

# BD - Resumo 1

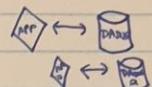
Ch 1

## • Conceito

- : Uma BD é uma coleção organizada de dados que estão relacionados e que podem ser partilhados por múltiplas aplicações

## • Processamento Isolado de dados

- : Cada App gera os próprios dados



- : Os mesmos dados podem estar replicados

- : Da origem a que diferentes organizações existam com diferentes formatos de dados

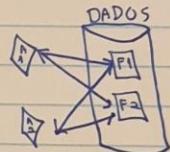
|| : Problemas de  
Sincronismo  
(Incoerências)

Evolução

"

## • Sistema de Gestão de Ficheiros

- : Dados organizados e armazenados em vários ficheiros partilhados por várias apps



- : Cada app acede diretamente aos ficheiros

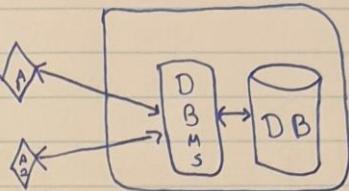
- : Cada app usa uma interface proprietária

|| : Problemas de acesso concorrente aos dados  
|| : Problemas de Integridade  
|| : Problemas de Segurança

## Sistema de Gerenciamento de BD

"DBMS is a general purpose software system that facilitates the processes of defining, constructing, manipulating & sharing DB among various users & apps"

Defining: Especificação do tipo de dados, estruturas de dados e restrições



Constructing: Processo de armazenamento de dados

Manipulating: Envolve operações como pesquisa e obtenção de dados

### Sharing

Sharing: Acesso simultâneo aos dados por parte de vários utilizadores e programas

### Características

Entidade única que opera com a BD  
(acesso à BD é sempre mediado pelo SGBD)

Existe uma interface de acesso que esconde os detalhes de armazenamento físico de dados

Elevada abstracção no nível aplicacional

Os dados estão integrados (nível lógico) numa mesma unidade de armazenamento

Supõe uma ou mais BD

Keyword: Data Independence

## : Vantagens !!

- ∴ Independência entre programas e dados
- ∴ Integridade dos dados (controle de alteração de dados de acordo com as regras de integridade definidas)
- ∴ Consistência dos dados (nos processos de transações e mesmo em failover de software/hardware)
- ∴ Eficiência no acesso aos dados (especialmente em cenários de manipulação de grandes quantidades de dados, por um ou mais utilizadores)
- ∴ Isolamento dos utilizadores (Cada user tem a "sessão" de seu único)
- ∴ Melhor gestão do acesso concorrential
- ∴ Serviços de segurança (Controlo de acesso/permisões Codificação de dados)
- ∴ Mecanismo de Backup e Recuperação de dados
- ∴ Administração de dados (Disponibilidade de ferramentas disponibilizadas pelo fabricante e/ou terceiros desenvolvidas)
- ∴ Linguagem de desenho e manipulação de dados

## : Desvantagens !!

- ∴ Maiores custos e complexidades na instalação e manutenção (específicos em soluções empresariais)
- ∴ Não respondem aos requisitos de alguns cenários aplicacionais (p.e.: pesquisa de texto)
- ∴ Centralização de dados pode ter problemas de tolerância a falhas (software e hardware) e de escalabilidade

## : Vista Simplificada - Sl 1.1

### : Utilizadores

• Fimais - Aquelas que usam o sistema com determinada finalidade com recurso a ferramentas disponibilizadas pelo fabricante do sistema ou aplicações de 3<sup>os</sup> entidades

• Programadores de Apps - Desenvolvem aplicações que permitem que os utilizadores interajam com a BD. Podem utilizar várias linguagens de programação

• Admin. de BD - Tratam dos processos de gestão e manutenção da BD

### : Dicionário de Dados

• Para além da BD, o SGBD Contém info. relativa à descrição (def) da própria estrutura de BD, incluindo as restrições (Metadados - dados sobre dados)

• Um dicionário contém

• Descrições de objetos de BD (tabelas, usuários, vistos, ...)

• Info sobre dados em uso e por quem (dados)

• Schemas e Mappings

## : Interfaces - Sl 1.1

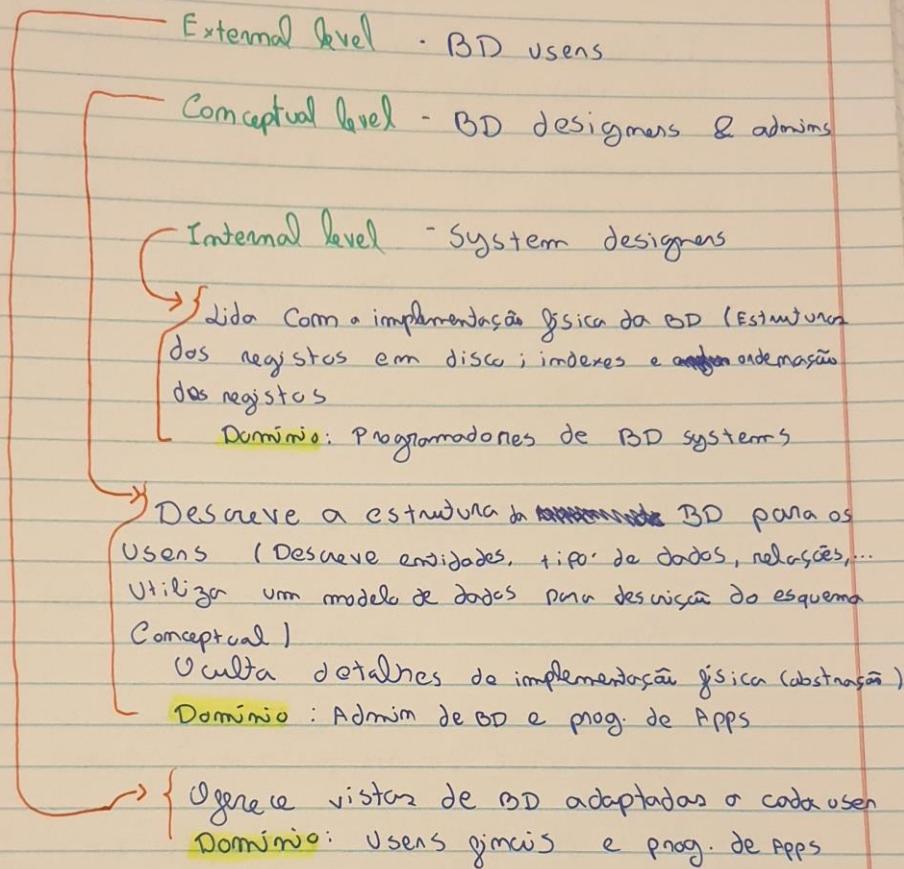
(...)

