



Universidade Federal do Ceará
Centro de Ciências
Departamento de Computação

Sistemas de Informação e Banco de Dados

Angelo Brayner
brayner@dc.ufc.br



1. Tecnologia de Banco de Dados - Histórico -

- Até final da década de 60
 - Sistema de arquivos
- Final da década de 60
 - Nascimento da tecnologia de banco de dados
 - Gerenciamento eficiente de dados
 - Representação de dados através de árvores
 - Modelo hierárquico
 - Representação de dados através de grafos
 - Modelo Redes



1. Tecnologia de Banco de Dados - Histórico -

• Anos 70

• Modelo Relacional

- Representar dados em um nível conceitual mais elevado
- Relações matemáticas
- Tabelas

- Impulsionou pesquisas na área de banco de dados
- Processamento de consultas
- Monitoramento no acesso concorrente aos dados



1. Tecnologia de Banco de Dados - Histórico -

• Década de 80

• Distribuição

- Sistemas de Bancos de Dados Distribuídos
- Sistemas Bancos de Dados Cliente-Servidor
- Sistemas de Bancos de Dados Paralelos

• Integração

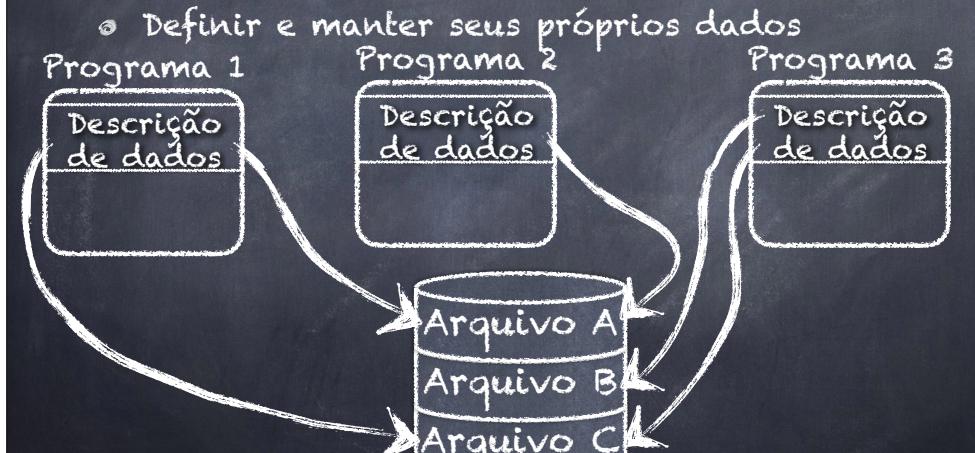
- Integrar Bancos de Dados
 - Distribuídos
 - Heterogêneos
 - Autônomos

1. Tecnologia de Banco de Dados - Histórico -

- Década de 90
 - Banco de Dados Objeto-Relacional
- (2000, ...)
 - Integração
 - Mobilidade
 - Fluxo de Dados
 - Sistemas de banco de dados clientes do hardware
 - Máquinas multi-cores
 - Memória de estado sólido

1. Tecnologia de Banco de Dados - Motivação -

- Sistemas de arquivos (anos 60 e 70)
 - Cada aplicação

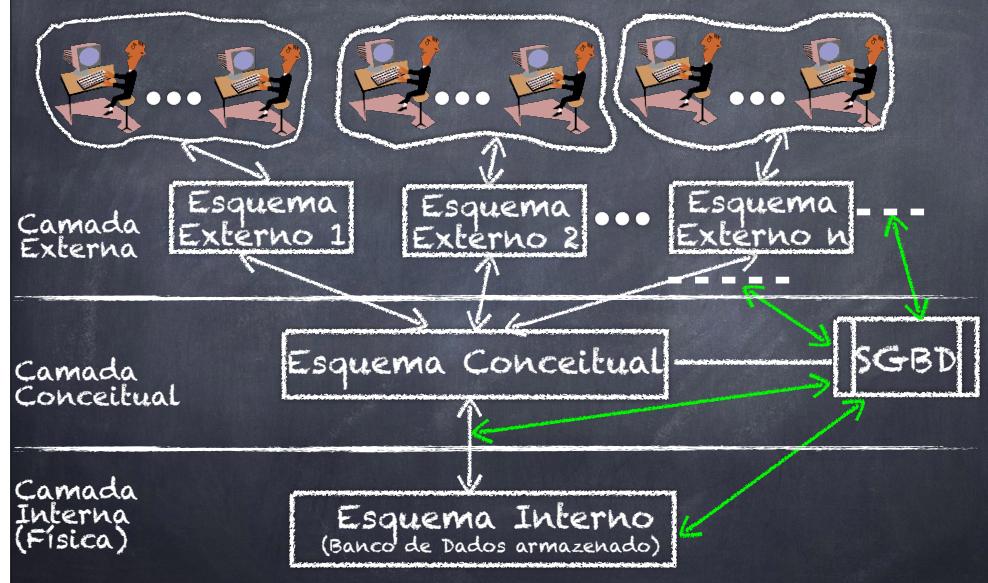


1. Tecnologia de Banco de Dados - Motivação -

• Propriedades

- Garantir independência de dados
- Alteração na organização Lógica ou física dos dados não implica na alteração de aplicações
- Eliminar redundância de dados
- Eliminar inconsistência de dados
- Facilitar acesso a dados através de uma Linguagem de consulta
- Evitar inconsistências produzidas pelo acesso concorrente
- Recuperar banco de dados após falhas

1. Tecnologia de Bancos de Dados - Arquitetura de Três Camadas -



1. Tecnologia de Bancos de Dados - Arquitetura de Três Camadas -

• Esquema Interno - Camada Interna

- Descreve como os dados estão fisicamente armazenados
 - Exemplo
 - Organização de arquivo
 - seqüencial-indexado, hash, seqüencial, heap
 - Alocação em disco
 - Estruturas de índices

• Esquema Conceitual - Camada Conceitual

- Descreve quais dados estão armazenados no banco de dados

• Esquema Externo - Camada Externa

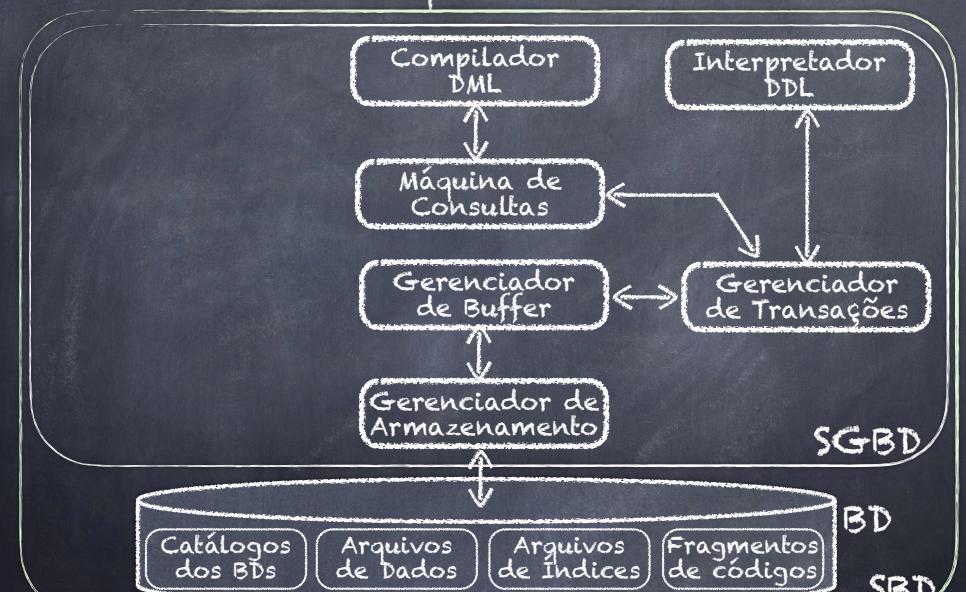
- Descreve parte do banco de dados
 - Simplificar a visão do usuário
 - "Ver" só o que interessa
 - Segurança

1. Tecnologia de Bancos de Dados - Sistemas de Banco de Dados -

• Sistema de Bancos de Dados (SBD)

- Banco de Dados (BD)
- Conjunto de dados relacionados
- Sistema Gerenciador de Bancos de Dados (SGBD)
- Componente de software
 - Acesso
 - Controle de Concorrência
 - Recuperação
 - Armazenamento

1. Tecnologia de Banco de Dados - Arquitetura SBD -



1. Tecnologia de Banco de Dados - Componentes da Arquitetura de SBDs -

- SGBD
 - Compilador DML
 - Analisa sintaticamente e semanticamente comandos DML expressos em uma Linguagem de consulta (SQL)
 - Traduz estes comandos para uma forma de representação interna (p. ex. álgebra relacional, cálculo relacional)
 - Pré-Compilador DML
 - Traduz comandos DML em chamadas a procedimentos na linguagem hospedeira
 - Interpretador DDL
 - Interpreta comandos DDL e os armazena no catálogo
 - Tabelas contendo meta-dados (Esquema)
 - Máquina de Consultas
 - Otimização de consultas e geração de planos de execução



1. Tecnologia de Banco de Dados - Componentes da Arquitetura de SBDs -

• SGBD

- Gerenciador de Transações
- Controle de concorrência
- Recuperação do banco de dados após falhas
- Gerenciador de Buffer
- Responsável por manter na área de buffer (memória principal) páginas mais acessadas
- Política de alocação de páginas em buffer
 - LRU, MRU, FIFO, ARC
- Gerenciador de Armazenamento
- Responsável pelo armazenamento físico em memória secundária



1. Tecnologia de Banco de Dados - Componentes da Arquitetura de SBDs -

• BD

- Arquivos de dados
- Armazena os dados
- Arquivos de Índices
- Estruturas de índices para os arquivos de dados
 - Índices ordenados
 - Árvores B+, Arquivos grade, árvores kd, Árvores R, Árvores quadrantes
 - Índices não ordenados
 - Índice hash

1. Tecnologia de Banco de Dados

- Componentes da Arquitetura de SBDs -

- BD
 - Catálogo
 - Armazena esquema do banco de dados (meta-dados)
 - Nomes das tabelas, atributos de cada tabela, definição de índice para uma tabela, etc...
 - Armazena informações estatísticas
 - Exemplo
 - Cardinalidade de uma tabela
 - Utilizadas na otimização de consultas
 - Fragmentos de código
 - Stored procedures, gatilhos (triggers), funções

1. Tecnologia de Banco de Dados

- Modelo De Dados -

- Definido por três componentes
 - Uma coleção de tipos de estrutura de dados
 - blocos de construção do banco de dados
 - Uma coleção de operadores
 - Podem ser aplicados a qualquer instância dos tipos de dados definidos para o modelo
 - Uma coleção de regras de integridade
 - Definem o conjunto de estados e consistentes do banco de dados
- Funcionalidade
 - Representar dados do mundo real
 - Capturar semântica e incorporá-la em um banco de dados
 - Através do modelo relacional representar os dados de uma universidade



UFC

CC

DC

1. Tecnologia de Banco de Dados

- Classificação de Sistemas de Bancos de Dados -

• Modelo de Dados

- Sistema de Banco de Dados Relacional
 - Modelo relacional
- Sistema de Banco de Dados Orientado a Objeto
 - Modelo orientado a objeto
- Sistema de Banco de Dados Objeto-Relacional
 - Modelo relacional + Modelo OO



UFC

CC

DC

1. Tecnologia de Banco de Dados

- Classificação de Sistemas de Bancos de Dados -

• Arquitetura

- Centralizada
 - Sistema de Banco de Dados Centralizados
 - Os componentes do SBD residem no mesmo host
- Distribuída
 - Critérios
 - Função, Controle, Dados
 - Sistema de Banco de Dados Cliente-Servidor
 - Distribuição de funções do SGBD entre clientes e servidor
 - Sistema de Banco de Dados Paralelos
 - Distribuição do controle de funções do DBMS entre diversos sistemas computacionais
 - Sistema de Banco de Dados Distribuídos
 - Distribuição de dados através de diversos SBDs homogêneos



UFC
CC
DC

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Modelo de dados MER

- Proposto por P. Chen em 1976
- Utilizado como modelo conceitual em projeto de BDs
 - Ferramenta para a modelagem de BDs
- Não é implementado por nenhum SBD

• Princípio básico

- Representar dados do mundo real através
 - Entidades
 - Relacionamentos
 - Entre entidades
- Atributos
- Propriedades de entidades ou relacionamentos



UFC
CC
DC

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Entidade

- Representação de um objeto do mundo real
 - Exemplos de entidades do mundo real
 - Objeto concreto
 - Um servidor, um professor, um estudante
 - Objeto abstrato
 - Uma empresa, uma conta bancária, uma disciplina
- Conjunto de entidades (tipo de entidade)
 - Entidades que apresentam características semelhantes
 - Estudantes, Professores, Disciplinas



UFC
CC
DC

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

- Atributos de uma entidade

- Propriedades que caracterizam uma entidade
 - Exemplos

- Atributos de empregados

- matrícula, nome, endereço, rg, cpf, data-nasc, salário, lotação, data-admissão

- Atributos de estudantes

- matrícula, nome, curso, rg, cpf, data-ingresso

- A cada atributo de uma entidade deve estar associado um valor



UFC
CC
DC

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

- Atributos de uma entidade (cont.)

