionadas aos mesmos para que seja possível caracterizar suas principais funcionalidades.
8. Explique o que significa redundância de dados.
9. Cite e associe os principais componentes de um SGBD.
10. Explique a(s) diferença(s) entre atributos, domínio de um atributo e valor de atributo?
11. Apresente os três modelos lógicos de SGBD baseados em registros.
12. Explique o **“Gerenciamento de Transações”** disponibilizado pelos SGBDs.
13. Relacione, explique e associe as etapas necessárias para a concepção de um sistema de informação.
14. Explique quais métodos ou técnicas para levantamento de requisitos podem ser utilizadas e como as mesmas podem ser aplicadas junto aos usuários.
15. Relacione e explique os conceitos pertinentes a modelagem EA – Entidade Associação.
16. O que significa um atributo ser **“Identificador”** ou **“Chave Candidata”**?
17. Cite alguns exemplos de atributos descritores que podem ser encontrados em entidades.
18. O que significa definir a **“Cardinalidade”** em um modelo conceitual?
19. Apresente as notações diagramáticas possíveis para a definição da **“Cardinalidade”** em um modelo conceitual.
20. Relacione os conceitos com suas respectivas definições:

(a) Codpessoa	( ) Técnica de levantamento de requisitos.
(b) Cardinalidade	( ) Modelo lógico de dados baseado em registros.
(c) Pessoa	( ) Ferramenta Case de Modelagem.
(d) Valores inteiros positivos múltiplos de 5	( ) Pode ser considerado um atributo identificador ou chave candidata.
(e) Documentação	( ) Representa uma Entidade no modelo Entidade Associação.
(f) System Architect	( ) Domínio de um atributo.
(g) Rede	( ) Número de ocorrências de uma entidade em relação a outra.

21. Defina os termos cardinalidade máxima e cardinalidade mínima.
22. O que é SQL? Qual sua vantagem?
23. Explique os conceitos do Modelo Relacional apresentados a seguir:
    a) Chave primária
    b) NULL
    c) Domínio
    d) Chave Estrangeira
24. Relacione e explique pelo menos dois tipos de integridade impostas e características do Modelo Relacional.
25. Pesquise um banco de dados relacional na internet e explique como o mesmo permite escrever procedimentos armazenados.
26. Em relação ao conceito de Chave Primária, assinale a afirmação correta.
    a) ( ) Pode conter atributos com valor nulo.
    b) ( ) É formada por, no máximo, um único atributo.
    c) ( ) É formada por, no mínimo, dois atributos.
    d) ( ) Identifica unicamente uma tupla.
    e) ( ) Identifica duas ou mais tuplas.
27. O Diagrama Entidade-Relacionamento, proposto por P. Chen, é uma ferramenta tipicamente utilizada para a elaboração do seguinte modelo de dados:
    a) ( ) físico
    b) ( ) interno
    c) ( ) externo
    d) ( ) conceitual
    e) ( ) hierárquico
28. *Atomicidade* é uma propriedade de transação de um SGBD relacional que garante que:
    a) ( ) uma transação seja realizada de forma independente de outras transações;
    b) ( ) uma operação de uma transação seja efetuada de forma independente de outras operações;
    c) ( ) nenhuma operação de uma transação seja subdividida em tarefas menores pelo SGBD;
    d) ( ) todos os atributos manipulados por uma transação sejam atômicos;
    e) ( ) todas as operações em um banco de dados, em uma transação, sejam executadas ou nenhuma delas o seja.
29. Os comandos SQL podem ser agrupados em categorias e ou componentes, de acordo com seu objetivo e funcionalidade. As principais categorias são:

    DML – Linguagem de Manipulação de Dados
    DDL – Linguagem de Definição de Dados
    DCL – Linguagem de Controle de Dados

    Observe os seguintes comandos SQL:
    CREATE TABLE
    DELETE
    SELECT
    INSERT
    ALTER TABLE

UPDATE

As categorias às quais pertencem os comandos SQL anteriores são, respectivamente:
a) ( ) DDL, DDL, DML, DML, DML, DDL
b) ( ) DDL, DML, DML, DML, DDL, DML
c) ( ) DML, DCL, DCL, DML, DDL, DML
d) ( ) DML, DCL, DML, DDL, DML, DML
e) ( ) DML, DDL, DCL, DML, DCL, DCL

30. As linguagens usadas para definir e manipular bancos de dados, respectivamente, são:

a) ( ) CDL e CML
b) ( ) DDL e DML
c) ( ) CDL e DML
d) ( ) CML e SGML
e) ( ) DDL e SGML

11.LISTA DE EXERCÍCIOS – ÁLGEBRA RELACIONAL

Considerando as tabelas: USUARIO, LIVRO, EMPRESTIMO e RESERVA.

Apresentar somente a expressão em álgebra.

