UFLA - Universidade Federal de Lavras
Professora Olinda Paes 2001 / 1
Banco de Dados – COM 111

Síntese do Conteúdo:

1. Conceitos básicos
   1.1. Arquitetura de bancos de dados
2. Modelos de dados
3. Modelagem e projetos de bancos de dados
   3.1. Modelagem usando o modelo ER
   3.2. Modelagem usando o modelo relacional
   3.3. Mapeamento ER / Relacional
   3.4. Normalização
4. Linguagens de consulta
5. Sistemas de bancos de dados
   5.1. Segurança
   5.2. Integridade
   5.3. Concorrência
   5.4. Recuperação após falhas
   5.5. Gerenciamento de transações
6. Bancos de dados orientados a objetos
7. Sistemas avançados de bancos de dados

1. Conceitos básicos

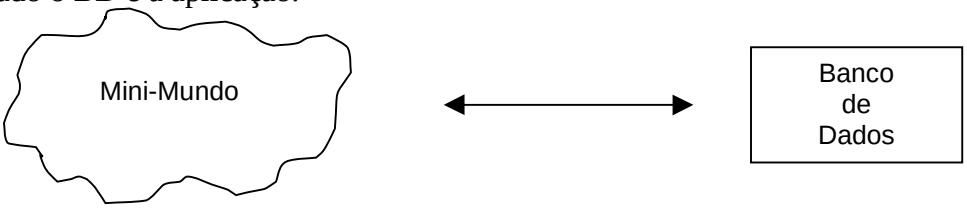
- Bancos de dados (BD) são conjuntos de dados relacionados e acessíveis.
- Sistemas gerenciadores de bancos de dados (SGBD ou DBMS – *Database Management System*) são sistemas que gerenciam BDs, ou são linguagens utilizadas para manter os BDs.
- Sistemas de BD são sistemas desenvolvidos com funções específicas, que usam BDs, desenvolvidos em SGBDs.

SGBD é um pacote de *softwares* que facilita a criação e manutenção de um BD. Sozinho um SGBD não significa nada, com um BD e um programa escrito para sua manipulação forma-se um sistema de BD.

Uma analogia sobre a diferença de um SGBD e um sistema de BD, pode ser por exemplo, um programa escrito em C e seu compilador, juntos formam uma aplicação.

Num BD os dados relacionados têm que possuir interesses comuns e têm que ser ligados à realidade. Os dados são matéria-prima de forma crua, fatos que podem ser gravados com significado implícito.

Mini-Mundo (Universo de Discurso) é a parte do Mundo real sobre o qual vai ser criado o BD e a aplicação.

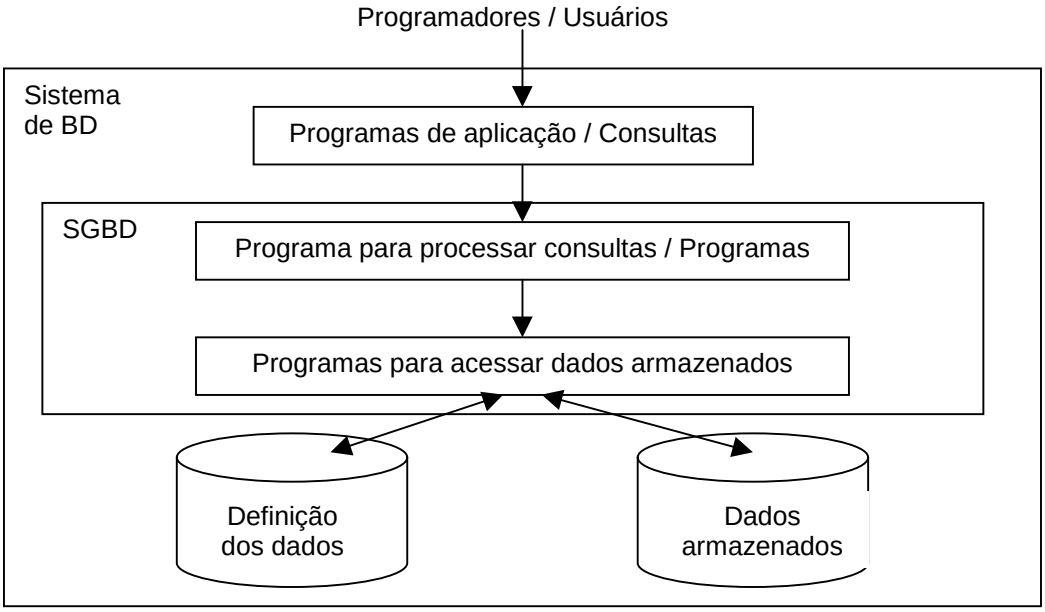


Melhor definição de BD:

Um banco de dados é um conjunto de dados armazenados, cujo conteúdo informativo representa, a qualquer instante, o estado de uma determinada aplicação.

Um banco de dados é um modelo de uma determinada parte da realidade, geralmente denominada de Universo de Discurso.

Ambiente de um sistema de banco de dados:



Principais características da tecnologia de BDs:

- Natureza "auto-contida" de um sistema de BD (catálogo que armazena o esquema do banco).
- Isolamento (independência) entre programas e dados .
- Abstração de dados (um modelo de dados é usado para esconder detalhes de armazenamento, com uma visão conceitual do BD).
- Múltiplas visões (capaz de suportar diferentes visões dos dados, a depender do usuário, somente as que interessam). Isto é importante tanto para simplificar para o usuário, quanto por motivos de segurança.

Classes de usuários:

- Administrador de bancos de dados (DBA – *Database Administrator*)
- Projetistas do banco de dados
- Analistas de sistemas
- Usuários finais: casuais, ingênuos e sofisticados.