- Definição formal

- $\forall E_i \in \text{Entidades}, \text{Atributo}: E_i \rightarrow v_k$, onde $v_k \in \text{dom(Atributo)}$

- Atributo nome de entidades Empregado

- nome(Empregado_1)=José

- Conjunto de entidades

- Entidades que apresentam mesmo conjunto de atributos

2. Modelo Entidade-Relacionamento

- Conceitos Básicos -

- Atributos chave de uma entidade

- Atributos que identificam univocamente uma entidade
- Seja f atributo chave para o conjunto de entidades E , $x \in E$ e $f(x)=v$, então $\forall e \in E \Rightarrow f(e) \neq v$, onde $e \neq x$

- Exemplo

- Matrícula é atributo chave para Estudante

- Tipos de atributos

- Atributo atômico
- Atributo composto
- Formado por vários atributos atômicos



2. Modelo Entidade-Relacionamento

- Conceitos Básicos -

- Tipos de atributos

- Atributo mono-valorado

- Atributo para o qual está associado um único valor

- Atributo multi-valorado

- Atributo para qual podem estar associados vários valores

- Exemplo

- Para o atributo telefone podem estar associados vários valores, como telefone residencial, comercial e celular

- Atributo derivado

- Atributo cujo valor pode ser derivado com base no valor de um outro atributo (atributo base)

- Exemplo

- Idade pode ser derivado do atributo data-nasc

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Relacionamento

- Abstração que representa associação entre entidades de diferentes conjuntos de entidades

◦ Exemplo

- Relacionamento que associa o empregado Bárbara com o departamento "Ciência da Computação"

• Conjunto de relacionamentos (tipo de relacionamento)

- Grupo de relacionamentos que representam o mesmo tipo de associação

◦ Conjunto de relacionamentos Lotação

- Todos relacionamentos entre empregado e departamento

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Conjunto de relacionamentos (cont)

- Seja R um conjunto de relacionamentos, representando associações entre os conjuntos de entidades E_1, E_2, \dots, E_n , então
- $R \subseteq E_1 \times E_2 \times \dots \times E_n$

- Seja $r \in R$, então $r = (e_1, e_2, \dots, e_n)$, onde $e_1 \in E_1, e_2 \in E_2, \dots, e_n \in E_n$

◦ Exemplo

- Lotação \subseteq Departamento \times Empregado
 - (Bárbara, Ciência da Computação) \in Lotação

◦ Papel de uma entidade no relacionamento

- Função de uma entidade no relacionamento
 - Papel de empregado
 - É Lotado (Bárbara é lotada em Ciência da Computação)
 - Papel de departamento
 - Lota (Ciência da Computação lota Bárbara)



UFC
CC
DC

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Grau de Relacionamento

- Número de entidades participantes no relacionamento
- Exemplos
 - Relacionamento binário
 - Relacionamento ternário
 - Exemplo
 - Relacionamento que associa Agência-Conta-Cliente



UFC
CC
DC

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Auto-relacionamento (relacionamento recursivo)

- Relacionamento envolvendo um único conjunto de entidades
- O conjunto de entidades apresenta diferentes papéis
- Considere o seguinte cenário:
 - Supervisores são responsáveis por subconjuntos de empregados de um mesmo departamento
 - Relacionamento supervisiona
 - Empregado desempenha dois papéis
 - é-supervisionado-por
 - é-supervisor-de

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Atributos de relacionamento

- Propriedades que descrevem um relacionamento
- Considere o atributo data-lotação
 - Representa a data em que um empregado foi lotado em um determinado departamento
 - data-lotação é atributo do relacionamento Lotação
- Considere o atributo nota
 - Descreve a nota de um aluno em uma disciplina
 - nota é um atributo do relacionamento cursa
 - Relacionamento entre os conjuntos de entidades Estudante e Disciplina

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Restrições estruturais de relacionamentos

- Cardinalidade de relacionamento
- Restrição de participação
- Cardinalidade de relacionamento
 - Indica o número de entidades que podem participar de um relacionamento
 - Seja R um relacionamento binário entre os conjuntos de entidades A e B
 - Cardinalidade Um para um (1:1)
 - Uma entidade de A está associada a uma só entidade de B
 - Uma entidade de B está associada a uma só entidade de A

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Cardinalidade de relacionamento

- Cardinalidade Um para muitos (1:N)
 - Uma entidade de A (B) está associada a qualquer quantidade de entidades de B (A)
 - Uma entidade de B (A) está associada a uma só entidade de A (B)
- Cardinalidade Muitos para muitos (N:N)
 - Uma entidade de A pode estar associada a qualquer quantidade de entidades de B
 - Uma entidade de B pode estar associada a qualquer quantidade de entidades de A

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Exemplos Cardinalidade de relacionamentos

- Cardinalidade do relacionamento lotação entre Departamento e Empregado 1:N
- Cardinalidade do relacionamento cursa entre Estudante e Disciplina N:N
- Cardinalidade do auto-relacionamento supervisiona 1:N
- Cardinalidade do relacionamento gerência entre Departamento e Empregado 1:1

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Restrição de participação

- Especifica a obrigatoriedade ou não de uma entidade e participar de um relacionamento com outra entidade
- Participação total
 - A participação de um conjunto de entidades A é total em R, se toda entidade de A participa de pelo menos um relacionamento em R
 - Dependência existencial
 - Considere o relacionamento Lotação, para o qual todo empregado deve estar lotado em algum departamento
 - A participação de Empregado em Lotação é total
- Participação parcial
 - A participação de um conjunto de entidades A é parcial em R, se apenas um subconjunto de entidades de A participa em R
 - Relacionamento cursa entre Estudante e Disciplina

2. Modelo Entidade-Relacionamento - Conceitos Básicos -

• Entidade fraca

- Entidade cuja existência depende de estar associada, via um relacionamento (relacionamento de identificação), com uma outra entidade (entidade forte)
- Considere o relacionamento dependência entre os conjuntos de entidades Empregado e Dependente (dependentes dos empregados)
 - A existência do dependente Bárbara
 - Condiciona-se existência do empregado Angelo e que Bárbara esteja relacionada a Angelo através do relacionamento dependência
 - Uma entidade fraca é identificada
 - Por estar relacionada com uma entidade forte
 - Pelo atributo chave da entidade forte
 - Atributos da própria entidade fraca
 - Chave parcial

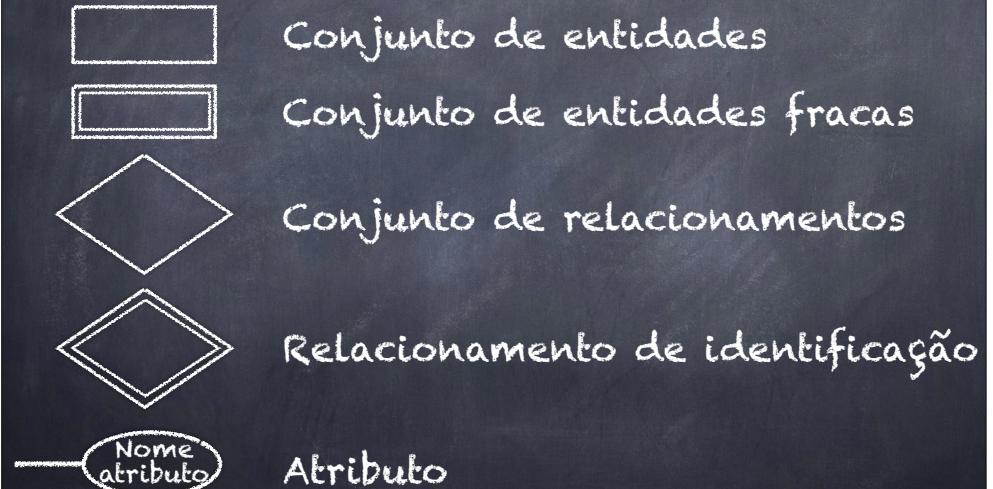
2. Modelo Entidade-Relacionamento - Diagrama ER -

- Ferramenta (gráfica) de projeto

- Capaz de capturar e representar graficamente toda estrutura lógica de um banco de dados
- Utilizada para modelagem de BDs
- Existem ferramentas
 - Fornecem interface gráfica para criar DERs
 - brmodelo, ERwin, visual paradigm
 - A partir do DER especificado, geram o esquema do BD relacional
 - ERwin, visual paradigm

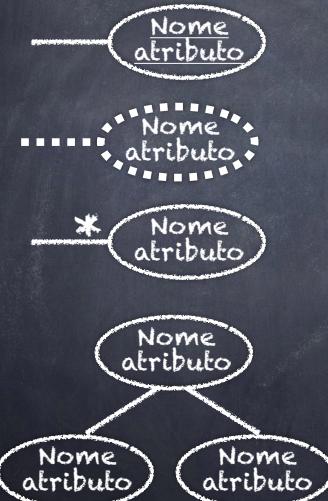
2. Modelo Entidade-Relacionamento - Diagrama ER -

- Notação



2. Modelo Entidade-Relacionamento - Diagrama ER -

● Notação



Atributo chave

Atributo derivado

Atributo multivalorado

Atributo composto

2. Modelo Entidade-Relacionamento - Diagrama ER -

● Notação



Atributo chave parcial
de uma entidade fraca



Cardinalidade 1:N



Participação total
de E2 em R

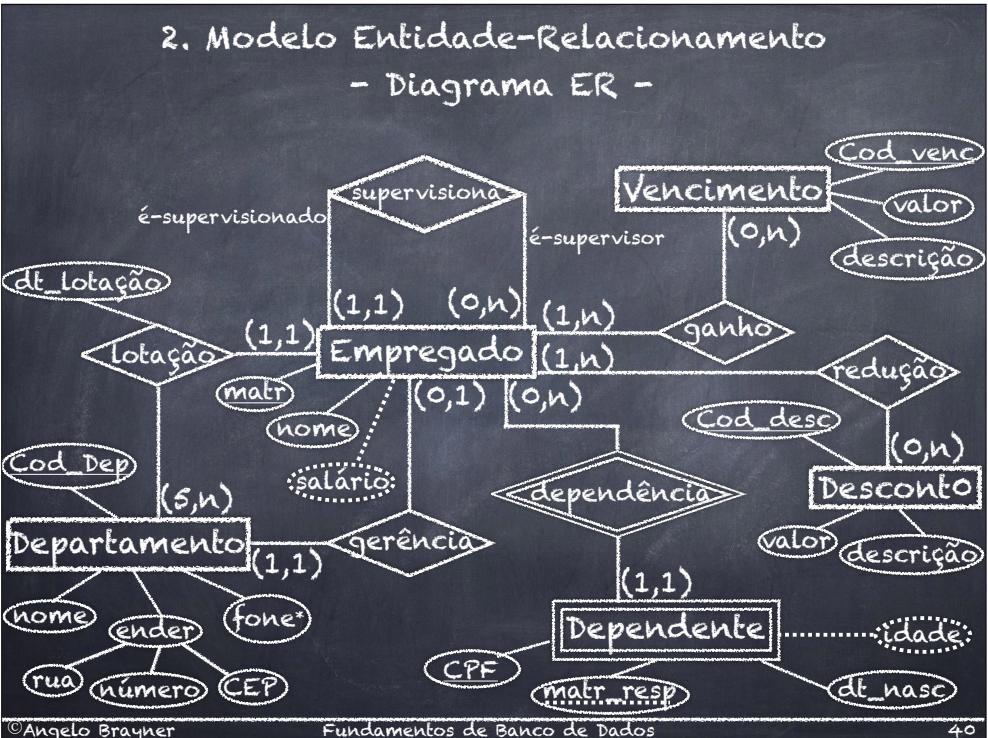


Restrição estrutural
de participação e
cardinalidade de E2
em R

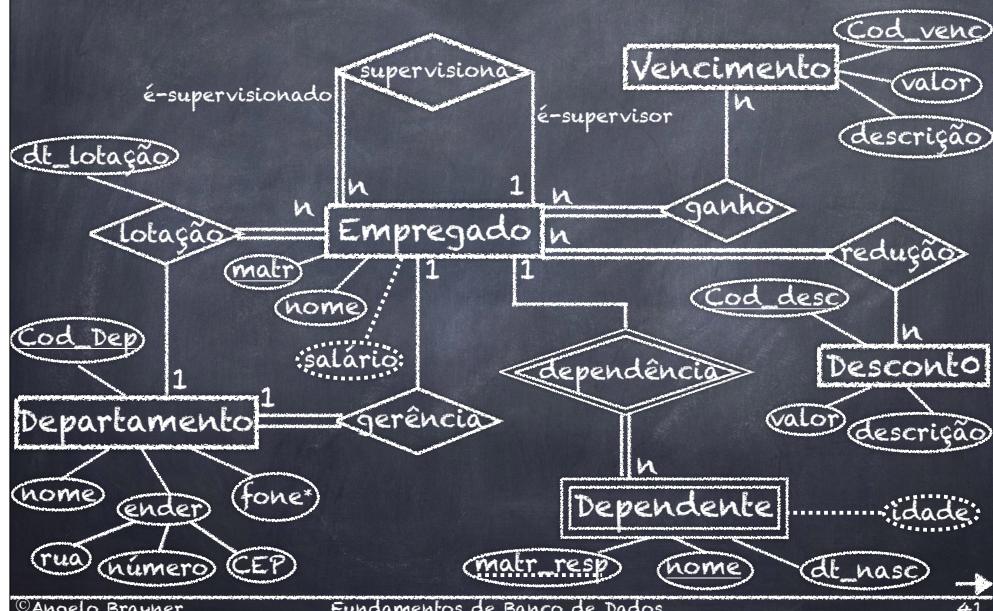
2. Modelo Entidade-Relacionamento - Diagrama ER -