31. Código e título dos livros da biblioteca 3?
32. Título de todos os livros?
33. Todos os empréstimos do dia 13/01/2008?
34. Título dos livros pertencentes a biblioteca 1 ou biblioteca 2?
35. O nome de todos os usuários que já emprestaram livros no dia 13/01/2008?
36. Nome dos usuários com código superior a 4?
37. Endereço do usuário 1?
38. Todos os empréstimos do usuário 6?
39. Todas as reservas do usuário 2?
40. Todas as reservas e empréstimos do usuário 4?
41. O título de todos os livros que o usuário 1 emprestou?
42. O título de todos os livros que o usuário 2 reservou?
43. O nome de todos os usuários que emprestaram o livro 1002?
44. O nome de todos os usuários que reservaram o livro 1002?
45. Todos os empréstimos da biblioteca 1?
46. Todos os usuários que realizaram reservas no mês 2 de 2006?
47. Todos os usuários que realizaram empréstimos no mês 1 de 2006?
48. Todos os usuários que já emprestaram livros com o título 'Livro 01' ou 'Livro 02'?
49. Todos os livros que já sofreram empréstimo?
50. Todos os usuários que já realizaram empréstimo?

Dado o esquema relacional a seguir, especificar as operações em álgebra relacional para obter os resultados das interrogações apresentadas:

FORNECEDOR(<u>CODFOR</u>, NOMFOR, CODCON, NOMCID)
PRODUTO(<u>CODPRO</u>, NOMPRO, DESCOR)
PEDIDO(<u>CODFOR, CODPRO</u>, QTDPED)

51. Denominação dos fornecedores que fornecem o produto 'P2'?
52. Denominação dos fornecedores que fornecem pelo menos um produto cuja cor é 'VERMELHO'?
53. Denominação dos produtos da cor 'AZUL' que tiveram pedidos com quantidade superior a 200?
54. Denominação dos fornecedores que fornecem todos os produtos?

Dadas as relações:

S

A	B	C	D
3	g	e	3
5	a	j	7
1	d	n	21
7	d	e	2
9	e	b	34
2	e	c	7
1	a	j	2
1	f	e	34
2	b	a	22

R

A	B	C	D
3	a	d	3
4	b	f	7
1	b	j	21
2	f	c	2
3	j	d	34
2	a	c	7
1	a	j	2
1	f	e	34
2	b	a	22

Resolva(demonstre):

55. $\pi_{B,C}(R)$
56. $\pi_{A}(\sigma_{B = 'e'}(S))$
57. $\pi_{A,D}\{[\sigma_{C = 'c'}(R)] \bowtie_{R.A = S.A} [\sigma_{B = 'e'}(S)]\}$
58. $\pi_{A}\{[\pi_{A,B}(\sigma_{B = 'b'}(S))] \cup [\pi_{A,B}(\sigma_{B = 'b'}(R))]\}$

Dado o esquema relacional a seguir, especificar as operações em álgebra relacional para obter os resultados das interrogações apresentadas:

NATURALIDADE(<u>CODNAT</u>, DESNAT)
ALUNO(<u>CODALU</u>, NOMALU, ENDALU, <u>CODNAT</u>)
DISCIPLINA(<u>CODDIS</u>, NOMDIS, CHADIS)
HISTORICO(<u>CODALU, CODDIS,</u> VLRNOT)

59. Quais alunos(NOME) tiveram nota superior a 7 nas disciplinas de 'MATEMÁTICA' e 'FÍSICA'?
60. Quais alunos(NOME) cursaram todas as disciplinas com carga horária maior que 60 horas, com nota superior a 6?
61. Quais disciplinas(NOME), todos os alunos que já a cursaram obtiveram nota superior a 5?
62. Quais alunos(NOME) naturais de 'JOINVILLE' obtiveram nota superior a 8 em disciplinas com carga horária superior a 45 horas?

Dado o esquema relacional a seguir, especificar as operações em álgebra relacional para obter os resultados das interrogações apresentadas:

FORNECEDOR(CODFOR, NOMFOR, NOMCID)
PECA(CODPEC, NOMPEC, PESO, COR)
EMBARQUE(CODFOR, CODPEC, QTDEMB)

63. Buscar os dados dos fornecedores de Porto Alegre.
64. Buscar os dados das peças que pesam mais do que 5 gramas E não são pretas.
65. Buscar o nome das peças que não são pretas.
66. Buscar o nome das peças fornecidas pela Ceval S.A.
67. Buscar o nome de todas as peças leves (pesam menos que 10 gramas).
68. Buscar as quantidades das peças fornecidas pelo fornecedor de nome Ceval S.A.
69. Buscar o nome dos fornecedores de Canoas que fornecem pregos.
70. Buscar o nome dos fornecedores que fornecem todas as peças.
71. Buscar o nome das peças que são fornecidas por fornecedores de Porto Alegre em quantidade superior a 300.
72. Buscar o nome e a cidade dos fornecedores que realizam embarques em quantidades maiores que 500.
73. Buscar o nome dos fornecedores que não forneçam nenhuma peça.