• GUIT • Manipulação visual de esquemas de BD com recurso a diagramas  
(...)

• PBMs Comunicação • Criar contas de usuários, personalizar o sistema, def/alterar estruturas de dados, ...  
• Usa linguagem própria - SQL

## : Arquitetura ANSI / SPARC

∴ Arquitetura de 3 Níveis



## : ANSI / SPARC - Independência dos dados

∴ A alteração do esquema de um nível **NÃO** tem impacto no esquema do nível acima

∴ 2 Níveis de Independência

∴ **Físico** - Alterações do nível físico não devem ter impacto no esquema conceptual (p.ex., se alterarmos a forma como armazenamos os dados no Filesys por razões de desempenho)

∴ **lógico** - Alterações no esquema Conceptual (mod de dados) não devem repercutir-se nos esquemas exteriores ou aplicações já desenvolvidas

: SGBD - Anq. Típica Sl 1.20

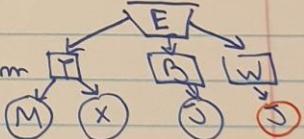
### • Modelo de Base de Dados

- :: Coleção de conceitos para descrição lógica de dados (Modelo lógico)
- :: Esquema - Descrição de um conjunto particular de dados com respeito a um determinado modelo
- :: É fundamental para garantir a independência dos dados
- :: Modelo relacional é dos mais usados

### • Modelo Hierárquico

- :: Dados estão armazenados numa estrutura hierárquica (árvore)
- :: Os nós da árvore designam-se como **registos** que estão ligados por **ponteiros** (links)

:: Um **Registo** é composto por um conjunto de atributos



:: Um **Link** é uma associação entre 2 registos do tipo **Pai-Filho**

:: Um registo **pai** encontra-se associado a **N** registos **filhos** (**1:N**)

:: Ex sl 1.24 / 1.25

~~QUESTION~~

## *:: Desvantagens*

*:: Adaptado a cenários de acesso sequencial aos dados (qg acesso passa seg. pelo seg. raiz; A maior parte das necessidades不易 requere acesso aleatório)*

*:: Redundância de Info (Desperdício de espaço e inconsistência de dados)*

*:: Restrições de Integridade (Ex.: elim. de um seg. pôr implicar remoção de todos os filhos assoc.)*

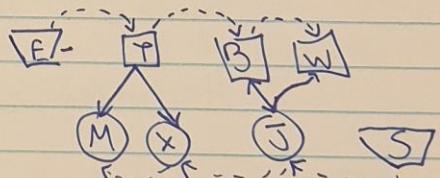
*:: Não permite estabelecer associações N:M  
(ver sl 1.27)*

## *:: Modelo de Rede*

*:: Extensão do modelo hierárquico*

*:: Permite que um registo esteja envolvido em várias associações*

*:: Melhorar a capacidade de manegagem  
ma estrutura de dados*



*:: Relações representadas através de grafos*

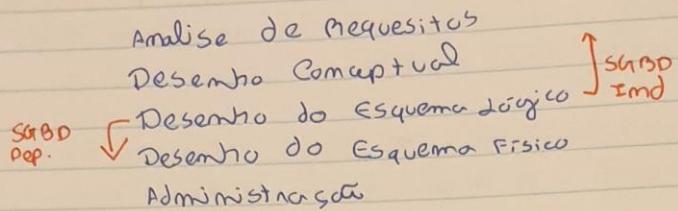
*:: Um Conjunto suporta associações entre registos do mesmo tipo*

*:: Melhoramento 1:N entre 2 tipos de registos*

*:: Ex e Comparaçõe sl 1.29 / 1.30*

## Ch 2

### • Desenho BD



### • Análise de Requisitos

: Obliga a um processo de comunicação com o cliente da solução de BD

1. Levantamento detalhado de toda a info essencial associada ao problema do mundo real
2. Filtragem da info: remoção de redundâncias e ruído (info pouco relevante)
3. Discussão para澄清各方面 aspectos dúbios e eventuais falhas no levantamento do ponto 1
4. Distinção entre dados e operações

### • Desenho Conceptual

: Modelo Conceptual - Conceptualização do mundo real

: Trata do mapeamento das entidades e relações do mundo real para conceitos de BD

