

```
(SELECT DISTINCT R.A
FROM R)
INTERSECT ( (SELECT S.A FROM S)
UNION
(SELECT T.A FROM T))
```

5

Roteiro

- União, interseção, diferença
- Subconsultas
- Agregação
- Valores null

Leitura do livro “SQL for Nerds”: capítulo 4, “More complex queries”

2

Para Conservar Duplicações

Compra (comprador, vendedor, loja, produto)
Pessoa(nome, tel, cidade)

```
(SELECT nome
FROM Pessoa
WHERE cidade=“Curitiba”)

UNION ALL

(SELECT nome
FROM Pessoa, Compra
WHERE comprador=nome AND loja=“Ponto Frio”)
```

Renomeação de Colunas

Produto	PNome	Preco	Categoria	Fabricante
	lapis	\$19.99	papelaria	Farber
	lapiseira	\$29.99	papelaria	Farber
	camera	\$149.99	fotografia	Canon
	televisao	\$203.99	eletronicos	Hitachi

```
SELECT Pnome AS nomeProd, Preco AS precoVenda
FROM Produto
WHERE Preco > 100
```



Consulta com renomeação

nomeProd	precoVenda
camera	\$149.99
televisao	\$203.99

3

Subconsultas

Compra (comprador, vendedor, loja, produto)
Pessoa(nome, tel, cidade, cpf)

Uma subconsulta produzindo um valor único:

```
SELECT Compra.produto
FROM Compra
WHERE comprador =
(SELECT nome
FROM Pessoa
WHERE cpf = '123456789');
```

Neste caso, a subconsulta retorna um único valor.

Se retornar mais de um, **erro em tempo de execução.**

7

União, Interseção, Diferença

Compra (comprador, vendedor, loja, produto)
Pessoa(nome, tel, cidade)

```
(SELECT nome
FROM Pessoa
WHERE cidade=“Curitiba”)

UNION

(SELECT nome
FROM Pessoa, Compra
WHERE comprador=nome AND loja=“Ponto Frio”)
```

Da mesma forma, você pode usar **INTERSECT** e **EXCEPT**. Os tipos/nomes dos atributos devem ser os mesmos. Caso contrário: rename.

4

Compra (comprador, vendedor, loja, produto)
Pessoa(nome, tel, cidade, cpf)

A mesma consulta pode ser escrita sem usar subconsulta:

```
SELECT Compra.produto
FROM Compra, Pessoa
WHERE comprador = nome AND cpf = '123456789'
```

Esta consulta é equivalente a anterior se cpf for uma chave e a chave '123456789' existir na base de dados; caso contrário são diferentes.

8

Subconsultas que retornam relações

Produto (nome, preço, categoria, fabricante)
 Companhia (numCia, nome, valorAcao, pais)
 Compra (comprador, vendedor, loja, produto)

Encontre o nome das companhias que fabricam produtos comprados por 'Joao'.

```
SELECT Companhia.nome
FROM Companhia, Produto
WHERE Companhia.numCia= Produto.fabricante
AND Produto.nome IN
    (SELECT Compra.produto
     FROM Compra
     WHERE Comprador = 'Joao');
```

Esta subconsulta retorna um conjunto de valores. Não oc mais erros em tempo de execução.

Subconsultas que retornam relações

Você também pode usar: s > ALL R
 s > ANY R
 EXISTS R

Produto (nome, preço, categoria, fabricante)
 Encontre todos os produtos que são mais caros que todos os produtos fabricados pela "Farber"

```
SELECT nome
FROM Produto
WHERE preço > ALL (SELECT preço
                   FROM Produto
                   WHERE fabricante='Farber')
```

Subconsultas que retornam relações

Equivalente a:

```
SELECT Companhia.nome
FROM Companhia, Produto, Compra
WHERE Companhia.numCia= Produto.fabricante
AND Produto.nome = Compra.produto
AND Compra.comprador = 'Joao'
```

Esta consulta é equivalente a anterior?