Características adicionais da tecnologia de BD:

- Controle de redundância
- Compartilhamento entre múltiplos usuários
- Restrição de acesso aos dados
- Diferentes tipos de interface para diferentes usuários.
- Representação dos dados com um nível grande de complexidade.
- Garante a restrição de integridade (manter dados íntegros).
- Mecanismos de *backup* e recuperação de dados.
- Flexibilidade na mudança das estruturas de dados.
- Redução do tempo de desenvolvimento da aplicação.
- Dados sempre atuais estão disponíveis.
- Economia de escala relacionado com a redução do tempo de desenvolvimento e usado uma vez continuará disponível para outras aplicações.

Quando não usar um SGBD:

- Principal custo do uso do SGBD
  - Grande investimento inicial
  - *Overhead* devido a uma variedade de controles que o SGBD tem que executar.

- Quando o SGBD não é necessário
  - Aplicações simples e que não necessitam de mudanças.
  - Requisitos de processamento que não podem ser garantidos pelo SGBD.
  - Não requer acesso múltiplo de usuários.

2. Modelos de Dados

Modelo de dados é um conjunto de conceitos que se usa para descrever a estrutura do BD e certas restrições que o banco deve garantir.

Exemplo: No modelo relacional as relação (ou tabelas) representam os dados.

EMP

C#	Nome	Endereço	...
001	Maria	Rua A	...
002	José	Rua B	...
...	...	...	...

Operações no modelo de dados:

Inserção
Deleção
Alteração
Recuperação

Categorias de modelos de dados:

Conceitual – baseado em entidades ou objetos. Descreve a estrutura dos dados de maneira abstrata sem se preocupar com a implementação física.

Físico – descreve aspectos físicos de implementação.

Implementação – modelos lógicos.

Esquema:

Descrição da estrutura de um BD. Pode ser textual ou gráfico.

Exemplo:

CREATE TABLE EMP (C# INTEGER, Nome CHAR, Endereco CHAR, ...)

ou

EMP

C#	Nome	Endereço	...

Instância:

Os dados atuais armazenados no BD em um momento particular. Também chamado estado do banco de dados.

Arquitetura de três-esquemas: (diferentes níveis de descrição dos dados)

- Diferentes níveis de mapeamento (independência dos dados).
- Diferentes visões dos dados.
- Independência física.
- Independência lógica.

Modelos conceituais → descrevem → o esquema conceitual (Exemplo: ER).
Modelos lógicos ou de implementação → descrevem → esquemas externos (Exemplo: Relacional).

Na prática não acontece desta forma. Já na modelagem conceitual usa-se modelos lógicos para descrever e são criados vários esquemas externos para um mesmo modelo relacional.

Exemplo:

Modelo lógico (Relacional)

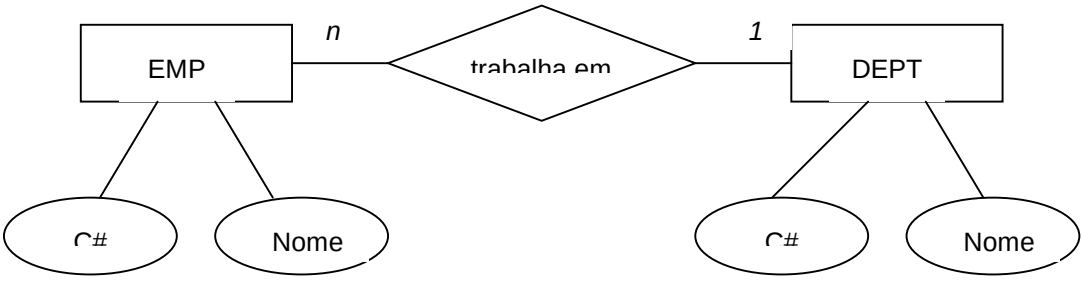
EMP

C#	Nome	Endereco	...	NumDep
21	João	Rua X	...	D1
...				

DEPT

C#	Nome	...
D1	DEX	...
...		

equivale a...
Modelo Conceitual (Entidade-Relacionamento)



Linguagens do SGBD:

- DCL (Data Control Language) – usada pelo DBA para controlar os dados.
- DDL (Data Definition Language) – descreve a estrutura do BD. Usada pelo DBA e pelos projetistas. Gera um catálogo a partir da descrição dos dados.
- DML (Data Manipulation Language) – permite especificar recuperação e alterações dos dados do BD. Exemplo: SELECT... FROM... WHERE... Pode ser embutida em uma linguagem hospedeira, tendo que ser compilada.
  Tipos de DML: procedimental e declarativa (não procedimental).

SQL = DDL + DML + DCL

Interfaces do SGBD:

- Interfaces stand-alone query language
- Linguagens embutidas
- Interfaces amigáveis
- Interfaces parametrizadas
- Geradores de relatório
- Interfaces para o DBA

Utilitários do SGBD:

- Carga de dados
- Ferramentas de backup
- Ferramentas de organização dos arquivos
- Geradores de relatório
- Ferramentas de gerenciamento (monitoramento) de desempenho
- Outras funções, como: ordenação, monitoramento de usuários, etc.

Utilitários do dicionário de dados:

- Funções para descrição do esquema
- Dicionário de dados ativo
- Dicionário passivo

Classificação dos SGBDs:

- Dependendo do modelo de dados usado:
  - Tradicionais: Relacional, Rede e Hierárquico.
  - Emergentes: orientado a objetos, semânticos, entidade-realcionamento.

