Bases de Dados

NORMALIZAÇÃO

Desenho de BDs relacionais

- Como é que se desenha uma boa BD relacional?
- Qual é o critério para quantificar a qualidade e funcionalidade de um modelo relacional?
- Porque é que um determinado agrupamento dos atributos em relações é melhor do que outro agrupamento?

Qualidade de um modelo relacional

- **Do ponto de vista lógico ou conceptual**: como é que os utilizadores interpretam o significado das relações e dos seus atributos.
  - Um bom modelo do ponto de vista lógico permite que os utilizadores compreendam claramente o significado dos dados e os possam manipular corretamente.

- **Do ponto de vista da implementação**: como é que os tuplos das relações são guardados e manipulados fisicamente na BD.
  - Um bom modelo do ponto de vista da implementação:
    - garante uma maior eficiência das operações de acesso aos dados,
    - minimiza o espaço necessário para guardar os tuplos das relações
    - evita informação incorreta ou supérflua.

Regras Para o Bom Desenho de BDs Relacionais – Regra 1

Os atributos de uma relação devem representar apenas uma entidade ou um relacionamento.

- Atributos de entidades ou relacionamentos diferentes devem estar separados o mais possível. Apenas chaves externas devem ser usadas para referenciar outras entidades.
- É mais fácil explicar o significado de uma relação se esta representar apenas uma entidade ou relacionamento. Evita ambiguidades no significado das relações.

- Exemplo de uma boa relação do ponto de vista lógico mas que viola a regra 1:

  EmpDep(nomeEmp, numBI, endereço, dataNasc, numDep, nomeDep, gerenteBI)

  Problemas com a relação EmpDep:

  - Os valores dos atributos nomeDep e gerenteBI aparecem repetidos para os empregados que trabalham num mesmo departamento.

Regras Para o Bom Desenho de BDs Relacionais – Regra 2

Evitar a possibilidade de ocorrerem anomalias nas operações de inserção, remoção ou alteração.

Se por razões de eficiência isso não for possível, garantir que os utilizadores/programas que manipulam a BD conhecem essas anomalias e as evitam.

EmpDep(nomeEmp, numBI, endereço, dataNasc, numDep, nomeDep, gerenteBI)

Exemplo de anomalias de **inserção** em EmpDep:

- Não é possível inserir um novo departamento a menos que seja associado a um empregado.
- Ao inserir um empregado é necessário garantir que os valores dos atributos nomeDep e gerenteBI são consistentes com os dos restantes empregados desse departamento.

Exemplo de anomalias de **remoção** em EmpDep:

- Se removermos o último empregado para um determinado departamento, então a informação desse departamento também é removida.

Exemplo de anomalias de **alteração** em EmpDep:

- A alteração do nome de um departamento leva a que essa alteração tenha que ser feita sobre todos os tuplos dos empregados que nele trabalham.

Regras Para o Bom Desenho de BDs Relacionais – Regra 3

Evitar atributos que possam ter valores NULL numa grande parte dos tuplos duma relação.

- Solução:
  - Colocar esse tipo de atributos em relações separadas juntamente com a chave primária.
  - Minimiza o espaço necessário para guardar os tuplos da relação e evita problemas no cálculo de funções de agregações sobre esses atributos.
- Exemplo:
  - Se apenas 5% dos empregados tiverem Gabinete individual não faz sentido incluir um atributo numGabinete na relação EmpDep. Uma melhor solução será criar uma relação Gabinete(empBI, numGabinete) para guardar essa informação.

Regras Para o Bom Desenho de BDs Relacionais – Regra 4

Evitar relações que não verificam a **propriedade de junção-não-aditiva.**

Exemplo:

nomeEmp	BI	nProj	nomeProj	local	Horas
José	123456	1	A	Porto	100
Maria	987654	2	B	Porto	200
Rui	456789	3	C	Braga	300

Se separado em:

nomeEmp	local
José	Porto
Maria	Porto
Rui	Braga

BI	nProj	nomeProj	local	Horas
123456	1	A	Porto	100
987654	2	B	Porto	200
456789	3	C	Braga	300

Voltando a juntar adiciona dados errados

nomeEmp	BI	nProj	nomeProj	local	Horas
José	123456	1	A	Porto	100
José	987654	2	B	Porto	200
Maria	123456	1	A	Porto	100
Maria	987654	2	B	Porto	200
Rui	456789	3	C	Braga	300

Normalização

Processo de análise que minimiza redundância de dados e minimiza anomalias nas operações de modificação dos dados. As relações que não satisfazem certas **propriedades** – **formas normais** – são sucessivamente decompostas em relações mais pequenas de modo a satisfazerem as propriedades pretendidas

Formas normais

- 1NF – Primeira forma normal
- 2NF – Segunda forma normal
- 3NF – Terceira forma normal
- BCNF – Forma normal de Boyce–Codd
- 4NF – Quarta forma normal
- 5NF – Quinta forma normal

- Nem sempre é necessário ou possível normalizar uma BD até à última formal normal (por vezes, 3NF ou BCNF é suficiente).

1NF – Primeira Forma Normal

Um esquema relacional está na primeira forma normal se todos os atributos forem atómicos (não divisíveis).

EmpDep(nomeEmp, numBI, endereço, dataNasc, numDep, nomeDep, gerenteBI)

- Decompor atributos compostos em atributos atómicos.
  - O atributo nomeEmp pode ser decomposto em (pNome, uNome).
  - O atributo endereço pode ser decomposto em (rua, cidade, codPostal).

Departamento(nome, num, {localização})

- Decompor atributos multi-valor em relação com chave externa.
  - A relação pode ser decomposta em Departamento(nome, num, localização)
  - mas a melhor solução é decompor em:
    - Departamento(nome, num) e
    - LocalizaçõesDep(numDep, localização) pois evita redundância.