Dado o esquema relacional a seguir, especificar as operações em álgebra relacional para obter os resultados das interrogações apresentadas:

DEPARTAMENTO(CODDEP, NOMDEP, CODGER)
EMPREGADO(CODEMP, NOMEMP, SALEMP, ENDEMP, DATNAS, CODDEP, CODSUP)
PROJETO(CODPRJ, NOMPRJ, CODDEP, LOCAL)
TRABALHA(CODEMP, CODPRJ, NUMHOR)
DEPENDENTE(NOMDEP, SEXDEP, DATNAS, PARDEP, CODEMP)
DEPARTAMENTO_LOCAL(CODDEP, LOCAL)

74. Quais os nomes dos gerentes de cada departamento.
75. Listar nome e endereço dos empregados que trabalham para o projeto ‘Banco de Dados’.
76. Para os projetos localizados em Belo Horizonte, listar seus números, o número do departamento que os controlam, e o nome, endereço e data de nascimento de seu gerente.
77. Quais identidades e nomes dos empregados que não têm dependentes?
78. Quais os nomes dos empregados e os números de departamento dos quais eles são gerentes, se o forem?
79. Listar os nomes dos empregados que trabalham em todos os projetos que o Manuel trabalha.
80. Quais empregados estão trabalhando menos de 40 horas por semana em projetos da empresa?

Dado o esquema relacional a seguir, especificar as operações em álgebra relacional para obter os resultados das interrogações apresentadas:

AUTOMOVEL(<u>CODAUT</u>, NOMFAB, DESMOD, ANOAUT, PAIS, PRECO)
REVENDORA(<u>CGCREV</u>, NOMREV, NOMPRO, NOMCID, UF)
CONSUMIDOR(<u>NUMIDE</u>, NOMCON)
NEGOCIO(<u>NUMIDE</u>, <u>CGCREV</u>, <u>CODAUT</u>, DATNEG, PRECO)
GARAGEM(<u>CGCREV</u>, <u>CODAUT</u>, QTDAUT)

81. Listar os nomes dos fabricantes dos automóveis na base de dados e os respectivos países de fabricação originalmente.
82. Listar os estados onde se vende o modelo Xantia, do fabricante Citröen.
83. Qual(is) o(s) carro(s) – codigo e ano - de preço de tabela mais caro?
84. Listar o nome das revendedoras e de seus respectivos proprietários que venderam em 2002 carros franceses ou italianos, fabricados em 2001, por valor até 30% acima do preço de tabela.
85. Quais os nomes das revendedoras que vendem todos os modelos do fabricante Peugeot?
86. Quais revendedoras não vendem automóveis de origem francesa?
87. Quais os nomes dos proprietários de revendedoras que já compraram algum carro nas revendedoras de outros proprietários?
88. Quais os consumidores (identidade e nome) que compram apenas carros italianos?

12.LISTA DE EXERCÍCIOS - SQL

Considerando as tabelas: EMPREGADO e DEPARTAMENTO.

89. Selecione todos os registros da tabela EMPREGADO com todas as colunas.
90. Selecione somente os nomes diferentes da tabela EMPREGADO.
91. Quem trabalha no departamento 20?
92. Onde está localizado o departamento 20 (utilize a tabela DEPARTAMENTO)?
93. Selecione todos os empregados com nome e data de admissão da tabela EMPREGADO e troque o nome da coluna data de admissão para admissão.
94. Quais empregados ganham mais de 2000?
95. Selecione o Nome, Salário e Nome do Departamento de cada empregado (utilize a tabela EMPREGADO E DEPARTAMENTO).
96. Quais empregados tem cargo diferente de ATENDENTE e VENDEDOR?
97. Selecione todos os empregados (Código e Nome) em ordem decrescente de código de empregado.
98. Existe algum empregado que tem o salário menor que a comissão?
99. Selecione o nome, salário e comissão da tabela EMPREGADO, selecionando apenas os empregados que tenham comissão.
100. Selecione o nome, salário, comissão e salário + comissão da tabela EMPREGADO.
101. Selecione o nome, salário, comissão e salário + comissão da tabela EMPREGADO selecionando apenas os empregados que tenham comissão.
102. Selecione o nome, salário, comissão e salário + comissão da tabela EMPREGADO selecionando apenas os empregados que tenham comissão e que a soma da comissão com o salário seja maior que 1500.
103. Selecione o nome, salário mensal e salário hora (salário/220) da tabela EMPREGADO.
104. Agora trunque o valor da divisão acima para quatro casas decimais.
105. Agora arredonde o valor da divisão da questão 15 para 4 casas decimais.
106. Selecione o maior salário da tabela EMPREGADO.
107. Conte quantos empregados estão lotados no departamento VENDAS.
108. Qual é a media salarial dos empregados?
109. Quais os empregados ganham entre 500 e 1500?
110. Quanto gasta a empresa em salários e comissão para todos os empregados?
111. Quanto gasta a empresa em salários por departamento?

Considerando as tabelas: USUARIO, LIVRO, EMPRESTIMO e RESERVA.

