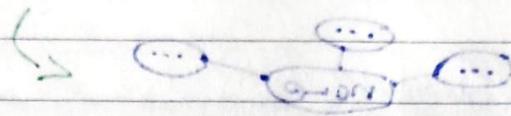


Modelo Entidade - Associação (EA)

- ## Nivel conceptual

- ↳ Durante a construção, não evite a preocupação de como será a implementação.

Entidade - tipo



- ↳ Abstrato feita para a descrição de um grupo de conceitos segundo determinadas características
 - ↳ Representada com um retângulo \rightarrow [NOME]
 - ↳ Nome maiúsculo, nem acentos nem opções

Atributo: color: glass are dark, transparent; very thin tibiae in both

- Propriedade usada para caracterizar uma entidade

- ↳ Representando con una eclisse → rome

- ↳ Note minusculo

Nota: no modelo EA não há informação sobre o seu domínio, mas deve ser indicado textualmente, porque é importante no modelo lógico.

Atributo-Chave

- ↳ Ter a capacidade de identificar se a subigualdade uma entidade
 - ↳ Principal meio de identificação de uma entidade - tipo
 - ↳ Os nomes ficam a sublinhado: → (nome)

Atributo Simples / Atômico

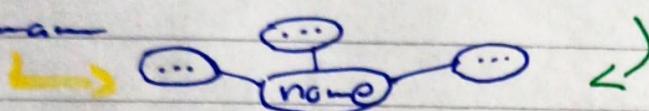
(A3) Exemplo - atributo nome

- ↳ não é possível dividir em vários campos → nome

Atributo Compósito

- ↳ pode ser decomposto em atributos simples

- ↳ possui campo valor concatenado dos valores dos atributos simples que o formam



Atributo Multivalor

- ↳ possui múltiplas valores para a mesma entidade → nome

Atributo Derivado

- ↳ não necessita de ser armazenado, dado que pode ser derivado a partir de outro atributo, de outras entidades → nome

Associação - tipo

- ↳ relações entre diversas entidades

- ↳ representa-se por um losango → Nome

→ Grande Associação, estratégia sup. proj., generalizações e extensões

- ↳ indica o nº de entidades que participa na associação

↓
• Unária = recursiva; 1ª uma entidade

• Binária = duas entidades → relações entre duas entidades

• Ternária = 3 entidades → relações entre três entidades

• N-ária = muitiplas entidades → relações entre mais de duas entidades

Modelo Entidade-Associação-Entidade (EER)

- Foram introduzidos outros conceitos no modelo EER:
- generalização
 - associação - tipo de exclusividade (1..n) → M - M ou 1..n - n
 - entidade associativa

Generalizações são relações de inclusão

Uma entidade E é uma generalização das entidades E₁, ..., E_n; em cada ocorrência de E₁, ..., E_n é também E

Representa-se com um círculo



Cobertura

- Total → entidade genérica é obrigatoriamente uma das sub-entidades (deve ter filha na sola)
- Parcial → genérica não corresponde com as sub-entidades

Propriedade (d) → genérica não é uma das sub-entidades

Sobreponer (o) → genéricas podem ser variadas sub-entidades

→ Cardinalidade da Associação
↳ especifica como as entidades estão relacionadas.

- um - para - um (1:1) → cada tuplo de A está associado a um tuplo de B.
- um - para - muitos (1:N) → cada tuplo de A está associado a muitos, nenhuma ou múltiplos de B.
- muitos - para - muitos (N:N)

→ Participação na Associação
↳ especifica como a existência de cada instância de uma entidade depende ou não das outras entidades da associação.