Tenha cuidado com duplicações!

10

INTERSECT e EXCEPT: utilizando EXISTS

```
(SELECT R.A, R.B
 FROM R)
INTERSECT
(SELECT S.A, S.B
 FROM S)
```

→

```
SELECT R.A, R.B
FROM R
WHERE
EXISTS(SELECT *
        FROM S
        WHERE R.A= S.A and R.B= S.B)
```

```
(SELECT R.A, R.B
 FROM R)
EXCEPT
(SELECT S.A, S.B
 FROM S)
```

→

```
SELECT R.A, R.B
FROM R
WHERE
NOT EXISTS(SELECT *
           FROM S
           WHERE R.A= S.A and R.B= S.B)
```

14

Evitando duplicações

Múltiplas cópias

```
SELECT Companhia.nome
FROM Companhia, Produto, Compra
WHERE Companhia.numCia= Produto.fabricante
AND Produto.nome = Compra.produto
AND Compra.comprador = 'Joao'
```

Uma cópia

```
SELECT DISTINCT Companhia.nome
FROM Companhia, Produto, Compra
WHERE Companhia.numCia= Produto.fabricante
AND Produto.nome = Compra.produto
AND Compra.comprador = 'Joao'
```

11

Condições em Tuplas

Produto (nome, preço, categoria, fabricante)
 Companhia (numCia, nome, valorAcao, pais)
 Compra (comprador, vendedor, loja, produto, preço)

```
SELECT DISTINCT Companhia.nome
FROM Companhia, Produto
WHERE Companhia.numCia= Produto.fabricante
AND (Produto.nome,preço) IN
    (SELECT Compra.produto, Compra.preço
     FROM Compra
     WHERE Comprador = "Joao");
```

15

Removendo Duplicações

```
SELECT DISTINCT Companhia.nome
FROM Companhia, Produto
WHERE Companhia.numCia= Produto.fabricante
AND Produto.nome IN
    (SELECT Compra.produto
     FROM Compra
     WHERE Comprador = 'Joao');
```

```
SELECT DISTINCT Companhia.nome
FROM Companhia, Produto, Compra
WHERE Companhia.numCia= Produto.fabricante
AND Produto.nome = Compra.produto
AND Compra.comprador = 'Joao'
```

Agora elas são equivalentes

12

Consultas Correlacionadas

Filme (título, ano, diretor, duracao)
 Encontre os filmes cujo título apareça mais de uma vez..

```
SELECT DISTINCT titulo
FROM Filme AS x
WHERE ano <> ANY
    (SELECT ano
     FROM Filme
     WHERE titulo = x.titulo);
```

correlação

Note :

- (1) escopo das variáveis
- (2) pode ser escrito com um único SFW

16

```
SELECT DISTINCT titulo
FROM Filme AS x
WHERE ano <> ANY
      (SELECT ano
       FROM Filme
       WHERE titulo = x.titulo);
```

Filme

titulo	ano	diretor	duracao
shall we dance	1996	Massayuki Suo	136
shall we dance before sunrise	2004	Peter Chelsom	106
before sunset	1995	Richard Linklater	105
	2004	Richard Linklater	80

17

Exercícios

Carregar o banco de dados "BDProduto"

Produto (pname, preco, categoria, fabricante)
 Compra (comprador, vendedor, loja, produto)
 Companhia (cnome, valorAcao, pais)
 Pessoa (nomePess, tel, cidade)

- Ex #6: Encontre todas as pessoas que compraram produtos japoneses, mas que não compraram produtos brasileiros.
 Ex #7: Encontre o nome das pessoas que compraram produtos japoneses e brasileiros.
 Ex #8: Encontre o nome e a cidade onde moram pessoas que são as únicas moradoras desta cidade cadastradas no sistema.
 Ex #9: Encontre o nome das pessoas que tenham outros moradores da mesma cidade cadastrados no sistema.

