REVISÃO DE SQL – PARTE 2

Sérgio Mergen

Esquema de Exemplo

Proj (idP, nome, duracao, custo, idD)

idD referencia depto

Func (*idFunc, nome, salario, idD, idChefe*)

idDepto referencia depto

idChefe referencia Func

Depto (idD, nome, predio, idDiretor)

idDiretor referencia Func

Aloc (idProj, idFunc, funcao)

idProj referencia proj

idFunc referencia func

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Junção Externa

- Junção externa (Outer Join): um registro de uma relação não necessariamente precisa ter uma correspondência
  - Left outer join
  - Right Outer Join
  - Full Outer Join

Junção Externa

assim

Junções com condição
left (outer) join **right (outer) join** **full outer join**

Condições de Junção
on <predicate> **using** (A_1, A_2, ..., A_n)

ou assim

Junções sem condição
natural left (outer) join **natural right (outer) join** **natural full outer join**

Junção Externa

Proj				
idP	**nome**	**duracao**	**custo**	**idD**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	2
3	Genesis	2	15.000	
4	Caos	10	100.000	1

Depto		
idD	**nome**	**predio**
1	TI	3
2	Marketing	2
3	RH	2

SELECT *
FROM **proj p** INNER JOIN **depto d** ON **p.idD = d.idD**

Resposta							
idP	**nome**	**duracao**	**custo**	**idD**	**idD**	**nome**	**predio**
1	ABC	3	12.000	1	1	TI	3
2	Lucrei	2	30.000	2	2	Marketing	2
4	Caos	10	100.000	1	1	TI	3

Junção Externa

Proj				
idP	**nome**	**duracao**	**custo**	**idD**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	2
3	Genesis	2	15.000	null
4	Caos	10	100.000	1

Depto		
idD	**nome**	**predio**
1	TI	3
2	Marketing	2
3	RH	2

SELECT *
FROM **proj p** LEFT OUTER JOIN **depto d** ON **p.idD = d.idD**

Resposta							
idP	**nome**	**duracao**	**custo**	**idD**	**idD**	**nome**	**predio**
1	ABC	3	12.000	1	1	TI	3
2	Lucrei	2	30.000	2	2	Marketing	2
4	Caos	10	100.000	1	1	TI	3
3	Genesis	2	15.000	null	null	null	nulll

Junção Externa

Proj				
idP	**nomeP**	**duracao**	**custo**	**idD**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	2
3	Genesis	2	15.000	null
4	Caos	10	100.000	1

Depto		
idD	**nomeD**	**predio**
1	TI	3
2	Marketing	2
3	RH	2

SELECT *
FROM **proj p** NATURAL RIGHT OUTER JOIN **depto d**

Resposta							
idP	**nomeP**	**duracao**	**custo**	**idD**	**idD**	**nomeD**	**predio**
1	ABC	3	12.000	1	1	TI	3
2	Lucrei	2	30.000	2	2	Marketing	2
4	Caos	10	100.000	1	1	TI	3
null	null	null	null	null	3	RH	2

Junção Externa

Proj				
idP	**nomeP**	**duracao**	**custo**	**idDepto**
1	ABC	3	12.000	1
2	Lucrei	2	30.000	2
3	Genesis	2	15.000	null
4	Caos	10	100.000	1

Depto		
idD	**nomeD**	**predio**
1	TI	3
2	Marketing	2
3	RH	2

SELECT * FROM ***proj p*** NATURAL FULL OUTER JOIN ***depto d***

Resposta							
idP	**nomeP**	**duracao**	**custo**	**idD**	**idD**	**nomeD**	**predio**
1	ABC	3	12.000	1	1	TI	3
2	Lucrei	2	30.000	2	2	Marketing	2
4	Caos	10	100.000	1	1	TI	3
3	Genesis	2	15.000	null	null	null	nulll
null	null	null	null	null	3	RH	2

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Conjuntos

- As operações com conjuntos **union, intersect,** e **except** operam sobre relações
  - ou sobre resultados de consultas