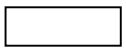
- Outras classificações:
  - Mono *X* Multiusuário
  - Centralizado *X* Distribuído
  - Custo
  - Tipo de acesso ao banco

3. Modelagem e projetos de bancos de dados

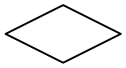
<u>Modelagem de dados usando o modelo E/R</u>

O projeto físico muitas vezes precisa de informações das especificações das operações básicas feitas pelos usuários, para ser desenhado.

Conceitos básicos do modelo E/R:

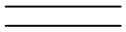


Tipo de entidade – são conjuntos de instâncias



Tipo de relacionamento – são ações que interagem com as entidades

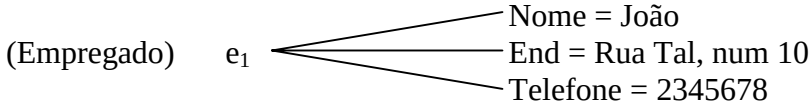
Atributos – são características comuns às instâncias que formam entidades



Totalidade – restrição imposta às entidades e aos relacionamentos, onde todas as instâncias têm que estar relacionadas a outras

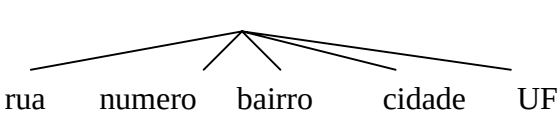
Em outras palavras:

ENTIDADES são objetos ou "coisas" do mundo real que possuem uma existência independente e são de interesse para uma determinada aplicação. ATRIBUTOS são propriedades usadas para descrever uma entidade. Por exemplo:



Os atributos podem ser dos seguintes tipos:

- **simples** (geralmente é só um conjunto de caracteres) ou **compostos** (com *n* atributos simples, exemplo: Endereco = rua+numero+bairro+cidade+UF)



- **monovalorados** (cada atributo possui uma instância) ou **multivalorados** (um atributo pode possuir *n* instâncias, por exemplo um departamento de uma empresa localizado em diversas cidades)

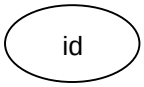
 Representação de atributo multivalorado

- **armazenados-básicos** (desenhados explicitamente no modelo E/R) ou **derivados** (não são armazenados e são representados por linha tracejada, por exemplo, quantidade de empregados)

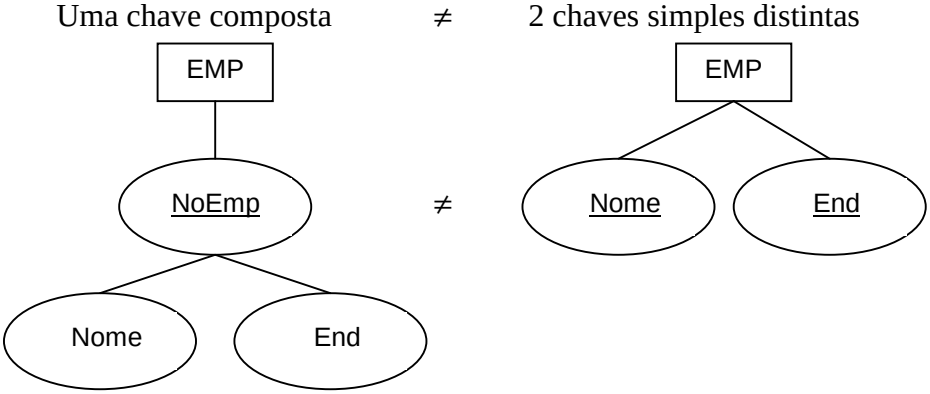
 Representação de atributo derivado

Tipos de entidades definem um conjunto de entidades que têm os mesmos atributos. Por exemplo, numa empresa o tipo de entidade Empregado.

CHAVES são identificadores de um tipo de entidade. Um atributo de um tipo de entidade que possui um valor único para cada entidade é chamado de chave. Por exemplo: número da identidade de um empregado. A chave é representada pelo nome do atributo sublinhado, por exemplo



Uma CHAVE COMPOSTA é formada pela combinação de vários atributos. Um tipo de entidade pode possuir uma chave simples, ou uma chave composta, ou várias chaves simples (distintas), ou várias chaves compostas.



O conceito de chave foi introduzido em 1976 por Peter Chen, com o motivo de identificar uma instância, porém a chave deve ser vista de uma forma mais ampla, como uma restrição de integridade. Para o conceito de chave deve-se entender que a instância é única.

Conjunto de valores (DOMÍNIO) de um atributo – cada atributo de um tipo de entidade possui um conjunto de valores, ou domínio, que define os valores que podem ser assumidos por esse atributo, para cada entidade individualmente.

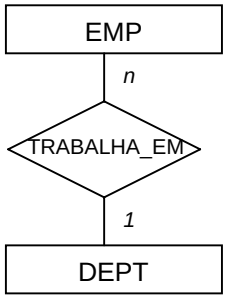
$$A: E \rightarrow P(V)$$

Dois atributos podem ter o mesmo domínio, mais amplo, como por exemplo: na entidade EMP, os atributos TelRes e TelCom possuem o mesmo domínio amplo dos números de telefone; e, DataNasc e DataAdms possuem o mesmo domínio amplo das datas

RELACIONAMENTOS ou tipos de relacionamentos – um relacionamento é uma associação entre duas ou mais entidades distintas(ou instâncias da entidade), com um determinado significado.