21

Consultas Complexas Correlacionadas

Produto (nome, preco, categoria, fabricante, ano)

- Encontre os produtos (e seus fabricantes) que são mais caros que todos os produtos feitos pelo mesmo fabricante antes de 1972.

```
SELECT DISTINCT nome, fabricante
FROM Produto AS x
WHERE preco > ALL (SELECT preco
                  FROM Produto AS y
                  WHERE x.fabricante = y.fabricante
                  AND y.ano < 1972);
```

18

Agregação

```
SELECT Avg(preco)
FROM Produto
WHERE fabricante="Toyota"
```

SQL dá suporte a diversas operações de agregação:

SUM, MIN, MAX, AVG, COUNT

22

Divisão

Produto (nome, preco, categoria, fabricante)
 Compra (comprador, vendedor, loja, produto)

- Encontre as pessoas que compraram todos os produtos cadastrados.

```
SELECT DISTINCT comprador
FROM Compra AS c1
WHERE NOT EXISTS ((SELECT nome
                  FROM Produto)
                 EXCEPT
                 (SELECT produto
                  FROM Compra c2
                  WHERE c1.comprador=c2.comprador))
```

Verifica se o conjunto de produtos comprados por uma pessoa inclui todos os produtos existentes.

19

Agregação: Count

```
SELECT Count(*)
FROM Produto
WHERE ano > 1995
```

Com exceção de COUNT, todas as outras operações de agregação aplicam-se a um único atributo.

23

Divisão – sem utilizar EXCEPT

Produto (nome, preco, categoria, fabricante)
 Compra (comprador, vendedor, loja, produto)

- Encontre as pessoas que compraram todos os produtos cadastrados.

```
SELECT DISTINCT comprador
FROM Compra AS c1
WHERE NOT EXISTS (SELECT nome
                  FROM Produto AS p
                  WHERE NOT EXISTS
                        (SELECT produto
                         FROM Compra c2
                         WHERE c1.comprador=c2.comprador
                         AND p.nome = c2.produto))
```

Verifica se **não existe** nenhum produto que **não tenha** sido comprado por uma pessoa.

20

Agregação: Count

COUNT considera duplicações a não ser que seja explicitamente definido para não considerá-las.

```
SELECT Count(categoria)    o mesmo que Count(*)
FROM Produto
WHERE ano > 1995
```

Assim é melhor:

```
SELECT Count(DISTINCT categoria)
FROM Produto
WHERE ano > 1995
```

24

Agregação Simples

Compra(prodoto, data, preco, quantidade)

Exemplo 1: encontre o total de vendas de toda a base de dados.

```
SELECT Sum(preco * quantidade)
FROM Compra
```

Exemplo 1': encontre o total de vendas de leite

```
SELECT Sum(preco * quantidade)
FROM Compra
WHERE produto = 'leite'
```

25

Primeiro execute as cláusulas **FROM-WHERE** (data > "1/10") e depois agrupe por produto:

Produto	Data	Preco	Quantidade
banana	19/10	0.52	17
banana	22/10	0.52	7
leite	20/10	0.85	20
leite	21/10	0.85	15

29

Agregações Simples

Compra

Produto	Data	Preco	Quantidade
leite	20/10	0.85	20
banana	22/10	0.52	7
banana	19/10	0.52	17
leite	21/10	0.85	15

26

Depois, calcule a agregação

produto	vendaTotal
leite	\$29.75
banana	\$12.48

```
SELECT produto, Sum(preco*quantidade) AS vendaTotal
FROM Compra
WHERE data > "1/10"
GROUPBY produto
```

30

Agrupamento e Agregação

Geralmente, nós queremos agregar algumas partes da relação:

Compra(prodoto, data, preco, quantidade)

Example 2: **encontre o total de vendas por produto após 1/10.**

```
SELECT produto, Sum(preco*quantidade) AS vendaTotal
FROM Compra
WHERE data > "1/10"
GROUPBY produto
```

Vamos ver o que isto significa...