:: Não é determinístico

:: Nem sempre é óbvio

: Uma visão abstrata da estrutura de dd que suporta os dados reais

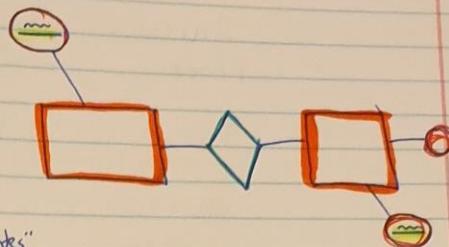
: Tipicamente usa-se **Modelo Entidade/Relação**

## • Modelo E/M

### : Elementos Básicos

- Entidade

- "Algo que Existe"



- Atributos

- "Propriedades das Entidades"

- Relações

- "Entre 2 ou + entidades")

As entidades têm um ou mais Atributos Chave que as identificam (aparecem sublinhados)

### : Entidades

- Fontes

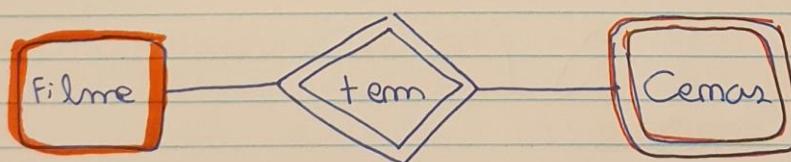
- Não dependem de outras entidades

- Fracas

- Dependem de outras entidades

• Não têm atributos chave, são identificados sendo relacionados a entidades específicas de outro entity type em combinação com um dos seus atributos

• Não pode ser identificado sem owner entity



### : Atributos

- Derivados

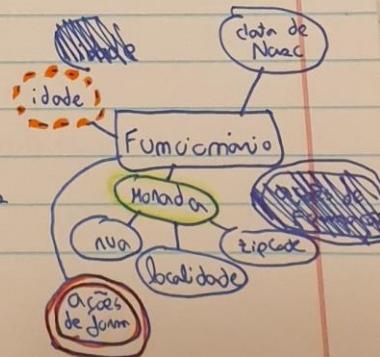
Atributo baseado noutro atributo

- Multivalor

Atributo que pode ter um ou mais valores

- Compostos

Atributos compostos por outros atributos



## : Relações

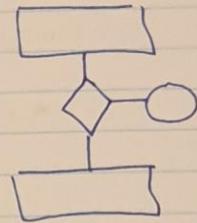
• Interações entre 2 ou mais entidades

• Podem ter atributos

• Clasificam-se como:  
• Granularidade

• Obligatoriedade

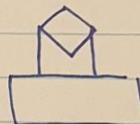
• Cardinalidade



## : Relacionamentos - Granularidade

• Número de entidades participantes na relação

• Unária

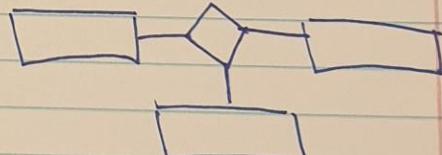


• Bimária



• Trinária

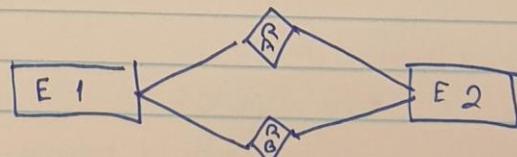
Podem ser convertidas em bimárias  
e em unárias



Dão Assos à Relações



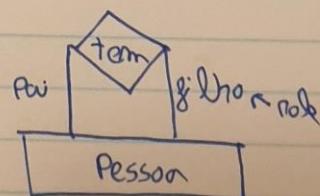
• Múltiplas



• Recursivas

Unárias

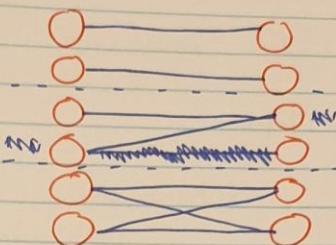
Assimétricas (é preciso espalhar os papéis)



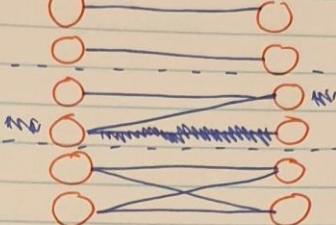
## Relacionamentos + Cardinalidade

Definição: Relação entre o # de ocorrências numa entidade com as respectivas ocorrências na outra com que tem relacionamento.

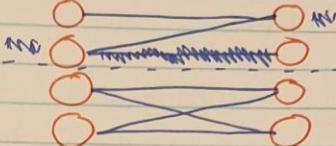
- 1:1



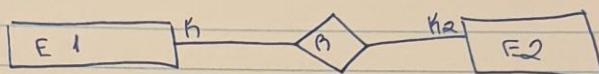
- 1:N



- N:M



## Notação de Chen



## Relacionamentos - Obrigatoriedade

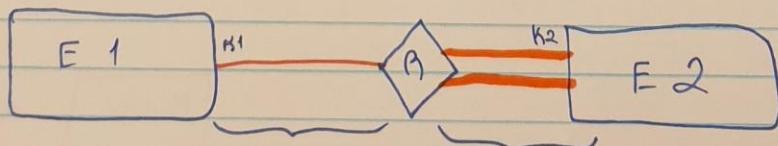
Obrigatoriedade: Da participação da entidade na relação

### Participação Total

Definição: Cada instância da entidade participa em pelo menos uma relação do conjunto de relações

### Participação Parcial

Definição: Algumas (s) instâncias da entidade podem não participar em qualquer relação do conjunto de relações



Nem todas as entidades E1 são obrigadas a participar na rel. R

Todos os entidades do tipo E2 participam em pelo menos 1 relação R

*Nota: Existe uma outra motivação → SL 2.18, 2.19  
Exemplos de Relações → SL 2.20 / 2.21 / 2.22*

## *: Previsões de Integridade*

*:: São invariantes que a BD deve garantir*

*:: Tipos*

*:: Atributos*

Cada atributo tem um só valor

Atributos chave são únicos

Atributo deve/pode ter um valor

Valor do atributo pode ter restrições ( $>$ ,  $<$ ,  $=$ , not null, ...)