Exercício

- Utilize o DER para modelar o BD para o seguinte cenário na empresa X
- Todo empregado deve estar lotado em um único departamento e é coordenado por um supervisor (empregado)
- Um departamento possui no mínimo 5 empregados, onde um deles é o gerente do departamento.
- Os dependentes dos funcionários devem possuir como atributos: nome, data-nasc. A idade limite para ser dependente de um empregado é 18 anos
- O salário de um empregado é calculado com base nos seus diversos vencimentos e descontos.
- Para cada tipo de vencimento ou de desconto, existe uma descrição e o valor correspondente



2. Modelo Entidade-Relacionamento - Diagrama ER -



2. Modelo Entidade-Relacionamento - Diagrama ER -

- Utilize o DER para modelar o BD com os dados de estoque da empresa X (**desafio: 0,5 na 1a. Avaliação**)
 - Os itens de estoque possuem um código, descrição, quantidade em estoque, preço de compra, preço de venda e o fornecedor.
 - Cada fornecedor terá um código, nome, endereço, telefones, cidade e estado.
 - Existem várias Lojas (filiais) vendendo produtos da empresa X. Cada loja tem um código identificador, nome, endereço, cidade de localização e um gerente (um vendedor da loja).
 - Cada vendedor terá uma matrícula, nome, RG, CPF, salário e filial de lotação.
 - Para cada venda, deve ser identificado o código do item vendido, a matrícula do vendedor que efetuou a venda, a quantidade vendida, o preço de venda, a data e hora em que a venda foi efetuada.

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

• Especialização

- Existem conjuntos de entidades compostos de subgrupos de entidades, onde
 - Cada subgrupo apresenta propriedades específicas
- Conjunto de entidades Empregado numa universidade
 - Pode-se identificar os seguintes subconjuntos
 - Professores
 - Titulação
 - Universidade de titulação
 - Regime de trabalho (DE, 40h, 20h)
 - Técnicos
 - Grau de instrução
 - Área de atuação (contador, secretária, etc...)
 - Engenheiros
 - Especialidade

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

• Especialização

- Processo de identificação de subgrupos de entidades dentro de um conjunto de entidades
 - Processo de especialização pode ser recursivo
- Um único conjunto de entidades pode ser especializado por mais de uma característica de diferenciação (especialização)
 - No exemplo de Empregado em uma universidade
 - Especialização por tipo de empregado
 - Especialização por tipo de contrato
 - RJU
 - Serviços prestados

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

- Especialização (cont.)

- Subgrupos identificados em um processo de especialização podem participar de relacionamentos que não se aplicam a todas entidades do conjunto de entidades de origem

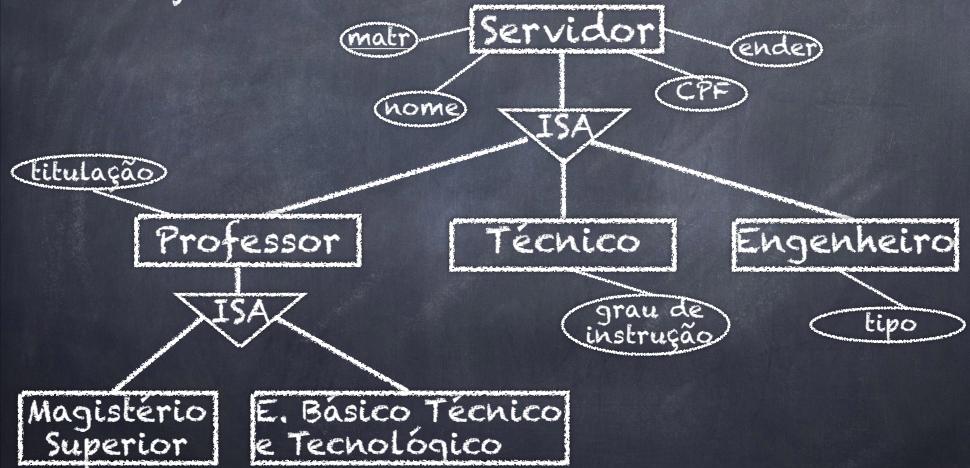
- Herança de propriedades

- Os subgrupos de entidades herdam todas as propriedades do conjunto de entidade de mais alto nível

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

- Especialização (cont.)

- Notação MER



2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

• Generalização

- Processo de identificação de conjuntos de entidades que possuem características em comum
- Mesmos atributos e participam de mesmos relacionamentos
- Formação de um único conjunto de entidades de mais alto nível
- É realizada sobre vários conjuntos de entidades
- Identificar um conjunto de entidades de mais alto nível

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

• Generalização (cont.)

- Ênfase nas similaridades entre diversos conjuntos de entidades
- Redução de redundância de representação
 - Atributos compartilhados só serão representados no conjunto de entidades de nível mais alto
 - Não serão repetidos
- Na prática
 - Generalização é o processo inverso da especialização

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

- Restrições para generalização e especialização
- Sejam E_1, E_2, \dots, E_n subconjuntos de entidades do conjunto de entidades E



2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

- Restrições para generalização e especialização
 - Disjunção
 - $\forall i, j \leq n, i \neq j : E_i \cap E_j = \emptyset$
 - Nenhuma entidade de E pode pertencer a mais de um subgrupo
 - Completeza
 - Total
 - $E = \bigcup_{i=1}^n E_i$
 - Parcial
 - $E \supseteq \bigcup_{i=1}^n E_i$

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

• Agregação

- Abstração que representa relacionamentos como entidades
- Utilizado para representar relacionamentos de relacionamentos

• Exemplo

- Modelagem de dados em um banco BX, onde clientes estão relacionados a agência e conta

• Estratégia 1



2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

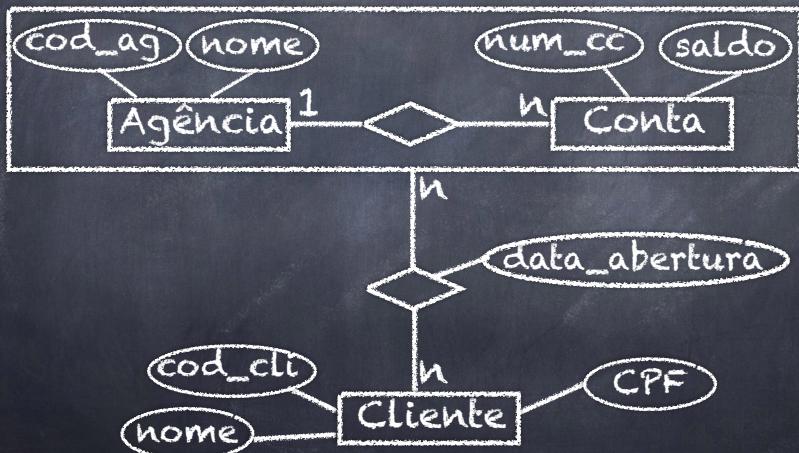
• Agregação

- Problemas na Estratégia 1
 - Não representa a estrutura lógica que deve ser modelada
 - Relacionamento de cliente com o relacionamento agência-conta
 - Na prática, para acessar todas as contas de uma agência
 - Terão que ser acessadas todas triplas (agência, conta, cliente)
 - Uma conta pode pertencer a vários clientes
 - Vários acessos redundantes

2. Modelo Entidade-Relacionamento - Propriedades Avançadas -

- Agregação

- Estratégia 2 (agregação)



3. Modelo Relacional - Introdução -

- Proposto em 1970 por Codd

- IBM

- Consolidou-se como principal modelo de dados para aplicações comerciais

- Modelo de dados para Sistemas de Banco de Dados Relacionais

- SBDs relacionais

- DB2 (IBM)

- Postgres

- MySQL

- Oracle

- SQL Server

3. Modelo Relacional - Introdução -

- Um banco de dados relacional consiste
 - Um conjunto de tabelas (relações)
- Tabelas
 - Conjunto de linhas (Tuplas)
 - Cada linha
 - Conjunto de colunas (atributos)

3. Modelo Relacional - Conceitos Básicos -

- Domínio
 - Conjunto de valores permitidos para um atributo
 - Valores são atômicos
 - Indivisíveis
 - Exemplo: Domínio do atributo matrícula
 - Conjunto de todos os valores válidos de matrícula
 - $\text{dom}(A)$ denota o domínio do atributo A

3. Modelo Relacional - Conceitos Básicos -

- Esquema de uma relação R
 - Descreve os dados uma relação (meta-dados)
 - Representado por $R(A_1, A_2, \dots, A_n)$, onde
 - A_1, A_2, \dots, A_n uma lista de atributos de R
- Instância $r(R)$ de uma Relação R (ou relação R)
 - Dado o esquema $R(A_1, A_2, \dots, A_n)$
 - $r(R) \subseteq \text{dom}(A_1) \times \text{dom}(A_2) \times \dots \times \text{dom}(A_n)$
 - subconjunto do produto cartesiano dos domínios dos atributos que definem R
 - cada tupla de $r(R)$ **relaciona** valores dos vários domínios
 - $r(R)$ representa um conjunto de tuplas de R

3. Modelo Relacional - Conceitos Básicos -

- Informalmente, uma relação (tabela) R , com esquema $R(A_1, A_2, \dots, A_n)$, consiste da
 - Instância $r(R)$, onde
 - $r(R) = \{t_1, t_2, \dots, t_k\}$
 - Cada t_i representa uma n -tupla de n valores $\langle v_1, v_2, \dots, v_n \rangle$, onde cada $v_j \in \text{dom}(A_j)$, $0 \leq j \leq n$
- Observação importante
 - $r(R)$ contém os dados
 - o esquema de R descreve os dados armazenados na relação

3. Modelo Relacional

- Conceitos Básicos -

- Esquema de um banco de dados relacional
 - Conjunto de esquemas de relação mais um conjunto de restrições de integridade IC
 - $S = \{R_1, R_2, \dots, R_n\}$ é um conjunto de restrições de integridade IC
- Instância de um banco de dados relacional
 - Seja o esquema S
 - Uma instância DB para o esquema S
 - $DB = \{r_1, r_2, \dots, r_n\}$, onde
 - Cada r_i é uma instância de relação de R_i e
 - Cada r_i satisfaz as restrições de integridade especificadas em IC

3. Modelo Relacional

- Restrições -

- Restrição de domínio
 - O valor de cada atributo A tem que ser um valor atômico de $\text{dom}(A)$
- Restrição de Chave
 - Uma relação R é definida como um conjunto de tuplas
 - Elementos de conjuntos são distintos entre si
 - Tuplas de R devem ser distintas entre si
 - Duas tuplas de R não podem ter a mesma combinação de valores para seus atributos
 - Geralmente existe um subconjunto SC de atributos em um esquema de relação R
 - Todas as tuplas de qualquer instância $r(R)$ apresentam uma combinação diferente de valores para os atributos de SC
 - $\forall t_i, t_j \in r \text{ } \forall i, j \leq n, i \neq j : t_i[SC] \neq t_j[SC]$
 - Super chave (superkey)

3. Modelo Relacional - Restrições -

○ Restrição de Chave

- Super chave pode apresentar atributos redundantes
 - Empregado(matr, nome, ender, cpf)
 - matr e cpf são atributos da super chave
 - Apenas um dos dois já garante a não existência de tuplas repetidas
- Chave candidata (candidate key)
 - Atributo da super chave que pode funcionar como chave da relação
 - Para Empregado existem duas chaves candidatas
 - matr ou cpf
- Chave primária (primary key)
 - Chave candidata escolhida como chave da relação
 - Garante a unicidade de uma tupla na relação