- Precisam ser *Union Compatible*
  - *Relações (ou consultas) envolvidas precisam ter*
    - *O mesmo número de colunas*
    - *Os mesmos tipos de dados para cada coluna*

Conjuntos

- Listar os códigos de funcionários que são chefes e diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

(SELECT **idChefe** FROM **func**)
INTERSECT
(SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
1

Conjuntos

- Listar os códigos de funcionários que são chefes ou diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

(SELECT **idChefe** FROM **func**)
UNION
(SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
3
1
2

Conjuntos

- Listar os códigos de funcionários que são chefes e não são diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

(SELECT **idChefe** FROM **func**)
EXCEPT
(SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
3

Conjuntos

- Cada uma das operações anteriores elimina as duplicatas automaticamente.
  - Para reter duplicatas deve-se utilizar as respectivas versões com **all**
    - **UNION ALL**
    - **INTERSECT ALL**
    - **EXCEPT ALL**

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Subconsultas

- SQL disponibiliza um mecanismo para aninhar consultas umas dentro de outras.

- Uma subconsulta
  - é uma expressão **SELECT-FROM-WHERE** que se encontra dentro de uma outra (sub)consulta.

- Pode aparecer dentro de diferentes partes
  - na cláusula **SELECT**
  - na cláusula **FROM**
  - na cláusula **WHERE**

Sumário

- Junção Externa
- Conjuntos
- Subconsulta
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Subconsultas dentro do SELECT

Template

```
SELECT
    (SELECT x FROM tab2 WHERE tab1.c = tab2.c) AS col1,
    (SELECT y FROM tab3 WHERE tab1.c = tab3.c) AS col2,
FROM
     tab1
```

- Cada uma das subconsultas deve retornar um valor simples (ex. x e y)
  - Ou seja, elas se comportam como funções

- Tipo de consulta é utilizada geralmente para formar registros compostos por colunas independentes
  - Encontradas a partir de uma tabela em comum (ex. tab1)

Subconsultas dentro do SELECT

Contar o número de projetos e o número de subordinados de cada funcionário

Func		
idF	**nome**	**idChefe**
1	Marcos	Null
2	Ana	1
3	João	1

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	2	Analista
2	2	Revisor

```
SELECT   nome,
     (SELECT count(*) FROM aloc a      WHERE f.idF =  a.idF)      AS proj,
     (SELECT count(*) FROM func sub WHERE f.idF =  sub.idChefe) AS sub
FROM func f
```

nome	proj	sub
Marcos	1	2
Ana	2	0
João	0	0

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Subconsultas dentro do FROM

Template

```
SELECT t1.x, ...
FROM (
        SELECT x
         FROM ...)    as t1
```

- Usada para criar uma tabela temporária (derivada) e usá-la como se fosse se fosse uma tabela qualquer
  - Pode-se dar um apelido para ela (ex. t1)
  - Seus atributos podem ser usados em qualquer lugar na consulta externa (ex. t1.x)

Subconsultas dentro do FROM

Listar a maior média salarial dentre todos os departamentos

Func			
idF	**nome**	**salario**	**idD**
1	Marcos	10000	1
2	Ana	6000	2
3	João	5000	1

```
SELECT  MAX(tab.media) AS maior_media
FROM   (SELECT  AVG(salario) AS media
            FROM func f
            GROUP BY   idD) AS tab
```

maior_media
7500

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Subconsultas dentro do WHERE

- São utilizadas habitualmente para efetuar algum tipo de comparações entre conjuntos de registros.