*:: Cardinalidade da relação*

Relação 1:1

" 1:N

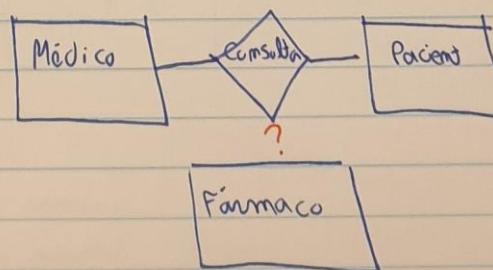
" N:M

*:: Obrigatoriedade de participação das entidades nas associações*

## *: Agregação*

As vezes temos a necessidade de modelar uma relação entre uma entidade e outra relação

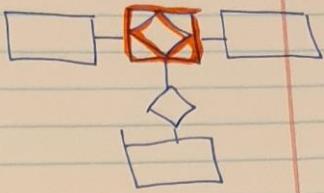
P. ex. Como associar Fármacos prescritos numa consulta



Solução: Tomar a relação numérica:

#### • Entidade Associativa

Permite associar entidades a relacionamentos



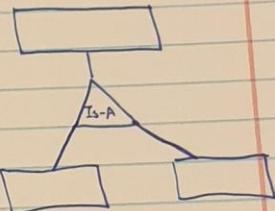
→ Exemplos Sl 2.28 | 2.29 | 2.30

#### : Generalização VS Especialização

• Classificações de entidades em hierarquia de classes

#### • Relação Is-A

As subentidades herdam os atributos das super-entidades

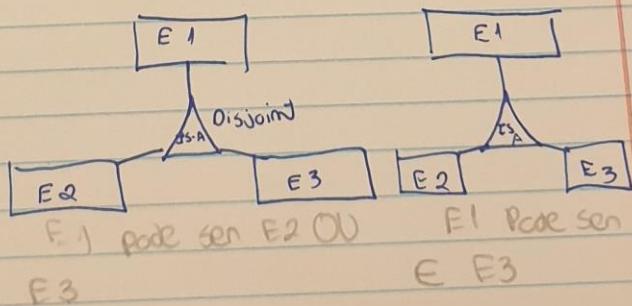


#### : Restrições (Tipos de Especialização)

##### Restrição de Overlapping

Disjuntos - Uma entidade só pode pertencer, no máximo, a uma subclasse de especialização

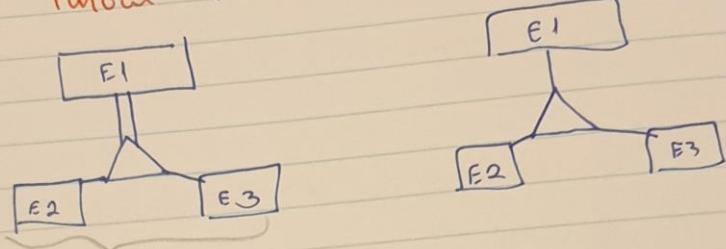
Sobrepostos - Uma ocorrência de entidade genérica pode ter mais de uma especialização



### Restrições de Covering

Total - uma entidade de nível superior TEM de pertencer a pelo menos uma subclasse de especialização

Parcial - Pode não pertencer a nenhuma



E1 Tem de Ser E2  
OU E3 (Nunca pode ser  
só E1)

### Ch 3

#### • Modelo Relacional (Naum Ch1, é um dos Modelos de BD)

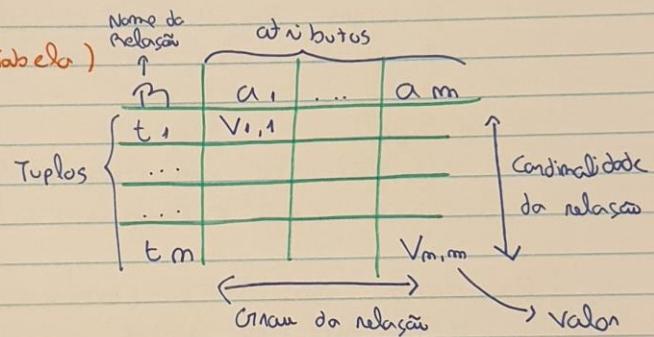
• Baseado na Teoria dos Conjuntos (modelo matemático rigoroso)

• Baseado na noção matemática de "Relação" representados por tabelas

• Dispõem de um sistema formal de manipulação de relações - **Algebra Relacional**

#### • Conceitos

• Relação (Tabela)



• Atributo ( $A_1, A_2, \dots$ )

• Representam o tipo de dados a armazenar

• O # de atributos de uma relação define o **grau da relação**

• Os atributos devem ter nomes distintos

• Domínio ( $D_1, D_2, \dots$ )

• tipo de Dados

• Grau de valores possíveis para determinado atributo

(p. ex : Nome { Adelmo, Diaz, Mafalda, Heolito } )

• Valores desconhecidos ou não existentes - **Null**

### : Esquema de Relação $R(A_1, A_2, \dots)$

“Relational Schema”

• Nome do esquema e lista de atributos (Ex: Pessoa|Name)