Dependências Funcionais

X → Y

Y depende funcionalmente de X

X chama-se o **determinante** da dependência funcional.

Considerando a seguinte relação: *EmpProj(empBI, numProj, horas, nomeEmp, nomeProj, localizaçãoProj)*

Dependências funcionais da relação EmpProj:

- empBI → nomeEmp
- numProj → {nomeProj, localizaçãoProj}
- {empBI, numProj} → horas

2NF – Segunda Forma Normal

Um esquema relacional está na segunda forma normal se:

- Está na 1NF.
- Todos os atributos não-chave dependem por completo da chave primária.

Na relação EmpProj(empBI, numProj, horas, nomeEmp, nomeProj, localizaçãoProj), as dependências funcionais

empBI → nomeEmp e

numProj→{nomeProj, localizaçãoProj} violam a 2NF.

Normalização 2NF

Decompor em relações com chave externa de forma a que, em cada relação, todos os atributos não-chave dependam por completo da chave primária correspondente.

EmpProj(empBI, numProj, horas, nomeEmp, nomeProj, localizaçãoProj)

Emp(empBI, nomeEmp)

EmpProj_1(empBI, numProj, horas, nomeProj, localizaçãoProj)

Proj(numProj, nomeProj, localizaçãoProj)

EmpProjHoras(empBI, numProj, horas)

3NF – Terceira Forma Normal

Um esquema relacional está na terceira forma normal se:

- Está na 2NF.
- Nenhum atributo não-chave depende por transitividade da chave primária.

Na relação EmpDep(nomeEmp, numBI, endereço, dataNasc, numDep, nomeDep, gerenteBI),

numBI → {nomeDep, gerenteBI} é uma dependência transitiva que viola a 3NF.

Normalização 3NF

Decompor em relações de forma a que, em cada relação, nenhum atributo não chave dependa por transitividade da chave primária correspondente.

EmpDep(nomeEmp, numBI, endereço, dataNasc, numDep, nomeDep, gerenteBI)

Emp(nomeEmp, numBI, endereço, dataNasc, numDep)

Dep(numDep, nomeDep, gerenteBI)

BCNF – Forma Normal de Boyce–Codd

- Um esquema relacional está na forma normal de Boyce–Codd se para qualquer dependência funcional X → A o determinante X é uma chave candidata.

- A diferença para 3NF é a inexistência da possibilidade de A ser um atributo de uma qualquer chave.
- Se um esquema relacional está na BCNF então também está na 3NF.
- O inverso pode não ser verdadeiro.

- Na prática, quando um esquema relacional está na 3NF, normalmente também está na BCNF.
- A exceção é quando na 3NF existe uma dependência X → A em que X não é uma chave e A é um atributo de uma chave.

Normalização BCNF

Na relação Propriedade, {numRegisto} e {concelho, numLoteamento} são chaves.

Propriedade(<u>numRegistro</u>, <u>concelho</u>, <u>numLoteamento</u>, taxa)

Registo(<u>numRegisto</u>, <u>numLoteamento</u>, taxa)

TaxaConcelho(<u>taxa</u>, concelho)

A normalização BCNF não verifica a propriedade de preservação das dependências, podendo originar a perda de dependências funcionais.

Dependências Multi-Valor (MVD)

$$X \twoheadrightarrow Y$$
$$X \twoheadrightarrow Z$$

- Restrição entre 2 atributos multi-valor independentes da mesma relação.
- Para manter a relação consistente, uma MVD obriga a repetir todos os valores de um atributo para cada valor do outro atributo.
- A dependência multi-valor resulta do facto de juntarmos dois relacionamentos 1:N na mesma relação.

Dependências Multi-Valor (MVD)

Relação EmpProjsDeps

- relaciona cada empregado com os projetos em que trabalha e os seus dependentes.
- Um empregado pode trabalhar em vários projetos e ter vários dependentes.

nomeEmp ↠ nomeProj

nomeEmp ↠ nomeDep

EmpProjsDeps		
nomeEmp	**nomeProj**	**nomeDep**
Silva	Proj_X	João
Silva	Proj_X	Maria
Silva	Proj_X	Alberto
Silva	Proj_Y	João
Silva	Proj_Y	Maria
Silva	Proj_Y	Alberto

4NF – Quarta Forma Normal

- Um esquema relacional está na quarta forma normal se:
  - Respeita BCNF
  - para cada dependência multi-valor X ↠ Y, X é uma chave.

- A 4NF evita os problemas de consistência e redundância relacionados com as dependências multi-valor.

- As dependências multi-valor da relação EmpProjsDeps violam a 4NF:
  - nomeEmp ↠ nomeProj
  - nomeEmp ↠ nomeDep

Normalização 4NF

EmpProjsDeps(nomeEmp, nomeProj, nomeDep)

EmpProjsDeps		
nomeEmp	**nomeProj**	**nomeDep**
Silva	Proj_X	João
Silva	Proj_X	Maria
Silva	Proj_X	Alberto
Silva	Proj_Y	João
Silva	Proj_Y	Maria
Silva	Proj_Y	Alberto

EmpProjs(nomeEmp, nomeProj)

EmpDeps(nomeEmp, nomeDep)

nomeEmp	nomeProj
Silva	Proj_X
Silva	Proj_Y

nomeEmp	nomeDep
Silva	João
Silva	Maria
Silva	Alberto

5NF – Quinta Forma Normal

Fornecimento(nomeForn, nomeComp, nomeProj)

R1(NomeForn, NomeComp)

R2(NomeForn, nomeProj)

R3(NomeComp, nomeProj)