112. Selecione todos os registros da tabela EMPRESTIMO, com todas as colunas.
113. Selecione o código e nome, quantidade de empréstimos do usuário 2.
114. Selecione a data(exiba a coluna com o nome “Data de Empréstimo”) do último empréstimo realizado.
115. Selecione o código e quantidade de empréstimos de cada biblioteca.

116. Selecione o código e quantidade de reservas de cada biblioteca.
117. Conte a quantidade de empréstimos de 2008.
118. Selecione o código, nome e quantidade de reservas de todos os usuários.
119. Selecione somente os usuários com duas ou mais reservas.
120. Qual o livro com o maior número de empréstimos?
121. Qual o usuário com o maior número de reservas?
122. Quais livros da biblioteca 1 já foram emprestados pelo usuário 3?
123. Em que cidade se encontra o livro 3?
124. Selecione o código e nome do usuário que possui empréstimos em 2006 ou reservas em 2006, na biblioteca 1.
125. Selecione o código e nome do usuário que possui empréstimos em 2006 e não possui reservas em 2006, na biblioteca 1.
126. Selecione os dados de empréstimo de cada usuário. Liste todas as colunas da tabela empréstimo e todas as colunas da tabela usuário. Coloque em ordem de nome de usuário e data de empréstimo.
127. Conte o número de empréstimos diários e calcule a média diária de empréstimos realizados na biblioteca 2.

Considerando o esquema relacional a seguir, resolva com SQL:

FORNECEDOR(<u>CODFOR</u>, NOMFOR, NOMCID)
PECA(<u>CODPEC</u>, NOMPEC, PESO, COR)
EMBARQUE(<u>CODFOR</u>, <u>CODPEC</u>, QTDEMB)

128. Conte o número de empréstimos diários e calcule a média diária de empréstimos realizados na biblioteca 2.
129. Buscar os dados dos fornecedores de Porto Alegre.
130. Buscar os dados das peças que pesam mais do que 5 gramas E não são pretas.
131. Buscar o nome das peças que não são pretas.
132. Buscar o nome das peças fornecidas pela Ceval S.A.
133. Buscar o nome de todas as peças leves (pesam menos que 10 gramas).
134. Buscar as quantidades das peças fornecidas pelo fornecedor de nome Ceval S.A.
135. Buscar o nome dos fornecedores de Canoas que fornecem pregos.
136. Buscar o nome dos fornecedores que fornecem todas as peças.
137. Buscar o nome das peças que são fornecidas por fornecedores de Porto Alegre em quantidade superior a 300.
138. Buscar o nome e a cidade dos fornecedores que realizam embarques em quantidades maiores que 500.
139. Buscar o nome dos fornecedores que não forneçam nenhuma peça.

Considerando o esquema relacional a seguir, resolva com SQL:

ARTIGO(<u>AUTOR</u>, <u>TITULO</u>, UNIORI, ORGFIN, NOTA)
EVENTO(<u>EVENTO</u>, LOCAL, MES, NUMPAR, UNIORG)
ACEITO(<u>EVENTO</u>, <u>AUTOR</u>, <u>TITULO</u>)

140. Selecionar o título e o autor de todos os trabalhos da UNOESC aceitos para apresentação em algum evento.
141. Selecionar, por ordem decrescente, a média das notas dos trabalhos de cada universidade e o nome da universidade.
142. Selecionar o título e o autor dos trabalhos que foram aceitos para eventos organizados pela própria universidade do autor do artigo.
143. Selecionar a programação de trabalhos que serão apresentados no mês de Outubro, ordenado por título de trabalho.
144. Selecionar o nome dos autores com mais de um trabalho aceito para apresentação.
145. Selecionar o nome das universidades que tenham artigos com notas maiores do que todas as notas obtidas por artigos da UNOESC.
146. Selecionar o nome dos autores e o número de trabalhos de sua autoria que foram aceitos para apresentação em qualquer evento.
147. Selecionar os nomes dos eventos, os títulos dos trabalhos aceitos para estes eventos, os órgãos financiadores dos trabalhos e os locais onde os eventos ocorrerão.
148. Selecionar os eventos onde foram aceitos trabalhos da UNOESC, mas não foram Aceitos trabalhos da UFRGS.
149. Selecionar o número total de participantes nos eventos realizados em Curitiba.
150. Selecionar o número de Artigos que contém os termos ‘banco de dados’ em seu título.
151. Selecionar os autores que tiveram trabalhos aceitos em TODOS os eventos cadastrados, ou seja, se existem três eventos cadastrados, os autores selecionados têm que ter tido trabalhos aceitos nos três eventos.
152. Selecionar os eventos que aceitaram artigos de TODAS as Universidades, ou seja, se existem dez Universidades que publicaram artigos, os eventos selecionados têm que ter artigos dessas dez Universidades.

Considerando as tabelas: EMPREGADO e DEPARTAMENTO.