• Opcionalmente: inclui o tipo de dados

### : Relação $n(R)$

• Estrutura bidimensional com determinado esquema e zero ou mais instâncias (tuplos)  
 $(n = \{ t_1, t_2, \dots \})$

• Formalmente é subconjunto do produto cartesiano  
 $n(R) \subseteq (\text{dom}(A_1) \times \text{dom}(A_2) \times \dots)$

### : Tuplo

• Linha de uma relação ( $t = \langle v_1, v_2, \dots \rangle$ )

• Devem ser distintos numera relações

• A ordem das linhas é irrelevante

• O # de tuplos define a cardinalidade da relação

### : Atomidade

• O valor de um atributo num tuplo é atómico (mão é composta / multi-valor)

### : Esquema da BD (DB Schema)

• Conjunto de todos os esquemas da relação da BD

•  $D = \{ R_1(A_1, \dots, A_n), \dots, R_m(X_1, \dots, X_n) \}$

### : Exemplo

Relação Estudante		
N McC	Nome	Curso
65022	Juan Sousa	MIECT
65023	Maria Costa	TIC
65024	José Pereira	L MAT
...	...	...

↑ Atributos  
↓ Cardinalidade da relação

←→ Ordem da relação

Tuplos

Estudante (N McC, Nome, Curso) ← Esquema da relação

## • Chaves

**Superchave** - Conjunto de atributos que identificam de forma única os tuplos da relação

**Chave Candidata** - Subconjunto de atributos de uma Superchave que não pode ser reduzido. Sem perder essa qualidade é superchave.

→ Cada relação tem pelo menos 1 superchave (O conjunto de todos os atributos)

→ Ex: Estudante (Nome, Email, NMec, Curso)

### Superchaves

- { Nome, Email, NMec, curso }
- { Nome, Email, NMec }
- { Email, Nome }
- { NMec, Nome }
- { Email, NMec }
- { Email }
- { NMec }

### Chaves Candidatas

- { Email }
- { NMec }

Basicamente não caso não podemos ter { Nome }, { Curso } ou { Nome, Curso } como superchave pois não é de forma única o tuplo

**Chave Primária** - Chave principal selecionada de entre as chaves candidatas

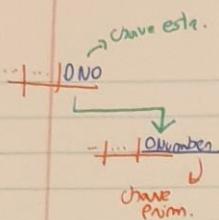
**Chave Única** - Chave Candidata não selecionada como primária

→ A escolha da primária de entre as candidatas é arbitrária

→ A chave primária Não Pode ter valor Null

→ Recomenda-se use de critérios na escolha (Como p.ex, atributo que nunca é alterado)

→ Ex: Primary → NMec Unique → Email



**Chave Estrangulina** - Conjunto de um ou mais atributos que é chave primária numa relação

→ Ex. Sl 3.14

### • Restrições de Integridade

: Visam garantir a integridade dos dados (garantidos pelo próprio SGBD)

#### : Tipos:

: **Domínio** (dos Atributos) - Forma primitiva elementos de integridade. Os campos devem obedecer ao tipo de dados e às restrições de valores admitidos para um atributo

: **Entidade** - Cada tuplo deve ser identificado de forma única com recurso a uma chave primária que não se repete e não pode ser null

: **Preferencial** - O valor de uma chave estrangeira ou é null, ou contém um valor que é chave primária na relação de onde foi importado

### • Regras de Codd

: Como verifico se um SGBD é relacional?

: Codd estabeleceu 12 regras que validam um sistema de modelo relacional

### 1. Representação da Info

- Numa BD rel.: todos os dados, incluindo o próprio dic. de dados, são representados de uma só forma  $\rightarrow$  Em tabelas Bidimensionais

### 2. Acesso garantido

- Cada elem de dados gica bem determinado pelo comb. do nome da tabela onde está armazenado, valor da chave primária e respectiva coluna

### 3. Supõe Sistematico de valores Nulos (null)

São supostos para rep. info não disp. ou não aplicável, indep. do domínio dos respectivos atributos.

### 4. Catálogo Ativo e Disponível

Metadados representados e armazenados da mesma forma que os próprios dados

5

## • Conversão do DFD em Modelo Relacional

: Um desenho conceptual de uma BD usando DFD pode ser rep. por um conjunto de relações (tabelas)

: Cada Conjunto de entidades e rel. do DFD irá gerar uma única relação com o nome do respectivo conjunto

## - Mapping Process - (Ex mas sl. after 3.20)

1. Para cada Entidade regular E, do esquema E/R, criar uma relação (tabela) R e incluir todos os atributos de E
  - Incluir os atributos compostos como elems singulares
  - Selecionar uma das chaves de E para primária de R

E:	Project
	PName   PNumber   Plocation

2. Cada Entidade fraca, do esquema é rep. por um relação (tabela) R que inclui os seus atributos assim como a chave primária da entidade dominante E que passará a ser chave estrangeira em R
  - Incluir os atributos compostos como elems singulares
  - Selecionar como chave primária de R, uma comb. da chave primária de E e da chave parcial de W

3. - Para cada Rel. 1:1 do esquema, envolvendo S e T:
  - = Escolher uma das relações (p.ex S) e incluir como chave estrangeira, a primária da outra
  - = Incluir em S, eventuais atributos de relacionamento.
  - ! Escolher como S uma relação com Participação Total

4. - Para cada rel. 1:N do E/R envolvendo as rel. S e T:

- = Escolher como S a rel. que rep. a entidade do lado N e como T a do lado 1.
- = Incluir em S como chave estrangeira, a primária de T
- = Incluir os atributos da rel. em S