- Parcial → não depende de outros
- Total → depende da existência (dupla linha)

Entidade - Fraca
↳ sempre relacionada com uma entidade - tipo
↳ ter sempre participação total na relação
↳ representada por um duplo retângulo → ||NOME||

↳ ter um atributo chave parcial → Nome
↳ o relacionamento identificador é representado por: → Nome

Regras de texto para ER
• nome = entidade
• verbo = associação
• adjetivo = atributo de entidade
• advérbio = atributo de associação

Normalização

Regras para um Bom Modelo Relacional

Regra 1: • ser fácil de explicar o seu significado
• não misturar atributos de regras distintas

Regra 2: • evitar possibilidade de ocorrerem anomalias nas operações de inserção, remoção ou alteração
• se não for possível garantir que os gestores de DB saibam da existência das anomalias

Regra 3: • evitar atributos que possam ter valor NULL

Regra 4: • evitar regras que tenham atributos relacionados com combinações de PK e FK

Decomposição

↳ a redundância pode ser ultrapassada decompondo as regras em regras mais pequenas

Dependência entre dados

- Funcionais (DF_1) \rightarrow 1NF, 2NF, 3NF, BCNF
 - Multiválor (DN) \rightarrow 4NF
 - de Sung (Dd_1) \rightarrow 5NF
- $\left. \begin{array}{l} \\ \\ \end{array} \right\}$ Regras

Integridade de Valor não nulo

Basta dizermos que

↳ determina se o campo pode ou não receber valor NULL

Dizemos que o campo é de tipo não nula

obrigatório ou o campo é facultativo: Existe
obrigatória quando o estudante tem que dar

ou não dá para dar, mas se der é sempre
obrigatório, se não der é facultativo.
se não der é facultativo, se der é sempre
obrigatório ou não dá para dar.

OBS: vale isto mesmo para estudante nativo: Existe

→ se não der é facultativo, se der é sempre
obrigatório para estudante nativo: Existe

existe se der é sempre obrigatório para estudante nativo: Existe
se não der é facultativo para estudante nativo: Existe

obrigatório para estudante nativo: Existe

campo?	70.08, 70.09, 70.10	(A) não é?
	70.11	(B) é?
	70.12	(C) é?

Modelo Relacional

↳ São os elementos principais:

- esquema de relações
- tabelas
- atributo
- domínio

Esquema de Relações

↳ Descreve as colunas da tabela

↳ Nome - R

↳ Atributo - A₁, ..., A_n: conjunto de atributos

↳ Domínio: conjunto de valores de A₁, ..., A_n em que cada valor

Exemplo:

Esquema de Relações: PESSOA (nome: string, idade: integer)

Relações: {<"André", 19>, <"Bruna", 20>, <"Neyke", 19>}

Atributo

↳ identifica uma propriedade da relação

Domínio

↳ conjunto de valores que cada atributo pode tomar

↳ " " " " : domínio que caracteriza o atributo

NULL → ausência de valor

Superchave e Chave

↳ subconjunto de 1 ou mais atributos que identificam um tuplo

Chave Primária

- ↳ chave candidata com maior significado e menor nº de atributos
- ↳ representa-se a sublinhado

Métricas de Integridade

- ↳ condições impostas ao esquema que restringem os dados que podem existir nas instâncias da base de dados

Integridade de Entidade → Chave Primária

- ↳ garante que o relações de entidades possuem uma chave única e que nenhum atributo dessa chave tem o valor NULL

Integridade Referencial → Chave Estrangeira

- ↳ envolve duas relações associadas; usada para manter a coerência entre os valores dessas relações

Chave Estrangeira

- ↳ conjunto de atributos FK que referenciam a chave PK de outra relação

Integridade de Domínio

- ↳ o domínio é um conjunto de valores, a partir dos quais os valores inseridos nos adicionais às colunas da tabela

Integridade de Coluna

- ↳ determinar valores aceitáveis para aquela coluna

Passo 8A: Multiplos heróis

PK(G) \rightarrow FK(T_i) G-geral

PK(G) \rightarrow PK(T_i) | T_i \rightarrow especialidade

atributo de especialidade

A.3 student

student3

studentA

studentB

Passo 8B

studentA

studentB

student3

Passo 8C

(..., student) SNOM : student3 - > student

Passo 8D(a) Guardar o endereço de email student, removendo student (↓)