3. Modelo Relacional - Restrições -

○ Restrição de Integridade de Entidade

- Especifica que nenhuma chave primária pode ter valor nulo

○ Restrição de Integridade Referencial

- Sejam dois esquemas de relação R e S
- Um conjunto de atributos FK de um esquema de relação R é chave estrangeira (foreign key) se
 - Os atributos em FK têm o mesmo domínio que a chave primária PK da relação S, e
 - Um valor de FK em uma tupla t_i de $r(R)$
 - Ou ocorre em S como valor da chave primária PK para uma tupla t_k em $s(S)$
 - $t_i[FK] = t_k[PK]$
 - ou é nulo

3. Modelo Relacional - Restrições -

- Restrição de Integridade Referencial
 - Exemplo
 - Departamento(cod_depart, nome, ender)
 - Empregado(matr, nome, ender, cpf, lotação)
 - Se Lotação for chave estrangeira (referenciando cod_depart), então
 - Deve ser garantido que o valor de lotação ou
 referecie um valor existente como chave
 primária em Departamento ou seja nulo
 - Garantida automaticamente pelos SGBD
 existente no mercado

3. Modelo Relacional - Álgebra Relacional -

- Coleção de operadores sobre relações
 - Relação expressão AR → Relação
- Linguagem de consulta procedural para BDs
 relacionais
 - Ferramenta Relax
- Operadores básicos

◦ Seleção	(σ)	Operadores unários
◦ Projeção	(Π)	
◦ União	(U)	Operadores binários
◦ Diferença	(-)	
◦ Produto cartesiano	(X)	

3. Modelo Relacional - Álgebra Relacional -

- Operação de seleção

- Seleciona um subconjunto de tuplas de uma relação com base em um predicado
- Filtro de tuplas

- Notação

- $\sigma_P(r)$

- r é uma relação
- P representa um predicado (condição de seleção)
- P é construído através de átomos
- $t[A_i] \theta t[A_k]$, $t \in r$ e A_i e A_k são atributos de r
- $t[A_i] \theta C$, onde C é uma constante
- θ denota um operador de comparação
 - $=, \neq, >, \geq, <, \leq$

- Átomos podem ser conectados por \wedge (and), \vee (or), \neg (not)

3. Modelo Relacional - Álgebra Relacional -

- Operação de seleção

- Exemplo

- Considere a relação Empregado

- Empregado(matr, nome, ender, cpf, salário, lotação)

- Listar todos os empregados que ganham salário maior que 5000

- $\sigma_{\text{salário} > 5000}(\text{Empregado})$

- Listar todos os empregados não lotados no departamento com código igual a 002 e que ganham salários entre 5000 e 10000

3. Modelo Relacional - Álgebra Relacional -

• Ferramenta Relax

- Banco de dados **Kemper Datenbanksysteme (en)**
- Construa o DER
- Consulta **C1**
- Construa e execute a expressão da AR, que retorna as disciplinas de quatro créditos, lecionadas pelo professor de ID igual a 2125

3. Modelo Relacional - Álgebra Relacional -

• Operação de seleção (cont.)

- Propriedades da seleção
 - $\sigma_{P_1 \wedge P_2}(r) \Leftrightarrow \sigma_{P_1}(\sigma_{P_2}(r))$
 - Distributividade
 - $\sigma_{P_1}(\sigma_{P_2}(r)) \Leftrightarrow \sigma_{P_2}(\sigma_{P_1}(r))$
 - Comutatividade
- Exercício
 - Mostre as propriedades acima, utilizando a consulta C1

3. Modelo Relacional - Álgebra Relacional -

• Operação de Projeção

- Seleciona um subconjunto de atributos de uma relação
- Filtro de colunas

• Notação

- $\Pi_{A_{i1}, A_{i2}, \dots, A_{in}}(r)$
- r é uma relação com esquema $R(A_1, A_2, \dots, A_n)$
- $\{A_{i1}, A_{i2}, \dots, A_{in}\} \subseteq \{A_1, A_2, \dots, A_n\}$

• Exemplo

- Listar o nome e salário de todos os funcionários
 - $\Pi_{\text{nome, salário}}(\text{Empregado})$
- Listar nome e salário de todos os empregado que ganham salário maior que 9000

3. Modelo Relacional - Álgebra Relacional -

• Exercício

- Construa e execute a expressão da AR, que retorna o nome das disciplinas (`lecture.title`) de quatro créditos, lecionadas pelo professor de ID igual a 2125

3. Modelo Relacional - Álgebra Relacional -

• Operação de União

- Executa a união de duas relações compatíveis
 - Duas relações $R(A_1, A_2, \dots, A_n)$ e $S(B_1, B_2, \dots, B_n)$ são compatíveis, se
 - Apresentam o mesmo número de atributos
 - $\text{dom}(A_i) = \text{dom}(B_i), \forall i, 0 \leq i \leq n$

• Notação

- $r \cup s$

• Exemplo

- Considere as seguintes relações
 - $\text{Empregado}(\text{matr, nome, ender, dt-nasc, cpf, salário, lotação})$
 - $\text{Dependente}(\text{nome-dep, data-nasc, matr-resp})$
- Listar nome e data de nascimento de todos os funcionários e dependentes na empresa
 - $\Pi_{\text{nome, dt-nasc}}(\text{Empregado}) \cup \Pi_{\text{nome-dep, data-nasc}}(\text{Dependente})$

3. Modelo Relacional - Álgebra Relacional -

• Operação de Diferença

- O resultado da operação $r - s$
 - relação que contém as tuplas de r que não pertencem a s
 - r e s são relações compatíveis

• Exemplo

- Listar matrícula de estudantes não matriculados em disciplinas (Relax)
 - $\Pi_{\text{student_id}}(\text{student}) - \Pi_{\text{student_id}}(\text{attends})$

3. Modelo Relacional - Álgebra Relacional -

• Exercícios

- Listar ID dos professores que não estão lecionando disciplinas
- Listar o nome de todos os professores e de todos os alunos

3. Modelo Relacional - Álgebra Relacional -

• Operação de Produto Cartesiano

- Sejam r e s com esquemas $R(A_1, A_2, \dots, A_n)$ e $S(B_1, B_2, \dots, B_m)$, respectivamente
- Resultado da operação $r \times s$ é uma relação T
 - $T(r.A_1, r.A_2, \dots, r.A_n, s.B_1, s.B_2, \dots, s.B_m)$
 - $(n+m)$ atributos
 - $t \in T \Leftrightarrow \exists v \in r \text{ e } \exists u \in s, \text{ tal que } t[A_i] = v[A_i], 0 \leq i \leq n, \text{ e } t[B_j] = u[B_j], 0 \leq j \leq m$
- Sejam n_r e n_s as cardinalidades de r e s , respectivamente
 - A cardinalidade de T é $n_r \cdot n_s$

3. Modelo Relacional
 - Álgebra Relacional -

• Operação de Produto Cartesiano

 • $r \times s$

r	r.A r.B
a1	b1
a1	b2
a2	b1

s	s.A s.B s.C
a3	b6 c5
a2	b3 c3
a2	b1 c4

 $r \times s$

r.A r.B s.A s.B s.C
a1 b1 a3 b6 c5
a1 b1 a2 b3 c3
a1 b1 a2 b1 c4
a1 b2 a3 b6 c5
a1 b2 a2 b3 c3
a1 b2 a2 b1 c4
a2 b1 a3 b6 c5
a2 b1 a2 b3 c3
a2 b1 a2 b1 c4

 3. Modelo Relacional
 - Álgebra Relacional -

• Exercício

- Execute o produto cartesiano entre as relações Professor e Lecture
- Explique o resultado apresentado

3. Modelo Relacional - Álgebra Relacional -

• Exercícios (Ferramenta Relax)

- Escreva as seguintes consultas sobre Banco de dados **Kemper Datenbanksysteme (en)**, utilizando a álgebra relacional
- Listar disciplinas lecionadas pelo professor com id igual a 2125
- Mostrar o nome do professor, com sua sala
- Listar matrícula dos alunos com nota menor que 2
- Listar nome de disciplinas com créditos menores que 4
- Apresentar id dos professores que não estão ministrando disciplinas
- Mostrar nome do professor e nome da disciplina ministrada por ele

3. Modelo Relacional - Álgebra Relacional -

• Operadores derivados

- Definidos a partir dos operadores básicos
 - Junção (\bowtie)
 - Semi-junção (\bowtie^*)
 - Interseção (\cap)
 - Divisão (\div)

3. Modelo Relacional - Álgebra Relacional -

• Operação de Junção (theta-join)

- $r \bowtie_P s = \sigma_P(r \times s)$
- P é a condição de junção, definido
 - P contém apenas operações de comparação entre atributos de r e s

3. Modelo Relacional - Álgebra Relacional -

• Operação de Junção (cont.)

- $r \bowtie_{r.B \neq s.B} s$

r r.A r.B	
a1	b1
a1	b2
a2	b1

s s.A s.B s.C		
a3	b6	c5
a2	b3	c3
a2	b1	c4

r $\bowtie_{r.B \neq s.B} s$				
r.A r.B s.A s.B s.C				
a1	b1	a3	b6	c5
a1	b1	a2	b3	c3
a1	b2	a3	b6	c5
a1	b2	a2	b3	c3
a1	b2	a2	b1	c4
a2	b1	a3	b6	c5
a2	b1	a2	b3	c3

3. Modelo Relacional - Álgebra Relacional -

• Exercícios (Ferramenta Relax)

- Escreva as seguintes consultas sobre Banco de dados **Kemper Datenbanksysteme (en)**, utilizando a álgebra relacional
- Apresentar nome dos professores que não estão ministrando disciplinas

renomeando a tabela professor por x

$$\pi_{\text{name}}((\pi_{x.\text{employee_id}}(p \times (\text{professor})) - \pi_{\text{lectured_by}}(\text{lecture})) \\ \times_{x.\text{employee_id}=\text{professor.employee_id}} \text{professor})$$

- Mostrar nome do professor e nome da disciplina ministrada por ele

3. Modelo Relacional - Álgebra Relacional -

• Operação de Semi-Junção (Semi-join)

$$r \ltimes_P s = \Pi_R (r \bowtie_P s)$$

• Utilizada para otimizar o processamento de consultas em SBDs Distribuídos

• Operação de Interseção

$$r \cap s = r - (r - s)$$

• r e s devem ser relações compatíveis

3. Modelo Relacional
 - Álgebra Relacional -

• Operação de divisão

 • Sejam r e s com esquemas

$r(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m) \in S(B_1, B_2, \dots, B_m)$

 • Resultado da operação $r \div s$

• É uma relação $T(A_1, A_2, \dots, A_n)$, onde

• Para uma tupla v pertencer a T , todos os valores de v precisam aparecer em R em associação com toda tupla de S

 3. Modelo Relacional
 - Álgebra Relacional -

• Operação de divisão

• Exemplo

r		
$r.A$	$r.B$	$r.C$
a1	b6	c3
a2	b3	c3
a2	b3	c1
a3	b3	c2
a2	b3	c2

s
$s.C$
c3
c1
c2

$r \div s$		
$r.A$	$r.B$	
a2	b3	

3. Modelo Relacional - Álgebra Relacional -

• Operação de divisão (cont.)

- Sejam r e s com esquemas
 - $R(A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_m)$ e $S(B_1, B_2, \dots, B_m)$
 - Conjunto de atributos de S é um subconjunto dos atributos de R (R -Div)
- $r \div s = \Pi_{R-(R \cap S)}(r) - \Pi_{R-(R \cap S)}((\Pi_{R-(R \cap S)}(r) \times s) - \Pi_{R-(R \cap S)}(s, r))$

retorna tuplas de r com falsas associações com s

3. Modelo Relacional - Álgebra Relacional -

• Operação de divisão

- Considere as tabelas
 - Emp-Desc(matr, cod-desc) e
 - Desconto(cod-desc, valor, descrição)
- Mostrar a construção do resultado da consulta que retorna os empregados que têm descontados todos os descontos existentes na empresa

Emp_desc	
cod-desc	matr
3	11
99	5
7	11
7	21
99	11

Desconto Garantir R-Div
Π_{cod-desc} (Desconto)

cod-desc	valor	descrição
3	30	IR
99	8.5	INSS
7	10	Seguro

3. Modelo Relacional - Álgebra Relacional -

• Operação de Atribuição

- Às vezes, é importante escrever uma expressão da álgebra relacional em diferentes partes

- Atribuir resultados das partes a relações temporárias

• Notação

- \leftarrow ou \leftarrow

• Exemplo

- $rel1 \leftarrow \Pi_{R-S}((\Pi_{R-S}(r) \times s) - \Pi_{R-S}(r))$

- $rel2 \leftarrow \Pi_{R-S}(r)$

- resultado := rel1 - rel2

3. Modelo Relacional - Álgebra Relacional -

• Funções Agregadas

- Funções aplicadas sobre uma coleção de valores do banco de dados

• sum

- Retorna o somatório dos valores de uma coleção

• avg

- Retorna a média dos valores de uma coleção

• max

- Retorna o maior valor de uma coleção de valores

• min

- Retorna o menor valor de uma coleção

3. Modelo Relacional - Álgebra Relacional -

• Funções Agregadas (cont)

- count

- Retorna o número de elementos de uma coleção

- count-distinct

- Algumas vezes, torna-se necessário eliminar repetições para o cálculo das funções agregadas

3. Modelo Relacional - Álgebra Relacional -

• Funções Agregadas (cont.)