5. - Para cada rel. N:M criar uma nova relação (tabela) P

- = Incluir como chave estrangeira os primários das relações que participam. A combinação destas será a primária da rel. P
- = Incluir os atributos da rel. em P

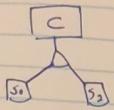
6. - Para cada atributo multi-valor A do E/R, criar nova relação (tabela) B

- = Incluir um atributo correspondente a A
- = Incluir a chave primária K da rel. que tem A como atributo
- = A chave primária de B é a comb de A e K

7. - Para cada relacionamento m-n (m>2) criar uma nova relação (tabela) P

- = Incluir como chaves estrangeiras as chaves primárias das rel. que rep. as entidades participantes
- = Incluir os eventuais atributos da rel.
- = A chave primária de P é, por norma, a combinação das chaves estrangeiras

### - D.E/A para relacionais - Especializações



- Há várias aproximações possíveis:

- Método 1

: Cria uma relações (tabela)  $L_1$  para a entidade de maior nível (C)

: Cria relações  $L_i$  para cada entidade de nível inferior. Incluir em cada uma destas a chave primária de C (para além dos atributos locais)

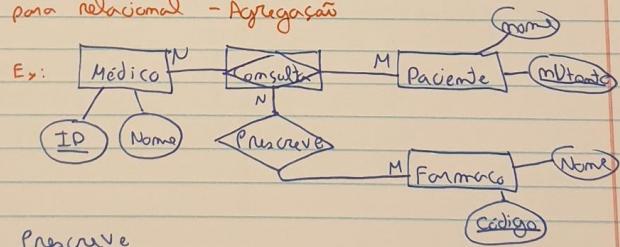
- Método 2

: Cria relações  $L_i$  para cada entidade de nível inferior. Incluir os atributos da superclasse e os locais

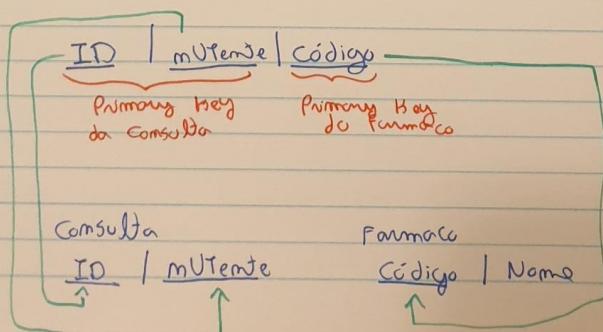
→ só funciona na esp. total

→ só recomendável em esp. disjunta (não sobrepõem bair duplação de info)

### - D.E/A para relacionais - Agregação



Prescreve



## Ch4

### SQL

: Structured Query Language

: Linguagem para definir, manipular e questionar uma Base de dados relacional

: Linguagem orientada ao processamento de conjuntos

: 2 Sublinguagens:

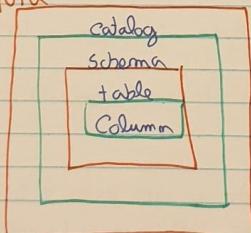
:: DDL - Data definition language

:: DML - Data Manipulation Language

: 1 Sublinguagem de controle BD

:: DCL - Data Control language

: Hierarquia



(Mas existem outros elementos)

: Notas

SQL usa: Tabela para se Relação  
Linha referir-se à Tuplo  
Coluna a Atributo

Cada instrução termina com ;

Comentários --

Comentários Multi-Linha /\* \*/

## • SQL - DDL

: Permite definir várias entidades da BD

: Utilizada para especificar a info acerca de cada relação

- o esquema de cada rel.
- o domínio de valores associados com cada atributo
- ...

: Há Comandos **NÃO** disponíveis em alguns SGBD

: Criar e eliminar uma BD.

- CREATE DATABASE dbname

- DROP DATABASE dbname

: Schema

: é um "máspac" que agrupa tabelas e outros elementos pertencentes à mesma app

- CREATE SCHEMA SchemaName [AUTHORIZATION user]

- DROP SCHEMA SchemaName

: Tipos de dados

: Podem variar de acordo com o SGBD

: Bóoleans Numbers (int; numeric (precisão, escala))

Character, Strings (char(m), varchar(m))

Date & Time (date, time)

Binary Object

Nota: Booleans não existem em  
SQL Server

Não  
disp.  
em  
SQL  
Server

### : Definição do domínio

:: O comando `CREATE DOMAIN` permite definir novos tipos de dados

:: Um domínio pode conter um valor de ~~default~~ (`ON NULL`) e restrições do tipo não null e check

- `CREATE DOMAIN domínio`

Ex: `Create domain compsalary Integer  
NOT NULL Check (compsalary > 475);`

### : Definição de novo Tipo

:: Alternativa mais limitada do que o domínio.