(..., student) SNOM : email student - > student

Sumarizado

(a.9) Inserindo, consultas e alterações a banco de dados e banco de alunos

consultas de {email} : email \rightarrow "end" do aluno

Modelo EA

entidade - tipo: professor, estudante, associado 1:1 ou 1:N
associado N:N

Atributo Simples

- " Cognato
- " Multivalor
- " char

Modelo Relacional

entidade - relação (professor, aluno) - Atributos

FK

relações entre 2 FKs

atributo

conjunto de atributos simples

relação FK

PK - importe para - Atividade

atividade

(iii)

(estudante, end, email) consulta : busca

estudante buscando endereço, no topo da página e removendo end.

(estudante, end, email) consulta : estudante : D. consulta

Passagem de Modelos

Passagem : AB para

Modelo EA

Entidade
Atributo
Chave

Modelo Relacional

Mais
Atributo
Pk, Fk e Ak

Passagem : (ST) 3NF -> (S) 3NF

AB para

Passagem

Passo 1 - Entidades : NOME (atributos, ..., ...)

↳ atributos simples, atributos compostos e chave (PK).

Passo 2 - Entidades Filhas : NOME (atributos, ...)

↳ atributos simples e compostos e chave parcial (PK)

↳ chave de "pai" e PK e FK : FK : {Nome} de NOME. nome

(1:N)

A.B obedece

Passo 3A - Chave estrangeira para a parte entidade obrigatória

U:U obedece

Passo 3B

U:U obedece

Judicar

U:U obedece

Passo 3C : Judicar obedece

(1:N)

U:U obedece

obrigado obedece

obrigado obedece

obrigado obedece

Passo 4A - Chave estrangeira da 1º é chave P. de 1

Passo 4B

(N:N)

Passo 4C : ASSOCIAÇÃO (pk1, pk2, atributos)

↳ chaves primárias e estrangeiras são as primárias dos elementos da relação

Passo 5 : Atributo Multivalor : ATRIBUTO (chave(clemano), Fk da entidade)

Passo 6

Dependências Funcionais (DFs)

↳ X determina Y se para cada valor de X existe um só valor de Y

$$X \rightarrow Y \quad \leftrightarrow$$

Nominação de DFs (mejores de Inféria)

1-Reflexividade $X \supseteq Y \Rightarrow X \rightarrow Y$

2-Aumento $X \rightarrow Y \Rightarrow [X, Z] \rightarrow [Y, Z]$

3-Transitividade $\{X \rightarrow Y, Y \rightarrow Z\} \Rightarrow X \rightarrow Z$

4-Decomposição $X \rightarrow Y, Z \Rightarrow X \rightarrow Y \text{ e } X \rightarrow Z$

5-União $\{X \rightarrow Y, X \rightarrow Z\} \Rightarrow X \rightarrow \{Y, Z\}$

6-Pseudotransitividade

$$\{X \rightarrow Y, \{Y, W\} \rightarrow Z\} \Rightarrow \{X, W\} \rightarrow Z$$

Caracterizações de DFs

• Parcial - remoção de ~~algum~~ atributo de X, não deixa de determinar Y

• Total - remoção de algum atributo de X deixa de determinar Y

• Transitiva - se existir um conjunto de atributos Z, que depende de X e determina Y:

$$\{X \rightarrow Z, Z \rightarrow Y\} \quad \leftrightarrow$$

Fecho

↳ O fecho de um conjunto de DFs F é o conjunto de todas as DFs que podem ser inferidas a partir de F

$$F^+ \quad \leftrightarrow$$

↳ O fecho de um conjunto de atributos X, é o conjunto de todos os atributos que podem ser inferidos por X

$$X^+ \quad \leftrightarrow$$

Algoritmo → Calcular:

1- $X^+ = X$

2- Para cada $X \rightarrow Z$: $X^+ = X^+ \cup Z$

↳ Repetir até não encontrar mais Zs

Exemplo:

$$F = \{A \rightarrow B, C \rightarrow \{D, E\}, \{A, C\} \rightarrow F\}$$

$$\{A\}^+ = \{A, B\}$$

$$\{B\}^+ = \{B\}$$

$$\{C\}^+ = \{C, D, E\}$$

$$\{A, C\}^+ = \{A, C, F, B, D, E\}$$

Cobertura

Sejam F e G dois conjuntos de DFs

Diz-se que G é cobertura de F se

G+ incluir todas as DFs de F

Verificar se é cobertura:

Para cada DF: $X \rightarrow Y$ em F:

1- Calcular X^+ com G

2- Se $Y \in X^+$ calculado, continuar
caso contrário, não é cobertura X



→ G+ e cobertura

Normalização

1NF

- todos os atributos são atômicos, isto é, valor único
- decompor atributos compostos em outros atributos
- decompor atributos multivalor em relações

2NF

- todos os atributos não chave dependem da chave
↳ da totalidade
- decompor relações com FK

3FN

- todos os atributos dependem apenas da chave e nunca de outros não chave
- decompor em relações

Equivalecia de DFs

↳ Sejam G e F dois conjuntos de DFs, diz-se q são equivalentes se

$$\underline{G^+ = F^+} \quad \Leftrightarrow \quad G \equiv F$$

↳ Se G é cobertura de F e F é cobertura de G, então são equivalentes

Cobertura Mínima (CM)

↳ Diz-se que G é uma CM de um conjunto de DFs F, equivalente a F.

$$|F \equiv CM| \quad \text{e} \quad |F^+ = CM^+|$$

Calcular CM

$$1 - G = F$$

2 - Substituir cada DF $X \rightarrow \{A_1, \dots, A_n\}$ por n DFs: $X \rightarrow A_1, \dots, X \rightarrow A_n$

3 - Para cada DF $X \rightarrow A_i$ em G com múltiplos atributos no lado esquerdo:

 Para cada atributo $B \in X$, verificar se $(X-B) \rightarrow A_i$ é nula, substituir $X \rightarrow A$ com $(X-B) \rightarrow A$

4 - Para cada $X \rightarrow A \in G$, calcular X^+ a partir de $\{G - \{X \rightarrow A\}\}$

 se $A \in X^+$, remover $X \rightarrow A$ de G

Encontrar Chaves

Seja R uma relação e F um conjunto de DFs:

$$1. K := R$$

2. Para cada atributo A $\in K$:

- calcular $\{K-A\}^+$ em relação a F
- se contém todos os atributos:
 $|K| := K - \{A\}$

Heurísticas para calcular chave

1. Todos os atributos q só aparecem à esquerda

2. Todos os que não aparecem

3. Calcular fecho desse conjunto de atributos e se contiver tudo, é chave.

Preservação de Informações (lossless join)

Seja F um conjunto de DFs de $R(A_1, \dots, A_n)$ e sua decomposição $D = \{R_1, \dots, R_m\}$

1 - Criar uma matriz S com m linhas (uma para cada bloco de D) e n colunas, uma para cada atributo de R

2 - Para cada entry, se o atributo estiver na relação da linha, colocar a_{ij} , sendo b_{ij}

continua →

→ Cont. loss less join

3- Para cada DF: $X \rightarrow Y \in F$:

• Para todos os linhas com os mesmos valores nas colunas de X, igualar os valores de Y:

i) Se um dos valores for "a", igualar outras a esse valor

ii) Se só houver "b", igualar a esse valor para ficar tudo igual

↳ Repetir 3. até não mudar

4- Se houver uma linha que com as, conclui-se que a decomposição é lossless join

Algoritmo de Síntese

↳ Passagem para a 3NF

↳ Presença das dependências

↳ Propriedade de lossless join

↳ Algoritmo:

1- Determinar G, CM de F

2- Decompor com base em G

| Ex: $G = \{A, B\} \rightarrow C, C \rightarrow D, D \rightarrow E\}$

| $D = \{P_1(A, B, C), P_2(C, D), P_3(D, E)\}$

3- Se a chave candidata aparece numa das PIs, então verifica a condição lossless join e está na 3NF

4- Senão, criar uma relação com a mesma:

| Ex: $Ak = \{B, F\}$

| $P_4(B, F)$

Outros

Atributo primo - pertence a uma Ak.