153. Inclua na tabela Empregado o seguinte registro:
    a. Numero: 1234
    b. Nome: Zé Carioca
    c. Cargo: aprendiz
    d. Data de Inicio: Hoje
    e. Salário: 200
    f. Departamento: 30
154. Atualize o cargo do Zé Carioca para Programador.
155. Atualize em todos os registros a data de inicio para ‘15/10/1999’.
156. Exclua os dados do Zé Carioca da Tabela.
157. Atualize todos os salários com base no cálculo: salário atual * 1.4 + 200.

158. Limpe o campo comissão para todos os empregados.
159. Atualize o campo comissão de todos os empregados com base na fórmula: salário atual / 2 + 100.
160. Atualize o valor da comissão de todos os empregados com base na fórmula: comissão = 45.6% do salário atual.
161. Para todos os códigos de empregados pares atualize o salário para duas vezes este mesmo valor.
162. Para os casos onde a comissão do empregado for menor que 40% do valor do salário, mudar seu cargo para 'VENDEDOR'.

Considerando as tabelas: USUARIO, LIVRO, EMPRESTIMO E RESERVA.

163. Exclua da tabela de empréstimo os empréstimos cuja data de empréstimo é igual a menor data de empréstimo de 2006.
164. Exclua da tabela de reserva as reservas cuja data de reserva é igual a menor data de reserva de 2005.
165. Exclua os dados do usuário 3.
166. Inclua um novo usuário na tabela de usuário chamado "teste"com o código 4.
167. Inclua um empréstimo na tabela de empréstimos para o usuário 4.
168. Passe todos os empréstimos de 2005 para o usuário 4.
169. Somente os livros da biblioteca 2 devem sofrer uma modificação em seu título, os mesmos devem ter o título no formato = título atual || codigo da biblioteca a qual pertence.
170. Inclua uma linha de reserva para o usuário 4 para o livro 3.

Considerando as tabelas: EMPREGADO, DEPARTAMENTO, USUARIO, LIVRO, EMPRESTIMO E RESERVA.

171. Relacione somente o nome de todos os usuários em ordem decrescente.
172. Exiba a lista de todos os empréstimos pares.
173. Exiba a lista de todas as reservas múltiplas de 2.
174. Quais são os usuários cujo nome começa com a letra 'A'?
175. Exiba somente os primeiros 5 caracteres do título de cada um dos livros.
176. Relacione todos os empregados cujo salário supera 1630.45.
177. Em que data a empresa contratou o primeiro empregado?
178. Exiba o código e nome de todos os empregados gerentes.
179. Quais são os livros que estão na lista (1,5,7)?
180. Quais são os livros cujo tamanho do título supera 20 caracteres?
181. Quais são os empréstimos realizados anteriores a 2006?
182. Relacione todos os departamentos cujo departamento é múltiplo de 3.
183. Mostre quais são os usuários que estão na lista (2,3,4).
184. Qual o endereço de usuário cadastrado mais longo?
185. Qual o nome de empregado mais longo?
186. Qual o nome de empregado mais curto?
187. Relacione o código, nome e a cidade do departamento dos empregados admitidos em 1984.

188. Relacione o nome e o salário de todos os empregados gerentes que não recebem comissão.
189. Qual gerente recebe mais de 2000 de comissão?
190. Qual biblioteca possui o livro com o título ‘MEU JACARÉ’?
191. Exiba a lista de livros com a quantidade de empréstimos. Ordene a lista do livro com mais empréstimos até o livro com menos empréstimos. Exiba a lista de usuários com a quantidade de empréstimos. Ordene a lista do usuário com mais empréstimos até o usuário com menos empréstimos.
192. Mostre o nome e o salário de cada empregado com o salário reajustado em 23.45%.
193. Exiba o nome dos usuários que não possuem empréstimos.
194. Qual o maior cep de usuário cadastrado?
195. Qual o menor cep de usuário cadastrado?
196. Qual o resultado da soma de todos os salários dos empregados dividido por 20?
197. Qual a hora atual do sistema?
198. Qual a data atual do sistema?
199. Qual a data/hora atual do sistema?
200. Qual o valor na tabela ASCII do caractere ‘W’?
201. Exiba o nome de todos os títulos dos livros em letras minúsculas.
202. Liste todos os livros cujo título termina com a letra ‘O’.
203. Conte o número de usuários pares.
204. Conte o numero de livros que falam sobre ‘BANCO DE DADOS’.
205. Conte o número de livros da biblioteca 1.
206. Exiba o nome de todos os empregados do departamento ‘CONTÁBIL’ e do departamento ‘OPERACIONAL’.
207. Qual a data da última reserva realizada em 2005?
208. Selecione o menor salário da tabela empregado.
209. Exiba o maior e o menor salário da tabela empregado.
210. Exiba somente o nome e o cep dos usuários.
211. Qual a média dos ceps dos usuários.
212. Exiba somente os usuários com cep maior que 60000000.
213. Exiba o título do livro, o usuário e a data de empréstimo de cada livro. Ordene a lista final por livro, data de empréstimo.
214. Exiba o título do livro, o usuário e a data de empréstimo de cada livro. Ordene a lista final por livro, data de empréstimo.
215. Exiba o título do livro, o usuário e a data de reserva de cada livro. Ordene a lista final por livro, data de reserva.
216. Exclua os empréstimos do usuário 1.
217. Exclua as reservas do usuário 2.
218. Atualize a data de empréstimo para ’12/01/2007’ de todos os empréstimos do usuário 3.
219. Todos os empregados atendentes devem ser promovidos a vendedores.
220. Todos os gerentes devem receber uma comissão equivalente a 2/3 do seu salário.