- Exemplos

- Considere a relação

- Empregado(matr, nome, ender, salário, cpf, lotação)

- matr é a chave primária

- Encontre o número de empregados lotados no departamento 001

- $\text{count}(\Pi_{\text{matr}}(\text{Olotação}=001 \text{ (Empregado)}))$

- Encontre o maior salário da empresa

- $\text{max}(\Pi_{\text{salário}}(\text{Empregado}))$

3. Modelo Relacional - Álgebra Relacional -

• Funções Agregadas (cont.)

• Exemplos

- Encontre o salário médio da empresa
 - $\text{avg}(\Pi_{\text{salário}}(\text{Empregado}))$
- Encontre a quantidade de salários distintos no departamento 001
 - $\text{count-distinct}(\Pi_{\text{salário}}(\text{Olotação}=001 \text{ (Empregado)}))$

• Desafio (0.2 na 1a. Aval.)

- Encontre o primeiro e segundo maiores salários da empresa

3. Modelo Relacional - Álgebra Relacional -

- Execute as seguintes consultas no Relax (Datenbanksystem)
 - Retornar matrícula (student_id) dos estudantes que não estão matriculados em alguma disciplina
 - Retornar o nome de todos alunos com os códigos das disciplinas em que estão matriculados

3. Modelo Relacional - Álgebra Relacional -

- Considere as seguintes relações
 - Vendedor(matr, nome, ender, salário, cpf)
 - Vendas(matr_vend, cod_item, qtd, pr_venda)
 - Listar histórico de vendas de cada vendedor. O resultado deve apresentar o seguinte esquema:
 - Res(nome, cod-item, qtd, pr-venda)
- $\Pi_{\text{Res}}(\text{Vendedor} \bowtie_{\text{matr}=\text{matr_vend}} \text{Vendas})$
- Consulta com perda de informação
- Não aparecerão no resultado Vendedores que não efetuaram vendas

3. Modelo Relacional - Álgebra Relacional -

- Outer join
 - Adicionar tuplas que não satisfazem a condição de junção ao resultado da junção
- Tipos
 - Junção externa à esquerda (left outer join)
 - Junção externa à direita (right outer join)
 - Junção externa completa (full outer join)

3. Modelo Relacional - Álgebra Relacional -

• Operação de Left Outer join

- $r \bowtie s$
- Calcula o resultado da junção de r com s
 - Adiciona ao resultado da junção
 - Tuplas da relação à esquerda (r) que não satisfazem à condição de junção
 - Atribui valores nulos aos atributos não definidos para estas tuplas

3. Modelo Relacional - Álgebra Relacional -

• Operação de Right Outer join

- $r \bowtie s$
- Calcula o resultado da junção de r com s
 - Adiciona ao resultado da junção
 - Tuplas da relação à direita (s) que não satisfazem à condição de junção
 - Atribui valores nulos aos atributos não definidos para estas tuplas

3. Modelo Relacional - Álgebra Relacional -

- Operação de Full Outer join

- $r \bowtie s$
- Calcula o resultado da junção de r com s
 - Adiciona ao resultado da junção
 - Tuplas das relações operando que não satisfazem à condição de junção
 - Atribui valores nulos aos atributos não definidos para estas tuplas

3. Modelo Relacional - Álgebra Relacional -

- Operação de outer-join

- left outer join (cont.)
- Considere as seguintes relações
 - Vendedor(matr, nome, salário)
 - Vendas(matr, cod-item, qtde, pr-venda)
- Listar o histórico de vendas de cada vendedor

Matr	nome	salário
9	Bárbara	4.000,00
3	Andre	3.500,00
11	Yami	1.500,00
5	Flor	1.000,00
7	Caio	2.500,00

Matr	cod-item	qtde	pr-venda
11	553	1	25,00
3	72	3	100,00
7	553	9	25,00

3. Modelo Relacional - Álgebra Relacional -

- Operação de outer-join

- Left outer join (cont.)

- Listar o histórico de vendas de cada vendedor
 - Vendedor \bowtie Vendas

Matr	nome	salário	matr	cod-item	qtdc	pr-venda
9	Bárbara	4.000,00	NULL	NULL	NULL	NULL
3	Andre	3.500,00	3	72	3	100,00
11	Yami	1.500,00	11	553	1	25,00
5	Flor	1.000,00	NULL	NULL	NULL	NULL
7	Caio	2.500,00	7	553	9	25,00

3. Modelo Relacional - Álgebra Relacional -

- Execute as seguintes consultas no Relax (Datenbanksystem)

- Retornar o nome de todos professores com as disciplinas que estão lecionando
- Retornar o código de todas disciplinas e o código de seus pré-requisitos (tabela requires)
- Retornar o nome de todas disciplinas e o nome de seus pré-requisitos

3. Modelo Relacional - Álgebra Relacional -

• Exercício

- Retornar o nome dos alunos matriculados em disciplinas, com o nome destas disciplinas e o nome do professor destas disciplinas
- No resultado deve aparecer nome do aluno, nome da disciplina e nome do professor da disciplina

```
π student.name, title, professor.name (((student ⋈attends.student_id=student.student_id attends)
                                         ⋈attends.lecture_id=lecture.lecture_id lecture) ⋈lectured_by=employee_id professor)
```

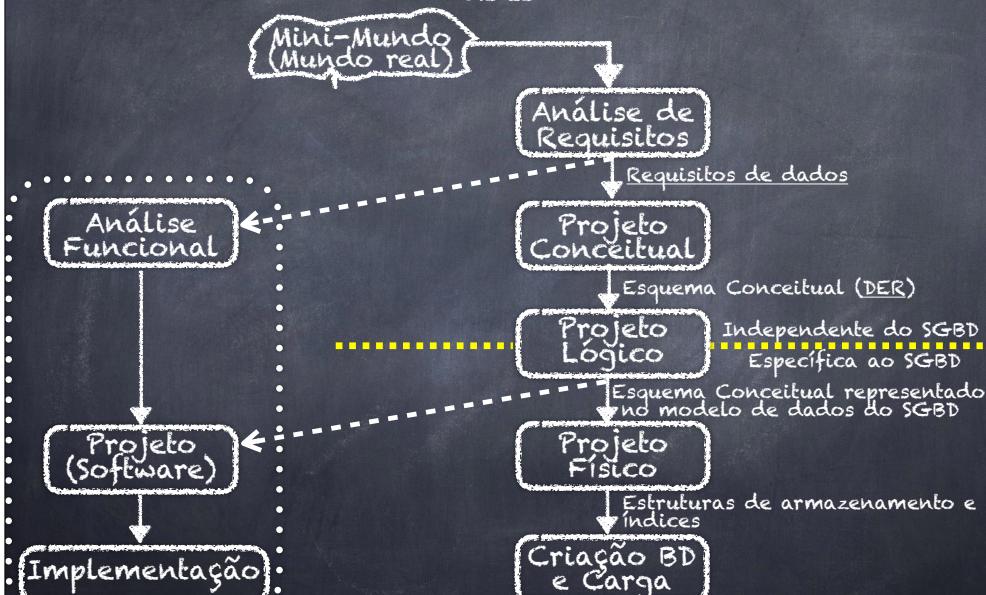
3. Modelo Relacional - Álgebra Relacional -

• Desafio (0.2 na 1a. Aval.)

- Retornar o nome de todos alunos, o nome das disciplinas em que estão matriculados, com o nome do professor das disciplinas
- No resultado, devem aparecer todos os alunos da tabela student.

4. Projeto de Bancos de Dados

- Fases -



4. Projeto de Bancos de Dados

- Fases -

• Análise (especificação) de requisitos

- Projetista de banco de dados deve realizar entrevistas com usuários prospectivos do banco de dados
- Requisitos de dados

• Projeto conceitual

- Com base nos requisitos de dados, criar um esquema conceitual para o banco de dados
- Modelo de dados conceitual (MER)
- Construir DER

• Projeto Lógico

- Com base no DER definido na fase anterior
- Criar um diagrama relacional
- Representação gráfica de um esquema relacional

4. Projeto de Bancos de Dados

- Fases -

- Projeto Físico

- Definir estruturas de armazenamento
 - Como e onde devem ser armazenadas as tabelas
 - Uma tabela em um arquivo ou várias tabelas em um único arquivo
 - Distribuir uma tabela em vários discos
- Definir caminhos de acesso
- Definir índices
 - Ordenado
 - Primário
 - Secundário
 - Hash
- Visões materializadas

4. Projeto de Bancos de Dados

- Fases -

- Implementação

- Com base no DR definido e as estruturas de armazenamento e caminhos de acesso definidos
 - Criar o banco de dados
 - Executar expressões DDL
 - Carregar os dados no BD

4. Projeto de Bancos de Dados

- Construção do Diagrama Relacional -

- Diagrama relacional (DR)

- Ferramenta gráfica utilizada para representar um esquema de banco de dados relacional

- Passo 1

- Para cada conjunto de entidades E, criar uma tabela com todos os atributos de E

- Escolher uma chave candidata para ser a chave primária da tabela

- Apenas os componentes atômicos de atributos compostos devem ser incluídos

- Notação de tabela no DR

-

Nome
Tabela

4. Projeto de Bancos de Dados

- Construção do Diagrama Relacional -

- Passo 2

- Para cada relacionamento binário 1:1 entre os conjuntos de entidades E1 e E2

- Escolher uma das tabelas, por exemplo E2, e incluir como chave estrangeira em E2 a chave primária PK da outra tabela (E1)

- Critério de escolha

- Entidade com participação total no relacionamento

- Atributos de relacionamentos devem ser incluídos na tabela com chave estrangeira

- Notação



4. Projeto de Bancos de Dados

- Construção do Diagrama Relacional -

• Passo 3

- Para cada relacionamento binário 1:N entre os conjuntos de entidades E1 e E2
- Identificar o conjunto de entidades que participa do lado N (suponha que seja E2)
- Incluir como chave estrangeira na tabela E2 a chave primária da outra tabela (E1)
- Atributos de relacionamentos devem ser incluídos na tabela com chave estrangeira

• Notação



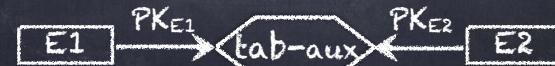
4. Projeto de Bancos de Dados

- Construção do Diagrama Relacional -

• Passo 4

- Para cada relacionamento binário N:N entre os conjuntos de entidades E1 e E2
 - Criar uma nova tabela auxiliar tab-aux para representar o relacionamento
 - Incluir como chaves estrangeiras na tabela tab-aux as chaves primárias de E1 e E2
 - Estes dois atributos comporão a chave primária de tab-aux (chave primária composta)
 - Atributos de relacionamentos devem ser incluídos na tabela tab-aux

• Notação



4. Projeto de Bancos de Dados - Construção do Diagrama Relacional -

• Passo 5

- Para relacionamento de grau maior que 2
 - Criar uma nova tabela auxiliar tab-aux para representar o relacionamento
 - Incluir como chaves estrangeiras na tabela tab-aux as chaves primárias das tabelas que participam do relacionamento
 - Estes atributos comporão a chave primária de tab-aux

• Passo 6

- Para cada conjunto de entidades fracas F
 - Cria uma tabela TFr com todos os atributos de F
 - Incluir como chave estrangeira de TFr a chave primária da tabela correspondentes ao conjunto de entidades fortes R
 - A chave primária de TFr será composta pela chave parcial de F e chave primária de R

4. Projeto de Bancos de Dados - Construção do Diagrama Relacional -

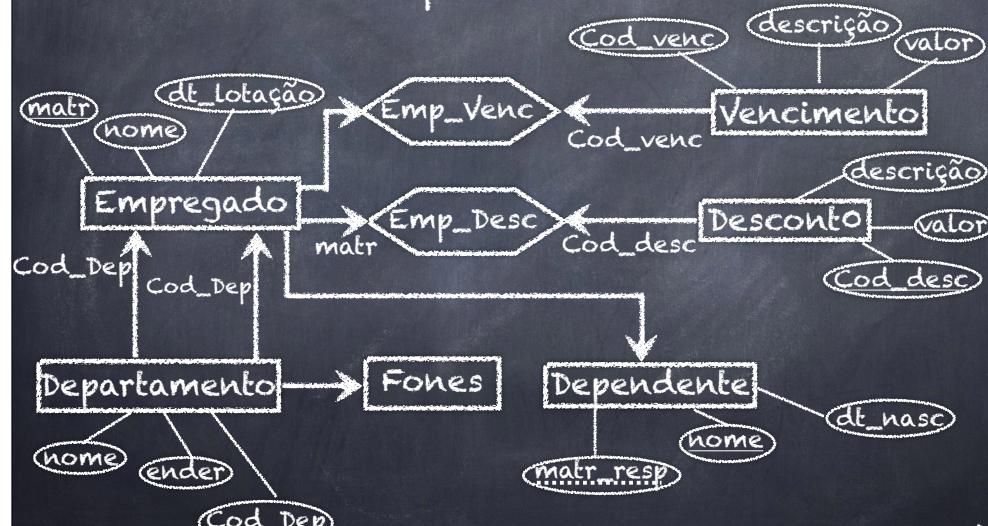
• Passo 7

- Para cada atributo multivalorado A de um conjunto de entidades E1
 - Criar uma tabela T com o atributo A
 - Incluir como chave estrangeira em T a chave primária de E1
 - A chave primária de T será composta do atributo A mais a chave primária de E1