Álgebra Relacional

L Coleção de operações de manipulação de relações no modelo relacional

L As operações produzem novas relações

L Uma sequência de operações é uma expressão

Operador	Operação	
σ	Seleção	Especificar para a BD
π	Projeção	
ρ	Renomear	
\bowtie	Junção	
J	Agregação	

Operador	Operação	
\cup	União	Sobre Conjuntos
\cap	Interseção	
-	Diferença	
\times	Produto Cartesiano	
\div	Divisão	

Seleção σ

L Seleciona todos os tuplos que satisfazem a condição

$$\sigma_{\text{condição de seleção}}(R)$$

↳ Atributos com comparadores ($=, \neq, \dots$)

↳ Cláusulas com lógicas (\wedge, \vee, \neg)

→ Comutativa

$$\sigma_{P_1}(\sigma_{P_2}(R)) = \sigma_{P_2}(\sigma_{P_1}(R)) =$$

$$= \sigma_{(P_1 \wedge P_2)}(R)$$

Projeto π

L Obtém uma nova relação com os atributos indicados

$$\pi_{(listadeatributos)}(R)$$

↳ atributos vêm na mesma ordem que estão na lista

Aviso: Remove tuplos repetidos caso o PK não estiver na lista

$$\pi_{A_1}(\pi_{A_2}(R)) = \pi_{A_1}(R)$$

se $A_2 \subseteq G$:

Renomear ρ

L Permite renomear a relação R ou os atributos

$$\rho_S(R) \text{ ou } \rho_{(B_1, B_2)}(R) \text{ ou } \rho_{S(B_1, B_2)}(R)$$

L Junção \bowtie \rightarrow Junção - Ø (tudo)

L Permite combinar tuplos de duas relações que obedecem a uma condição de atributo

$$R \bowtie_\theta S$$

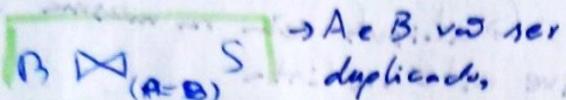
Ø condição de junção

$$R \bowtie_\theta S \Leftrightarrow \sigma_\theta(R \times S)$$

Equi-Junção

- Junção em que apenas se usa o operador " $=$ "