221. Todos os empregados devem passar a ter o sobrenome ‘DA SILVA’.
222. Exclua o empregado 1201.
223. Exclua o departamento 40.
224. Exclua o usuário 1.
225. Exclua todos os livros que possuem no título a palavra ‘PROGRAMAÇÃO’.
226. Insira um novo livro com o título ‘XML’.
227. Insira um novo empregado com todos os dados iguais aos do empregado 4356.
228. Exclua o usuário 4356.
229. Remova o salário de todos os empregados.
230. Atualize os salários dos empregados com base no ano de admissão: antes de 80 – 3000, entre 80 e 90 – 2500, entre 91 e 2000 – 2000, depois de 2000 – 1000.
231. Selecione todos os livros da biblioteca 2 cujo código do livro seja maior que 7.
232. Selecione todos os livros cujo título do livro possui os caracteres ‘BANCO’.
233. Selecione todos os usuários cujos códigos estejam na lista (2,5,6,7,8).
234. Selecione todos os usuários que já emprestaram livros da biblioteca 1.
235. Selecione todos os empregados do departamento CONTÁBIL ou do departamento de PESQUISA.
236. Selecione os departamentos e conte os empregados por departamento. Conte somente os empregados que possuem comissão.
237. Selecione todos os livros que pertencem a biblioteca 2.
238. Selecione todos os usuários que realizaram empréstimos no mês de ABRIL.
239. Selecione todos os usuários que realizaram reservas entre o dia 05 e 14 do mês.
240. Selecione todos os usuários que possuem empréstimos e reservas.
241. Selecione os usuários que não possuem nenhum empréstimo no mês de JUNHO.
242. Selecione o usuário e a quantidade de empréstimos que o mesmo possui por biblioteca.

Considerando o esquema relacional a seguir, resolva com SQL:

MEDICO(<u>CODMED</u>, NOMMED, DATNAS, ESPECIALIDADE)
PACIENTE(<u>CODPAC</u>, NOMPAC, DATBAS, PROBLEMA)
CONSULTA(CODMED, CODPAC, DATCON)

243. Crie as tabelas do esquema relacional.
244. O nome dos médicos com idade > 22 ou com especialidade diferente de traumatologia.
245. O nome e o problema dos pacientes com menos de 24 anos.
246. As consultas para o dia 26/09/96 após as 15h.
247. Buscar todas as especialidades dos médicos.
248. Buscar todas as datas de consultas com horário após as 10h.
249. Buscar todas as idades dos médicos.
250. Buscar o nome de todos os pacientes que começam com a letra C.
251. Buscar dados dos médicos que não têm especialidade.
252. Buscar o nome dos médicos com consultas marcadas para horários mais tarde que todas as consultas da médica Maria.
253. Buscar os dados de todas as consultas do paciente Carlos, ordenadas de forma decrescente pela

hora da consulta.

254. Buscar todas as datas de consultas e o total de consultas para esta data.
255. Buscar somente as datas e o total de consultas para horários após as 14 horas
256. Buscar somente as datas e o total de consultas para as datas onde haja mais de uma consulta marcada.
257. Buscar, para a tabela de médicos, todas as idades e o total de médicos com a mesma idade.
258. Buscar o nome dos médicos com mais de uma consulta marcada.
259. Mostrar todos os dados da tabela de consultas.
260. Mostrar os dados dos médicos classificados por ordem de código.
261. Obter os nomes e códigos de todos os médicos cirurgiões.
262. Fornecer os nomes e códigos de todos os médicos e seus respectivos dias de consulta.
263. Fornecer os nomes dos médicos que possuem alguma consulta marcada com o paciente 4.
264. Mostrar os nomes dos médicos que não têm consulta marcada com a paciente 4.
265. Mostrar os nomes dos médicos que não tem consulta marcada com a paciente Maria.
266. Mostrar o código dos pacientes que tem consulta marcada para o dia 23/09/96 ou com médicos pediatras.
267. Mostrar os nomes dos médicos que não fornecem consultas para os pacientes de obstetras.

