



UNIVERSIDADE  
LUSÓFONA

# Base de Dados

SQL: Junções Horizontais

2022/2023

# Sumário

## Correção Trabalho de Casa

- Integridade Referencial
- Foreign Key

## Hoje

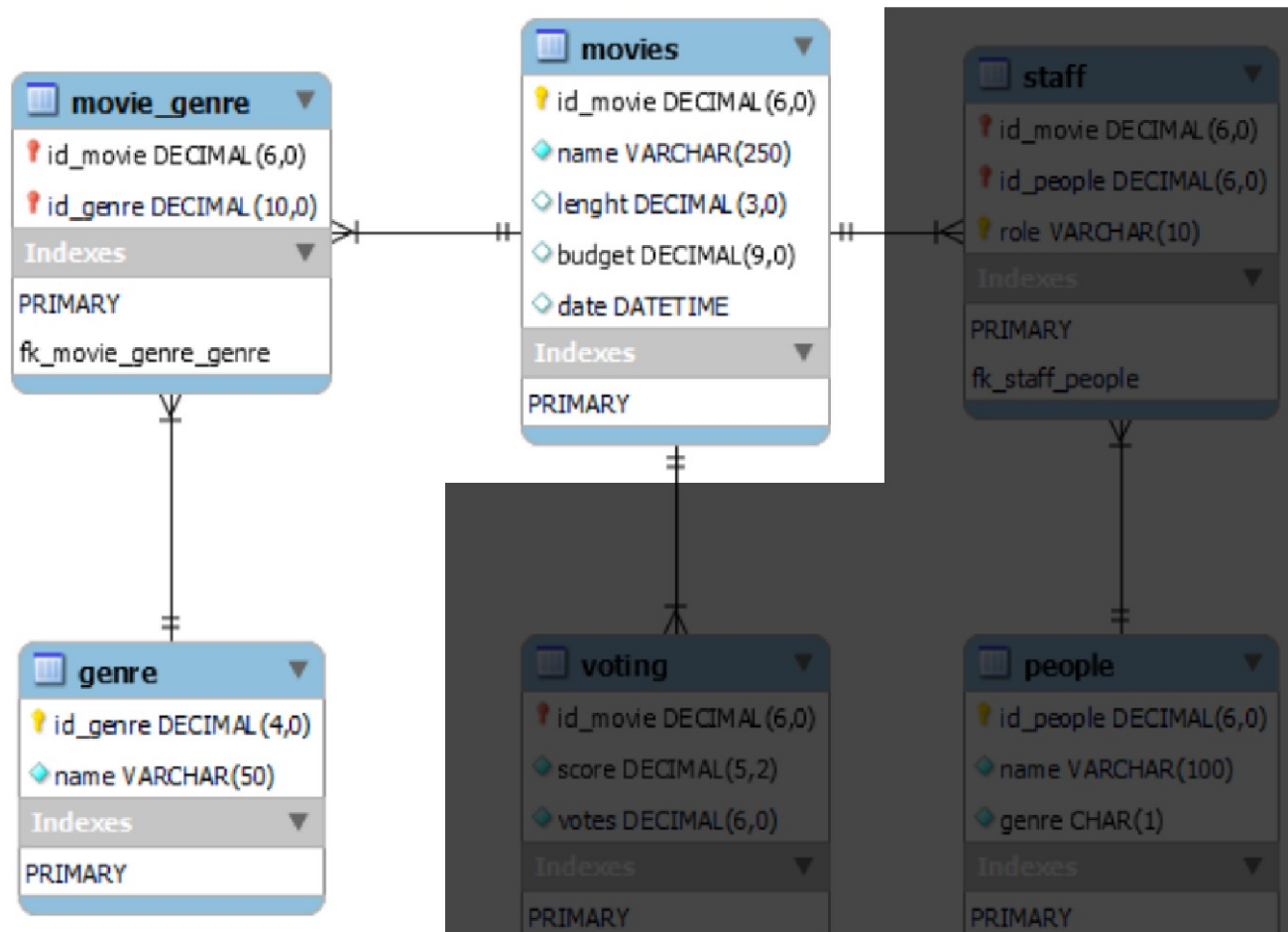
- Junção Horizontal (JOIN)
  - Interna (INNER)
  - Externa (OUTER)
  - Extra: nested queries