- Alguns operadores para comparação
  - Comparação direta
  - IN e EXISTS
  - NOT IN e NOT EXISTS
  - SOME/ANY
  - ALL

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Comparação Direta

Template

```
SELECT  ...
FROM  tab1
WHERE col1 OP (SELECT col2 FROM ...)
```

OP: Os operadores de comparação típicos podem ser usados (=, >, >=, <, <=, <>, !=, <=>)

A subconsulta deve devolver apenas um registro, e esse registro deve possuir a mesma quantidade de colunas que for usada na comparação (ex. col1 OP col2)

Comparação Direta

- Funcionários com salário superior ao maior custo de projeto

Func		
idF	**nome**	**salario**
1	Marcos	5.000
2	Ana	7.000
3	João	30.000

Proj		
idP	**nome**	**custo**
1	Lucrei	12.000
2	Caos	25.000
3	ABC	20.000

SELECT **nome** FROM **func**
WHERE **salario** > (SELECT MAX(**custo**)
FROM **proj**)

Resposta
f.nome
João

Esse tipo de comparação direta não pode ser realizada usando apenas junções

Comparação Direta

- Funcionários que tenham exatamente duas alocações

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista
2	1	Coordenador

```
SELECT nome FROM func f
WHERE 2 = (SELECT COUNT(*)
                FROM aloc a WHERE f.idFunc = a.idFunc )
```

Resposta
f.nome
Marcos

Pode-se comparar uma subconsulta com valores escalares (ex.2)

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

IN / EXISTS

Template do operador IN

```
SELECT ...
FROM ...
WHERE X IN (
  SELECT Y
  FROM ...
)
```

- A consulta devolve tuplas da consulta externa se o(s) atributo(s) X estiverem presentes em tuplas devolvidas pela consulta interna

- Os atributos devolvidos pela consulta interna (Y) devem ser equivalentes aos atributos (X)
  - O número de atributos deve ser igual
  - Os atributos devem ser de um mesmo tipo

IN / EXISTS

Template do operador EXISTS

```
SELECT ...
FROM t1, ...
WHERE EXISTS (
        SELECT ...
        FROM t2, ...
        WHERE t1.x = t2.y
)
```

- A consulta devolve tuplas da consulta externa se a consulta interna devolver um número de tuplas maior do que zero
- A consulta interna pode usar os atributos da consulta externa (variáveis de correlação)
  - Ex. t1.x

IN / EXISTS

- Listar os funcionários que estejam alocados em projetos.

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

IN / EXISTS

- Listar os funcionários que estejam alocados em projetos.

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Projeto	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

SELECT DISTINCT **nome**
FROM **func** NATURAL JOIN **aloc**

Resposta
f.nome
Marcos
João

Usando junção normal

Deve-se usar DISTINCT para garantir que o resultado não contenha duplicatas

IN / EXISTS

- Listar os funcionários que estejam alocados em projetos

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

```
SELECT   nome
FROM     func
WHERE idF IN  (SELECT idF
                      FROM aloc)
```

Resposta
f.nome
Marcos
João

Usando IN

O resultado fica diretamente livre de duplicatas

IN / EXISTS

- Listar os funcionários que estejam alocados em projetos

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

```
SELECT nome
FROM   func f
WHERE EXISTS  (SELECT * from aloc a
               WHERE f.idF = a.idF)
```

Resposta
f.nome
Marcos
João

Usando EXISTS

Forma alternativa que usa variáveis de correlação

IN / EXISTS

- O EXISTS tem um poder de expressão maior do que o IN
  - EXISTS permite usar qualquer tipo de expressão para testar existência
  - IN só permite comparações por igualdade

IN / EXISTS

- No exemplo
  - Deseja-se descobrir projetos que tenham o custo menor do que o salário de algum funcionário
    - Com IN
      - Só podemos saber se o projeto tem o custo igual ao salário, mas não se ele é menor

```
SELECT nomeP
FROM proj P
WHERE EXISTS
      (SELECT 1 FROM func f
      WHERE p.custo < f.salario )
```

```
SELECT nomeP
FROM proj p
WHERE custo IN
            (SELECT salario
             FROM  func)
```

Sumário

- Junção Externa
- Conjuntos
- Subconsultas
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

NOT IN / NOT EXISTS

- Pode-se usar o operador NOT junto com os operadores IN e EXISTS

- NOT IN
  - Devolve tuplas da consulta externa cujos atributos não tenham correspondência com tuplas da consulta interna

- NOT EXISTS
  - Devolve tuplas da consulta externa para os casos em que a consulta interna não retornar nenhuma tupla

NOT IN / NOT EXISTS

- Listar os funcionários que **não** estejam alocados em projetos.