4. Projeto de Bancos de Dados

- Construção do Diagrama Relacional -

- Construa o DR para o DER do slide 42

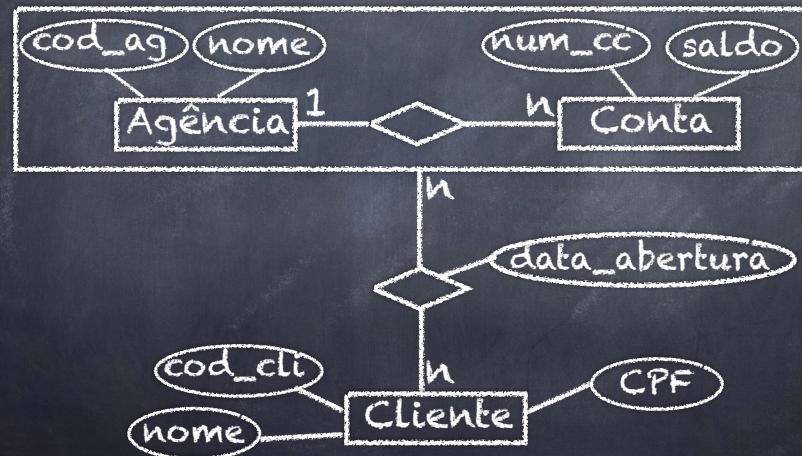


4. Projeto de Bancos de Dados

- Construção do Diagrama Relacional -

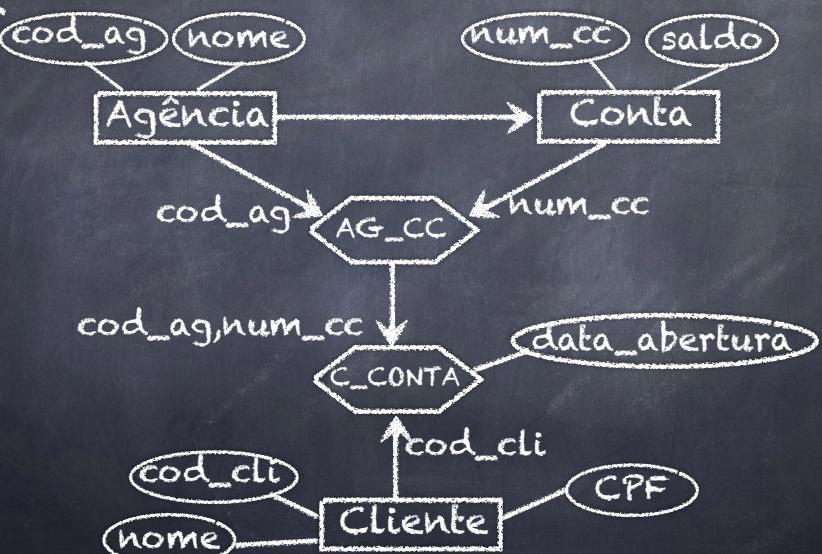
- Exercício

- Construa o DR para o seguinte DER



4. Projeto de Bancos de Dados - Construção do Diagrama Relacional -

- DR



5. SQL - Histórico -

- Structured Query Language - SQL

- Desenvolvida pela IBM
- Structured English Query Language - Sequel
- Linguagem de consulta para o sistema R
- Especificar consultas de forma interativa
- Consultas ad hoc
- Dois conjuntos de operações
- Data Description Language (DDL)
- Operações para a definição do banco de dados
- Data Manipulation Language (DML)
- Operações para o acesso e manipulação dos dados

5. SQL

- Histórico -

- Padrão

- ANSI
- SQL 86
- SQL 89
- SQL 92
- SQL 99
 - Propriedade de SBDs objeto-relacional

5. SQL

- Operações da DDL -

- Esquema de um banco de dados relacional
 - Conjunto de esquemas de relação mais um conjunto de restrições de integridade IC
- Expressões DDL do SQL permitem especificar
 - Esquema de relações (tabelas)
 - Domínio de valores associados a cada atributo
 - Restrições de integridade
 - Conjunto de índices a serem mantidos para cada relação
 - Estrutura de armazenamento físico de cada relação em disco
 - Autorização de acesso para cada relação

5. SQL

- Operações da DDL -

• Criando um banco de dados

CREATE DATABASE nome-BD

• Abrindo um banco de dados

USE nome-BD

5. SQL

- Operações da DDL -

• Criando tabelas

CREATE TABLE nome-tabela

(nome-coluna tipo-de-dados [not null],
 [nome-coluna tipo-de-dados [not null] ...],

Definição de
 restrições

[CONSTRAINT nome-restrição]

UNIQUE nome-coluna

| PRIMARY KEY(nome-coluna {, nome-coluna})

| FOREIGN KEY (nome-coluna {, nome-coluna})

 REFERENCES nome-tabela

 [ON DELETE CASCADE |

 SET NULL | NO ACTION],

 [ON UPDATE CASCADE],

 | CHECK (predicado)

)

5. SQL

- Operações da DDL -

• Criando tabelas (cont.)

- Alguns tipos de dados suportados pelo SQL 92
 - char(n)
 - string de caracteres de tamanho fixo n
 - varchar(n)
 - string de caracteres de tamanho variável (máximo n)
 - integer
 - smallint
 - decimal(p,d)
 - numérico com p dígitos
 - Dos p dígitos, d dígitos representam casas decimais após a vírgula
 - real
 - numérico ponto flutuante

5. SQL

- Operações da DDL -

• Criando tabelas (cont.)

- Alguns tipos de dados suportados pelo SQL 92
 - date
 - data de calendário
 - datetime
 - data e hora
 - vários formatos
 - YYYY-MM-DD HH:MM:SS
 - Função getdate()
 - Retorna data e hora corrente



5. SQL

- Operações da DDL -

• Atividade de laboratório

- Construir o Diagrama Relacional para o banco de dados Lojas
- Utilize o DER já construído
- Criar o banco de dados Lojas no SQL Server



5. SQL

- Operações da DDL -

• Removendo tabelas

- Estrutura básica
- `DROP TABLE nome-tabela [CASCADE | RESTRICT]`
- Remove as tuplas da tabela e sua definição do catálogo
- CASCADE remove as restrições do tipo foreign key das tabelas que referenciam a tabela removida
- Oracle, Postgres, DB2
- SQL Server
- `DROP TABLE nome-tabela`

5. SQL

- Operações da DDL -

- Alterando tabelas

ALTER TABLE nome-tabela

[ADD nome-coluna tipo-dado]

[ALTER COLUMN nome-coluna tipo-dado [not null]]

[DROP column nome-coluna]

[ADD CONSTRAINT nome-restrição]

[DROP CONSTRAINT nome-restrição]

[DROP PRIMARY KEY]

[repetir ADD ou DROP em qualquer ordem]

5. SQL

- Operações da DDL -

- Alterar o BD Lojas, com as seguintes restrições

• O salário de cada vendedor deve ser maior que 1200

• Ao alterar o código de uma filial, o valor do atributo codfil para todos empregados da filial deve ser automaticamente alterado

• Incluir a coluna cnpj em Fornecedores

• Não podem existir valores de cnpj repetidos

5. SQL

- Operações da DML -

- Incluindo tuplas em uma tabela

- Cláusula Insert

- Sintaxe

```
INSERT [INTO] nome_tabela  
[(lista_de_colunas)]  
{VALUES ( {DEFAULT | NULL | expr }[,...n] )  
| subquery }
```

5. SQL

- Operações da DML -

- Atualizando tuplas de uma tabela

- Cláusula Update

- Sintaxe

```
UPDATE nome_tabela  
SET nome_coluna = {expr | NULL | (subquery) }  
{, nome_coluna = {expr | NULL | (subquery) }}  
WHERE predicado
```

- Removendo tuplas de uma tabela

- Cláusula Delete

- Sintaxe

```
DELETE FROM nome_tabela  
WHERE predicado
```

5. SQL

- Operações da DML -

• Consultas simples sobre o banco de dados

```
SELECT [ALL | DISTINCT] {*} |  
    expr [[AS] c_alias]  
    {, expr [[AS] c_alias] ... }  
FROM nome-tabela [[AS] alias]  
    {, nome-tabela [[AS] alias] ... }  
WHERE predicado
```

5. SQL

- Operações da DML -

• Consultas simples sobre o banco de dados

• SELECT

- Representa a operação de projeção da álgebra relacional
 - ALL (default)
 - Retorna todas as tuplas, inclusive repetidas
 - DISTINCT
 - Retorna apenas tuplas não repetidas
 - *
 - Retorna todos os atributos da(s) tabela(s)
 - expr
 - Atributo ou expressão matemática envolvendo atributos das tabelas
 - salario
 - salario*1.40

5. SQL

- Operações da DML -

- Consultas simples sobre o banco de dados
- FROM
 - Representa a operação de produto cartesiano da álgebra relacional
 - Tabelas referenciadas
- WHERE
 - Corresponde a operação de seleção da álgebra relacional
 - Predicado
 - Condição da seleção

5. SQL

- Operações da DML -

◦ Exemplo

- Listar o nome e o salário de todos os vendedores, lotados na filial 01. No resultado, os nomes das colunas devem ser Nome do Vendedor e Salário Líquido

```
Select nome as 'Nome do Vendedor', salario  
as 'Salário Líquido'  
from vendedores  
where codfil=01
```

- Listar o código dos itens que tiveram vendas

```
Select distinct coditem  
from historico
```



5. SQL

- Operações da DML -

- Predicados com operações sobre strings
 - Identificação de padrão
 - %
 - Casa com qualquer substring
 - _
 - Casa com qualquer caracter
 - Operador
 - Like



5. SQL

- Operações da DML -

- Predicados com operações sobre strings
 - Exemplos (na cláusula **where**)
 - nome Like 'inf%'
 - Retorna strings que iniciam pelo substring inf
 - nome Like '%si_'
 - Retorna strings que contenham 'si' como substring e terminem com um caracter qualquer após 'si'
 - Listar todos vendedores com sobrenome 'brayner'
`Select nome from vendedores
where nome Like '%brayner%'`

5. SQL

- Operações da DML -

- Consultas simples sobre o banco de dados
 - **Junção**

```
select v.nome as 'Nome Vendedor',  

       f.nome as 'Nome Filial'  

  from vendedores v, filiais f  

 where codfil=cod
```

projeção

cartesiano

seleção

5. SQL

- Operações da DML -

• Exercícios

- Listar vendedores com salários maior que 1000 e menor que 2000
- Listar nome dos vendedores com o nome de sua filial de lotação e uma simulação de seu salário com um aumento de 15%
- Listar nome do vendedor e o nome de sua filial de lotação, para todos vendedores que ganham mais que 4500
- Listar nome dos fornecedores que tiveram itens vendidos por vendedores que ganham mais que 4500

5. SQL

- Operações da DML -

• Exercícios

- Listar nome das filiais que venderam itens de fornecedores localizados na cidade de São Paulo

5. SQL

- Operações da DML -

• Operador de união (tabela compatíveis)

- UNION
 - União de duas tabelas (resultados de duas consultas)
 - Sem repetições
- UNION ALL
 - União de duas relações, com repetições
- Exemplo
 - Liste o nome e cpf de todos os vendedores e dependentes existentes na empresa

```
select nome, cpf from vendedores UNION
select nome, cpf from Dependente
```

5. SQL

- Operações da DML -

- Consultas com o operador de interseção
 - INTERSECT
 - Interseção entre duas tabelas
 - Sem repetições
 - INTERSECT ALL
 - Interseção entre duas tabelas, com repetições
- Consultas com o operador de diferença
 - EXCEPT
 - Diferença entre duas tabelas
 - Sem repetições
 - EXCEPT ALL
 - Diferença entre duas relações, com repetições

5. SQL

- Operações da DML -

• Exercícios

- Listar matrícula dos vendedores que não possuem vendas

```
Select matr From vendedores  
EXCEPT  
Select matrvend From historico
```
- Listar matrícula dos vendedores que tiveram vendas

```
Select matr From vendedores  
INTERSECT  
Select matrvend From historico
```