# Trabalho de Casa Aula 9

# Trabalho de Casa

Escreva o código SQL que permite criar as relações (movies, movie\_genre, genre) definidas no modelo 1 do enunciado do projeto, incluindo as respectivas restrições de integridade referencial definidas no esquema físico:



# Trabalho de Casa (2)

```
CREATE TABLE genre (  
    id_genre DECIMAL(4,0),  
    name VARCHAR(50),  
    PRIMARY KEY(id_genre)  
);
```

```
CREATE TABLE movies (  
    id_movie DECIMAL(6,0),  
    name VARCHAR(250),  
    length DECIMAL(3,0),  
    budget DECIMAL(9,0),  
    date DATETIME,  
    PRIMARY KEY(id_movie)  
);
```

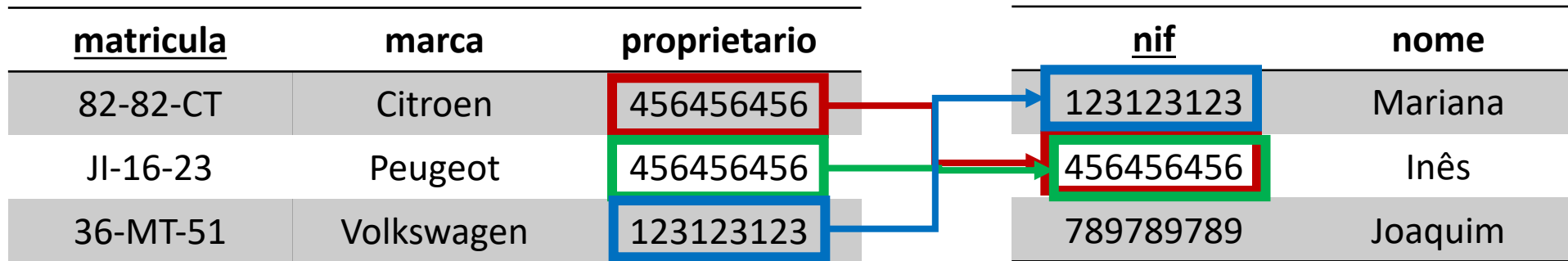
```
CREATE TABLE movie_genre (  
    id_genre DECIMAL(4,0),  
    id_movie DECIMAL(6,0),  
    PRIMARY KEY(id_movie, id_genre),  
    CONSTRAINT fk_movie_genre_genre FOREIGN KEY (id_genre) REFERENCES genre(id_genre),  
    CONSTRAINT fk_movie_genre_movie FOREIGN KEY (id_movie) REFERENCES movies(id_movie)  
);
```



# Hoje

SQL: INNER JOIN, OUTER JOIN, nested query

# SQL – Junções (JOIN)



- Conseguimos recuperar informação de cada relação:
  - SELECT \* FROM veiculo;
  - SELECT \* FROM proprietario;
- Como selecionar veículos e os respectivos proprietários?
  - Utilizando JOIN (junção)

# SQL – INNER JOIN

- Recuperar informação de veículos e respetivos proprietários:

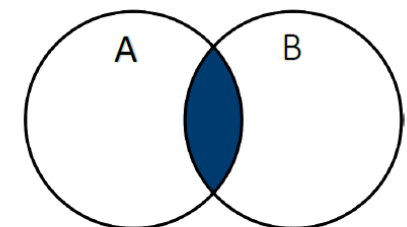
```
SELECT
    matricula, marca, proprietario.nif, proprietario.nome
FROM
    veiculo
JOIN
    proprietario
ON
    veiculo.nif = proprietario.nif;
```

- Atributos indicados nif = nif nas duas relações são associados

<u>matricula</u>	marca	nif
82-82-CT	Citroen	456456456
Jl-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123

<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim

matricula	marca	nif	nome
82-82-CT	Citroen	456456456	Inês
Jl-16-23	Peugeot	456456456	Inês
36-MT-51	Volkswagen	123123123	Mariana