Por exemplo: EMP João *TRABALHA_EM* DEPT pesquisa

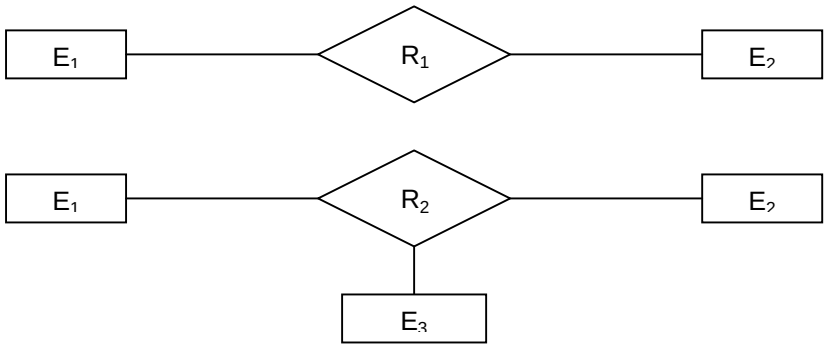
Um tipo de relacionamento *R* entre *n* tipos de entidades $e_1, e_2, ..., e_n$ define um conjunto de associações entre entidades desse tipo. Por exemplo:



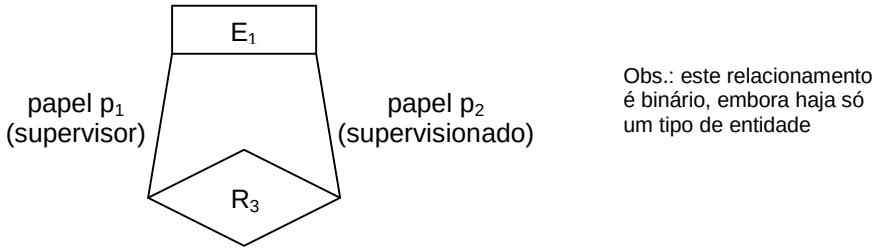
Matematicamente, *R* é um conjunto de instâncias de relacionamentos r_i, onde r_i cada associa *n* entidades (instâncias de entidades) $e_1, e_2,..., e_n$ e cada e_j em r_i pertence ao tipo de entidade E_j, onde $i \leq j \leq n$.

Assim: $R \subseteq E_1 \times E_2 \times ... \times E_n$ (produto cartesiano)

GRAU de um tipo de relacionamento é o número de tipos de entidades participantes de um tipo de relacionamento. Na maioria das vezes os relacionamentos são binários (grau 2). Alguns autores defendem que não deva existir relacionamentos com grau maior que 2, porém na prática isso não acontece. No caso de relacionamentos ternários, na maioria das vezes é possível transforma-lo em dois relacionamentos binários.



As entidades participantes de um relacionamento atuam com um determinado PAPEL nesse relacionamento. O significados desse papel é dado por um nome a ele atribuído. Um relacionamento é dito RECURSIVO quando este relaciona uma única entidade, através de dois papéis distintos. Por exemplo:



Restrições sobre tipos de relacionamento:

- Relação de cardinalidade – especifica o número de instâncias de um tipo de relacionamento que uma entidade pode participar. Para relacionamentos binários pode ser: *1* para *1* (*1:1*), *1* para *n* (*1:n*), *n* para *1* (*n:1*) ou *n* para *n* (*n:n*).
- Relação de participação – especifica se a existência das entidades dependem de ela estar associada ou não a outra entidade. Pode ser chamada de **obrigatória** (total) ou **opcional** (parcial).

CARDINALIDADE + PARTICIPAÇÃO = RESTRIÇÕES ESTRUTURAIS

As restrições estruturais definem como as entidades serão representadas, ou seja, a participação mínima e máxima das instâncias dos tipos de entidades nos relacionamentos.

<u>Modelagem de dados usando o modelo Relacional:</u>

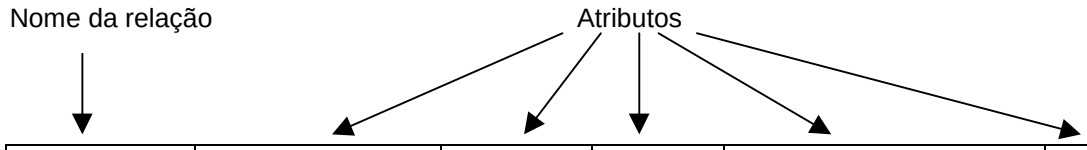
Num modelo relacional as estruturas são tabelas (relações), as restrições de integridade são chaves, restrições de domínio, de entidade e referencial, e as operações são álgebra relacional.

Conceitos:

Um banco de dados estruturado de acordo com o modelo relacional corresponde a uma coleção de relações. Informalmente, uma relação é uma tabela na qual cada linha expressa ema coleção de dados relacionados, cujos valores podem ser interpretados como um fato que descreve uma entidade ou um relacionamento.

Terminologia Relacional:

As linhas de uma relação, ou tabela, são chamadas de TUPLAS e o cabeçalho das colunas de ATRIBUTOS. O conjunto de valores que podem aparecer em cada coluna é chamado de DOMÍNIO.



ESTUDANTE	Nome	RG	Telefone	Endereco	Idade
	Sara Pereira	5.555.555	821-1234	Rua da Bahia, 90	21
	João Paulo Moura	9.999.999	821-0099	Av. Brasil, 1004	20
Tuplas	Marcos da Silva	4.444.444	821-8324	Rua Progresso, 370	18
	Adndre Carvalho	2.222.222	821-5002	Rua da Paz, 503	21
	Paula de Almeida	6.666.666	821-7890	Av. José Faria, 238	22

ESQUEMA DE RELAÇÃO é uma expressão da forma: $R(A_1,A_2,...,A_n)$ onde:
R – é o nome de uma relação;
A_i – é o nome de um atributo que representa um papel de um domínio D em R – $dom\ (A_i)$;
n – é o grau da relação.