5. SQL

- Operações da DML -

• Consultas ordenadas

- ORDER BY coluna-resultado [ASC | DESC]
{, coluna-resultado [ASC | DESC] ...};
- Listar vendedores ordenados por salário na ordem decrescente e por nome na ordem crescente

```
Select *  
from vendedores  
order by salario desc, nome
```

5. SQL

- Operações da DML -

• Funções Agregadas

- Funções aplicadas sobre uma coleção de valores (colunas) do banco de dados, retornando um escalar
- sum
 - Retorna o somatório dos valores de uma coleção
- avg
 - Retorna a média dos valores de uma coleção

5. SQL

- Operações da DML -

• Funções Agregadas

- max

- Retorna o maior valor de uma coleção de valores

- min

- Retorna o menor valor de uma coleção de valores

- var/varp

- Retorna a variância de uma amostra/população

- stdev/stdevp

- Retorna o desvio padrão de uma amostra/população

5. SQL

- Operações da DML -

• Funções Agregadas

- min

- Retorna o menor valor de uma coleção

- count

- Retorna o número de elementos de uma coleção

- Sintaxe

- nome-da-função (ALL | DISTINCT nome-coluna) | count(*)

- Não podem ser utilizados na cláusula WHERE

- Não é permitido o uso de função composta

5. SQL - Operações da DML -

• Exercícios

- Encontre o número de vendedores lotados na filial Recife

```
select count(*) from vendedores e, filiais d  
where e.codfil=d.cod and d.cid like 'Recife'
```

- Encontre o montante da folha de pagamento da empresa

```
select sum(salario) from vendedores
```

- Encontre o salário médio pago pela empresa

```
select cast (avg(distinct salario) as decimal(6,2))  
from vendedores
```

5. SQL - Operações da DML -

• Cláusula GROUP BY

- Agrupar tuplas por valores de atributos
 - um ou mais atributos
- Aplicar funções agregadas a diferentes grupos de tuplas
- Exemplo
 - Quantidade de vendedores que cada filial tem
 - Agrupar tuplas de vendedores por código da filial
 - Um grupo cada valor de codfil
 - Aplicar a função count a cada grupo

5. SQL

- Operações da DML -

• Exemplo Group by

• Listar a quantidade de vendedores por filial

```
select codfil as 'Filial', count(*) as 'Número de
Vendedores'
from vendedores
group by codfil
```

A função count é aplicada para o conjunto de tuplas de cada grupo, no caso, codfil

Filial	Número de Vendedores
1	4
2	2

5. SQL

- Operações da DML -

• Atenção

• Todas colunas que aparecem na cláusula select têm que aparecer na cláusula group by

• Exceto os argumentos das funções agregadas

• Exemplo de sintaxe incorreta

```
select codfil, matr, count(*) from vendedores
group by codfil
```

5. SQL

- Operações da DML -

• Exercício

- Listar maiores e menores salários de cada filial

```
select f.nome, max(v.salario) as 'Maior Salario',  
       min(v.salario) as 'Menor_Salario'  
  from filial f, vendedores v  
 where cod=codfil  
 group by codfil, f.nome
```

5. SQL

- Operações da DML -

• Exercício

- Mostrar a quantidade de itens vendidos por vendedor e por item, ordenado pelo nome do vendedor

```
select v.nome as 'Nome Vendedor', e.ref as 'Item',  
       sum(h.qtd) as 'Unidades Vendidas', count(*)  
           as 'Número de Vendas'  
  from estoque e, vendedores v, historico h  
 where e.cod=h.coditem and v.matr=h.matrvend  
 group by matrvend, v.nome, coditem, e.ref  
 order by v.nome
```

5. SQL

- Operações da DML -

 • Cláusula **having**

- Filtro de grupos
- Selecionar grupos

• Exemplo

- Listar nome das filiais com média salarial maior que 2000, em ordem decrescente de média salarial

```
select f.nome, avg(salario)
from filial f, vendedores v
where cod=codfil
group by f.nome
having avg(v.salario)>2000
order by avg(salario) desc /* order by 2 desc */
```

5. SQL

- Operações da DML -

 • **Atenção**

- Consulta com where e having
- Predicado da cláusula where é avaliado primeiramente
 - Tuplas que satisfazem o predicado são agrupadas pelo group by
 - **Filtro de tuplas**
- Predicado da cláusula having é avaliado
 - Grupos que satisfazem o predicado aparecem no resultado
 - **Filtro de grupos**

5. SQL

- Operações da DML -

• Exercícios

- Listar nome e média salarial das filiais que possuem mais de 6 vendedores

```
select d.nome, avg(e.salario) as 'Média Salarial'  
from filial d, vendedores e  
where d.cod=e.codfilial  
group by d.nome  
having count(matr)>=6
```

5. SQL

- Operações da DML -

• Exercícios

- Listar nome e quantidade de vendedores das filiais cuja média salarial é maior que 5000

```
select f.nome, count(*) as 'Número Vendedores'  
from filial f, vendedores v  
where cod=codfilial  
group by f.nome  
having avg(salario)>=5000
```

5. SQL

- Operações da DML -

• Exercícios

- Gerar relatório com nome de cada vendedor e quantidade de vendas efetuadas por ele, em ordem decrescente de quantidade de vendas. Só devem aparecer no relatório vendedores com volume de vendas superior a 1,000,00

```
select v.nome, count(*) as 'Total de Vendas'  
from Vendedores v, historico h  
where v.matr=h.matrvend  
group by matr,v.nome  
having sum(h.qtd* h.prven) > 1000  
order by 2 desc
```

5. SQL

- Operações da DML -

• Checando valores nulos

- Predicado IS NULL
- Exemplo
 - select * from Empregado
 - where dt-nasc is null
- Atribuindo um valor a atributos com conteúdo NULL
 - ISNULL(nome_atributo,0)

5. SQL

- Operações da DML -

- Consulta SQL aninhada (sub-consulta)
- Consulta SQL embutida dentro de outra consulta SQL
- Comporta-se de forma semelhante a uma sub-rotina (função ou método) dentro de um programa

5. SQL

- Operações da DML -

- Consulta SQL aninhada (subconsulta)

- Exemplo (BD Lojas)

- Listar vendedores com salário maior que a média salarial da empresa

```
select v.nome  
from vendedores v  
where salário > (select avg(salário) from  
vendedores)
```

subconsulta retorna um valor
para a consulta mais externa

5. SQL

- Operações da DML -

• Consulta SQL aninhada (cont.)

- Listar o primeiro e segundo maiores salários da empresa

Desafio: 0.2 na 1a. Aval

```
select max(salário) from Vendedores
```

```
union
```

```
select max(salario) from Vendedores
where salário <> (select max(salário)
from Vendedores)
```

```
select max(salário) as 'Maior salário',
(select max(salario) from Vendedores
where salário <> (select max(salário)
from Vendedores)) as 'Segundo Maior
salário' from Vendedores
```

5. SQL

- Operações da DML -

• Consulta SQL aninhada (cont.)

- Listar nome das filiais com média salarial maior que a média salarial da empresa

```
select f.nome, avg(salario)
from filiais f, vendedores v
where f.cod=v.codfil
group by codfil, f.nome
having avg(v.salário) > (select avg(salário)
from vendedores)
```



5. SQL - Operações da DML -

• Predicado IN

- Verifica a pertinência de elementos em um conjunto

• Sintaxe

- `expr [NOT] IN (subconsulta) | expr [NOT] IN (val [val ...])`

• Exemplo

```
select nome  
from vendedores  
where matr in (1,5,8,9)
```



5. SQL

- Operações da DML -

- Listar os vendedores lotados nas filiais localizadas na cidade de Fortaleza

```
select nome from vendedores  
where codfil in (select cod from filiais  
where cidade='Fortaleza')
```

5. SQL

- Operações da DML -

• Consulta correlacionada

- Listar empregados que possuem salário maior que a média salarial de sua filial

```
select nome from vendedores
where salário > (select avg(salário)
from vendedores where codfil=?????)
```

A subconsulta precisa do valor do atributo codfil de cada tupla da consulta mais externa, como parâmetro de entrada

5. SQL

- Operações da DML -

• Consulta correlacionada

- Listar empregados que possuem salário maior que a média salarial de suas filiais

```
select nome from vendedores v
where v.salário > (select avg(salário)
from vendedores vend
where vend.codfil = v.codfil)
```

Executada para cada tupla da consulta mais externa

Variável de correlação
 Variável da consulta mais externa utilizada pela consulta mais interna

5. SQL

- Operações da DML -

• Predicados SOME, ANY e ALL

- Implementam os quantificadores existencial e universal
- Listar vendedores que ganham salários maior ou igual a média salarial de pelo menos uma filial

• Sintaxe

- $\text{expr} \theta \{ \text{SOME} \mid \text{ANY} \mid \text{ALL} \}$ (subconsulta)
- $\theta \in \{ <, \leq, \geq, =, \neq \}$

5. SQL

- Operações da DML -

• Predicados SOME, ANY e ALL (cont.)

- θSOME (subconsulta) e θANY (subconsulta)
 - Retornam verdade se e somente se
 - Para pelo menos um elemento s retornado pela subconsulta, $\langle \text{expr} \theta s \rangle$ é verdade
 - Implementam o quantificador existencial
 - São equivalentes
- Listar empregados com salários maior ou igual a média salarial de pelo menos uma filial


```
select nome from vendedores
      where salário >=some (select avg(salário) from
      vendedores group by codfil)
```

5. SQL

- Operações da DML -

- Predicados SOME, ANY e ALL (cont.)
- θ ALL (subconsulta)
 - Retorna verdade se e somente se,
 - Para todo elemento s retornado pela subconsulta, $\langle\text{expr } \theta s\rangle$ é verdade
 - Implementa o quantificador universal
 - Listar vendedores com salários maior ou igual a média salarial de cada filial


```
select nome from vendedores
  where salário >=all (select avg(salário) from
  vendedores group by lotação)
```

5. SQL

- Operações da DML -

- Predicados SOME, ANY e ALL (cont.)
- Listar filial com maior média salarial
 - Não é permitido função agregada composta


```
select f.nome
  from vendedores v, filiais f where f.cod=v.codfil
  group by f.nome
  having avg(salário) >=all (select avg(salário)
  from vendedores group by codfil)
```

5. SQL - Operações da DML -

• Predicado EXISTS

- Sintaxe
- [NOT] EXISTS (subconsulta)
- EXISTS (subconsulta)
- Retorna verdade se e somente se
 - O conjunto retornado por subconsulta não é vazio
- NOT EXISTS (subconsulta)
- Retorna verdade se e somente se
 - O conjunto retornado por subconsulta é vazio

5. SQL - Operações da DML -

• Exercício

- Listar nome do fornecedor, para o qual não houve vendas de seus produtos

```
select f.nome
from fornecedores f
where not exists (select * from estoque e, historico
where f.cod=codfor and e.cod=coditem)
```

5. SQL

- Operações da DML -

• Desafio

- Listar nome da filial com empregados ganhando duas vezes mais que a média salarial da filial

```
select f.nome
from filial f
where exists (select * from vendedores v
where v.codfil=f.cod and
salário > (2*(select avg(salário) from vendedores
where codfil=v.codfil)))
```

5. SQL

- Operações da DML -

• Predicado Between

- Sintaxe
 - $expr1 [NOT] BETWEEN expr2 and expr3$
- Exemplo
 - $matr \text{ between } 2 \text{ and } 10 \Leftrightarrow matr \geq 2 \text{ and } matr \leq 10$

5. SQL

- Operações da DML -

◦ Formas de Junção em SQL

- Listar nome dos departamentos com empregados com salários maior que 7000

```

  select d.nome ← projeção      produto
        from Empregado e, Departamento d ← cartesiano
        where d.cod_depart=e.lotação ←
              and salario>7000           seleção
  
```

$$\sigma_{d.cod_depart=e.lotação}(\text{Departamento} \times \text{Empregado})$$
 \Downarrow

$$\text{Departamento} \bowtie_{d.cod_depart=e.lotação} \text{Empregado}$$

5. SQL

- Operações da DML -

◦ Sintaxe da cláusula FROM

```

  [ FROM {<tabela_fonte>} [,...n] ]
  <tabela_fonte> ::=

    nome_tabela [ [AS] qualificador ] [ WITH ( <table_hint>
    [,...n] ) ]
    | nome_tabela [ [AS] qualificador ] [ (column_alias [,...n] ) ]
    | (subquery) [AS] qualificador [ (column_alias [,...n] ) ]
    | nome_visão [AS] qualificador [ (column_alias [,...n] ) ]
    | <tabela_fonte> <tipo_junção> <tabela_fonte> ON
      <condição_junção>

  <tipo_junção> ::=

    [ INNER | { { LEFT | RIGHT | FULL } [OUTER] } ] [ <join_hint> ]
    JOIN
  
```

5. SQL

- Operações da DML -

◦ Formas de Junção em SQL (cont.)