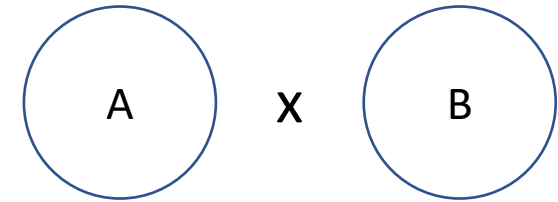


**INNER JOIN**



# JOIN – How does it work?

- Junção é obtido através do produto cartesiano de conjuntos
  - Todas as combinações de elementos de A e elementos de B
- veículo x proprietario =



$A = \{a1, a2\}$  e  $B = \{b1\}$

$A \times B = \{(a1,b1), (a2,b1)\}$

<u>matricula</u>	marca	nif
82-82-CT	Citroen	456456456
JI-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123

x

<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim

# JOIN – How does it work?

- Produto cartesiano veiculo X proprietario

matricula	marca	veiculo.nif	proprietario.nif	nome
82-82-CT	Citroen	456456456	123123123	Mariana
JI-16-23	Peugeot	456456456	123123123	Mariana
36-MT-51	Volkswagen	<u>123123123</u>	<u>123123123</u>	Mariana
82-82-CT	Citroen	<u>456456456</u>	<u>456456456</u>	Inês
JI-16-23	Peugeot	<u>456456456</u>	<u>456456456</u>	Inês
36-MT-51	Volkswagen	123123123	456456456	Inês
82-82-CT	Citroen	456456456	789789789	Joaquim
JI-16-23	Peugeot	456456456	789789789	Joaquim
36-MT-51	Volkswagen	123123123	789789789	Joaquim


Quais as correspondências **veiculo.nif = proprietario.nif**?

# JOIN – How does it work?

- Produto cartesiano *veiculo* X *proprietario* e seleção de valores iguais para *nif*

<b>matricula</b>	<b>marca</b>	<b>veiculo.nif</b>	<b>proprietario.nif</b>	<b>nome</b>
36-MT-51	Volkswagen	123123123	123123123	Mariana
82-82-CT	Citroen	456456456	456456456	Inês
JI-16-23	Peugeot	456456456	456456456	Inês

produto cartesiano

  
*veiculo* × *proprietario*

# JOIN – How does it work?

- Produto cartesiano veiculo X proprietario e seleção de valores iguais para nif

matricula	marca	veiculo.nif	proprietario.nif	nome
36-MT-51	Volkswagen	123123123	123123123	Mariana
82-82-CT	Citroen	456456456	456456456	Inês
Jl-16-23	Peugeot	456456456	456456456	Inês

produto cartesiano

$\sigma_{\text{veiculo.nif}=\text{proprietario.nif}}(\text{veiculo} \times \text{proprietario})$

selecao de tuplos onde valores nif são iguais

# JOIN – How does it work?

- Produto cartesiano `veiculo` X `proprietario` e seleção de valores iguais para `nif`

matricula	marca	veiculo.nif	proprietario.nif	nome
36-MT-51	Volkswagen	123123123	123123123	Mariana
82-82-CT	Citroen	456456456	456456456	Inês
JI-16-23	Peugeot	456456456	456456456	Inês

Em A.R. uma junção pode ser representada de forma mais simples utilizando o símbolo  $\bowtie$

$veiculo \bowtie_{veiculo.nif=proprietario.nif} proprietario$

É equivalente a:

$\sigma_{veiculo.nif=proprietario.nif}(veiculo \times proprietario)$

# SQL – NATURAL JOIN

- Recuperar informação de veículos e respetivos proprietários:

```
SELECT
FROM      matricula, marca, veiculo.nif, proprietario.nome
          veiculo
          NATURAL JOIN proprietario;
```

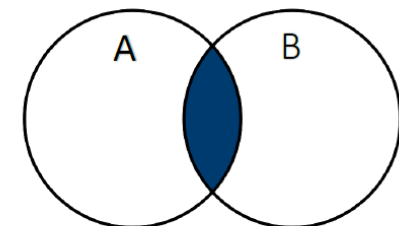
- Atributos comuns nas duas relações são associados

*veiculo* ⋈ *proprietario*

matricula	marca	nif	nome
82-82-CT	Citroen	456456456	Inês
JI-16-23	Peugeot	456456456	Inês
36-MT-51	Volkswagen	123123123	Mariana

<u>matricula</u>	marca	nif
82-82-CT	Citroen	456456456
JI-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123