Relação (ou INSTÂNCIA de uma relação) $r(R)$ é um conjunto de n-tuplas (ou tuplas) $r = \{t_1, t_2,..., t_n\}$ onde cada n-tupla t é uma lista ordenada de n valores $t=\langle v_1, v_2,...,v_n\rangle$, sendo cada v_i um elemento do $dom(A_i)$ ou um valor especial nulo.

Formalmente, $r(R) \subseteq (dom(A_1)\ x\ dom(A_2)x...x\ dom(A_n))$, ou seja, um subconjunto do produto cartesiano dos domínios de atributos, que representam fatos do mundo real.

Características de uma relação:

- As tuplas de uma relação não são ordenadas.
- Uma tupla é uma lista ordenada de valores, portanto, a ordem dos valores em uma tupla, e consequentemente, dos atributos na definição de um esquema de relação é importante.
- valor de cada atributo em uma tupla é atômico.
- Um esquema de relação pode ser interpretado como uma declaração, ou seja, uma tupla satisfaz ou não uma relação.

Esquema de um BD relacional:

Um esquema de BD relacional S define um conjunto de esquemas relação $R = \{R_1, R_2,...,R_n\}$ e um conjunto de restrições de integridade I. Portanto $S=(R,I)$.

Uma instância B de S é um conjunto de relações $B = \{r_1, r_2,...r_n\}$, tal que, cada r_i é uma instância de R_i e satisfazem as restrições de integridade especificadas em I.

Restrições de integridade:

➢ Restrições de domínio

Especificam que o valor de cada atributo *A* de uma relação deve ser um valor atômico do *dom(A)*.

➢ Restrições de chaves

Um conjunto de atributos *SK* de *R*, tal que, para duas tuplas quaisquer t_1 e t_2 de *r(R)*, $t_1[SK] \neq t_2[SK]$ (t_1 e t_2 identificam de maneira única) é chamado de SUPERCHAVE de *R*.

Uma CHAVE de *R* é uma superchave com a propriedade adicional de que nenhum de seus subconjuntos também seja uma superchave de *R*. Em outras palavras, chave é o conjunto mínimo de atributos para identificar de forma única uma tupla numa tabela.

Em geral, um esquema de relação *R* pode ter mais de uma chave, cada uma dessas chaves é chamada de CHAVE CANDIDATA. Dentre as chaves candidatas, uma delas é indicada como CHAVE PRIMÁRIA (aquela que será escolhida para identificar unicamente uma tupla), as demais constituem as chaves alternadas (estas não devem ser sublinhadas).

➢ Restrições de integridade de entidade

Nenhum componente de uma chave primária pode ser NULO.

➢ Restrições de integridade referencial

Seja *FK* um conjunto de atributos de um esquema de relação R_1 definido sobre os mesmos domínios dos atributos da chave primária *PK* de um outro esquema de relação R_2.

Então, para qualquer tupla t_1 de R_1 uma das seguintes situações ocorre:

i) $t_1[FK] = t_2[PK]$, onde t_2 é alguma tupla de R_2, ou

ii) $t_1[FK]$ é nulo.

A restrição de integridade referencial entre R_1 e R_2 pode ser expressa pela notação $R_1[FK] \rightarrow R_2[FK]$, onde *PK* é a chave primária de R_2 e *FK* é a chave ESTRANGEIRA de R_1.

Restrição de integridade referencial (RIR)– Opções de Remoção:

Associada a uma RIR: $R_1[x] \rightarrow R_2[y]$ é possível definir uma opção de remoção de valores da chave primária.

São três as opções:

n → Tornar nulo o campo referente na chave estrangeira.

p → Propagar a remoção, ou seja, remover todas as referências na tabela que contém a chave estrangeira. Atitude bastante drástica, geralmente utilizada em tabelas que são entidades fracas no modelo E/R.

b → Bloqueio da remoção. Não permite excluir itens da chave primária que contêm referências na tabela da chave estrangeira. Bastante utilizada.

Operações sobre relações:

De acordo com o modelo relacional as operações sobre um BD relacional podem ser classificadas em:

- operações de recuperação (consulta)
- operações de atualização (INSERT, DELETE, MODIFY tupla)

Restrições de integridade não podem ser violadas pelas operações de atualização. Atualizações podem ser propagadas automaticamente para manter restrições de integridade, por exemplo, no caso da remoção de tuplas que violem a restrição de integridade referencial.

Álgebra Relacional:

A álgebra relacional é um conjunto de operações básicas usadas para manipular relações em um BD relacional.

As operações da álgebra relacional podem ser divididas em fundamentais ou derivadas (formadas por mais de uma operação fundamental). As operações são geralmente divididas em dois grupos: operações de conjuntos ou específicas.

- Operações de conjuntos: UNIÃO, INSERÇÃO, DIFERENÇA e PRODUTO CARTESIANO.
- Operações específicas: SELEÇÃO, PROJEÇÃO, JUNÇÃO e DIVISÃO.

Obs.: estudaremos algumas destas operações, as mais importantes.

Uma característica importante é que toda operação da álgebra relacional tem como resultado uma relação.

Operação de seleção: seleciona um subconjunto de tuplas de uma relação, que satisfazem uma condição de seleção (expressa no predicado).