◦ Sintaxe da cláusula FROM (cont.)

◦ WITH (<table_hint> [,...n])])

◦ Especifica estratégias (dicas) para o otimizador de consultas

- Índices

- tipo e granularidade de bloqueio (lock)

◦ (column_alias [,...n])

◦ Especifica alias para colunas retornadas de

- uma tabela ou subconsulta

◦ Um alias para cada coluna especificada na lista do select da subconsulta

◦ join_hint

- Indica qual o algoritmo de junção deve ser executado

- (Nested) Loop join, merge join ou hash join (SQL Server 2005)

5. SQL

- Operações da DML -

◦ Formas de Junção em SQL (cont.)

◦ Sintaxe da cláusula FROM (cont.)

◦ Tipos de junção

◦ Junção theta

◦ INNER JOIN

◦ Junção externa à esquerda

◦ LEFT OUTER JOIN

◦ Junção externa à direita

◦ RIGHT OUTER JOIN

◦ Junção externa completa

◦ FULL OUTER JOIN

5. SQL

- Operações da DML -

• Junção theta

- Listar nome dos vendedores com o nome do respectivo filial

```
select e.nome, d.nome
from vendedores e inner join filiais d
on e.codfil=d.cod
```

5. SQL

- Operações da DML -

• Junção theta (INNER JOIN)

- Para os vendedores que têm salário maior que 900, Listar nome com o nome da respectiva filial

```
select empregado, d.nome as 'Filial'
from filiais d inner join
(select nome, codfil from vendedores where
salário>900) e (empregado,filial)
on cod=filial
```

alias para colunas nome e
 lotação (resultado da subconsulta)



5. SQL

- Operações da DML -

• LEFT OUTER JOIN

- Calcula o resultado da junção
- Adiciona ao resultado tuplas da relação à esquerda, que não satisfazem a condição de junção
- Atribui valores nulos aos atributos não definidos para estas duplas
- Listar o histórico de vendas de cada vendedor (BD Lojas)

```
select v.nome, e.referência,  
d.qtde,d.qtde*d.pr_venda  
from Vendedores v left outer join (histórico  
d inner join Estoque e on d.cod_item=e.cod_  
item) on v.matr=d.matr
```



5. SQL

- Operações da DML -

• Formas de Junção em SQL (cont.)

• RIGHT OUTER JOIN

- Para cada empregado, listar nome dele, nome do departamento de lotação e nome dos dependentes

```
select e.nome, d.nome, p.nome  
from Departamento d inner join  
(Dependente p right outer join Empregado e on  
p.matr_resp=e.matr) on d.cod_dep=e.lotação
```

5. SQL

- Operações da DML -

• Formas de Junção em SQL (cont.)

• FULL OUTER JOIN

• Calcula o resultado da junção

• Adiciona ao resultado da junção

• Tuplas das relações envolvidas na junção que não satisfazem a condição de junção

• Atribui valores nulos aos atributos não definidos para estas tuplas

5. SQL

- Operações da DML -

• Exercícios

- Listar referência, quantidade e preço de compra de itens sem movimentação de saída
- Ordenado de forma decrescente, por preço de compra

```
select ref, qtd, prcom from estoque e
where not exists (select cod from estoque inner
join historico on cod=coditem where cod=e.cod)
order by prcom
```

```
select ref, e.qtd, prcom from estoque e left
outer join historico on e.cod=coditem
where coditem is NULL
order by prcom
```

5. SQL - Operações da DML -

• Exercícios

- Listar referência, preço de compra e número de vendas de cada item de estoque
- Ordenado de forma decrescente, por número de vendas

```
select ref, prcom, count(coditem) from estoque e
left outer join historico on cod=coditem
group by coditem, ref, prcom
order by prcom desc
```

5. SQL - Visões -

• Acesso a um banco de dados

- Requer conhecimento do esquema
- Indesejável
 - Para usuários inexperientes
 - Desenvolvedores de aplicativos que acessam o BD
- Por questões de segurança e privacidade
 - Grupos de usuários devem ter acesso a dados de interesse
 - O acesso a todo o banco de dados é perigoso



UFC

CC

DC

5. SQL - Visões -

- Janelas sobre o banco de dados
 - Cada janela mostra parte do banco de dados
 - Diferentes visões sobre o banco de dados
- Visões (views)
 - Definidas sobre tabelas do banco de dados
 - Tabelas base
- Tipos de visões
 - Virtual
 - Materializada



UFC

CC

DC

5. SQL - Visões -

- Visão virtual
 - A definição da visão é armazenada
 - Dados da visão não são persistentes
- Sempre que referenciada
 - Os dados são materializados
 - Custo praticamente igual a cada materialização
- Quanto ao acesso
 - Somente leitura
 - Visões que só permitem acesso de leitura
 - Permitem atualização
 - Visões que permitem atualizações nas tabelas base

5. SQL - Visões -

- Visão materializada
 - Dados e definição são persistentes
 - Problema de atualização dos dados da visão
 - Sempre que há uma atualização nas tabelas base da visão
 - Recalculada
 - Atualizada
 - Com intervenção humana
 - Automática
 - Reduz custos de materialização de resultado
 - Visões somente para leitura
 - Aplicações
 - Implementação Data Warehouse
 - Integração de fontes de dados heterogêneas

5. SQL - Visões -

- Definição de visões (SQL Server)

```
CREATE VIEW nome_da_visão [(nome_coluna {, nome_coluna ...})]
[WITH <propriedade_da_visão> [ ,...n ] ]
AS expressão_SQL [WITH CHECK OPTION]
<propriedade_da_visão> ::= {[SCHEMABINDING]
[ENCRYPTION] [VIEW_METADATA] }
```
- WITH CHECK OPTION
 - Atualizações (INSERT ou UPDATE) na tabela base através da visão só serão permitidas se
 - Resultam em tuplas visíveis para a visão



UFC
CC
DC

5. SQL - Visões -

• SCHEMABINDING

- Associa a visão ao esquema das tabelas base da visão
- Não permite que as tabelas base sejam alteradas (ou removidas), caso a definição da visão seja afetada
 - Remover coluna da tabela base que também aparece na definição da visão
- Para permitir alteração das tabelas base
 - Alterar definição da visão ou remover a visão
- Obrigatório para visões materializadas
- Na expressão SQL da definição da visão
 - Nomes de tabelas, visões, funções têm que ser na forma <schema.objeto>
 - dbo.empregado



UFC
CC
DC

5. SQL - Visões -

• ENCRYPTION

- Criptografa a definição da visão em sys.syscomments

• VIEW_METADATA

- Não permite a visualização do nome das tabelas base da visão



5. SQL - Visões -

• Exemplos

- Definir uma visão, que retorna o nome da filial em que o vendedor está lotado e o nome do vendedor

```
create view V1 (nome_filial, nome_vendedor)
as
select f.nome, v.nome
from filiais f inner join vendedores v
on cod=codfil
```

```
select * from V1
```



5. SQL - Visões -

• Exemplos

- Definir uma visão, que retorna o nome da filial e a quantidade de vendedores lotados na filial

```
create view V2 (nome_filial, num_vend)
as
select f.nome,
(select count(*) from vendedores where
cod=codfil)
from filiais f
```

```
select * from V2
```



5. SQL - Visões -

• Exemplos

- Definir visão que retorna matrícula, código da filial e salário dos vendedores que ganham menos que 2000

create view V3 (matrícula, filial, salário)
as
select nome, codfil, salario from vendedores
where salário<2000 with check option

select * from V3



5. SQL - Visões -

- Acessando o banco de dados através da visão V3

matrícula	filial	salário
13	2	1500
99	3	1100

- Atualizando o banco de dados através de visões

- update v3 set salário=salário+400
 - select * from V3

- update v3 set salário=salário+200

Erro!!

tuplas a serem alteradas vão deixar de ser
visíveis para V3



5. SQL - Visões -

• Visão materializada - SQL Server

- Definir uma visão, que retorna o nome da filial e a quantidade de vendedores lotados na filial

```
Create view V4 (filial, num_vend)
with schemabinding
as
```

```
select f.nome, count_big(*)
from dbo.vendedores inner join dbo.filiais f
on cod=codfil
group by f.nome
```

```
Create unique clustered index I_V4
on V4 (filial)
```



5. SQL - Visões -

- Visões que permitem atualizações apresentam as seguintes restrições na subconsulta
 - Não estão especificadas as cláusulas group by e having
 - A palavra reservada distinct não está especificada
 - A cláusula where não contém subconsulta que referencia qualquer tabela na cláusula from diretamente ou indiretamente (via visões)
 - Todas as colunas da subconsulta são colunas simples
 - Não são permitidas colunas do tipo avg(salário) ou expressões aritméticas

6. Restrições de Integridade - Triggers -

- Primeiras gerações de SGBDs
 - Armazenavam dados de forma passiva
 - Nenhuma ação era executada pelo SGBD
 - Ações sobre dados só eram executadas quando especificadas por transações
 - Muitas restrições de integridade deveriam ser mantidas por aplicativos
 - Total de salários por departamento deve ser menor que K
 - Ao se atualizar salários
 - Garantir a restrição de total de salários por departamento
 - Para cada novo valor K'
 - Alterar todos os programas que atualizam salários
 - Restrições que envolvam mais de uma tabela
 - Um dependente não pode estar associado a mais de um empregado

6. Restrições de Integridade - Triggers -

- Regras ativas
 - Mecanismos que especificam ações a serem executadas quando certas condições são satisfeitas
 - Definidas inicialmente para sistemas de bancos de dados ativos
 - Armazenam dados e regras
 - Disparam e executam as ações definidas nas regras
- Regras E[C]\A
 - Na ocorrência de um EVENTO
 - Se CONDIÇÃO
 - Então execute AÇÃO
- Quando um evento ocorre
 - Uma ou mais ações podem ser disparadas, caso a condição seja satisfeita
- Um evento pode representar uma atualização sobre dados do BD

6. Restrições de Integridade - Triggers -

- Triggers

- Mecanismos de regras ativas implementados por SGBDs Relacionais existentes no mercado
- São armazenados no banco de dados
- Eventos
 - Insert, Delete, Update
- Execução das ações
 - Antes ou depois da ocorrência do evento
 - SQL Server
 - Depois
 - A condição de execução seja satisfeita
 - Definida dentro de um trigger
- Definidos através de uma expressão DLL
 - Create trigger

6. Restrições de Integridade - Triggers -

- Sintaxe da cláusula CREATE TRIGGER (SQL Server)

```
CREATE TRIGGER nome_trigger ON nome_tabela
{
  {FOR | AFTER | INSTEADOF { [DELETE] [,] [INSERT] [,]
  [UPDATE] } [NOT FOR REPLICATION]
  AS
  sentença [ {sentença ... } ] }

  {FOR { [INSERT] [,] [UPDATE] } [NOT FOR REPLICATION]
  AS
  { IF UPDATE (nome_coluna)
  [{AND | OR} UPDATE (nome_coluna)]
  [ { {AND | OR} UPDATE (nome_coluna) ... } ]
  sentença [ {sentença ... } ] }
}
```

6. Restrições de Integridade - Triggers -

- **FOR**

- Gatilho é disparado junto da ação

- **AFTER**

- Disparo se dá somente após a ação que o gerou seja concluída

- **INSTEAD OF**

- faz com que o trigger seja executado no lugar da ação que o gerou.

6. Restrições de Integridade - Triggers -

- **NOT FOR REPLICATION**

- Trigger não deve ser disparado quando o mecanismo de replicação altera a tabela envolvida no trigger

- **Sentença**

- Especifica a condição e ação do trigger

- Sentença SQL

- Update

- Sentença da linguagem proprietária do SGBD

- T-SQL (SQL Server)

- PL/SQL (Oracle)

- **UPDATE(nome_coluna)**

- Indica que a ação do trigger deve ser disparada para uma atualização sobre a coluna nome_coluna

6. Restrições de Integridade - Triggers -

• Observação importante

- O SQL Server implementa as tabelas temporárias
- Inserted
 - Armazena cópias das tuplas afetadas pelos comandos `insert` e `update`
- Deleted
 - Armazena cópias das tuplas afetadas pelo comando `delete`

6. Restrições de Integridade - Triggers -

• Exemplos

- Garantir que menor salário de vendedores será 1000

```
create trigger tr1 on Vendedores
  for insert,update
  as
    if (select salario from inserted)<1000
      begin
        raiserror('salario invalido', 16,1)
        rollback transaction
      end
```

6. Restrições de Integridade - Triggers -

• Exercícios Desafios (0,25 cada na 2a. prova)

- Especificar as seguintes restrições de integridade no SQL Server
 - O volume de salários pagos em uma filial não pode ser maior que 15000
 - Salário de um vendedor não pode ser maior que duas vezes a média salarial da filial em que trabalha