<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim



**INNER JOIN**

# SQL – INNER JOIN

- Recuperar informação de veículos e respetivos proprietários:

```
SELECT      matricula, marca, proprietario.nif, proprietario.nome
FROM        veiculo
JOIN        proprietario
ON          veiculo.proprietario = proprietario.nif;
```

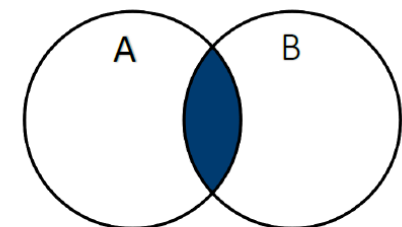
- Atributos indicados proprietario = nif nas duas relações são associados

*veiculo* ⋈<sub>nif=proprietario</sub> *proprietario*

matricula	marca	nif	nome
82-82-CT	Citroen	456456456	Inês
JI-16-23	Peugeot	456456456	Inês
36-MT-51	Volkswagen	123123123	Mariana

matricula	marca	proprietario
82-82-CT	Citroen	456456456
JI-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123

nif	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim



**INNER JOIN**

# SQL – OUTER JOIN

- Recuperar informação de proprietarios e respetivos veículos (caso existam):

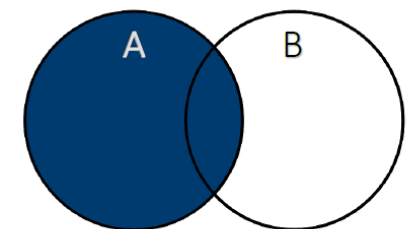
```
SELECT
FROM      proprietario
LEFT OUTER JOIN
      veiculo
ON        veiculo.proprietario = proprietario.nif;
```

- Atributos indicados proprietario = nif nas duas relações são associados

<u>matricula</u>	marca	proprietario
82-82-CT	Citroen	456456456
JI-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123

<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim

matricula	marca	nif	nome
82-82-CT	Citroen	456456456	Inês
JI-16-23	Peugeot	456456456	Inês
36-MT-51	Volkswagen	123123123	Mariana
NULL	NULL	789789789	Joaquim



**LEFT OUTER JOIN**



# SQL – OUTER JOIN

- Recuperar informação de proprietários sem veículos associados:

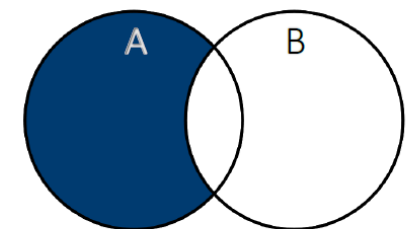
```
SELECT      matricula, marca, proprietario.nif, proprietario.nome
FROM        proprietario
LEFT OUTER JOIN
            veiculo
ON          veiculo.proprietario = proprietario.nif;
WHERE veiculo.proprietario IS NULL
```

- Atributos indicados proprietario = nif nas duas relações são associados

<u>matricula</u>	marca	proprietario
82-82-CT	Citroen	456456456
Jl-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123
20-XZ-23	Tesla	145987456

<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim

matricula	marca	nif	nome
NULL	NULL	789789789	Joaquim



**LEFT EXCLUSIVE OUTER JOIN**

# SQL – OUTER JOIN

- Recuperar informação de veiculos sem proprietarios associados:

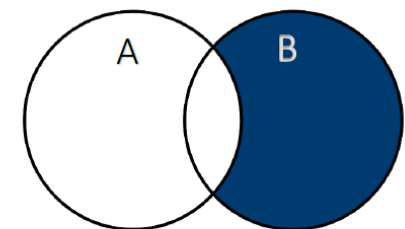
```
SELECT      matricula, marca, proprietario.nif, proprietario.nome
FROM        proprietario
            RIGHT OUTER JOIN
            veiculo
            ON      veiculo.proprietario = proprietario.nif;
WHERE proprietario.nif IS NULL
```

- Atributos indicados proprietario = nif nas duas relações são associados

<u>matricula</u>	marca	proprietario
82-82-CT	Citroen	456456456
Jl-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123
20-XZ-23	Tesla	145987456