Notação: $\sigma_{\langle COND \rangle}(\langle \text{nome da relação} \rangle)$

Exemplos:

1) $\sigma_{DNO=4}$ (EMPLOYEE)
2) $\sigma_{SALARY=30000}$ (EMPLOYEE)
3) $\sigma_{(DNO=4 \text{ AND } SALARY=30000)}$ (EMPLOYEE)

A operação de seleção é comutativa, ou seja:

$$\sigma_{\langle COND1 \rangle}(\sigma_{\langle COND2 \rangle}(R)) = \sigma_{\langle COND2 \rangle}(\sigma_{\langle COND1 \rangle}(R))$$

Operação de projeção: projeta as tuplas de uma relação sobre um determinado conjunto de atributos (elimina colunas).

Notação: $\pi_{\langle ATRIBUTOS \rangle}(\langle \text{nome da relação} \rangle)$

Exemplos:

1) $\pi_{LNAME,FNAME,SALARY}$ (EMPLOYEE)
2) $\pi_{SEX,SALARY}$ (EMPLOYEE)

Combinando as duas operações:

$$\pi_{\langle ATRIBUTOS \rangle}(\sigma_{\langle COND \rangle}(\langle \text{nome da relação} \rangle))$$

Exemplo: $\underbrace{\pi_{\text{LNAME,SALARY}}}_{\text{projeção}} \underbrace{(\sigma_{\text{DNO=4}}(\text{EMPLOYEE}))}_{\text{seleção}}$

Seqüência de operações: várias operações podem ser combinadas para formar uma expressão da álgebra relacional, que é uma consulta.

Alternativamente, pode-se especificar relações temporárias para cada passo da consulta. Por exemplo: $\text{DEPS\_EMPS} \leftarrow \sigma_{\text{DNO=5}}(\text{EMPLOYEE})$

$\text{RESULT} \leftarrow \pi_{\text{FNAME,LNAME}}(\text{DEPS\_EMP})$

Os nomes resultantes dos atributos da nova relação de uma expressão da álgebra relacional podem ser alterados. Por exemplo:

$\text{RESULT}_{\text{PRIMNOME,ULTNOME}} \leftarrow \pi_{\text{FNAME,LNAME}}(\text{DEPS\_EMP})$

Operações de conjuntos:

UNIÃO – efetua a união de duas relações compatíveis ($R \cup S$)

DIFERENÇA – efetua a diferença entre duas relações compatíveis ($R - S$)

INTERSEÇÃO – efetua a interseção de duas relações compatíveis ($R \cap S$)

Duas relações $R(A_1, A_2,..., A_n)$ e $S(B_1, B_2,..., B_n)$ são compatíveis se elas tiverem o mesmo grau n e se $dom(A_i)=dom(B_i)$ para $1 \leq i \leq n$.

Exemplos: $\text{DEPS\_EMP} \leftarrow \sigma_{\text{DNO=5}}(\text{EMPLOYYE})$

$\text{RESULT1} \leftarrow \pi_{\text{SSN}}(\text{DEPS\_EMP})$

$\text{RESULT2}_{\text{SSN}} \leftarrow \pi_{\text{SUPERSSN}}(\text{DEPS\_EMP})$

$\text{RESULT} \leftarrow \text{RESULT1} \cup \text{RESULT2}$

Produto cartesiano: combina as tuplas de duas relações ($R \times S$).

Exemplo: $\text{FEMALE\_EMP} \leftarrow \sigma_{\text{SEX='F'}}(\text{EMPLOYEE})$

$\text{EMP\_NAMES} \leftarrow \pi_{\text{FNAME,LNAME,SSN}}(\text{FEMALE\_EMP})$

$\text{EMP\_DEPENDENTS} \leftarrow \text{EMP\_NAMES} \times \text{DEPENDENT}$

$\text{ACTUAL\_DEPENDENT} \leftarrow \sigma_{\text{SSN=ESSN}}(\text{EMP\_DEPENDENTS})$

$\text{RESULT} \leftarrow \pi_{\text{FNAME,LNAME,DEPEND\_NAME}}(\text{ACTUAL\_DEPENDENT})$

Junção: combina as tuplas de duas relações que satisfazem uma determinada condição de junção ($R \bowtie_{\langle \text{COND} \rangle} S$).

O mesmo exemplo anterior poderia ser:

$\text{FEMALE\_EMP} \leftarrow \sigma_{\text{SEX='F'}}(\text{EMPLOYEE})$

$\text{EMP\_NAMES} \leftarrow \pi_{\text{FNAME,LNAME,SSN}}(\text{FEMALE\_EMP})$

$\text{ACTUAL\_DEPENDENT} \leftarrow \text{EMP\_NAMES} \bowtie_{\text{SSN=ESSN}} \text{DEPENDENT}$

A condição de junção é da forma *⟨COND1⟩ AND ⟨COND2⟩...⟨COND3⟩*, onde cada *⟨COND i⟩* é da forma $A\theta B$, sendo A um atributo de R, B um atributo de S e θ um dos operandos de comparação $\{=, <, \leq, >, \geq, \neq\}$. Esta forma geral é chamada de junção TETA.

Uma operação de junção que envolva apenas condição de igualdade é chamada de EQUI-JUNÇÃO. Uma junção NATURAL é uma equi-junção na qual o segundo atributo de cada condição de igualdade é eliminado da relação resultante. Em geral, uma junção natural define atributos de mesmo nome.

Notação: R∗S ou $R*_{A,B}S$ (forma geral), onde *A* é o atributo de *R*, *B* é o atributo de *S* de mesmo domínio de *A*; que será eliminado (*A*=*B* é subentendido).

Mapeamento ER / Relacional:

Passos:

1. Para cada tipo de **entidade regular** *E*, crie um esquema de relação *R* contendo todos os **atributos simples** de *E*. Defina a **chave primária** de *R* a partir de uma das chaves (identificadores) de *E*.