Considerando o esquema relacional a seguir, resolva com SQL:

EMPREGADO(CODEMP, SALEMP, FUNEMP, CODDEP)
DEPARTAMENTO (CODDEP, NOMDEP, CIDDEP)
JOGA (CODEMP, CODTIM, POSICAO)
TIME(CODTIM, NOMTIM, ENDTIM)

268. Mostrar os códigos dos empregados que ganham acima da média dos salários.
269. Mostrar os departamentos que não tem empregados.
270. Mostrar os departamentos com média salarial inferior a 500.
271. Mostrar os departamentos que possuem mais de 10 programadores.
272. Mostrar o nome do time do empregado de maior salário.
273. Mostrar o maior salário dos empregados que jogam na ponta direita.
274. Mostrar a posição e o código dos empregados de maior salário em cada departamento.

13.REFERÊNCIAS

CHEN, Peter. **Modelagem de dados:** a abordagem Entidade-Relacionamento para Projeto Lógico. São Paulo: Makron Books, 1990. 80p.

CHEN, Peter. **The Entity-Relationship Model-Toward a Unified View of Data**. ACM Transactions on Database Systems, Vol. 1, No. 1. March 1976, Pages 9-36.

CODD, E.F. **Does Your DBMS Run by the Rules**. Computerworld, Out 21, 1985a.

CODD, E.F. **Is Your DBMS Really Relational?**. Computerworld, Out 14, 1985b.

CODD, E.F. **Relational Database**: A Practical Foundation for Produtivity. *CACM* 25, Nº 2, Fev.1982.

CONNOLLY, Thomas; BEGG, Carolyn. **Database Systems**: a practical approach to design, implementation, and management. 4 ed. England: Pearson Education Limited, 2005.

COUGO, Paulo. **Modelagem conceitual e projeto de bancos de dados.** 3. ed. Rio de Janeiro: Campus, 1997. 284 p. ISBN 8535201580

DATE, C. J.. **Introdução a sistemas de bancos de dados**. Tradução da 8.ed. americana, 2. tiragem. Rio de Janeiro: Campus, 2004. 674p. ISBN 8535212736

DOUGLAS, Korry; DOUGLAS, Susan. **PostgreSQL**: the comprehensive guide to building, programming, and administering PostgreSQL database. 2. ed. USA: Developer's Library, 2006. 1006 p. ISBN 0672327562.

ELMASRI, Ramez; NAVATHE, Shamkant B.. **Sistemas de banco de dados**. 4.ed. São Paulo: Addison Wesley Longman, 2006. 724 p. ISBN 8588639173

GANE, Chris. **Desenvolvimento rápido de sistemas.** Rio de Janeiro: LTC, 1988. [xi], 170 p. : (Ciência da computação) ISBN 8521606125

GARCIA-MOLINA, Hector; ULLMAN, Jeffrey D.; WIDOM, Jennifer. **Implementação de sistemas de bancos de dados**. Rio de Janeiro: Campus, 2001. 690p. ISBN 853520749X

GETTING Joins. 2007. Disponível em: <http://www.khankennels.com/blog/index.php/archives/2007/04/20/getting-joins>. Acesso em: 20 jan. 2008.

GILLENSON, Mark L.. **Fundamentos de sistemas de gerência de banco de dados**. Rio de Janeiro: LTC, 2006. 304p. ISBN 8521614977

GONZAGA, Jorge Luiz. **Dominando o PostgreSQL**. Rio de Janeiro: Ciência Moderna, 2007. 228 p. ISBN 9788573935592.

HARRINGTON, Jan L.. **Projeto de banco de dados relacionais**: teoria e prática. Rio de Janeiro: Campus, 2002. 360p. ISBN 8535211128

HEUSER, Carlos Alberto. **Projeto de banco de dados**. 5. ed. Porto Alegre: Sagra Luzzatto, 2004. 236 p. (Livros Didáticos ;v.4) ISBN 8524105909

KROENKE, David. **Banco de dados**: fundamentos, projeto e implementação. 6. ed. Rio de Janeiro: LTC, 1998. 382 p.

MACHADO, Felipe Nery Rodrigues. **Banco de dados**: projeto e implementação. São Paulo: Érica, 2004. 398p. ISBN 8536500190

MANZANO, José Augusto N. G. **PostgreSQL 8.3.0**: interativo : guia de orientação e desenvolvimento para windows. 1. ed. São Paulo: Érica, 2008. 240 p. : ISBN 9788536501987

MAUSS, Jason. **Database Naming Conventions Version 1.1**. 2004. Disponível em: <http://weblogs.asp.net/jamauss/pages/DatabaseNamingConventions.aspx>. Acesso em: 30 jul. 2009.

ORACLE Naming Conventions. 2009. Disponível em: <http://ss64.com/ora/syntax-naming.html>. Acesso em: 30 jul. 2009.