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

NOT IN / NOT EXISTS

- Listar os funcionários que **não** estejam alocados em projetos.

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

SELECT **nome**
FROM **func** NATURAL LEFT JOIN **aloc a**
WHERE **a.idP** IS NULL

Resposta
f.nome
Ana

Usando junção externa

Deve-se incluir uma condição IS NULL na cláusula WHERE

NOT IN / NOT EXISTS

- Listar os funcionários que **não** estejam alocados em projetos.

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

```
SELECT nome
FROM   func
WHERE idF NOT IN
          (SELECT idF FROM aloc)
```

Resposta
f.nome
Ana

Usando NOT IN

Nenhuma condição extra é necessária

NOT IN / NOT EXISTS

- Listar os funcionários que **não** estejam alocados em projetos.

Func	
idF	**nome**
1	Marcos
2	Ana
3	João

Proj	
idP	**nome**
1	Lucrei
2	Caos

Aloc		
idP	**idF**	**função**
1	1	Coordenador
1	3	Analista

```
SELECT nome
FROM func f
WHERE NOT EXISTS
        (SELECT * FROM aloc a
              WHERE f.idF = a.idF)
```

f.nome
Ana

Usando NOT EXISTS

Forma alternativa que usa variáveis de correlação

NOT IN / NOT EXISTS

- NOT IN é **semanticamente** diferente de NOT EXISTS

- O NOT IN realiza comparações contra uma lista de valores calculada pela consulta
  - Caso essa lista contenha algum valor nulo
    - O NOT IN retornará um resultado vazio

NOT IN / NOT EXISTS

Ex. Encontre departamentos que não possuam projetos

Proj				
idP	**nomeP**	**duracao**	**custo**	**idD**
2	Lucrei	2	30.000	2
3	Genesis	2	15.000	null
4	Caos	10	100.000	2

Depto		
idD	**nomeD**	**predio**
2	Marketing	2
3	RH	2

select idD, *nomeD*
from *depto*
where not exists *(**select** 1 **from** proj p*
where *d.idD = p.idD)*

A consulta com NOT EXISTS devolve um registro

Resposta	
idD	**nomeD**
3	RH

NOT IN / NOT EXISTS

Ex. Encontre departamentos que não possuam projetos

Proj				
idP	**nomeP**	**duracao**	**custo**	**idD**
2	Lucrei	2	30.000	2
3	Genesis	2	15.000	null
4	Caos	10	100.000	2

Depto		
idD	**nomeD**	**predio**
2	Marketing	2
3	RH	2

select *nomeD*
from *depto*
where *idD* ***not in*** *(****select*** *idD* ***from*** *proj)*

Resposta	
idD	**nomeD**

A consulta com NOT IN devolve um resultado vazio

Isso ocorre devido a presença de um idD com valor NULO

Sumário

- Junção Externa
- Conjuntos
- Subconsulta
  - Dentro do SELECT
  - Dentro do FROM
  - Dentro do WHERE
    - Comparação direta
    - IN/EXISTS
    - NOT IN / NOT EXISTS
- Subconsulta vs conjuntos

Subconsultas em vez de conjuntos

- Em SQL, pode existir várias alternativas para encontrar o resultado desejado
  - Um exemplo é o uso de subconsultas em vez de operadores de conjunto

- Os próximos slides mostram exemplos de situações em que é possível trocar um pelo outro

  - Listar os códigos de funcionários que são chefes **ou** diretores

  - Listar os códigos de funcionários que são chefes **e** diretores

  - Listar os códigos de funcionários que são chefes e **não são** diretores

Subconsultas em vez de conjuntos

- Em SQL, pode existir várias alternativas para encontrar o resultado desejado
  - Um exemplo é o uso de subconsultas em vez de operadores de conjunto

- Os próximos slides mostram exemplos de situações em que é possível trocar um pelo outro
  - Listar os códigos de funcionários que são chefes **ou** diretores
  - Listar os códigos de funcionários que são chefes **e** diretores
  - Listar os códigos de funcionários que são chefes e **não são** diretores