2. Para cada tipo de **entidade fraca** *W*, crie um esquema de relação *R* contendo todos os atributos simples de *W*. Além disso, inclua como atributos de *R* os atributos que formam a chave primária das relações correspondentes aos tipos de entidade que identificam *W*. Defina a chave primária de *R* como a combinação destes atributos e dos atributos derivados da chave parcial de *W*.

3. Para cada tipo de relacionamento binário *R*, **1:1**, escolha um dos esquemas de relação correspondentes aos tipos de entidades participantes de *R*, por exemplo, *S* e inclua como **chave estrangeira** de *S* a chave primária do esquema de relação correspondente ao outro tipo de entidade participante de *R*, definindo-a como chave alternativa de *S*. Inclua em *S* os atributos simples de *R*.

4. Para cada tipo de relacionamento binário *R*, **1:n**, inclua no esquema de relação *S* correspondente ao tipo de entidade participando do "lado n" de *R*, como chave estrangeira, a chave primária do esquema de relação correspondente ao tipo de entidade participante do "lado 1" de *R*. Inclua os atributos de *R* como atributos de *S*.

5. Para cada tipo de relacionamento binário *R*, **m:n**, crie um esquema de relação *S*. Inclua como chave estrangeira de *S* as chaves primárias dos esquemas de relação correspondentes aos tipos de entidade participantes de *R*. Defina como chave primária de *S* a combinação das chaves estrangeiras. Além disso, inclua como atributos de *S* os atributos simples de *R*.

6. Para cada **atributo multivalorado** *A*, crie um esquema de relação *R*, que inclua um atributo correspondente a *A* e, como chave estrangeira, a chave primária *K* do esquema de relação correspondente ao tipo de entidade ou relacionamento que contém *A*. Defina como chave primária de *R* a combinação de todos os seus atributos.
7. Para cada tipo de relacionamento n-ário (n>2) *R*, crie um esquema de relação *S*. Inclua como chave estrangeira de *S* as chaves primárias dos esquemas de relação correspondentes aos tipos de entidade participantes de *R*. Defina a chave primária de *S* como uma combinação das chaves estrangeiras, de acordo com a relação de cardinalidade de *R*.

Observação:
Para cada chave estrangeira incluída em um esquema de relação deve ser especificada uma restrição de integridade referencial com a sua respectiva opção de remoção.

Exemplos:

EMPLOYEE[SUPERSSN] $\xrightarrow{n}$ EMPLOYEE[SSN]

EMPLOYEE[DNO] $\xrightarrow{b}$ DEPARTMENT[DNUMBER]

DEPT_LOCATIONS[DNUMBER] $\xrightarrow{p}$ DEPARTMENT[DNUMBER]

QUADRO DE CORRESPONDÊNCIA ENTRE OS CONCEITOS DOS MODELOS ER E RELACIONAL

MODELO ER	**MODELO RELACIONAL**
Tipo de entidade	Esquema de relação
Tipo de relacionamento 1:1 ou 1:n	Esquema de relação ou chave estrangeira
Tipo de relacionamento m:n	Esquema de relação com 2 chaves estrangeiras
Atributo composto	Conjunto de atributos
Atributo multivalorado	Esquema de relação + chave estrangeira
Tipo de relacionamento n-ário	Esquema de relação + n chaves estrangeiras
Chave	Chave primária ou alternativa
Tipo de entidade fraca	Esquema de relação + chave(s) estrangeira(s)

Normalização:

> *"Técnicas de racionalização das estruturas de dados de um sistema, eliminando redundâncias, problemas de manipulação e armazenamento."*

A normalização é a ferramenta para auxiliar no projeto físico, porém só é suficiente teoricamente. Na prática provou-se que a normalização não é suficiente.

A teoria da normalização serve para verificar se os esquemas do projeto físico satisfazem algumas características básicas. Se tornou muito mais um método de verificação, do que de definição como foi proposto.

NORMALIZAÇÃO é um processo através do qual esquemas de relação que não sejam "satisfatórios" são decompostos em esquemas menores que satisfaçam certas propriedades desejáveis.

Medidas de qualidade para o projeto de um esquema de relação:

1 – correta representação semântica
2 – redução de valores redundantes
3 – redução de valores nulos
4 – não geração de tuplas espúrias (sem sentido)

Quando ocorre uma decomposição de tabelas, por exemplo:

$R_{1.1}$ ($\underline{A_1}$, A_2, A_3)

R_1 ($\underline{A_1}$, A_2, A_3, A_4)

$R_{1.2}$ ($\underline{A_1}$, A_4)

Se for recompor $R_{1.1}$ e $R_{1.2}$ o resultado desejado seria a tabela R_1, mas nem sempre isto é conseguido. Podem ser geradas tuplas sem sentido (espúrias), nesta recomposição. A decomposição deve ser feita de forma muito cuidadosa.

O atributo que deve ser o pivô da decomposição deve ser chave de uma das tabelas envolvidas ou de ambas.

Diretrizes do projeto:

D1 – Projete um esquema de relação de tal forma que seja fácil explicar o seu *significado semântico*.

D2 – Projete um esquema de relação de forma a evitar *anomalias de atualização*.

D3 – Sempre que possível, projete um esquema de relação de forma a evitar atributos que possam assumir *valores nulos*.

D4 – Projete esquemas de relação de tal forma que as junções entre as relações correspondentes possam ser feitas através de condições de igualdade sobre atributos que são chaves primárias ou chaves estrangeiras de forma a garantir a não geração de *tuplas espúrias*.