- A relação resultante tem pelo menos um par de atributos com o mesmo valor



Junção Natural \bowtie ou *

- Junção com pelo menos um par de atributos com o mesmo nome

- Pode ser necessário renomear antes da junção

- Quando não omitidos os atributos, são considerados os como mesmo nome e tipo

$R * S$

Semi-junção

- à esquerda $R \bowtie_p S$

- mantém todos de R e mete a NULL nos que não satisfazem

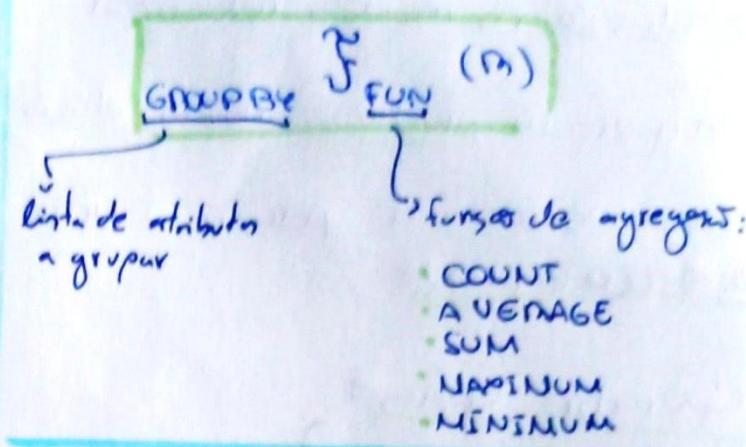
- à direita $R \bowtie_p S$

- completa $R \bowtie_p S$

- mantém todos os atributos mete NULL nos que não satisfazem

Agregação

- Agrupar tuplos e numericas informacionais



Relações Encadeadas

- Operações numa única expressão

- só aplicada uma de cada vez em resultados intermedios:

$\text{RESULTADO} \leftarrow \sigma_{\dots}(\dots)$

Operações sobre Relações (conjuntor)

- Única (U) → todos os tuplos (remove repetidos)

- Intersection (\cap) → and

- Diferença ($-$) → $(A \cap \bar{B})$

- Produto Cartesiano (\times) - combina todos os tuplos de 2 relações linha por linha

- Divisão (\div) - variares de uma relação combinada com os tuplos de outra

Outros

- Relações compatíveis têm o mesmo nº de atributos e o mesmo domínio

SQL

DDL - Data Definition Language

- CREATE
- ALTER
- DROP

DML - Data Manipulation Language

- SELECT
- INSERT
- UPDATE
- DELETE

TCL - Transaction Control Language

- BEGIN TRANSACTION
- COMMIT
- ROLLBACK

Criar Tabela

```
CREATE TABLE tablename (
    columnname datatype [constraint]
    ( )
);
```

PRIMARY KEY
NOT NULL
CHECK (condition)

⚠ Multiplos primary key ou
PRIMARYKEY (..., ...)

PREFERENCES tablename (column)

Primary key must
 be unique
 and non null

Remover Tabela

```
DROP TABLE IF EXISTS tablename  

CASCADE;
```

Alterar Tabela

```
ALTER TABLE tablename {  

  ADD columnname datatype  

  DROP columnname CASCADE  

  ALTER columnname SET value  

};
```

Inserir numa Tabela

```
INSERT INTO tablename  

VALUES (...);
```

Atualizar Tabela

```
UPDATE tablename  

SET column = value  

WHERE condition;
```

Limpar Tabela

```
DELETE FROM tablename  

[WHERE condition]
```

Consultas na Tabela

```
SELECT columns  

FROM tablename  

[WHERE condition]
```

Tipos de dados

- integer
- varchar(n) → até size n
- boolean
- BIT
- date
- char(n) → size fixo
- decimal(n,m)
 - n casas, m delas são decimais

Aggregadas

SELECT ... Function(column) [AS ...]
FROM ...
GROUP BY ... → os mesmos

COUNT
SUM
AVG
MAX / MIN

Juntas

INNER JOIN ○○

LEFT JOIN ○○

RIGHT JOIN ○○

FULL JOIN ○○

NATURAL JOIN → não usa ON

↳ junta columns com mesm nome

Criar Vistas

CREATE VIEW name
AS SELECT ...
FROM ...

Apagar Vistas

DROP VIEW name

Outros

→ Enum:

CREATE TYPE name AS ENUM(listofvalues);

DISTINCT → remove duplicados

AS → renomear

BETWEEN (...) AND (...) → return Boolean

LIKE corresponde padrão. '%': nº de asteriscos → 1 ou caractere

IS [NOT] NULL → verificar se é NULL

ORDER BY columns [ASC / DESC]

WHERE ssn IN (list)

IN (SELECT ...)

LIKE "..."

BETWEEN 0 AND 10

CAST(bdate AS CHAR(20))

COUNT
SUM
AVG
MAX
MIN

} Função de Agregadas

→ Pode ser usada no SELECT e no HAVING

GROUP BY (...)

HAVING ... (ex: COUNT(*) > 2;)