<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim

matricula	marca	nif	nome
20-XZ-23	Tesla	NULL	NULL



**RIGHT EXCLUSIVE OUTER JOIN**

# SQL – OUTER JOIN

- Recuperar informação de veiculos e respetivos proprietarios (caso existam):

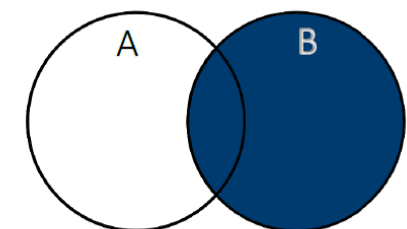
```
SELECT
    matricula, marca, proprietario.nif, proprietario.nome
FROM
    proprietario
RIGHT OUTER JOIN
    veiculo
ON
    veiculo.proprietario = proprietario.nif;
```

- Atributos indicados proprietario = nif nas duas relações são associados

matricula	marca	nif	nome
82-82-CT	Citroen	456456456	Inês
JI-16-23	Peugeot	456456456	Inês
36-MT-51	Volkswagen	123123123	Mariana
20-XZ-23	Tesla	NULL	NULL

matricula	marca	proprietario
82-82-CT	Citroen	456456456
JI-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123
20-XZ-23	Tesla	145987456

nif	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim



**RIGHT OUTER JOIN**

# SQL – OUTER JOIN

- Recuperar informação de veiculos sem proprietários e proprietarios sem veículos associados:

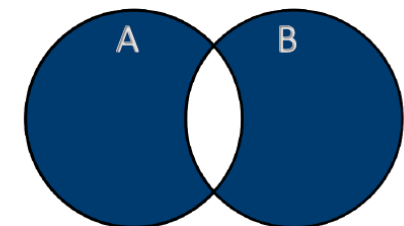
```
SELECT
    matricula, marca, proprietario.nif, proprietario.nome
FROM
    proprietario
LEFT OUTER JOIN
    veiculo
ON
    veiculo.proprietario = proprietario.nif;
WHERE veiculo.proprietario IS NULL
UNION ALL
SELECT
    matricula, marca, proprietario.nif, proprietario.nome
FROM
    proprietario
RIGHT OUTER JOIN
    veiculo
ON
    veiculo.proprietario = proprietario.nif;
WHERE proprietario.nif IS NULL
```

- Atributos indicados proprietario = nif nas duas relações são associados

matricula	marca	nif	nome
20-XZ-23	Tesla	NULL	NULL
NULL	NULL	789789789	Joaquim

matricula	marca	proprietario
82-82-CT	Citroen	456456456
JI-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123
20-XZ-23	Tesla	145987456

nif	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim



**FULL OUTER EXCLUSIVE JOIN**

# SQL – OUTER JOIN

- Recuperar informação de veiculos e proprietários com ou sem associações:

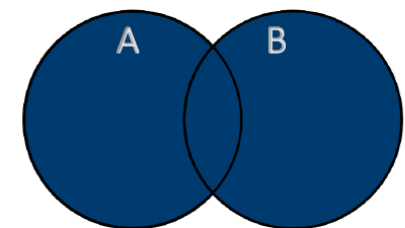
```
SELECT      matricula, marca, proprietario.nif, proprietario.nome
FROM        proprietario
LEFT OUTER JOIN
            veiculo
ON          veiculo.proprietario = proprietario.nif

UNION
SELECT      matricula, marca, proprietario.nif, proprietario.nome
FROM        proprietario
RIGHT OUTER JOIN
            veiculo
ON          veiculo.proprietario = proprietario.nif;
WHERE proprietario.nif IS NULL
```

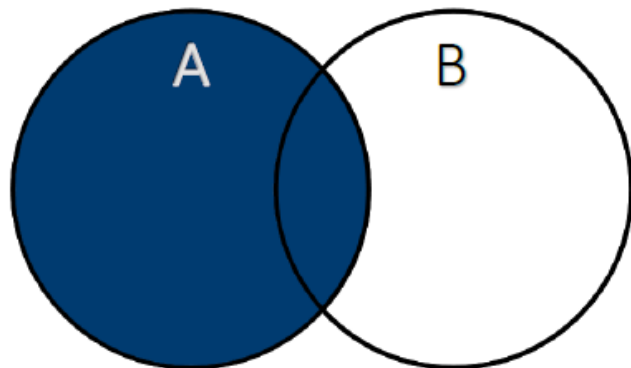
- Atributos indicados proprietario = nif nas duas relações são associados