Dependências funcionais:

Uma dependência funcional entre dois conjuntos de atributos, *x* e *y*, de um esquema de relação *R*, denotada por $x \rightarrow y$ é uma restrição que estabelece que para quaisquer tuplas t_1 e t_2 de uma instância *r* de *R*, tal que, se $t_1[x] = t_2[x]$, então $t_1[y] = [y]\ t_2$.

Exemplos:

1. SSN $\rightarrow$ {ENAME, BDATE, ADDRESS, DNUMBER}

$t_1[SSN]=t_2[SSN] \Rightarrow$

$t_1[ENAME,BDATE,ADDRESS,DNUMBER]=t_2[ENAME,BDATE,ADDRESS, DNUMBER]$

2. DNUMBER $\rightarrow$ {DNAME, DMGRSSN}

$t_1[DNUMBER]=t_2[DNUMBER] \Rightarrow t_1[DNAME,DMGRSSN]=t_2[DNAME,DMGRSSN]$

Quando se define uma dependência funcional (DF), esta regra deve valer para todas as instâncias da relação. É como se fosse uma restrição de integridade.

Não se pode deduzir a existência de uma dependência funcional baseado no conteúdo de uma tabela, porém a ausência da DF pode ser definida.

Por exemplo:

PROFESSOR	DISCIPLINA	PERÍODO
Olinda	Bancos de Dados	4
Olinda	Sistemas de Informação	6
Wilian	Arquitetura de Computadores	6
Wilian	Sistemas Digitais II	5
Antônio Maria	Bancos de Dados	4

Posso dizer: DISCIPLINA $\not\rightarrow$ PROFESSOR
DISCIPLINA $\not\rightarrow$ PERÍODO
PROFESSOR $\not\rightarrow$ DISCIPLINA
Não posso dizer: DISCIPLINA $\rightarrow$ PERÍODO (embora pareça verdade)

Formas Normais baseadas em chaves primárias:

O processo de normalização é realizado gradativamente através de três formas normais, definidas a partir do conceito de DF.

1FN → 2FN → 3FN

Propriedades do processo de normalização através de decomposição:
1. Junções sem perda
2. Preservação de dependências

Primeira Forma Normal (1FN):

Um esquema de relação *R* está na 1FN se **todos** os seus atributos forem **atômicos** e **monovalorados**, ou seja, não possuem valores que formam atributos compostos.
Exemplos:
1. FUNCIONÁRIOS = {CÓDIGO + NOME + ENDEREÇO + CARGO} e
   ENDEREÇO = {RUA + NUMERO + BAIRRO + CIDADE + UF}
   Para colocar na 1FN faz:
   FUNCIONÁRIOS = {CODIGO+NOME+RUA+NUMERO+BAIRRO+CIDADE+UF+CARGO}
2. FUNCIONÁRIOS = {CODFUNC + NOME + CARGO + {PROJETO + DATAINI + DATAFIM}}
   Para colocar na 1FN faz:
   FUNCIONÁRIOS = {CODFUNC + NOME + CARGO}
   FUNC_PROJ = {CODFUNC + PROJETO + DATAINI + DATAFIM}

Observação: todas as tabelas são relações na 1FN.

Segunda Forma Normal (2FN):

Dependência funcional total ou completa: Uma DF $x \rightarrow y$ é **total**, se não existir nenhum atributo *A* em *x* , tal que $(x - \{A\}) \rightarrow y$, para qualquer $A \in x$, ou seja se retirarmos esta atributo *A* da relação *x* a DF deixa de existir. Caso contrário, $x \rightarrow y$ é **parcial**.

Definição da 2FN: Um esquema de relação está na 2FN se: estiver na 1FN e, além disso, todo atributo que não pertença a alguma de suas chaves for **totalmente dependente** da sua chave primária.

Em outras palavras, para que uma relação esteja na 2FN é preciso que esteja na 1FN e todos os dados que não são chaves dependem de toda a chave primária.
Exemplo:

ESTOQUE = {PRODUTO+ALMOX+END_ALMOX+UNID_EST+QTD+PRECO}

Não está na 2FN porque alguns dados não chave, como END_ALMOX dependem de parte da chave, só de ALMOX, ou UNID_EST depende só de PRODUTO.

Normalizando ficaria:

ESTOQUE1 = {PRODUTO + UNID_EST}
ESTOQUE2 = {ALMOX + END_ALMOX}
ESTOQUE3 = {PRODUTO + ALMOX + QTD + PRECO}

Terceira Forma Normal (3FN):

Dependência funcional transitiva: Uma DF $x \rightarrow y$ é **transitiva** em um esquema de relação *R* se existir um conjunto de atributos *z*, que não seja um subconjunto de alguma chave de *R*, e as DFs $x \rightarrow z$ e $z \rightarrow y$ forem válidas em *R*.

Definição da 3FN: Um esquema de relação está na 3FN se: estiver na 2FN e, além disso, nenhum atributo que não pertença a alguma das suas chaves for **transtivamente dependente** da sua chave primária.

Em outras palavras, para que uma relação esteja na 3FN é preciso que esteja na 2FN e se todo atributo, que não pertença a alguma chave for não dependente de algum outro atributo, que também não pertença a alguma chave.

Exemplo:

MÚSICA = {CÓDIGO + TÍTULO + GÊNERO + PAÍS_ORIGEM}

Neste exemplo, o PAÍS_ORIGEM se refere ao GÊNERO musical e não a música, sendo assim, apesar de estar na 2FN, não está na 3FN pois existe dependência entre GÊNERO e PAÍS_ORIGEM.

Normalizando ficaria:

MÚS_1 = {CÓDIGO + TÍTULO + GÊNERO}
MÚS_2 = {GÊNERO + PAÍS_ORIGEM}