PLEW, Ronald R.; STEPHENS, Ryan K.. **Aprenda em 24 horas SQL, segunda edição**. 2. ed. Rio de Janeiro: Campus, 2000. 394 p. ISBN 8535206647

POMPILHO, S.. **Análise essencial:** guia prático de análise de sistemas. Rio de Janeiro: Ciência Moderna, 2002. 269p. ISBN 8573932023

RAMALHO, José Antonio Alves. **SQL**: a linguagem do banco de dados. São Paulo: Berkeley, 1999. 627p. ISBN 8572515011

SHASHA, Dennis. **Database tuning**: principles, experiments, and troubleshooting techniques. 5. ed. San Francisco: Elsevier, 2007. 414 p.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN, S.. **Sistema de banco de dados**. 3. ed., rev. São Paulo: Makron Books, 2007. 778 p. ISBN 8534610738

TOFFLER, Alvin; TOFFLER, Heidi. **Criando uma nova civilização**: a política da terceira onda. Rio de Janeiro: Record, 1994. 142 p. ISBN 850104458X

WATT, Justin. **Essential Database Naming Conventions (and Style).** 2004. Disponível em: <http://justinsomnia.org/writings/naming_conventions.html>. Acesso em: 20 jan. 2008.

YOURDON, Edward. **Análise estruturada moderna.** Rio de Janeiro: Campus, 1990. 836p. : (Yourdon Press) ISBN 8570016158

ANEXOS

ANEXO 01

CONVENÇÃO PARA NOMEAÇÃO DE OBJETOS DE UMA BASE DE DADOS

1. **Quanto ao Estilo**
   a) **Use caracteres minúsculos**
      i. Elimine problemas de erros com caracteres, principalmente em sistemas *case sensitive*;
      ii. aumenta velocidade na digitação e a corretude dos termos;
      iii. diferencie nomes de tabelas e colunas das palavras SQL(escritas em maiúsculo).
   b) **Separe palavras usando o caractere *underline*(_), nunca use espaços**
      i. promova a legibilidade(ex.: nome_pessoa vs. Nomepessoa);
      ii. evite o uso de caracteres especiais isolando nomes(ex.: [nome_pessoa]);
      iii. ofereça uma certa independência de plataforma.
   c) **Evite usar números**
2. **Nomes de tabelas**
   a) **Escolha nomes curtos, sem ambiguidade, usando não mais que uma ou duas palavras**
      i. Facilita a distinção entre as tabelas;
      ii. Facilita a nomeação de colunas(nomes únicos) utilizando abreviações do nome da tabela.
   b) **Atribua nomes no singular, nunca no plural**
      i. Promove consistência na nomeação de campos chave primária;
      ii. Facilita a organização alfabética das tabelas;
      iii. Melhora a gramática SQL(Ex.: SELECT pessoa.nome_pessoa).
   c) **Evite abreviações, concatenações ou acrônimos**
      i. Promove a auto-documentação;
      ii. Facilita a leitura e o entendimento tanto do leitor como do desenvolvedor.
   d) **Identifique tabelas relacionadas com o nome da tabela a qual esta se relaciona:**
      i. Facilita o agrupamento das tabelas(ex.: pessoa_endereco);
      ii. Previne a duplicidade de nomes entre tabelas relacionadas.
   e) **Para tabelas relacionadas a mais de uma tabela, utilize como nome a concatenação de todas as tabelas as quais esta se relaciona**
      i. Facilita o relacionamento das tabelas;
      ii. Expressa que a tabela é composta.

3. **Colunas/Atributos/Campos**
   a) **A chave primária pode ser nomeada concatenando-se o nome da tabela e a abreviação “pk” ou “id”**
      i. Facilita a dedução do nome da chave primária;
      ii. Torna consistentes o nome da chave primária em relação a chave estrangeira;
      iii. Previne a necessidade do uso de nomes próprios gerados pelo SGBD.
   b) **Coloque como prefixo para cada nome de coluna o nome da tabela(ex. pessoa_nome)**
      i. Evita que nomes de colunas sejam confundidos com palavras reservadas;
      ii. Facilita a definição de nomes únicos de coluna;
      iii. Torna os nomes de coluna consistentes para a nomeação de chaves primárias;
      iv. Diferencia chaves estrangeiras de colunas nativas da própria tabela;
      v. Mantêm a semântica dos nomes das colunas;
      vi. Previne que uma coluna receba o mesmo nome de uma tabela;
      vii. Esta opção pode ser ignorada caso existam mais de 30 tabelas no Banco de Dados, ou caso a tabela possua mais de 30 colunas ou os comandos SQL sejam sempre formados pelo conjunto Nome_Tabela.Nome_Coluna.
   c) **O nome da chave estrangeira deve ser idêntico ao nome da chave primária da tabela a qual se refere.**

Fonte: Adaptado de Watt(2004).

ANEXO 02

PADRÃO PARA DEFINIÇÃO DAS TABELAS DO MODELO RELACIONAL(MODELO E-R)

Nome da Tabela
Coluna: Tipo(*) (PK) ou (FK)

(*)Como tipos de dados válidos podem ser utilizados: Alfa(Varchar), Inteiro(Integer), Real(Decimal), Data(Date), Hora(Time), Lógico(Boolean) e Binário(Blob ou Bytea no Postgres).

Notação para os relacionamentos: utilizar notação Engenharia da Informação.