<u>matricula</u>	marca	proprietario
82-82-CT	Citroen	456456456
Jl-16-23	Peugeot	456456456
36-MT-51	Volkswagen	123123123
20-XZ-23	Tesla	145987456

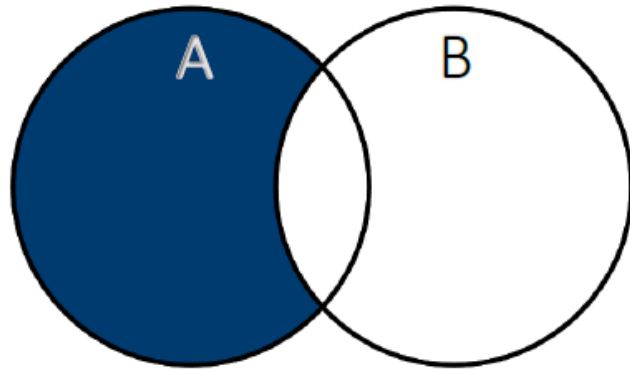
<u>nif</u>	nome
123123123	Mariana
456456456	Inês
789789789	Joaquim



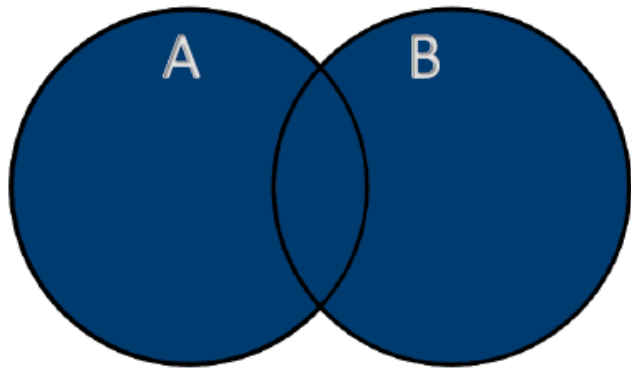
**FULL OUTER JOIN**



**LEFT INCLUSIVE**

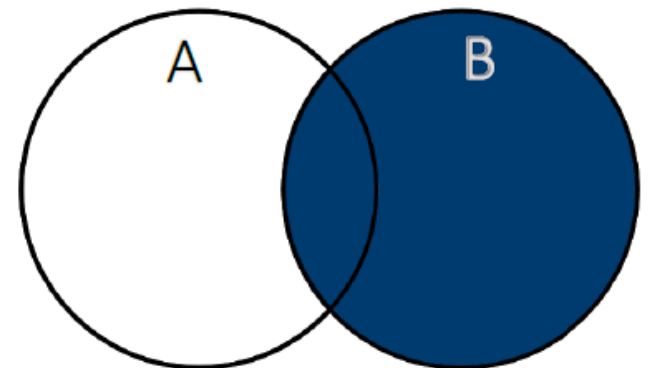


**LEFT EXCLUSIVE**

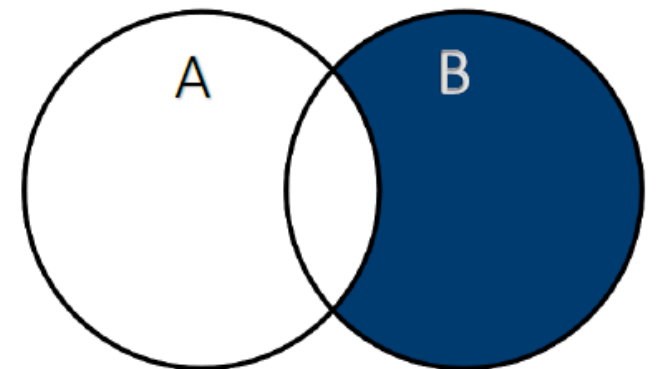


**FULL OUTER INCLUSIVE**

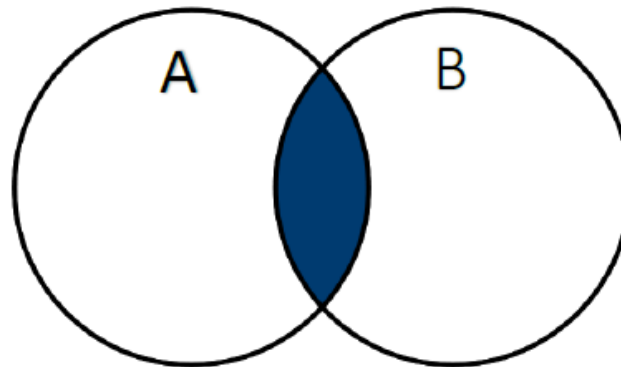
SQL JOINS	
<b>LEFT INCLUSIVE</b> SELECT [Select List] FROM TableA A LEFT OUTER JOIN TableB B ON A.Key= B.Key	<b>RIGHT INCLUSIVE</b> SELECT [Select List] FROM TableA A RIGHT OUTER JOIN TableB B ON A.Key= B.Key
<b>LEFT EXCLUSIVE</b> SELECT [Select List] FROM TableA A LEFT OUTER JOIN TableB B ON A.Key= B.Key WHERE B.Key IS NULL	<b>RIGHT EXCLUSIVE</b> SELECT [Select List] FROM TableA A LEFT OUTER JOIN TableB B ON A.Key= B.Key WHERE A.Key IS NULL