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes ou diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

(SELECT **idChefe** FROM **func**)
UNION
(SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
3
1
2

Usando UNION
(como já vimos)

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes ou diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

SELECT **idF** FROM **func**

WHERE **idF** IN (SELECT **idChefe** FROM **func**)

OR **idF** IN (SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
3
1
2

Usando IN

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes ou diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

SELECT DISTINCT **f.idF** FROM **func f, func sub, depto dir**

WHERE **f.idF** = **sub.idChefe**

OR **f.idF** = **dir.idDiretor**

Resposta
idChefe
3
1
2

Usando JUNÇÃO

Subconsultas em vez de conjuntos

- Em SQL, pode existir várias alternativas para encontrar o resultado desejado
  - Um exemplo é o uso de subconsultas em vez de operadores de conjunto

- Os próximos slides mostram exemplos de situações em que é possível trocar um pelo outro
  - Listar os códigos de funcionários que são chefes **ou** diretores
  - Listar os códigos de funcionários que são chefes **e** diretores
  - Listar os códigos de funcionários que são chefes e **não são** diretores

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes e diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

(SELECT **idChefe** FROM **func**)
INTERSECT
(SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
1

Usando INTERSECT (como já vimos)

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes e diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

SELECT **idChefe** FROM **func**

WHERE **idChefe** IN (SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
1

Usando IN

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes e diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

SELECT DISTINCT **idChefe**

FROM **func** JOIN **depto** ON **idChefe** = **idDiretor**

Resposta
idChefe
1

Usando JOIN

Subconsultas em vez de conjuntos

- Em SQL, pode existir várias alternativas para encontrar o resultado desejado
  - Um exemplo é o uso de subconsultas em vez de operadores de conjunto

- Os próximos slides mostram exemplos de situações em que é possível trocar um pelo outro
  - Listar os códigos de funcionários que são chefes **ou** diretores
  - Listar os códigos de funcionários que são chefes **e** diretores
  - Listar os códigos de funcionários que são chefes e **não são** diretores

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes e não são diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

(SELECT **idChefe** FROM **func**)
EXCEPT
(SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
3

Usando EXCEPT (como já vimos)

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes e não são diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

SELECT **idChefe** FROM **func**
WHERE **idChefe** NOT IN (SELECT **idDiretor** FROM **depto**)

Resposta
idChefe
3

Usando NOT IN

Subconsultas em vez de conjuntos

- Listar os códigos de funcionários que são chefes e não são diretores

Func		
idF	**nome**	**idChefe**
1	Marcos	null
2	Pedro	3
3	João	1

Depto		
idD	**nome**	**idDiretor**
1	TI	1
2	RH	2

SELECT **idChefe**

FROM **func** LEFT JOIN **depto** ON **idChefe** = **idDiretor**

WHERE **idDiretor** IS NULL AND **idChefe** IS NOT NULL

Resposta
idChefe
3

Usando LEFT JOIN

Subconsultas em vez de conjuntos

- Alguns bancos dão suporte à UNION, mas não a INTERSECT ou EXCEPT
- Isso ocorre porque existem situações em que apenas a operação de UNION consegue resolver
- São situações em que não existe uma única origem que contenha todos os dados de interesse
- Ex. Como recuperar uma lista contendo todos nomes de departamentos e nomes de projetos?
  - A única forma de resolver é com UNION

Atividade Individual

- Os exercícios a seguir são baseados em dados representados pelo esquema abaixo

movie (movie_id, title, release_year)

person (person_id, person_name)

Movie_cast (movie_id, person_id, character_name, cast_order)
    movie_id referencia movie
    person_id referencia person

- O banco de dados está disponível no moodle

Atividade Individual

1. Exibir nomes de artistas que contracenaram em filmes. Só deve ser exibido o nome se todos os seus filmes forem anteriores a 1950

2. Exibir a quantidade de atores que contracenaram em mais do que 5 filmes

3. Exibir os nomes de personagens que são nomes de filmes.

REVISÃO DE SQL – PARTE 2

Sérgio Mergen