- `CREATE TYPE typename FROM datatype constraints`

### : Criar uma Tabela

- `CREATE TABLE tbname (A1 Datatype, A2 Datatype, ...,  
(integrity - constraint 1),  
...,  
(integrity constraint n));`

Ex: `Create Table Employee`

fname	Varchar(15),
ssn	Char(9),
bdate	DATE,
);	

### : Valores por Omissão

Usando o termo `default` podemos definir valores por omissão para cada coluna

Ex: `Salary Decimal(10,2) default 0,`

## : Restrições de Integridade

### Check (P)

- Impõe uma regra a um atributo
- Ex: Salary Decimal(10,2) Check (Salary > 12)
- Restrição aplicada a cada atributo referenciado
- Sempre que tuplo é adicionado ou modificado

### Not Null

- Atributo NÃO pode ser null

### Primary Key (A<sub>1</sub>, ..., A<sub>N</sub>)

- Define a chave primária → não pode ter valores repetidos
- Apenas uma por tabela
- Ex: Primary Key (Nome, eMail);

### Unique (A<sub>1</sub>, ..., A<sub>N</sub>)

- Usado para as restantes chaves candidatas
- Pode ter valores null (mas não repetidos)
- Ex: DName Varchar(15) Unique Not Null;

### Foreign Key

- Usado para declarar chaves estrangeiras
- Deve referenciar uma chave primária ou única
- Ex: super\_ssn Char(9) References Employee(ssn),
- Ex 2: Foreign Key (super\_ssn) references Employee(ssn);

Nota: Pode ocorrer uma violação da Integridade

→ Não deixa registrar ou -restrição  
→ não pode os registos associados  
ou alterar a chave estrangeira → Cascade

Podemos definir as ações alternativas  
On Delete e On Update com as opções

Chave estrangeira  
passa o null → Set Null  
passa a ter valor → Set Default  
Por omisão

**Nota:** As restrições podem ser de

- **Coluna** - Regulam-se apenas a uma coluna e descreve o que deve ser feito com a coluna

- **Tabela** - Referenciam mais do que uma coluna e ficam separados da def. das colunas

### : Restrições - Atribuição de nome

- Imaginemos que queremos alterar uma restrição de uma tabela devemos baptizar a restrição com um nome próprio para podermos referenciá-la

- Ex: Constraint EMPK Primary Key (SN)

### : Tabela Drop

:: O comando `drop table` remove da BD toda a info sobre tabela e os dados (tuplos)

- `DROP TABLE tablename`

:: Caso haja violação de rest. de integ. reg. a operação é negada

:: O cascade elimina a tabela e os elems referenciados **Não** existe em SQL Server !!  
(Termos de elim. lº o Constraint)

### : Tabela Alter

Usado para modificar o esquema da tabela ou restrições existentes

Add atributos - `ALTER TABLE nome ADD attr. Domains`

Add restrições - `ALTER TABLE nome ADD CONSTRAINT attr_name`

Elim Atributos - `ALTER TABLE nome DROP COLUMN attr_name`

Drop Restn. - ALTER TABLE tablename DROP CONSTRAINT

Alt. Attr. - ALTER TABLE name ALTER attr domain

Ex. sl 4.31.00/4.32

## Ch 5

### • Linguagem de Consulta / Interrogação de BD

#### : Algebra Relacional

- Linguagem formal do Modelo Relacional
- Conjunto básico de Operações

: Oferecem uma base teórica para a Linguagem de Consulta utilizada na prática

#### : Linguagem prática do Modelo Relacional - SQL

### • Algebra Relacional

#### : Questões

- Como formular interrogações?
- Que tipo de interrogações existem?
- Como são os resultados?

#### : Exemplos:

- Sequências de operações de Algebra relacional
- Permitam formular pedidos básicos de manipulação de info sobre uma ou mais relações

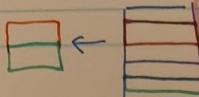
: Para formalizar uma interrogação necessitamos de um Conjunto de operadores que operam sobre as relações e devolvem uma nova relação

### • Algebra Relacional - Operações Básicas

#### → Seleção - $\sigma_{\text{selection condition}}(R)$ -

: Seleciona um subconjunto de tuplos da relação ( $t \in R$ ) que satisfazem a expressão booleana 'selection condition'.

Rel 2  $\sigma_{\text{sel.com}}(R_1)$  :



### : Op. de Comp:

- Permitem Comparar 2 atributos ou 1 atr e um valor

- $=, =/, \leq, \geq, <, >$

- Ex  $\sigma_{DNO=4} (Employee)$

### : Cond. Booleanas

- AND, OR, NOT

- Ex  $\sigma_{(DNO=4 \text{ AND } sal > 2500) \text{ OR } (DNO=5 \text{ AND } sal \leq 3000)} (Employee)$

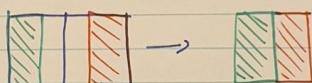
### → Projeção

- $\Pi_{\text{attr. list}} (R)$

(Attr list) Contém atributos da rel R

Resultado é uma nova rel só com os k atr. selecionados

Removidas as linhas duplicadas do resultado



### → Renomeação

- $R_{B_1, B_2, \dots, B_n}(A_1)$  -

- ou  $R_{A_2}(A_1)$  -

- ou  $R_{(B_1, B_2, \dots, B_m)}(A_1)$  -

1º Caso - ~~No caso~~ Resultado é uma nova rel, R2, com os atributos renomeados ( $A_1, A_2, \dots, A_m$ )

2º Caso - Só renomeamos a relação

3º Caso - Só renomeamos os atributos

$\rightarrow \text{União} - P \cup S = \{t : t \in P \vee t \in S\}$

- As tabelas têm de ser compatíveis (mesmo # attr. e attr com domínios compatíveis)



- Resultado é uma relação que inclui todos os atributos de P e S

$\rightarrow \text{Intersecção} - P \cap S = \{t : t \in P \wedge t \in S\}$

- As tabelas têm de ser compatíveis



- Resultado é uma relação que inclui os tuplos que existem em P e S simultaneamente