<b>FULL OUTER INCLUSIVE</b> SELECT [Select List] FROM TableA A FULL OUTER JOIN TableB B ON A.Key = B.Key	<b>FULL OUTER EXCLUSIVE</b> SELECT [Select List] FROM TableA A FULL OUTER JOIN TableB B ON A.Key = B.Key WHERE A.Key IS NULL OR B.Key IS NULL
<b>INNER JOIN</b> SELECT [Select List] FROM TableA A INNER JOIN TableB B ON A.Key = B.Key	



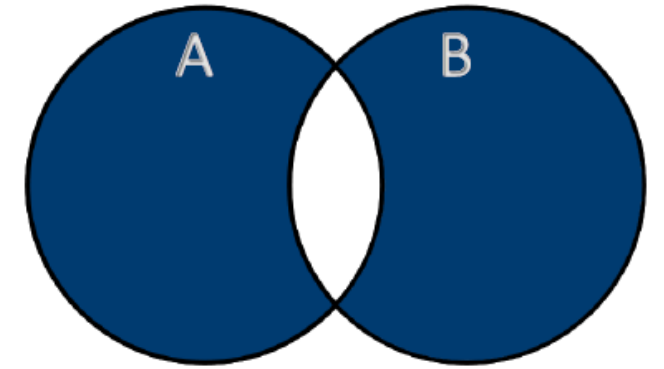
**RIGHT INCLUSIVE**



**RIGHT EXCLUSIVE**



**INNER JOIN**



**FULL OUTER EXCLUSIVE**

# SQL: Nested Queries

- Queries que contém outras queries.
- São muitas vezes difíceis de otimizar
- As queries interiores podem aparecer em várias partes diferentes

```
SELECT nome  
FROM estudante  
WHERE eid IN (SELECT eid FROM inscrito);
```



Query  
Exterior

Query  
Interior

# SQL: Nested Queries

- Obter lista de nomes de alunos inscritos à UC 215

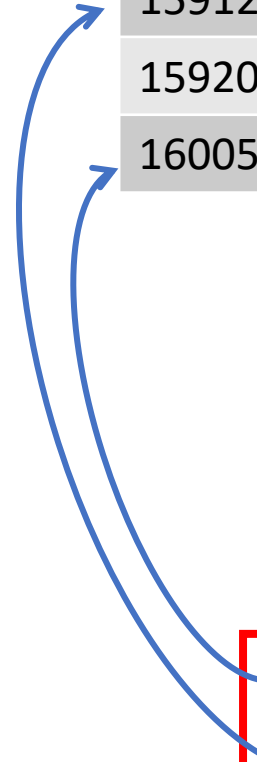
```
SELECT eid  
FROM inscrito  
WHERE uid=215
```

**estudante**(eid, nome, login, idade)

eid	nome	idade	Login
15912	Mariza	19	Mariza@lem
15920	Rui	31	rui@li
16005	Mafalda	23	Mafalda@lem