27

GROUP BY v.s. Consultas Aninhadas

```
SELECT produto, Sum(preco*quantidade) AS vendaTotal
FROM Compra
WHERE data > "1/10"
GROUPBY produto
```

```
SELECT DISTINCT x.produto,
(SELECT Sum(y.preco*y.quantidade)
FROM Compra y
WHERE x.produto = y.produto
AND y.data > '1/10') as total
FROM Compra x
WHERE x.data > "1/10"
```

Agrupamento e Agregação

1. Execute as cláusulas **FROM** e **WHERE**.
2. Agrupe as linhas pelo valor dos atributos em **GROUPBY**
3. Selecione uma tupla em cada grupo e aplique a operação de agregação

SELECT pode ter (1) atributos agrupados ou (2) agregados.

28

Outro Exemplo

Produto	totalVenda	qtdeMax
banana	\$12.48	17
leite	\$29.75	20

Para cada produto, qual o total de vendas e a maior quantidade vendida?

```
SELECT produto, Sum(preco * quantidade) AS totalVenda
FROM Compra
GROUP BY produto
Max(quantidade) AS qtdeMax
```

A Cláusula HAVING

Qual o total de vendas por produto, considerando somente aqueles que pelo menos 30 unidades foram vendidas.

```
SELECT produto, Sum(preco * quantidade)
FROM Compra
WHERE data > "1/9"
GROUP BY produto
HAVING Sum(quantidade) > 30
```

A cláusula HAVING contém condições sobre agregados.

33

Autor(idAutor, nome)
Autoria(idAutor, idDoc)

- Encontre os autores que escreveram pelo menos 10 documentos:
- Tentativa 1: com consulta aninhada

```
SELECT DISTINCT Autor.nome
FROM Autor
WHERE count(SELECT Autoria.idDoc
FROM Autoria
WHERE Autor.idAutor= Autor.idAutor)
> 10
```

Consulta escrita por um novato

A forma geral para Agrupamento e Agregação

```
SELECT S
FROM R1,...,Rn
WHERE C1
GROUP BY a1,...,ak
HAVING C2
```

Por que ?

S = pode conter atributos a₁,...,a_k e/ou agregados, mas NÃO PODE CONTER NENHUM OUTRO ATRIBUTO
C1 = qualquer condição sobre atributos de R₁,...,R_n
C2 = qualquer condição sobre agregados

34

Autor(idAutor, nome)
Autoria(idAutor, idDoc)

- Encontre os autores que escreveram pelo menos 10 documentos:
- Tentativa 2: com GROUP BY

```
SELECT Autor.nome
FROM Autor, Autoria
WHERE Autor.idAutor= Autoria.idAutor
GROUP BY Autor.idAutor, Autor.nome
HAVING count(Autoria.idDoc) > 10
```

Escrita por um expert

Não há necessidade de DISTINCT: automático devido o GROUP BY

38

A forma geral para Agrupamento e Agregação

```
SELECT S
FROM R1,...,Rn
WHERE C1
GROUP BY a1,...,ak
HAVING C2
```

Passos de avaliação:

8. Execute a parte FROM- WHERE part, e obtenha uma relação com todos os atributos em R₁,...,R_n
9. Agrupe pelos atributos a₁,...,a_k
10. Calcule os agregados em C2 e mantenha apenas os grupos que satisfazem C2
11. Calcule os agregados em S e retorne o resultado

35

Autor(idAutor, nome)
Autoria(idAutor, idDoc)
Vocabulario(idDoc, palavra)

- Encontre os autores que tem um vocabulário acima de 10000

```
SELECT Autor.nome
FROM Autor, Autoria, Vocabulario
WHERE Autor.idAutor= Autoria.idAutor
AND Autoria.idDoc= Vocabulario.idDoc
GROUP BY Autor.idAutor, Autor.nome
HAVING count(distinct Vocabulario.palavra) > 10000
```

Preste atenção nas duas últimas consultas: embora elas possam ser escritas utilizando consultas aninhadas, é melhor utilizar GROUP BY.