$\rightarrow \text{Diferença} - P - S = \{t : t \in P \wedge t \notin S\}$

- As tabelas têm de ser compatíveis



- Resultado é uma relação que inclui os tuplos de P que não existem em S

#### - Notas -

União e Intersecção são commutativas

$$P \cup S = S \cup P \quad P \cap S = S \cap P$$

Diferença não é comutativa

$$P - S \neq S - P$$

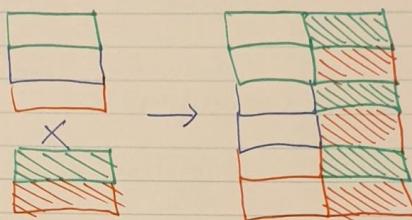
União e intersecção são associativas

$$P \cup (S \cup T) = (P \cup S) \cup T \quad P \cap (S \cap T) = P \cap (S \cap T)$$

### → Produto Cartesiano - $R \times S$ -

- Permite-mos combinar tuplos de relações diferentes

- O resultado é uma nova relação ( $Q$ ) que combina cada elemento (tuplo) de uma relação ( $R$ ) com um elemento (tuplo) da outra relação ( $S$ )



### → Junsão (Theta Join) - $R \bowtie_c S$ -

- Pode ser visto como o resultado das operações:

$$m_3 \leftarrow R_1 \times R_3 \\ O_c(m_3)$$

- C é uma <join condition> e toma a forma <cond> AND <cond> AND <cond> AND ...

- Em cada <cond> podemos usar operações de comparação (=, ≠, >, <, ...)

• Ver sl 5.19

### → Equi-Join

- Usado o operador "=" na cond de junsão
- Vamos ter sempre 2 colunas repetidas

→ Juntçāo Natural:  $R \bowtie S$

- Condicāo Implícita: Igualdade dos Atributos com o mesmo nome //!

- Atributos repetidos sāo normalizados

Ex:

X	Y	Z
a	c	x
b	d	y
g	e	z

$\bowtie$

X	Y	Z
b	d	y
g	e	z

? → Divisāo -  $R \div S$  -

- Dadas as relaçōes  $R(A_1, \dots, A_n, B_1, \dots, B_k)$  e  $S(B_1, \dots, B_k)$  o resultado inclui todos os tuplos de  $R_1(A_1, \dots, A_n)$  que tenham corrsp. com todos os tuplos de  $S$  em  $R_2(B_1, \dots, B_k)$

- H adj.  $R > \# \text{adj } S$

- Não existe em SQL, temos de fazer:

$$\rightarrow \text{π}_{R-S}(R) = \text{π}_{R-S}((\text{π}_{R-S}(R) \times S) - R)$$

Ex:

A	B		A
a	c		c
b	c		d
a	d	÷	
b	e		
a	a		
a	e		

## \* Algebra Relacional - Operações Estendidas

→ Semi-Join

Left Semi Join -  $R \times S = \pi_R(R \bowtie S)$

- Projeção dos attr. de R na junção natural de R com S

R	S
$\otimes$ $\oplus$	$y$ $z$
a   c	$\ominus$ g
b   d	e   h

$$R \rightarrow x y z$$
$$\begin{matrix} b \\ c \\ e \end{matrix} \rightarrow \begin{matrix} y \\ z \\ g \\ h \end{matrix}$$
$$\pi_R(S) = \begin{matrix} y \\ z \\ g \\ h \end{matrix}$$

Right Semi Join -  $R \times S = \pi_S(R \bowtie S) = (S \times R)$

- Proj. dos attr. de S na junç. natural de R com S

R	S
x   y	y   z
a   c	<del>d   g</del>
b   d	e   h

→ Inner VS Outer Join

Inner → Os tuplos que não estão relacionados são descartados (incluindo os valores null nos attr. da relação)

Outer → Incluimos no resultado todos os tuplos de uma (ou ambas) das relações. Compreender:  
- Os atributos que não são matched são preenchidos com NULL

→ Outer Join

Left Outer Join -  $R \bowtie S$

R		S	
A <sub>1</sub>	A <sub>2</sub>	B <sub>1</sub>	B <sub>2</sub>
a	c	d	g
b	d	e	h

A <sub>1</sub>	A <sub>2</sub>	B <sub>1</sub>	B <sub>2</sub>		
a	c	d	g	null	null
b	d	e	h	j	g

Right Outer Join -  $R \bowtie S$

R		S	
A <sub>1</sub>	A <sub>2</sub>	B <sub>1</sub>	B <sub>2</sub>
a	c	d	g
b	d	e	h

A <sub>1</sub>	A <sub>2</sub>	B <sub>1</sub>	B <sub>2</sub>		
b	d	d	g	null	null
		e	h		

Full Outer Join -  $R \bowtie S$

R		S	
A <sub>1</sub>	A <sub>2</sub>	B <sub>1</sub>	B <sub>2</sub>
a	c	d	g
b	d	e	h

A <sub>1</sub>	A <sub>2</sub>	B <sub>1</sub>	B <sub>2</sub>		
a	c	d	g	null	null
b	d	d	g	null	null
		e	h		

Nota: parsons sl 5.30

3 3 3

→ Agregação - grouping attr,  $\exists$   $\text{group by } (P)$  -

• Operações sobre vários tuplos da relação P

• Lista de funções:

- **avg** - média dos valores
- **min** - mínimo dos valores
- **max** - máximo dos valores
- **sum** - Soma dos valores
- **count** - Número dos valores

• Também podem ser usadas em projeções

Ous. atib. mās agregados sāo agrupados de  
forma q. mās haver valores repetitivos

- Ex:  $\Pi_{A_1, A_2, M=\text{avg}(A_3)} (R)$  → Ver Sl 5.32

- Ex csgrouping → Ver Sl 5.33

VI. Index (Exemplos) - Proj