**inscrito**(eid, uid, nota)

eid	uid	nota
15912	222	16
15920	218	18
15920	230	12
16005	215	9
15912	215	14





# SQL: Nested Queries

- Obter lista de nomes de alunos inscritos à UC 215

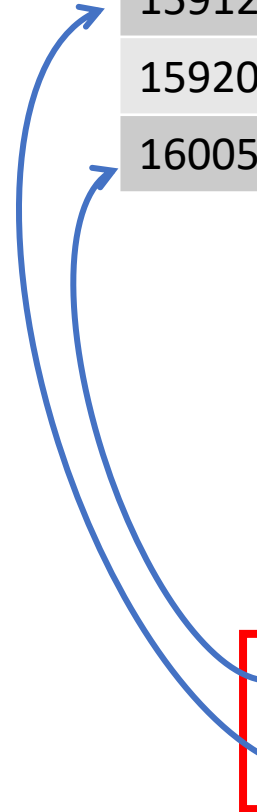
```
SELECT nome
FROM estudante
WHERE
eid IN (SELECT eid
        FROM inscrito
        WHERE uid=215);
```

**estudante**(eid, nome, login, idade)

eid	nome	idade	Login
15912	Mariza	19	Mariza@lem
15920	Rui	31	rui@li
16005	Mafalda	23	Mafalda@lem

**inscrito**(eid, uid, nota)

eid	uid	nota
15912	222	16
15920	218	18
15920	230	12
16005	215	9
15912	215	14



# SQL: Nested Queries

- ALL -> Necessita satisfazer a expressão para todos os tuplos resultado da subquery
- ANY -> Necessita satisfazer a expressão para pelo menos um tuplo resultado da subquery
- IN -> Equivalente a '=ANY()' (valor de atributo contido na subquery)

# SQL: Nested Queries

- Obter lista de nomes de alunos inscritos à UC 215

```
SELECT nome
FROM estudante
WHERE
eid = ANY(SELECT eid
          FROM inscrito
          WHERE uid=215);
```

**estudante**(eid, nome, login, idade)

eid	nome	idade	Login
15912	Mariza	19	Mariza@lem
15920	Rui	31	rui@li
16005	Mafalda	23	Mafalda@lem

**inscrito**(eid, uid, nota)

eid	uid	nota
15912	222	16
15920	218	18
15920	230	12
16005	215	9
15912	215	14

# SQL: Nested Queries

- Encontrar eid de alunos com notas iguais ou superiores à melhor nota de BD (UC=222)

```
SELECT eid
FROM inscrito
WHERE
    nota > ALL(SELECT nota
                FROM inscritos WHERE
                uid = 222);
```

**estudante**(eid, nome, login, idade)

eid	nome	idade	Login
15912	Mariza	19	Mariza@lem
15920	Rui	31	rui@li
16005	Mafalda	23	Mafalda@lem

**inscrito**(eid, uid, nota)

eid	uid	nota
15912	222	16
15920	218	18
15920	230	12
16005	215	9
15912	215	14

# SQL: Nested Queries

- Encontrar o aluno com o eid mais alto inscrito a alguma UC

```
SELECT eid, nome
FROM
    estudante,
WHERE eid IN (
    SELECT MAX(eid)
    FROM inscritos);
```

**estudante**(eid, nome, login, idade)

eid	nome	idade	Login
15912	Mariza	19	Mariza@lem
15920	Rui	31	rui@li
16005	Mafalda	23	Mafalda@lem

**inscrito**(eid, uid, nota)

eid	uid	nota
15912	222	16
15920	218	18
15920	230	12
16005	215	9
15912	215	14

# Exercícios – GitHub aula10

- Exercícios publicados no repositório git:  
<https://github.com/ULHT-BD/aula10>

## aula07

---

Nas aulas anteriores utilizámos a linguagem SQL para efetuar consultas de dados, recuperando dados e efetuando operações sobre estes de forma a obter a informação desejada. Nesta aula olhamos para a cláusula CREATE e respetica sintaxe, um comando que nos permite criar novas relações de acordo com o esquema definido. Bom trabalho!

### 0. Requisitos

### 1. CREATE TABLE

### 2. Tipos de Dados

### 3. Restrições de Integridade



Obrigado.