39

Agregação

Autor(idAutor, nome)
Documento(idDoc, titulo)
Autoria(idAutor, idDoc)
Vocabulario(idDoc, palavra)

36

Exercícios

Carregar o banco de dados "BDProduto"

Produto (pname, preco, categoria, fabricante)
Compra (comprador, vendedor, loja, produto)
Companhia (cnome, valorAcao, pais)
Pessoa (nomePess, tel, cidade)

- Ex #10: Para cada vendedor, escreva o seu nome e a soma das vendas realizadas.
- Ex #11: Para cada comprador, escreva o seu nome e a média de compras realizadas por categoria de produto.
- Ex #12: Para cada categoria de produto, obter o seu nome, sua média de preços, maior e menor preço.

40

Valores Null

Filme

titulo	ano	diretor	duracao
shall we dance	1996	Massayuki Suo	<i>null</i>
shall we dance	2004	<i>null</i>	106
before sunrise	<i>null</i>	Richard Linklater	105
before sunset	2004	Richard Linklater	80

Um *null* pode significar que:

- o valor não existe OU
- o valor existe mas ainda não é conhecido

41

Valores Null

É possível testar se um valor é nulo explicitamente:

- x IS NULL
- x IS NOT NULL

```
SELECT *
FROM Pessoa
WHERE idade < 25 OR idade >= 25
      OR idade IS NULL
```

Agora o resultado contém todas as pessoas.

45

Valores Null

- Se o valor de x for *Null* então a expressão $4*(3-x)/7$ resulta em *Null*
- Se o valor de x for *Null* então o teste $x = \text{“Joao”}$ resulta em **DESCONHECIDO**
- Em SQL há 3 valores booleanos:
 - FALSO = 0
 - DESCONHECIDO = 0.5
 - VERDADEIRO = 1

42

Valores Null em Junções

Produto(nome, categoria)
Compra(nomeProd, loja)

```
SELECT Produto.nome, Compra.loja
FROM Produto JOIN Compra ON
      Produto.nome = Compra.nomeProd
```

O mesmo que:

```
SELECT Produto.nome, Compra.loja
FROM Produto, Compra
WHERE Produto.nome = Compra.nomeProd
```

Mas produtos que nunca foram vendidos não aparecem no resultado !

46

Valores Null

- $C1 \text{ AND } C2 = \min(C1, C2)$
- $C1 \text{ OR } C2 = \max(C1, C2)$
- $\text{NOT } C1 = 1 - C1$

```
SELECT *
FROM Pessoa
WHERE (idade < 25) AND
      (altura > 150 OR peso > 80)
```

P.ex.
idade=20
altura=NULL
peso=90

Regra em SQL: inclui somente tuplas que resultam em VERDADEIRO

43

Valores Null e Outerjoins

Left outer joins em SQL:

Produto(nome, categoria)
Compra(nomeProd, loja)

```
SELECT Produto.nome, Compra.loja
FROM Produto LEFT OUTER JOIN Compra ON
      Produto.nome = Compra.nomeProd
```

47

Valores Null

Comportamento não intuitivo:

```
SELECT *
FROM Pessoa
WHERE idade < 25 OR idade >= 25
```

Algumas pessoas não são incluídas no resultado !

44

```
SELECT Produto.nome, Compra.loja
FROM Produto LEFT OUTER JOIN Compra ON
      Produto.nome = Compra.nomeProd
```

Produto

nome	categoria
lapis	papelaria
camera	fotografia
televisao	eletronicos

Compra

nomeProd	lojae
lapis	Cultura
camera	Ritz
camera	Ponto Frio

Resultado

nome	loja
lapis	Cultura
camera	Ritz
camera	Ponto Frio
televisao	<i>NULL</i>

48

Outer Joins

- **Left outer join:**
 - sempre inclui a tupla da esquerda
- **Right outer join:**
 - sempre inclui a tupla da direita
- **Full outer join:**
 - sempre inclui tanto a tupla da esquerda como da direita